Power BI Development Recommended Practices

Based on the Post: Power BI Development Best Practices - RADACAD

When developing a reporting solution for an organization in the Power BI Platform, it is important to consider practices that consider the optimization of the different resources. Reza Rad from Radacad published recently a post with best practices for developing Power BI solutions, which I found very aligned with our way working in DSI, so I would like to share them with you.

Use Dataflow to Re-use tables

As a Power BI developer, you know everything starts with the process of adding a query to Power BI by doing some transformations to a source table in Power Query. If you are working within an organization, the probability that that query will be used in another dataset is high. So, to avoid doing the same transformations over and over in different datasets, it is recommended that the table is created within Dataflow instead of directly in Power BI desktop. View Original text ->

Use Power Query functions to re-use data transformations

Again, we try to avoid re-work, now when we want to repeat the same transformation to different data sources that have the same structure. Instead of writing the same code for different sources, we write it one time as a function and the call it for the different sources. View Original text ->

Avoid the need of making changes to the Power Query code with Parameters

If you believe some variables used in the transformation process within Power Query, it is recommended the use of parameters, instead of hard coding those variables in the code. This way if that variable changes in the

future, you only need to change the parameter within Power Bi Service, without the need of opening the dataset. View Original text ->

One Dataset to Re-use DAX Calculations

Like the previous point, here we try to avoid the need of adding the same DAX calculations over and over, and the solution for that is having one dataset that contains all the DAX measures needed and create reports that connect to that dataset. The you can share this dataset with different analysts withing the company, allowing them to create reports much quicker, since all the measures are already available. View Original text ->

Use Groups to Organize your Queries within Power Query

If you are using one dataset that contains a tens or hundreds of queries you know that it would be a very tedious task to find the corresponding query that you need to edit. If you have these queries organized in a logic way (it may be by source, by area of the organization, etc.) by using groups and sub-groups, this task will be much less painful, and consume less time. View Original text ->

Load only tables you need

Temp tables used only during the transformation process in Power Query, but that are not needed in Power BI for the analysis, can be disabled to load. This way the model will weigh less without affecting the analysis process. View Original text ->

Filter data you don't need in Power Query

If in the data model you are building there's data you don't need, like certain old dates, or categories, etc., try filtering that out in Power Query, and using parameters to manage that without the need of editing the dataset to change the filtering rules. View Original text ->

Try using "Reference" and "Duplicate" in the correct place

In Power Query it is important to understand the difference between adding a "Reference" and a "Duplicate" of a query. A "Duplicate" will create an exact copy of the query with all the same transformation steps the original query had, and the "Reference" will be a link to the last step of the original query. So, depending on the need you have, it may be better to choose one or another, always trying to minimize the duplication of steps, since this will optimize the transformation process. View Original text ->

Multi-Layer Architecture Everywhere

We have discussed in several tips here the benefits of re-using elements, with Dataflow, centralized datasets, functions within Power Query, queries referencing, etc. These are all examples of multi-layer architecture, and a good practice is to think in general in methods that can make re-usability better, in all areas of your architecture. View Original text ->

Create a Date Table

A regular recommendation for most of Power BI Models is to create a date table, that will be able to slice multiple fact tables at the same time, and in which you can add different date fields, like different weeks, etc. View Original text ->

Use Aggregations for big tables

Aggregation tables is a nice feature when your report is slow when reading data for very big tables. An automated aggregation table is created that will be smaller, hence read faster. When a visualization can read the data from the aggregated table, then that one will be used, otherwise the big table will be used. View Original text ->

Use a Star Schema for your Model

One of the most important things to consider in a data model is the relation between the different tables, and the recommendation is the use of a star schema, which considers a central fact table that is related to different dimension tables with a many to one relationship. View Original text ->

Use Incremental Refresh and Hybrid tables

Refreshing a complete dataset can take a long time and create potential connection errors. These can be reduced drastically if you implement Incremental Refresh in big fact tables, that could be either imported or hybrid. The idea is that the refresh is done only for the data that can potentially change, and what is historic and never change, remove from the refresh process. Refresh time can be reduced from hours to a few minutes. View Original text ->

Avoid both-directional relationship

Both direction relationship causes a decrease in performance and errors caused by ambiguity in the relations between tables. In general, start schema models don't need both direction relationships, but if it is needed in some report section, both direction relationship could be avoided by using functions like crossfilter. View Original text ->

