POWER BI

SUCCINCTLY

BY **PIERSTEFANO TUCCI**

SUCCINCTLY E-BOOK SERIES

Syncfusion®

Power BI Succinctly

By
Pierstefano Tucci

Foreword by Daniel Jebaraj

Technical Reviewer: James McCaffrey
Copy Editor: John Elderkin
Acquisitions Coordinator: Morgan Weston, social media marketing manager, Syncfusion, Inc.
Proofreader: Jacqueline Bieringer, content producer, Syncfusion, Inc.

Table of Contents

The Story Behind the Succinctly Series of Books **7**

About the Author **9**

Chapter 1 What is Power BI? **10**

Power BI Scenario 11

Chapter 2 Service **14**

Data sources 14

The supported browser and mobile applications 14

Building blocks 15

Service types 18

An example 20

Visual gallery 33

Natural language capability 37

Sharing and cooperation 38

Cooperate 39

Import data sources: focus 46

Chapter 3 Desktop **54**

An overview of the Power BI site 54

Power BI Desktop 54

Development method 54

Step definition of a query: the controls 59

Configure relations 61

Data model definition 62

Data model definition: calculations 63

Calculated column 63

Measure 63
Design report 64
Publication on Power BI 64
A work example with Power BI Desktop app 64
Chapter 4 Mobile Apps 81
Windows 8 and Windows 10 81
App for Android 89
App for iPhone 90
App for Windows Phone 91
Chapter 5 Solution Template 98
Content pack 100
Data refresh and schedule 104
Power BI Gateway 107
Chapter 6 Developer 113
Power BI REST API 113
Power BI custom visual 121
Chapter 7 Power BI Embedded 130
Power BI Tiles 130
Power BI Embedded 133
What is Power BI Embedded 133
Licensing for Microsoft Power BI Embedded 133
Workspace Collection 134
Workspace 134
Cached datasets 135
Authentication and authorization through the app tokens 135
Configure the example application 135
Run the sample web app 138

Explore the sample code 139

Chapter 8 Power BI Gateway: Data Security 145

The Story Behind the *Succinctly* Series of Books

Daniel Jebaraj, Vice President
Syncfusion, Inc.

Staying on the cutting edge

As many of you may know, Syncfusion is a provider of software components for the Microsoft platform. This puts us in the exciting but challenging position of always being on the cutting edge.

Whenever platforms or tools are shipping out of Microsoft, which seems to be about every other week these days, we have to educate ourselves, quickly.

Information is plentiful but harder to digest

In reality, this translates into a lot of book orders, blog searches, and Twitter scans.

While more information is becoming available on the Internet and more and more books are being published, even on topics that are relatively new, one aspect that continues to inhibit us is the inability to find concise technology overview books.

We are usually faced with two options: read several 500+ page books or scour the web for relevant blog posts and other articles. Just as everyone else who has a job to do and customers to serve, we find this quite frustrating.

The *Succinctly* series

This frustration translated into a deep desire to produce a series of concise technical books that would be targeted at developers working on the Microsoft platform.

We firmly believe, given the background knowledge such developers have, that most topics can be translated into books that are between 50 and 100 pages.

This is exactly what we resolved to accomplish with the *Succinctly* series. Isn't everything wonderful born out of a deep desire to change things for the better?

The best authors, the best content

Each author was carefully chosen from a pool of talented experts who shared our vision. The book you now hold in your hands, and the others available in this series, are a result of the authors' tireless work. You will find original content that is guaranteed to get you up and running in about the time it takes to drink a few cups of coffee.

Free forever

Syncfusion will be working to produce books on several topics. The books will always be free. Any updates we publish will also be free.

Free? What is the catch?

There is no catch here. Syncfusion has a vested interest in this effort.

As a component vendor, our unique claim has always been that we offer deeper and broader frameworks than anyone else on the market. Developer education greatly helps us market and sell against competing vendors who promise to “enable AJAX support with one click,” or “turn the moon to cheese!”

Let us know what you think

If you have any topics of interest, thoughts, or feedback, please feel free to send them to us at succinctly-series@syncfusion.com.

We sincerely hope you enjoy reading this book and that it helps you better understand the topic of study. Thank you for reading.

Please follow us on Twitter and “Like” us on Facebook to help us spread the word about the *Succinctly* series!

About the Author

Pierstefano Tucci is a developer and data analytics consultant. He was born in Italy and has a strong DB and BI background as well as a special interest in languages such as SQL, U-SQL, R, .NET, HTML, JS, and CSS. He has a bachelor's degree in economics and management along with a master's degree in IT security and computer forensics. He is currently studying for a fourth degree in computer engineering. He works in the IT department of an international company.

Pierstefano is enthusiastic about technology, especially Microsoft technology. He is grateful to the Syncfusion team for the opportunity to write this e-book.

Chapter 1 What is Power BI?

