1 Introduction

This document describes best practices for building SAP BPC 5.0 reports. Before starting with the best practices, some discussion about EvGET and EvDRE might be helpful.

EvGET is the original function that was used to retrieve data from the database. To increase performance, an enhanced formula, EvDRE, was developed. For more information about EvDRE, see the WebExcel Help.

An EvGET formula always creates an MDX query, which queries the application. An EvDRE formula creates an MDX query, a SQL query, or both, depending on
the type of members being retrieved. By default, a SQL query is faster than an MDX query. However, sometimes EvDRE creates SQL queries that turn out to be less efficient than an MDX query.

EvDRE does not always perform better than an EvGET function in all cases. In this document, we discuss guidelines for when to use EvGET and when to use EvDRE. However, to be completely sure which formula is the best, you should try both and measure the query time.

2 General best practices for reporting

2.1 Think about the measures you want to retrieve

Try to use the measure in which the data is stored in the database. However, if the application is not big and the concurrency level is low you will not benefit much from this setup.

However, in test situations with high concurrency level (>50 concurrent users) and a big OLAP application (> 2 gigabyte), a huge performance improvement was achieved when using this approach. So if you need Periodic data in an YTD application it is better to retrieve the YTD data for time T and time T-1 and subtract these values from each other. The calculations are performed in Excel (on the client side) which increases performance.

1. If the Account-type dimension is included in the expansion, retrieve the account type using EvPRO, for example, (=EVPRO ('Application Name', <'Account'>, "ACCTYPE")). The account type indicates the nature of the account (AST, LEQ, INC, EXP). For AST and LEQ no subtraction is needed.

2. If the time period is JAN, subtract the previous period.

3. Use the following type of formula to calculate the periodic value:

   \[=\text{IF}(\text{OR}((\text{RIGHT}(E8, 3)="JAN"), T17="AST", T17="LEQ"), I17, I17-U17)\]

The following table describes the parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(RIGHT(E8,3))</td>
<td>Links to the Time period</td>
</tr>
<tr>
<td>$T17</td>
<td>Links to the cell from which the Account Type is retrieved</td>
</tr>
<tr>
<td>I17</td>
<td>Links to the cell from which the data for period T is retrieved</td>
</tr>
<tr>
<td>U17</td>
<td>Links to the cell from which the data for period T-1 is retrieved</td>
</tr>
</tbody>
</table>

2.2 Do not use a lot of comment formulas

In general, the use of EvCOM functions slows down the report performance significantly. The EvCOM function has a lot of overhead time. It takes about five to ten seconds to refresh a report with only one EvCOM formula. So if you have a lot of comments in your application and want to retrieve all, the best way is to build one report that retrieves all the text data, instead of having numerous reports with only a few comment lines in each report. Consolidating the comments to one report will save you the overhead time.

If using one report to retrieve all comments is not practical for you, then replace all the EvCOM functions with another function name such as: 'EvCON' (make sure that the function name is not used in your application). Then add a macro to rename all the EvCON to EvCOM when sending and retrieving data.
2.3 Prevent retrieving MDX calculated formulas as much as possible

Reports that contain KPIs and other calculated members are typically the main bottleneck for reporting performance. To achieve the best possible performance, we recommend that you avoid retrieving accounts that contain dimension formulas.

For KPI reports this is a bit difficult, but there are two ways to minimize the pain. The first way is to retrieve a sheet with the base data that you need for your calculations (this can be both base-level data as parent members) and calculate the KPIs in Excel formulas. This approach generates the best performance.

The second way is to prevent using dimension formulas in your reports. Analysis Services creates locks when calculating formulas. For example, if you have twenty reports that have five KPIs each, and only one user is retrieving data, then Analysis Services creates 100 (20*5) locks during data retrieval process. More numbers of locks slows down the performance and the overall report performance worsens with the increase in concurrency. Therefore, try to avoid the use of dimension formulas in your KPIs.

2.4 Turn off the automatic restore option during refresh

In 4.2 SP3 QFE2 there is a new workbook option "Allow cancellation of Expand and Refresh." This option enables the automatic backup of all the sheets in the workbook when using the restore option. However, it is better to turn this option off. During testing, copying all the worksheets at each refresh took seven minutes and deleting all the double sheets took the same amount time. So you can achieve fourteen minutes of performance improvement by just selecting a check box in the workbook options. This is very helpful if you have large workbooks.

