POWER BI STANDARDS

Created for:	EBIS Ltd
Version:	1.00
Last modification:	2022-12-30 14:57
Authors:	Dominik Dębowski, Łukasz Balcerzak
Company:	EBIS Ltd
Document status:	Confidential / ORIGINAL

1 History changes

Version	Date	Author	Description
1.0	2022-12-30	Dominik Dębowski, Łukasz Balcerzak	Initial version

Content

2 Data Model

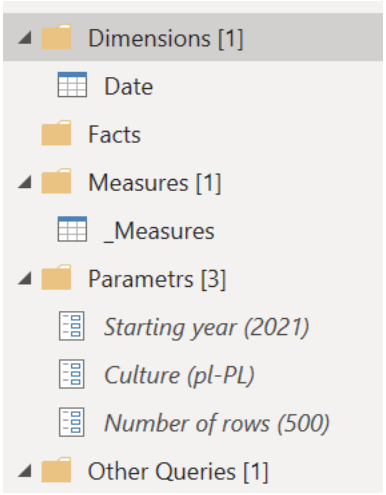
2.1 Data load

Model should not contain unnecessary data. Please avoid loading as much as possible data and keep rules below. Tables names should be in language consistent with customer expectations.

2.1.1 General rules

Queries should be grouped into folders in a meaningful way. Please use structure presented below.

- Every dimension table should be stored in **Dimensions** folder
- Every fact table should be placed in **Facts** folder
- Measures table should be placed in **Measures** folder
- Each parameter should be placed in **Parameters** folder
- Additional tables should be placed in **Other Queries** folder



2.1.2 Parameters

Use parameters to make the model easy to maintain. Examples of when to use parameters:

- Data sources – i.e. connection strings, paths to files in SharePoint or local drive, Database names..
- Filters – when you want to limit the data loaded to the model (e.g. only one brand, only X segment, only data newer than 2021-01-01)
- Other parameters - Culture

Always add a description to the parameter explaining it' s impact on the data model.

How to create a parameter in power query? - https://learn.microsoft.com/en-us/power-query/power-query-query-parameters

An example of parameter used in M code:

```
let
  FromYear = #"Starting year",
  ToYear = Date.Year(DateTime.LocalNow()) + 1,
  Culture = Culture,
```

2.1.3 Limit the amount of data loaded

In some cases you can limit the data which is loaded to Power BI Desktop data model to avoid problems with memory consumption on local machine and long processing time – e.g. in development environment. For this scenario additional parameter and additional step in query power query is required. It's recommended to set this logic in every large table.

```
  #"Changed Type" = Table.TransformColumnTypes(Source, {{"ID", Int64.Type}}),
  #"Kept First Rows" =
    if #"Number of rows" <= 0 then
      #"Changed Type"
    else
      Table.FirstN(#"Changed Type", #"Number of rows")
in
  #"Kept First Rows"
```

How to set this up: https://www.youtube.com/watch?v=_zYvybVMk7k

This logic requires changing the value of parameter in the final dataset used for reports consumed by end-users.

- **In Power BI Pro** workspace the value of parameter is published together with the whole model, so you need to change the value of parameter in Power BI desktop, reload the model in PBI Desktop and then publish it.
- **In Power BI Premium** workspace parts of the datasets can me published selectively using ALM Toolkit. You can publish the model without the value of parameter (so you keep limited number of rows in PBI Desktop and have full data in model hosted in PBI service).

It makes more sense to use this technique when you work with model which will be hosted in Power BI Premium workspace.

2.2 Naming convention[1]

Names for all objects – especially for tables, columns and measure which will be used by end-users should be human-readable. If your data model is sourcing data from data warehouse probably there are a lot o prefixes like Dim and Fact and a lot of names with camel or Pascal cases e.g. CustomerName. The best practice is to avoid this and provide human-readable names for Power BI data model objects (e.g. by creating views on database using friendly names).

2.2.1 Dimension tables

Names of dimension tables should be in singular form e.g. Date, Customer, Product. Avoid using prefixes like Dim, DIM, vDim etc.