Use Hierarchies

There are cases of dimensions that have a hierarchical relationship between them; for those cases you can create the hierarchy within your model, making it easier to add these fields into visualizations in one simple click. View Original text ->

Use Themes and Background images

Within an organization, it is normally very important to follow design guidelines, like colors and fonts. To save time and have the correct colors and fonts within all reports, it is recommended to use Theme files and background images, so all reports created follow the same look and feel. View Original text ->

Sync Slicers

Syncing slicers is generally the recommended way of handling the slicing within a report, so unless the requirement is not to do so, always sync slicers. You can even, if the space is not a constraint withing the report page, you can create a Slicer page or element, and add a button/bookmark that will open and close the slicer page or element. View Original text ->

Use DAX for Conditional formatting

Adding color codes to certain values in the report is useful to move the attention of the analyst to the most important ones. Doing conditional formatting hard coded is an option, but the recommended way is using DAX measures that will contain the formatting, since it will create consistency between different visuals and maintenance will be easier. Also, color codes can be stored in a table for maintenance and configuration can be done easily. View Original text ->

Organize Measures in folders within dedicated tables

Finding measures in a model could be difficult, mostly if the measures are created within the facts and dimension tables. The recommended way of organizing them is inside dedicated empty tables, that can be created in Power Query or Power BI Desktop. Also, you can organize the different measures in folders, which is quite helpful for big models with hundreds of measures. View Original text ->

Set the correct aggregation mode for the numeric fields

When you have a numeric field in a table Power BI will automatically assign a sum aggregation to it, but it won't necessarily be the correct aggregation you need, so it is important that you review your numeric fields and specify the correct aggregation, or even remove the aggregation if it not required. View Original text ->

Create a mobile specific design

Since people are spending a lot of time in mobile devices, it makes sense to use the mobile design feature in Power BI to add a version that can be optimized graphically for these; it will create a friendlier experience for users. View Original text ->

Regular Model Optimization

Reviewing your model regularly for optimization purposes is a good idea. There are tools that can help with the task of finding improvement opportunities. View Original text ->

Original Post

Power BI development best practices
Regardless of the job function, you have with Power BI (Developer, consultant, architect), following certain practices ensures a good quality solution. In this article and video, I explain some of those best practices, why they are helpful, and links to how to use them. These tips are related to the development of a Power BI solution, not the deployment, publishing or sharing of the solution.

Video

https://youtu.be/jzjRWvigF9c

Re-use tables generated in Power Query

If you are using Power Query to connect to the data source and transform the data, then it is likely that you will create a table that you may need in another file in the future. If you use the Power Query inside Power BI Desktop for transformation, then re-using the table in other PBIX files will be a challenge. Your only choice would be to copy and paste the codes into the new file. This would bring another issue; the redundancy of the Power Query code.

The proper way of connecting to the data source and transforming it in a way that can be reused in other Power BI files is by using Dataflows. Dataflow is a Power Query process that runs on the cloud, and the data will be stored in a destination such as Azure Data Lake Storage or Dataverse. If a table is generated in the Power Query of Dataflow, it can be used easily in multiple Power BI files.

If you have a table that you may need to reuse in multiple files, then consider generating that table using a Dataflow.
The wrong way: copy and paste Power Query tables between PBIX files

The right way: create Power Query tables in Dataflow, and get data from them in PBIX files

Re-use DAX calculations

DAX is the language of writing calculations in Power BI. We use DAX to write calculations such as year-over-year change and percentage, or percentage of the total or rank of customers by their yearly revenue. Writing calculations in DAX takes time, and you may likely need to re-use a calculation in multiple reports.

Creating copies of the PBIX file every time for reusing the calculation is not ideal. The better approach is to have a shared dataset created by DAX calculations and then create thin reports with live connections to the shared Power BI dataset. Using a shared dataset ensures that all the reports are using the same DAX calculations. If a change is needed, it is only needed in the shared dataset. Maintaining a solution like this would be much easier.
The wrong way: copy and paste DAX calculations between PBIX files

The right way: create a shared Power BI dataset, and connect live to it as Power BI thin reports

Power Query Functions for re-usable data transformation

There are likely a specific set of data transformation steps for multiple data sources. For example, the Sales data for all the branches of a company has the same structure. Each Sales data is in a different database (or in a different Excel or CSV file). You can develop a set of data transformations and re-use them inside the Power Query using Custom Functions.