2.5 Use EvTIM for comparable time periods

A lot of reports compare data from different periods with each other in the columns. To prevent a lot of maintenance, it is easy to create reports for comparable periods depending on the time you have in the Current View for your current reporting period. To do this, enter the EvCVW function for the Time dimension in the column that retrieves the current period. For the other columns you enter an EvTIM function at the top of the column. This function generates a valid time ID based on another ID and you can specify an “offset.” So by using the EvCVW function as the basis and entering the offset on “-1” you can retrieve the current month and the previous month.

3 Best practices for using EvDRE

3.1 Be careful with multiple dimensions in Row and ColKeyRange

Performance degrades rapidly in a large application if you have more than 1x2 dimensions in rows x columns. Reports with two dimensions in both the row and column become very slow when the concurrency level in the system is medium or high. Reports with more then 2 x 2 dimensions in the row and column may not finish, even if you refresh them during peak concurrency.

Analysis Services generates more locks when you have multiple dimensions in rows and columns. As a result, BPC users may experience poor performance.
Therefore, it is a good idea to reduce the number of row and column dimensions using multiple EvDRE functions. However, if you are not able to decrease the number of column or row dimensions, it is better to use EvGTS instead of EvDRE. See Best practices for more information.

3.2 Sum base members via the PagekeyRange to create SQL queries

When a report contains only base members (members at which data input is done), use EvDRE because this is definitely the fastest way. The use of base members also applies to the PagekeyRange. If a calculated member is mentioned in the PagekeyRange a less efficient MDX query is created. You can bypass a calculated member by stating the base members in the PagekeyRange. For detailed instructions see the example below.

T_REP_IFRS is a calculated (datasrc) member. The base members are I_IFRS, A_LC_IFRS, A_LC_IFRS and A_HQ_IFRS.

Instead of entering the member T_REP_IFRS in the PagekeyRange, you can sum the base members by stating them in the PagekeyRange. This results in a SQL query that is faster than an MDX query.

Note that this works only if all dimension members in the ColKeyRange, RowKeyRange and PagekeyRange are base members.

3.3 Use the correct options in EvDRE

Always use the correct options when using the EvDRE function. For a list of available options, refer to the WebExcel Help.

You can use multiple options by separating them with commas. Sometimes it can be useful to build one EvDRE that only does the expansion (with the ExpandOnly option) and build a few static EvDRE functions to retrieve data.

By default, EvDRE displays the Column and RowKeyRanges, but in a lot of reports you may want to display only the description. In this situation you should use the HideColKeys and HideRowKeys options.
3.4 Remove parent members from the key range when using Excel formulas for parent calculations

If you have a report with, for example, Accounts in the rows, and you have all base members in your PagekeyRange, and some calculated members in the RowKeyRange, EvDRE creates an MDX query for these calculated members. However, if you create a formula, for example, a SUM formula, and do not remove the calculated members from the RowKeyRange, you will not benefit from the SUM formula, because EvDRE will generate a MDX query.

To benefit from the SUM formula in Excel, you should remove the calculated members from the PageKeyRange, as displayed below. In this case the description of the account can still be retrieved, but EvDRE will generate only a SQL query.

3.5 Number of EvDRE functions in one sheet

It is possible to use more than one EvDRE in a worksheet, but the performance should always be tested very carefully. There are situations where multiple EvDRE functions can increase performance, because instead of building one EvDRE with two column dimensions and one row dimension, you can build two EvDREs with one row dimension and one column dimension. This makes the query easier and in most of the situations it is faster. The next two examples make this clearer.
In the report below there are two dimensions in the columns, Category and Time, with two members each.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
<th>K</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A</strong></td>
<td><strong>B</strong></td>
<td><strong>C</strong></td>
<td><strong>D</strong></td>
<td><strong>E</strong></td>
<td><strong>F</strong></td>
<td><strong>G</strong></td>
<td><strong>H</strong></td>
<td><strong>I</strong></td>
<td><strong>J</strong></td>
<td><strong>K</strong></td>
</tr>
<tr>
<td><strong>TIME</strong></td>
<td><strong>TIME</strong></td>
<td><strong>TIME</strong></td>
<td><strong>TIME</strong></td>
<td><strong>TIME</strong></td>
<td><strong>TIME</strong></td>
<td><strong>TIME</strong></td>
<td><strong>TIME</strong></td>
<td><strong>TIME</strong></td>
<td><strong>TIME</strong></td>
<td><strong>TIME</strong></td>
</tr>
<tr>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
</tr>
<tr>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
</tr>
</tbody>
</table>