2.2.2 Fact tables

Names of fact tables should be in plural form e.g. Orders, Sales etc. Avoid using prefixes like Fact, FACT, vFact etc.

2.2.3 Measures

All measures should be placed in _Measures table and should have names in correct business terminology and be as descriptive as possible. If measures calculates stock value it should be called Stock value.

2.2.3.1 Variables[2]

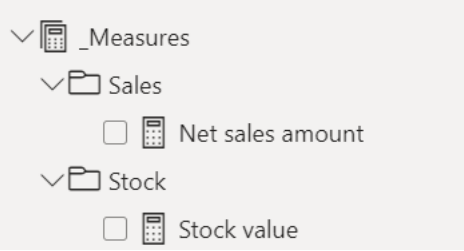
Two underscores __

2.2.3.2 Temporary columns[3]

Prefix with @

2.2.3.3 Folders

Measures should be grouped into folder hierarchy in a meaningful way.



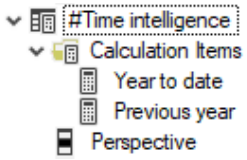
1 https://blog.crossjoin.co.uk/2020/06/28/naming-tables-columns-and-measures-in-power-bi/
2 https://www.sqlbi.com/blog/marco/2019/01/15/naming-variables-in-dax/
3 https://www.sqlbi.com/articles/naming-temporary-columns-in-dax/

2.2.4 RLS

Create row level security rules in data model if it's feasible. If role is created in a dynamic way then it should be called "Dynamic role". If it is a static one then it should has a name related to the filtered area or dimension e.g. "Team A", "Team B", "Country USA", "Country Poland" etc.

2.2.5 Calculation groups

Every calculation group should has descriptive name and items. Names of calculation groups tables should start from "#" like #Time intelligence.



2.3 Formatting

"If it is not formatted… it is not DAX!". Proper formatting style is important and use special tools for this if you are not familiar with general rules related to formatting.

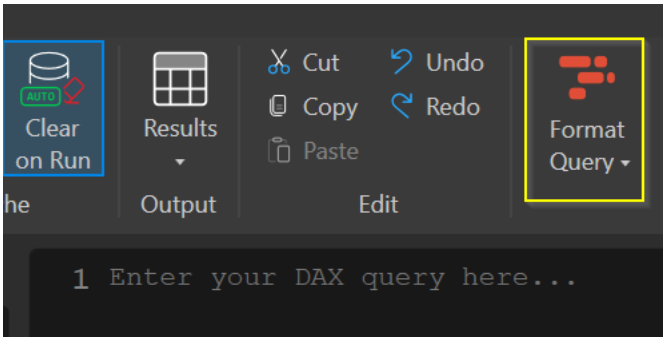
Not only DAX can be formatted. Queries in Power Query also should be formatted.

2.3.1 DAX Formatter

There are many ways to format DAX properly. You can use:

- https://www.daxformatter.com/
- Tabular Editor

- DAX Studio



2.3.2 Power Query Formatter

To format Power Query queries please use:

https://www.powerqueryformatter.com/formatter

2.3.3 Keys and values

In an ideal data model all keys (without any business value) and values should be hidden. Only attributes and measure should be visible for end-users.

2.4 Calculation groups

For repetitive calculation logic - especially for time intelligence - calculation groups should be created.

2.5 Advanced scripts

Advanced scripts are helpful for automating work. There are a lot of uses of these scripts. For example we can create script that will create specific measures based on some calculation group for each base measure e.g. Year to date for every single base measure.

Helpful scripts are available in main folder.

2.5.1 Custom Actions

Custom actions are saved advanced scripts and are available from the context menu (right click over an object in Model view) in Tabular Editor. A few custom actions are available in the main folder. To import them copy the MacroActions json file and paste it in this location C:\Users\UserName\AppData\Local\TabularEditor.
If you already have other Custom Actions saved in this location, combine those two files.
The names of custom actions should be self-explanatory.