Power Query custom functions are re-usable pieces of Power Query transformation steps. A function can have input parameters (or it may not have any parameters if it is a generator function) and generate an output. The output of the custom function may then be used in other queries.
Custom functions will reduce the need for re-writing a piece of code or transformation. As a result, the maintenance of that code will be easier. If there is a change needed, there is only one place that needs the change; inside the Power Query custom function. You can re-use the custom function in multiple places inside the Power BI file.

If you use a custom connector, then you can include the custom function in it so that you can use that custom function even in multiple PBIX files.
The wrong way: copy and paste the Power Query transformation between tables

The right way: create a Power Query custom function and re-use it in multiple places

Parametrize the Data Transformation Process using Power Query Parameters

Assume that the data source will likely change from one server to another. Or the folder that the Power BI gets the Excel files from it is likely to move to a shared folder on a server. Or the email address that the Power BI gets data from the Exchange might change. In any of these cases, changing it from the previous value to the new value would be a hassle of opening the Power BI file in the Power BI Desktop, changing the data source settings, or even going to Power Query Editor to make some changes.

If you use Power Query parameters in the places where you think the value might change, then you can change the parameter value even outside of the Power Query. If your file is already published to the service, then under the dataset settings, there is a place where you can change the values of parameters before the next refresh of the Power BI dataset.
Using Power Query parameters will enable you to parameterize the values in the data transformation. These can be the data source names, path, server names, table or column names, and many other values that are likely to change.

The wrong way: using values (such as server name, folder path etc., which are likely to change) directly in Power Query.

The right way: Create Power Query parameters for values likely to change and use Parameters in the Power Query.

Categorize Power Query using Power Query Groups (or Folders)

Categorize Power Query objects (tables, parameters, functions, lists, records etc.) in folders. In Power Query, these folders are called Groups. You can create as many groups as you want. The method by which you create the group is dependent on you. Some may prefer creating groups per data source (such as all the tables from SQL Server data source under one folder), and some may prefer creating groups per table functions (such as all product tables from all data sources under one folder). Some generic groups, such as final tables (the group that includes all enable-load queries) or temp tables (the group that includes all disable-load queries), are common.

Make sure to determine your standard for defining groups and use them to categorize your Power Query object structure.

Disable the load of temp tables in the Power Query

Power BI loads everything into the memory. If you have a table in Power Query that you are not using directly, but using to load data into another table, then consider disabling its load. For example, Suppose you are building the product table from the merge of Product category, subcategory and product details. In that case, those three tables can be disabled the load, especially if the data that you need is going to be all available in the product table at the end.
Disabling the load of a table won't load it into the memory of the Power BI dataset. The table will still be part of the refresh process in the Power Query. Disabling the load of temp tables is a significant performance improvement in the Power BI model. Adding unnecessary tables into Power BI is not only affecting the performance, but it also makes the model complicated and confusing (you may select the Product ID from a table that is not the primary product table).

Just load what you need for reporting, Filter the data

If you are loading 20 years of sales transactions just in the hope that someday, someone will use them in visualization, rethink your approach. It might be better to filter the data for the last few years that you need (such as the last three years) and then create a Power Query parameter to change it if required.

Reducing the amount of data loaded into Power BI is a significant performance improvement. Just load the data that you need.
The same rule also applies to tables with hundreds of columns. If you need five columns from a table, don't import the entire 200 columns. This is a common mistake in getting data from CRM or ERP systems where the tables are wide. You can use Power Query to load more columns or tables in the future if needed.

Use the "Reference" and "Duplicate" in their rightful places

If you need a table copy, There are two ways of doing it in Power Query; "Duplicate", and "Reference" give you a copy of the table. "Duplicate" will create an identical copy of the table without any link to the original table; You can change this new table without any concerns about the main query. "Reference" will keep the transformation in the primary table, and the new table will be a link to the original table. This is used when you want to apply a set of transformations on a currently existing query and still follow the transformations of the primary query. Choosing between "Reference" and "Duplicate" is a crucial choice.

Multi-Layer Architecture Everywhere

A best practice in re-using components is to design them in layers. When you have a component in a layer, you can replace it easily, change it easily without needing heavy maintenance, and you can re-use it easily. Dataflows used for shared Power Query tables or Shared Dataset for Power BI thin reports are examples of multi-layer architecture.

However, the multi-layer architecture is not just about the Dataflow and Dataset. Anywhere in your architecture, think of methods that can make re-usability better. For example, if the data source is likely to change from SQL Server or Oracle, then it might make the change easier if you create a Data staging layer using Dataflows. The staging layer can then be easily changed without much change needed in the transformations or other layers.
Datasets can be in layers themselves. If you get data from a dataset and apply some changes to it, you create a chained dataset. This way, you are creating layers of Power BI datasets.