The second report has two EvDREs where the Category dimension is no longer in the ColKeyRange, only the Time dimension is in the column. This generates two much more efficient queries in comparison with the query in the first example.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
<th>K</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A</strong></td>
<td><strong>B</strong></td>
<td><strong>C</strong></td>
<td><strong>D</strong></td>
<td><strong>E</strong></td>
<td><strong>F</strong></td>
<td><strong>G</strong></td>
<td><strong>H</strong></td>
<td><strong>I</strong></td>
<td><strong>J</strong></td>
<td><strong>K</strong></td>
</tr>
<tr>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
</tr>
<tr>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
<td><strong>ACTUAL</strong></td>
</tr>
</tbody>
</table>

The above sample is a simple case. In reality you should always test for the fastest way. If you divide one EvDRE into eight separate EvDREs, it is possible that you can improve performance (for example, if you go from one 1x3 dimension EvDRE to eight 1x1 EvDREs), but the more EvDREs you add, the more you have to test the performance.

Also be careful when creating workbooks containing multiple sheets with multiple EvDREs in them. When using "Refresh by Sheet" option for a workbook containing multiple sheets with multiple EvDREs, only data from the first column is retrieved. This is a known issue.
3.6 Avoid overlapping ranges when refreshing

Overlapping ranges causes the last executed EvDRE to overwrite the data of the EvDRE functions that were executed before and have the same overlapping data range. Overlapping ranges can be used for sending data (although not recommended, but sometimes necessary), but never for refreshing.

3.7 Conditional use of EvDRE

Sometimes when you have a workbook with multiple EvDRE functions, in certain cases you have to execute some of the EvDREs conditionally. For example, if you have a workbook with four sheets for four different divisions, but the only sheet to refresh should be the sheet of the division to which the current entity belongs. This can be done in two different ways.

3.7.1 Nesting EvDRE within an Excel formula

If you have a property that you retrieve in a cell, you can evaluate this cell with an IF formula, that has the EvDRE function included like:

```
=IF(B1="Y",EvDRE($B$3,A14:B20,A23:C29),"NO EvDRE")
```

In the first versions of EvDRE this approach did not work very well and generated errors, but for 4.2 SP2 later it seems to work fine.

3.7.2 Using a VBA to create EvDRE function

A way that was used to work around the first option when it was still an issue, was working with VBA. The same evaluation you make in an Excel formula can also be made in VBA code. By default, the EvDRE formulas do not appear in the worksheets, but the code pushes the formula with the right parameters to the right place in the sheet. The code looks like this:

```vba
If Range("Division") = "EXPRESS" Then
    Sheets("PAG_3A").Range("a1") =
    "=EvDRE($B$3,A16:B22)"

    Sheets("PAG_8A").Range("a1") =
    "=EvDRE($B$3,A16:B22)"

End If
```

3.8 Avoid the CellKeyRange option, if possible

When you have a complex report where you need to retrieve, for example, accounts that are not the same for the whole column or the whole row, it becomes difficult to create a report without using the cellkeyrange. This function allows you to define individual members for each cell in the datarange of EvDRE. But the query is less efficient in comparison to a “normal” EvDRE where you only use row and ColKeyRanges.

3.9 Data dump sheet & Excel-based calculations

In some cases, for performance reasons it may be an option to set up one big data dump sheet that is generated by one or more EvDRE functions. For example, if you have a workbook with ten sheets and all the sheets retrieve data from the same accounts, but for different entities per sheet. In this situation, instead of making ten EvDREs in the whole workbook, you can dump all your data in one hidden sheet and build your reports with links to the data dump sheet.

Also setting up easy KPIs like “Account1/Account2” via Excel formulas can increase the performance of your reports as it is faster than MDX queries.
However, when building a report this way, keep in mind to choose a unified approach throughout the whole report, because maintaining these data dump reports is a bit harder than maintaining a single report that retrieves its own data.

3.10 Text within EvDRE datarange

If you want to present text within an EvDRE datarange, use: ="[yourtext]" because otherwise the text disappears after a refresh. EvDRE overrules all cells that do not have formulas in them. So if you enter text in a cell, it is empty after refreshing and you have to enter everything again.