How to use custom actions - https://docs.tabulareditor.com/te2/Custom-Actions.html

2.6 Date table

2.6.1 Create Date Table

Standard Date Table must have a Date column that meets requirements:[4]

- Must contain unique values
- Must not contain BLANK
- Must not have any missing dates (is continuous)
- Must span full years (calendar year or fiscal year, or any other definition)

4 https://learn.microsoft.com/en-us/power-bi/guidance/model-date-tables

2.6.2 Adjust date table to your scenario

Standard time analysis is performed at year, quarter and month levels in a calendar or fiscal years periods – standard date table is perfectly fine for these tasks.

For more complex scenarios (week-related analysis, adding 13th month) you may be forced to customize Date table. Do it on Power Query or database level.

Here are some guidance on building date tables: https://www.daxpatterns.com/time-patterns/.

2.6.3 Mark as Date Table[5]

Date table should be marked as Date table – it handles Time Intelligence DAX function regardless of the data type of Date column. (i.e. PK in Date table can be integers). It is also a best practice.

2.6.4 Turn off Auto/date time

Just do it.[6]

2.7 Template date table and time intelligence

This document is accompanied by template pbix file that includes date dimension and time intelligence calculation groups already created.

2.7.1 Template date table

Standard date table is provided in the template .pbix file. It is designed to allow time intelligence analysis as described in Standard time-related calculations daxpattern[7].

This table consists of many columns to cover most of typical use cases. You should only load columns that you actually need in your data model. A description of template date table is provided in a separate xlsx file placed on root folder containing Power BI Standards.

5 https://www.sqlbi.com/articles/mark-as-date-table/
6 https://www.sqlbi.com/articles/automatic-time-intelligence-in-power-bi/
7 https://www.daxpatterns.com/standard-time-related-calculations/

In order to use template table follow these steps:

1. Add your Fact table in the power query editor
2. Set parameters in power query editor
    a. Date – Culture -> determines the language of day and month names
    b. Date – End date type
        i. Current Date – calendar ends on the 31st of December of current year
        ii. Max Fact Date – calendar ends on the 31st of December of last year present in fact table(s)
        iii. Custom Date – you can set the end year of calendar in Date query
    c. Date – Start date type
        i. Current Date – 5 years – calendar starts on the 1st of January of current year minus 5 years
        ii. Min Fact Date – calendar end on the 1st of January of first year present in fact table(s)
        iii. Custom Date – you can set the start year of calendar in Date query
3. Adjust queries
    a. 'Date – Max Fact Date' → substitute sample fact table with the reference to the date column in your fact table (e.g. 'Sales'[Order date])
        i. If there are multiple fact tables (or multiple columns in one fact table that should be included in this logic) add date column from all fact tables in this query like this

```
let
    Source = List.Max({
                List.Max( SampleFactTable[Order Date] ), // first fact table
                List.Max( SecondFact[Order Date] )  // second fact tbale
                })
in
    Source
```

    b. 'Date – Min Fact Date' → substitute sample fact table with the reference to the date column in your fact table
    c. 'Date' – inside the query set the following parameters
        i. CustomStartYear
        ii. CustomEndYear
        iii. FirstFiscalMonth
        iv. In the last step of the query -> Select columns you need to load in the data model
4. Disable load or delete query – SampleFactTable