Sync Slicers

If you are using the same slicer across multiple pages, then it makes sense to sync slicers. Syncing slicers means that if you change the values of the slicer on one page, you don't have to re-do that change on another page. The slicers will be synced.
Syncing slicers can also help in creating a slicer page. This is particularly helpful if the number of fields you want to define a slicer on is too much to be part of a report page. In this case, you can create a single page with all the slicers, then sync it with other pages, and use buttons and bookmarks to go back and forth to the slicer page.

Use a Theme file

There are company color codes in many organizations. You may also want to set some pre-defined styles for font sizes, colors, borders and other visual effects in your report. It is not good to set it for each page and visual one by one. Maintenance of a solution like that won't be easy, and on the other hand, if someone else wants to follow your standards, they would have a hard time following.

You can create a Power BI Theme for visual standards. A Power BI Theme can include colors, fonts, or other generic visual properties you want to synchronize across the pages. The theme can then be stored as a JSON file and be re-used for other PBIX files too. All you need to re-use it is to browse for the Theme and apply.
If you create a Theme for a team of developers, then it is a good idea to keep the file in a shared folder so that everyone can access it and re-use the theme file.

Background image for report pages

In addition to the theme file, it is common to use background images for report pages. You can design a background image with the border, color, header and footer you want and design sections for visual settings. Then this background image can be easily used in report pages to give them a professional yet common look. The report user won't be distracted when moving from one page to another page with a different look. Your entire report will look like an application with a constant look and feel.

Conditional formatting using DAX, parameters, and tables

Visualization is more than just beautiful charts and graphs; it is the art of conveying the right message to the user. The right message can sometimes be passed by color-coding values of a visual like a table or matrix visual.

In Power BI, you can use conditional formatting to make some values more visible than others by color coding.
Conditional formatting can be done by hard-coding values directly in the visual configuration (which is not recommended), or it can be done using DAX measures, parameter tables and what-if parameters. The latter is a better approach for conditional formatting because if you do the conditional formatting based on a DAX measure, you can use the same DAX measure in other pages and visuals and have consistent color-coding across your reports. The color codes can be stored in a table for more straightforward configuration and maintenance too.

Measure table

Measures can get lost inside a Power BI data model with many tables. It can sometimes be hard to find which table you have created measures under. If you cannot remember the measure name, then you will expand each table and scan the measure names gradually. This can be enhanced by using a measure table.

A measure table is a blank table (can be created anywhere; Power Query, data source, DAX etc.) with no data. The valuable objects in this table are measures. Once you have created this table, you can move all the measures under it. This way, you can find measures all in one place faster.
You can also use the Display Folders in addition to the Measure table. Display folders give you folders and subfolders for better categorization.

Create and Use Hierarchies

Suppose you have a Product category, subcategory and product name. You may want to combine these three fields in multiple visuals. And perhaps in some of them, you want to have the ability to drill down and up in these three levels of product. Instead of going to each product and dragging and dropping these three fields one by one, you can create a hierarchy of these three fields and re-use that hierarchy in other visuals.
Although you can achieve the same outcome with drag and drop of the three fields separately, creating the hierarchy will give you consistency across your report and makes it easier to use it in other visuals. If you want to use two levels of hierarchy in one visual, you can easily adjust it at the visual level. Hierarchies are part of the data model that helps unify and conform the usage of fields.

Set Auto Summarization at the field level

Numeric fields are auto-aggregatable in Power BI unless they are used in a relationship or some other conditions are applied to them. This is very helpful for fact fields, such as Sales or Budget. However, If you use a Date table with the Year column numeric, or if you have a parameter table with the age column on it, you won't want the Year or the Age columns to be automatically summarized when using it in Power BI reports.

It is recommended to set the auto summarization (or auto aggregation) on each field adequately based on that field at the model level so that when you use it in visuals, they follow the right summarization. This is a simple configuration to apply, but it helps a lot in future reporting and visualization from your model.

Consider using a custom Date Table

Although Power BI comes with a default Date table, for many advanced use cases, you will need extensions of a Date table. If you want to slice and dice data by weekdays vs weekends, holiday data analysis, or many other scenarios will require a custom Date table.
You can create the custom Date table using Power Query, DAX, or even back in the data source. If you use Power Query to create it, then it would be better to do it in a Dataflow so that you can re-use it in other files. Once you have created the custom Date table, make sure to Mark it as a Date table so that the Time Intelligence calculations work correctly, and disable the Auto-Date/Time option in the Power BI.