3.11 EvDRE and named ranges

The use of named ranges generally decreases performance. Excel has to do a lot of calculations to convert the named range to a real cell reference. The use of named ranges in combination with EvRNG or EvDRE functions may work with a few workarounds, but is not recommended.

Although the named ranges are bad for performance, they make it very easy to reuse ranges within a workbook that are the same for a lot of EvDREs, like an Entity that is chosen on a start sheet, which is used in every EvDRE in every sheet in the workbook. Its more user friendly to point to the named range "Entity," then pointing to a cell like "Home!$C$25," so in some cases you may want to use this approach.

The named ranges can be used in both the EvDRE and the EvRNG function. In the above picture, the formulas are made visible to show how named ranges can be used within EvDRE.
First of all you can fill the application parameter in the EvDRE function with a named range, like in the example “Menu.App.ID” contains “FINANCE.”

<table>
<thead>
<tr>
<th>Menu.pkr</th>
<th>MTB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>27</td>
<td></td>
</tr>
<tr>
<td>28 App</td>
<td>FINANCE</td>
</tr>
<tr>
<td>29 ACCOUNT</td>
<td>MTB</td>
</tr>
<tr>
<td>30 CATEGORY</td>
<td>MTB</td>
</tr>
<tr>
<td>31 DESCRIPTION</td>
<td>MTB</td>
</tr>
<tr>
<td>32 ENTITY</td>
<td>MTB</td>
</tr>
<tr>
<td>33 EQUIPMENT</td>
<td>MTB</td>
</tr>
<tr>
<td>34 GROUPS</td>
<td>MTB</td>
</tr>
<tr>
<td>35 HELCO</td>
<td>MTB</td>
</tr>
<tr>
<td>36 MEASURES</td>
<td>MTB</td>
</tr>
<tr>
<td>37 PROJECTS</td>
<td>MTB</td>
</tr>
<tr>
<td>38 SPECIFICATIONS</td>
<td>MTB</td>
</tr>
<tr>
<td>39 FLOW</td>
<td>MTB</td>
</tr>
<tr>
<td>40 TIME (Hour)</td>
<td>MTB</td>
</tr>
</tbody>
</table>

For the PagekeyRange, the named range “Menu.PKR” is used. This PagekeyRange contains a list of cells that normally is the PagekeyRange within the EvDRE. But since this PagekeyRange is the same for the whole workbook, there is a PagekeyRange on a central sheet that all the EvDREs use via the named range. The EvRNG function in this situation contains the named range directly, like EvRNG (named range).

Another way to work with named ranges within the EvRNG function makes use of the INDIRECT function of Excel. If you have named ranges that may change from time to time it is probably more efficient to enter them as text somewhere in the worksheet. In this case you do not have to look in to all the formulas, but you can just rename the text in that cell and all your formulas will automatically change. To use this approach within EvRNG, you just have to enter an INDIRECT function containing the named range within the EvRNG function like: EvRNG (INDIRECT(named range)).

The example above uses this approach for the ColKeyRange and the RowKeyRange and has a few boxes of text above it where the range names are maintained.

3.12 Turn off the “automatic calculation before refresh” in big workbooks

For workbooks with a lot of sheets that contain multiple EvDREs, it is a good practice to create a custom refresh button with VBA code behind it. The code looks like this.

```vba
Application.Run ("MNU_eAnalyze_Refresh")
Application.Calculation xlCalculationAutomatic
```

This code turns the automatic Excel calculation off before refreshing and enables it after the refresh.

You should use this approach if you have workbooks that contain references to named ranges in the EvDRE function. If not, you may get "unable to get the text property of the range class" error during refresh. Although the EvDRE looks OK before and after the refresh, it still does not refresh the right data.

3.13 Working with add/delete-line functions within EvDRE ranges

In some cases you need to have the option of an "add-line" or "delete-line" functionality in your schedule to insert lines that you need, instead of expanding all the available members. Most of the time this is done via custom VBA code which copies a hidden line into the data input part of the schedule. The add-line function copies the EvSND formula when you insert a new line.
EvDRE works a little bit differently, because the RowKeyRange is a reference to line numbers. If you insert a new line via VBA, the reference in the EvRNG formula needs to be updated. In some cases this happens automatically, but it may fail. To make it work, add the code “Application.CalculateFull” at the end of the insert line macro.