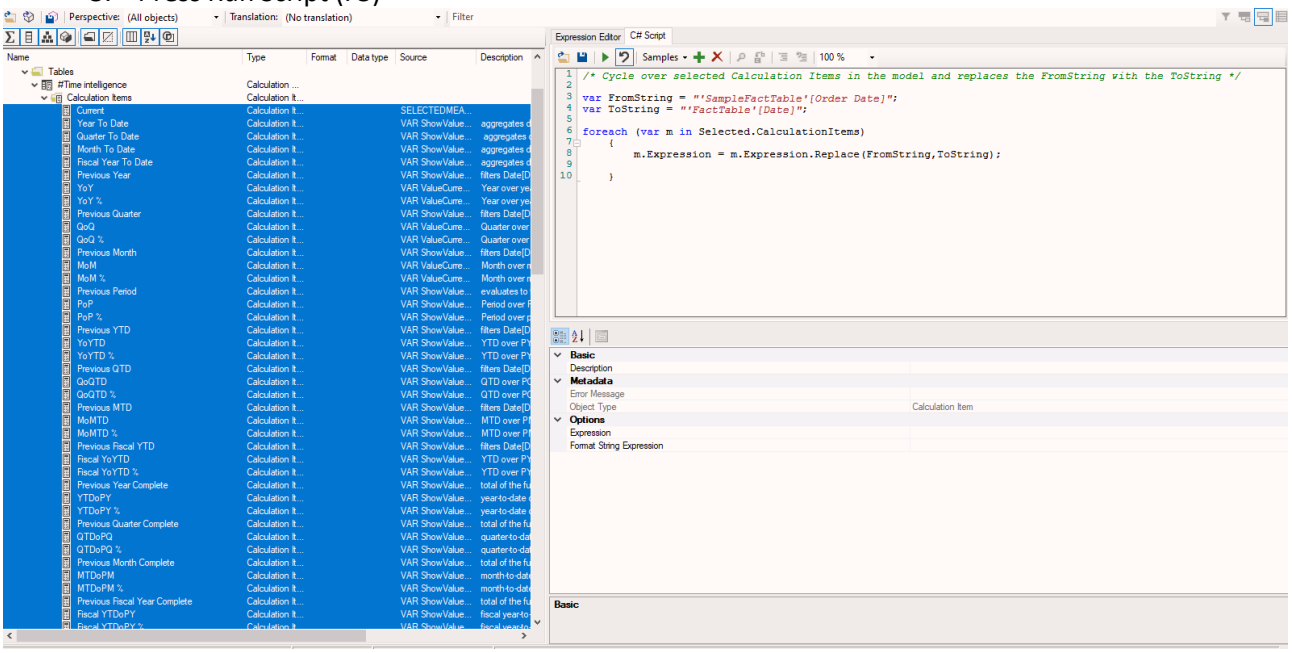
2.7.2 Time Intelligence calculation groups

The template pbix file includes time intelligence calculation groups based on the Standard time-related calculations daxpattern – a full description can be found here https://www.daxpatterns.com/standard-time-related-calculations/.

2.7.2.1 Setup

Before using this calculation group, you need to modify definitions of calculation items in Tabular Editor.

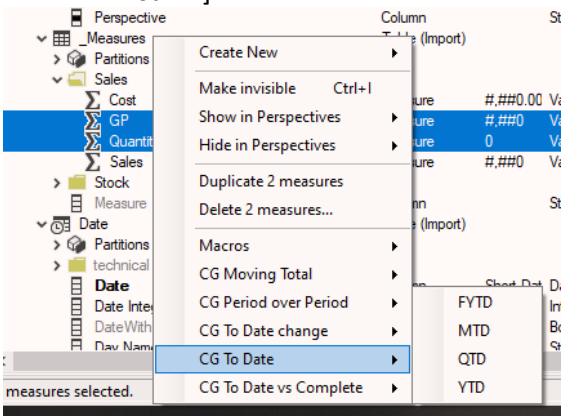
1. In Tabular Editor - go to C# script window
2. From Sample dropdown select Macros and find ‘Replace DAX in Calculation Item – CHANGE PARAMETERS’
3. Change parameters
    a. FromString – should be "'SampleFactTable'[Order Date]"
    b. ToString – should be a reference to date column in your fact table
4. Select all calculation items in #Time Intelligence calculation group
5. Press Run Script (F5)



2.7.2.2 Using time intelligence calculation group

Calculation group can be used ‘as is’ in a report to change the logic of measures visible in a report or to can be used to create specific DAX measures. To create a time intelligence measure (based on calculation group) follow these steps:

1. In Tabular Editor select base measures for which you want to create time intelligence measures
2. Right click on one of the selected measures
3. Select a group of calculation and select specific calculation
4. Measures should be added in the same folder with a name [base measure name + suffix]



A list of available calculation group, and their Custom action folder is provided in a table.