Design Mobile reports

Power BI reports are designed for interactivity. The new era of reporting and dashboarding is not relying on offline reporting methods, such as printed papers discussed in board meetings. Nowadays, using technologies such as Power BI, report users can access live, interactive reports and dashboards from their mobile devices everywhere. This is not only reducing the need for printing, but it also helps in users' contribution to the reports and sending feedback back to the developers.

Although Power BI reports are mobile-friendly by default, however, in order to view them in their best mode on a mobile device, the report developer has to design the mobile view of the report and dashboard. This is a simple yet effective process. Report pages can have different configurations and settings for the visual layout for the desktop vs mobile view (such as different font sizes on mobile or a different layout of visuals on the mobile view).
Creating mobile-friendly views for your reports and dashboards is a big step towards the adoption of Power BI in your organization.

Use Aggregations for big tables

If you are working with large tables and the reports are slow because of the enormous amount of rows in tables, then aggregations are something you need to consider. Aggregations can be done on top of the tables that are big (either DirectQuery or even Imported). Aggregation is a grouped version of the original table based on a few columns. This grouped table (an aggregated table) is a layer between visualization and the main table. Because the aggregated table is smaller in size, then querying data from it will be faster. If the visualisation values can be fetched from the aggregated table, then the aggregated table will be used; otherwise, the main table will be. This way, the performance of calculations and visualizations will increase by using smaller aggregation tables. You can have multiple layers of aggregations to support different visualizations.
Defining aggregations on big tables in Power BI is a crucial performance-tuning step in Power BI modeling. Aggregations can also be automatically generated based on the usage of the fields; the auto-aggregation can be enabled as a Power BI Premium functionality.

Use Incremental Refresh and Hybrid tables Where Possible

Loading the entire data of a table might sometimes take a long time, especially if the table has many rows. Instead of re-loading the entire data of the tables every time, you can only re-load the part of the data that is changing, and the historical data can be loaded once. This process is called Incremental Refresh. Using Incremental Refresh, you can load historical data (let's say for the past ten years) only once and then re-load the last period (let's say the last year) every time the dataset refreshes. This process will make the dataset refresh time much faster. This is mainly helpful with the data sources supporting query folding because, in that case, instead of reading the 11 years of data, you only read and transfer one year of the data each time.

Hybrid tables can also accompany incremental Refresh. Hybrid tables are tables that have part of their data in real-time using DirectQuery to the source table (usually the most recent period of the data) and the remaining data to be imported. Hybrid tables are suitable for scenarios the data freshness is also needed in a table with many rows where incremental refresh is set up already.

Design Star Schema Models

Designing the structure of tables and their relationships is one of the primary tasks of every BI system. A common best practice in designing the table structure is called Star-Schema. In this type of design, tables are categorized into two types; Dimension and Fact tables. Fact tables include the numeric and additive measures and the dimensions, including the descriptive fields that slice and dice the data of the fact table. The relationship between the Dimension and Fact tables is one-to-many from the Dimension table to Fact table. There can be multiple dimensions per fact table, and there can be multiple fact tables inside the model. The design of dimension and fact tables are dependent on the reporting requirements.

Avoid both-directional relationship

A vital performance consideration in a Power BI dataset is avoiding both-directional relationships. The direction of a relationship in Power BI is how the filter propagates between the tables. Sometimes to get the values of a field by another field in another table, you may feel the need to change the direction of the relationship. However, changing a relationship to both-directional comes at the big cost of performance reduction and ambiguity and confusion in the data model.

A star-schema-designed model usually doesn't need many both-directional relationships. However, on rare occasions, if required, it can be evaluated separately, and the solution can be designed without a both-directional relationship. Usage of DAX functions such as CrossFilter inside a measure can also be a remedy for scenarios where the both-directional relationship seems the only method to work. Using the measure instead of changing the direction of the relationship can result in better performance because that measure may not be used in all the report pages, so the performance reduction may not occur all the time.

Model cleanup

It is a good idea to review your model over time. In the review, you may find some fields that are not used anymore and some measures and calculations that are created for test purposes and can be removed. Cleaning up your model is a task that can be done over time. Fortunately, many community tools can help you with that. I use the Power BI Helper, which is a free tool for this purpose and many other helpful Power BI development features.