3.14 Using different data retrieval functions within one workbook (EvDRE & EvGET)

When (re)building reports, you can use both EvDRE and EvGTS functions within one workbook. For example, if you have a large report that retrieves finance data from the Finance application using EvDRE and rates data from the Rate application using the EvGTS function. In some cases performance may be better when you use a combination of both these functions. However, this does not mean that it is a good idea to mix up EvDRE and EvGET in all your reports. Since BPC needs to start up two different retrieve processes, there is some extra overhead time. We recommend you avoid the use of both these functions within one workbook for two reasons.

- To prevent possible performance issues. All reports that using mix-up functions should be tested in detail.
- To avoid maintenance. The more you have mix-up of functions, the more difficult it will be to maintain your report in the future, because first you have to investigate which data is retrieved in which way before you can really change your report.

3.15 Mixing different functions to create custom expands (EvEXP, EvNXP, and EvDRE)

When you are performing a very large expansion with two nested row dimensions (including suppress zero), it can be very useful in terms of performance to combine the original expand functions with EvDRE.

The EvNXP and EvEXP functions take care of the expansion part (including the suppress zero) as these functions seem to work faster in most cases when no data is retrieved.

The EvDRE function sometimes has some problems with the suppress function when using nested expansions, which results in an “error retrieving data from the web server” error. So by using the best of both worlds, a better performance can be achieved, where EvNXP and EvEXP take care of the expansion including the suppress (but without EvGTS formulas in the data grid) and EvDRE takes care of the data retrieval part. This workaround is also very useful when you want to play with the number of levels that you want to expand in your report. Currently, EvDRE cannot control the number of levels in an expansion, however, EvEXP has that capabilities. You can combine these two functions to take advantage of the filter options of EvEXP and the data retrieval performance of EvDRE.

Note: When using this method, you should use a custom button for the refresh that contains the macro “MNU_eTools_EXPANDANDREFRESH.” Otherwise you may not retrieve the correct data.

3.16 About the hidden EvDRE data cache sheet

When you change the workbook options of an EvDRE report and set the workbook type from “Report” to “Input Schedule” and expand it for the first time, BPC creates a hidden EvDRE_DATACACHE sheet. You can view this sheet using the VBA editor of Excel. To view this sheet you have to change the visible property in the VBA editor. This sheet contains cells with a lot of comma separated values. These values are used to evaluate which data is changed in the input schedule and should be sent to the database and which data did not
change. To prevent EvDRE to fail or to prevent a wrong evaluation, its better not to touch this EvDRE_DATACACHE sheet at any time. In certain cases you may want to reuse your input template for offline use and therefore you do not want to update the cache every time. For example, if you have data in January that does not change for February, you want to change the Time member in your sheet and send it. But if the datacache is filled, the system thinks the data is already in the database for February (which in fact is not true, but the datacache does not recognize the change of the Time member within the sheet) and does not send the data values for February. To prevent this behavior, uncheck the option “refresh after data send” in the workbook options. In this case the datacache will not be filled and users can use their offline template throughout the year.

4  Best practices for using EvGET/EvGTS

4.1 Using EvGTS instead of EvGET

These two functions basically do the same thing, they retrieve data from the database. EvGTS has the ability to use a scaling value. If you design a report with EvGTS and decide to use scaling, then you just change the scaling value in the worksheet. When you build the report with EvGET you add a divide function to all the formulas in your worksheet. EvGTS has no visible performance decrease, so it is always better to use EvGTS function. When you do not need scaling, you just use one (1) as the scaling factor.

4.2 Specifying all dimension members in the EvGTS function

If you have an application with eleven dimensions, make sure you specify a member for all of those eleven dimensions in your formula. If you do not do this, BPC looks at the current view and if this current view is not correct, you retrieve wrong data. You can easily check this by counting the number of dimensions in your current view and comparing that with the number of members specified in your EvGTS. Also check to make sure you are not using the same member twice. You may retrieve the wrong data if you use the same member more than once.

4.3 Fixing the dimensions that you do not change

If you use the traditional report wizard template, use the “Override Member” column in the control panel. If you have an EvDRE report and you want to rebuild it as an EvGTS report, just override the members in the PagekeyRange and point your EvGTS formulas to this PagekeyRange.

4.4 Developing “dynamic reports”

Building dynamic reports takes only a few extra minutes to think about how you want to use and reuse your reports and how you want to create it. But in the end it always saves a lot of time. For example, you can use the same dynamic report as a basis for building reports for different periods.

4.5 Reviewing the eTools Help

The help file contains a full list of functions and provides descriptions and examples.