L.p	Calculation Group	Abbreviation	Description	Custom Action
1	Current			
2	Year To Date	YTD	Year-to-date	CG To Date
3	Quarter To Date	QTD	Quarter-to-date	CG To Date
4	Month To Date	MTD	Month-to-date	CG To Date
5	Fiscal Year To Date	FYTD	Fiscal Year-to-Date	CG To Date
6	Previous Year	PY	Previous year	CG PoP change
7	YoY	YoY	Year-over-year	CG PoP change
8	YoY %	YoY %	Year-over-year %	CG PoP change
9	Previous Quarter	PQ	Previous quarter	CG PoP change
10	QoQ	QoQ	Quarter-over-quarter	CG PoP change
11	QoQ %	QoQ %	Quarter-over-quarter %	CG PoP change
12	Previous Month	PM	Previous month	CG PoP change
13	MoM	MoM	Month-over-month	CG PoP change
14	MoM %	MoM %	Month-over-month %	CG PoP change
15	Previous Period	PP	Previous period (automatically selects year, quarter	CG PoP change
16	PoP	PoP	Period-over-period (automatically selects year, qua	CG PoP change
17	PoP %	PoP %	Period-over-period (automatically selects year, qua	CG PoP change
18	Previous YTD	PYTD	Previous year-to-date	CG To Date change
19	YoYTD	YoYTD	Year-over-year-to-date	CG To Date change
20	YoYTD %	YoYTD %	Year-over-year-to-date %	CG To Date change
21	Previous QTD	PQTD	Previous quarter-to-date	CG To Date change
22	QoQTD	QoQTD	Quarter-over-quarter-to-date	CG To Date change
23	QoQTD %	QoQTD %	Quarter-over-quarter-to-date %	CG To Date change
24	Previous MTD	PMTD	Previous month-to-date	CG To Date change
25	MoMTD	MoMTD	Month-over-month-to-date	CG To Date change
26	MoMTD %	MoMTD %	Month-over-month-to-date %	CG To Date change
27	Previous Fiscal YTD	PFYTD	Previous Fiscal year-to-date	CG To Date change
28	Fiscal YoYTD	Fiscal YoYTD	Fiscal Year-over-year-to-date	CG To Date change
29	Fiscal YoYTD %	Fiscal YoYTD %	Fiscal Year-over-year-to-date %	CG To Date change
30	Previous Year Complete	PYC	Previous year complete	CG To Date vs Complete
31	YTDoPY	YTDoPY	Year- to-date-over-previous-year	CG To Date vs Complete
32	YTDoPY %	YTDoPY %	Year- to-date-over-previous-year %	CG To Date vs Complete
33	Previous Quarter Complete	PQC	Previous quarter complete	CG To Date vs Complete
34	QTDoPQ	QTDoPQ	Quarter-to-date-over-previous-quarter	CG To Date vs Complete
35	QTDoPQ %	QTDoPQ %	Quarter-to-date-over-previous-quarter %	CG To Date vs Complete
36	Previous Month Complete	PMC	Previous month complete	CG To Date vs Complete
37	MTDoPM	MTDoPM	Month-to-date-over-previous-month	CG To Date vs Complete
38	MTDoPM %	MTDoPM %	Month-to-date-over-previous-month %	CG To Date vs Complete
39	Previous Fiscal Year Complete	PFYC	Previous Fiscal Year Complete	CG To Date vs Complete
40	Fiscal YTDoPY	FYTDoPYC	Fiscal Year- to-date-over-previous-year	CG To Date vs Complete
41	Fiscal YTDoPY %	FYTDoPYC %	Fiscal Year- to-date-over-previous-year %	CG To Date vs Complete
42	Moving Annual Total	MAT	Moving annual total	CG Moving Total
43	Previous Year MAT	PYMAT	Previous year moving annual total	CG Moving Total
44	MAT change	MATG	Moving annual total growth	CG Moving Total
45	MAT change %	MATG %	Moving annual total growth %	CG Moving Total
46	AVG 30D non-additive	AVG 30D	Moving average 30 days	CG Moving Total
47	AVG 30D additive	AVG 30D	Moving average 30 days	CG Moving Total
48	AVG 3M non-additive	AVG 3M	Moving average 3 months	CG Moving Total
49	AVG 3M additive	AVG 3M	Moving average 3 months	CG Moving Total
50	AVG 1Y additive	AVG 1Y	Moving average 1 year	CG Moving Total
51	AVG 1Y non-additive	AVG 1Y	Moving average 1 year	CG Moving Total

2.8 Data modeling

Data modeling is very important phase of Power BI Implementation. To keep it on high level it is necessary to keep some rules.

2.8.1 Data reduction[8]

Model should not contain unnecessary data. Please avoid loading everything "As Is" from the data source and keep rules below. An ideal situation is when:

- Fact tables consist of only foreign key columns (that relate to existing dimension in the model) and metric columns (columns with values to aggregate)
- Dimension tables consists of primary key and attributes columns that are used in the model
- Dimension tables are connected only to Fact tables (a star schema or snowflake schema)

2.8.1.1 Remove unnecessary columns and unnecessary rows

Keep only columns that server some purpose (i.e. contain values to aggregate, are keys in relationships, are used to filter or slice the data).

Keep only rows that will be presented in reports- for example, there's no point in loading data from 2018 and before, if reporting requirement is to present result from 2019.

2.8.1.2 Cardinality reduction

Group tables to the level of granularity used in a report. For example if reporting is done only at month level and never at day or week level, then you can aggregate fact table to the month level.

Remove key/attribute columns from fact table, that don't correspond to any dimension and are not used in the report.

2.8.1.3 Optimize column data types[9]

Consider using floating data type (Decimal point in Power BI), especially for financial data. Opt for Fixed decimal point data type in Power Bi, especially for financial (transactional) data.

2.8.1.4 Calculated column

If you need to create a calculated column, prefer creating the column as far upstream as possible, and as far downstream as necessary.[10] Avoid adding column as calculated column in DAX, choose creating a column in Power Query instead.

2.8.2 Row Level Security

How to set up static RLS: *https://learn.microsoft.com/en-us/power-bi/enterprise/service-admin-rls#using-the-username-or-userprincipalname-dax-function*

How to set up dynamic RLS: *https://radacad.com/dynamic-row-level-security-with-power-bi-made-simple*

8 https://learn.microsoft.com/en-us/power-bi/guidance/import-modeling-data-reduction

[9] *https://www.sqlbi.com/articles/choosing-numeric-data-types-in-dax/*

[10] *https://ssbipolar.com/2021/05/31/roches-maxim/*

Remember that RLS follows active relationships in a data model. For example: when you filter Dim1 with RLS, users will see limited Dim1 table and Fact table, but will be able to see other dimensions in full.

2.8.3 Other recommendations

Power BI data model should be as optimize as possible. All measures should be formatted in a proper way – for example one decimal places, thousand separator etc..

There are some things that should be avoid:

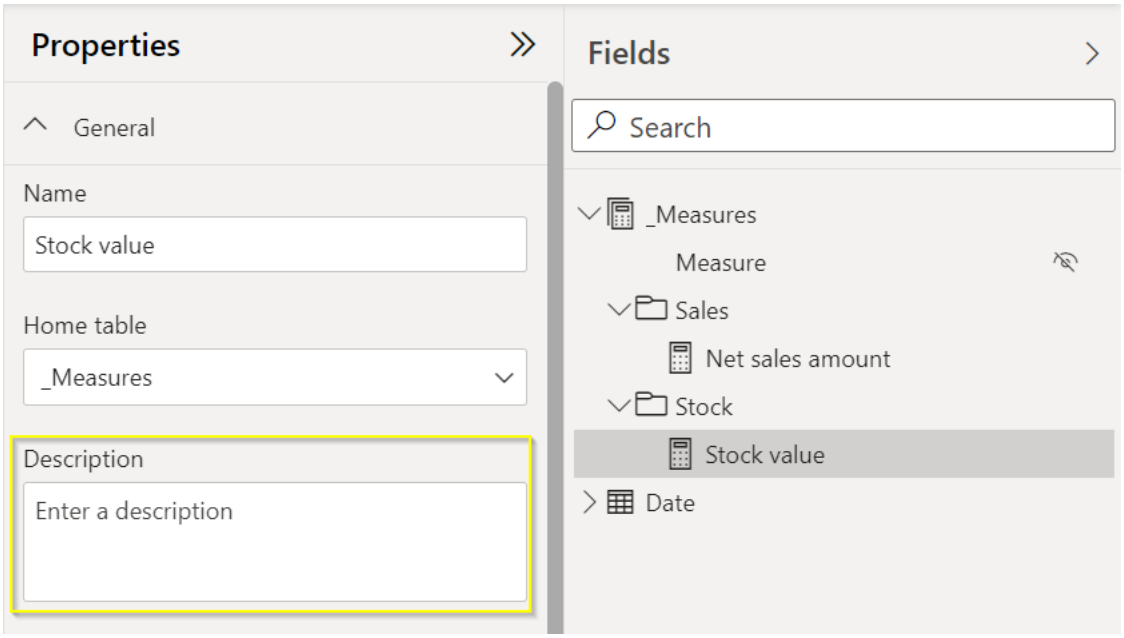
- Bi-directional relationships

2.8.4 Layouts

Every data model should have layouts for specific area of data model in model view.

2.8.5 Descriptions

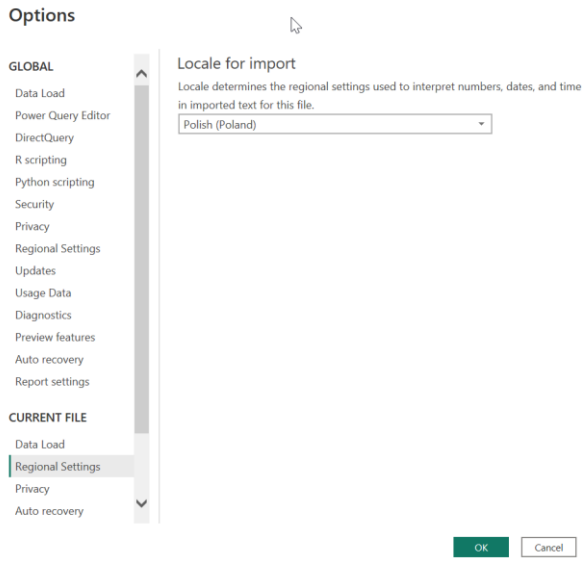
For complicated measures it is worth to create descriptions to keep business knowledge inside the model.



2.9 Power BI PBIX Template

Please use attached PBIX template for data models that will be created during the implementation. This template contains base calculation groups, structure for data loading, specific setting and it should save your time at the beginning of the project.

Below setting should be consider and can be changed.



3 Reports

3.1 General rules

Every report should be a separate file with specific visualization. Name of pages should be understood by end users. Do not keep “Page 1” on production report.

Report file should be separated from data model file. Script that you can use for this is available in main folder.

3.2 Background and theme

Every report should have specific background. Use Power Point to create file with background and save it in SVG format. All static elements like shapes, logos etc. should be included in this file. Only elements that should be interactive can be on the Power BI report level.

If it is possible please create JSON theme and use it on every report to keep consistency between all reports inside one project if there is no specific requirements or differences.

4 Version control

Version control for Power BI Reports or data models it is very important. There are two possible solution to keep the history of changes.

4.1 SharePoint

Every file on SharePoint has version history. Keep files on SharePoint folder to have possibility to restore necessary file.

4.2 DevOps

You can create Git repository and then create repository via pbi-tools.

5 Contact

Dominik Dębowski
Phone: +48 531 772 907
E-mail: dominik.debowski@ebisgroup.com

Łukasz Balcerzak
Phone: +48 531 177 221
E-mail: lukasz.balcerzak@ebisgroup.com

Confidentiality of information

All information contained in this document should be considered as confidential. This applies especially to information for economic, technical and commercial. Customer undertakes, as well as persons providing work in its favor in any form, to preserve the confidential nature of the information contained in this offer.

Copyright