Power BI is a cloud-based business analytics service that gives us a single view of the most critical business data. Using a live dashboard and creating rich, interactive reports, Power BI allows access to data for monitoring the health of a business. Essentially, it is a tool for creating our own BI reports and dashboard. Power BI is not complicated—it is powerful, easy, and for everyone.

Over time, there have been three BI waves.

1st Wave—Technical BI (IT people-to-end users): In order to achieve BI results, in the past companies had to rely on consultants and IT professionals in order to develop Analysis Server Models. These professionals would develop a data warehouse and then create reports and analysis in order to give end users options for making sound business decisions. However, there was a problem: there were not enough IT people. And, actually, this is the problem we still have today. Because of a lack of IT professionals, oftentime companies, which are always looking for reports, must wait too long while assuming the additional risk that what they eventually obtain is not still valid.

2nd Wave—Self-service BI (analysts-to-end users): This wave included PowerPivot Excel, which allowed users to make their own data analysis. This worked well. Data analysis and data scientist consultants began to buzz in the BI world. However, this wave was not as innovative as first expected because, despite improvements, the data analysis was still difficult.

Data analysis requires a flexible mindset that comes from traveling a long path of rethinking both our mental and data approaches.

3rd Wave—BI wave (end user BI). The aim is to give business intelligence to all. What is the difference between the first wave and the second wave? In the first, we have tools such as ClickView and PowerPivot. In the third wave, end users not only have a tool, but a tool series. Different tools for different users. The idea is that each user has a unique platform with many choices available for their unique needs.

Developers who take care of ops integration; data scientists who take care of data analysis; visualization experts who take care of data visualization and choose the best visualization option—each professional has a role in developing data analysis in Power BI.

However, some issues remain.

The first problem involves access to data. Data is often stored in a database, and on-premises data decreases over time. Data is typically stored in the cloud, in an Excel file, or on websites. We need to collect information and bring it into our own analysis systems. In order to do that, we also need to have data access and the correct tools.

In general, our data is stored in a database, but often we do not know exactly where it is. If we use a web service, we will have the access credentials to the portal, but we still won't know where the data is saved. For example, Google Analytics allows to us analyze data access for websites, but we do not know where the database is. We can't plug a direct cable into the Google databases and collect or select data. We need a tool to simplify this process, one that starts from Google Analytics and uses its own process or application.

The second problem involves security and accessibility. We need our data stored in a virtual machine, otherwise that data is available for some users (or user groups) but not for everyone. We need to implement safety mechanisms and rules—user mechanisms that allow particular groups to view different information—in order to achieve this security goal.

We can solve all these problems with Power BI.

Power BI Scenario

In order to understand the Power BI scenario, we must understand clearly who the content creators are and who uses the content.

Generally, the following positions are present within an organization:

The **Executive** needs heavily aggregated information in order to see a high-level image of the company's state or a particular functional area. This information for the executive is often presented in the form of a dashboard or a scorecard, and it is bound to raise strategic questions.

The **Analyst** needs raw, or slightly summarized, data with the aim of creating a detailed analysis of a specific problem or business opportunity. The analyst presents the data in the form of spreadsheets, presentations, or ad hoc reports that focus on the specific problems they have been called upon to solve.

The **Manager** needs data, which provides a detailed analysis of an area, a specific business, or a function. The data for the manager is used to evaluate past performances or to plan future activities. The data must have a detail level that can define specific actions, for example a sales plan or a marketing activity for the launch of a product. The data for the manager is usually supplied as reports or scorecard details.

The **Operational Users** deal with data on a transactional level. For example, an operating report in which the invoice registers information that comes from bookkeeping or a daily production plan. The operational data is usually shared in the form of reports. More and more frequently, the data for the operational users is shared online through an Intranet or directly through mobile devices.

All of the people in these positions work and cooperate by using tools of Business Intelligence.

The data visualization and discovery tool is important for several reasons. First, it allows us to use a very strong and sensitive organ—our eyes. Through our eyes we interpret the information correctly. It is a sophisticated instrument, so much so that machine learning algorithms still rely on it.

Second, the data can deceive us sometimes.

In 1973, Frank Anscombe produced four collections of information that he called "the quartet," to demonstrate this idea. He drew 11 points in each quadrant of his diagrams. The points have very similar statistical features, and they represent variance, average values, correlation, and linear regression. The values vary by a few percentage points. The way in which the points give us completely different information is shown in these graphs. The graphs allow us to identify the outliers—the individual points that are meaningfully different from others and therefore provide us with the information needed to write our analysis.

I		II		III		IV	
x	y	x	y	x	y	x	y
10.0	8.04	10.0	9.14	10.0	7.46	8.0	6.58
8.0	6.95	8.0	8.14	8.0	6.77	8.0	5.76
13.0	7.58	13.0	8.74	13.0	12.74	8.0	7.71
9.0	8.81	9.0	8.77	9.0	7.11	8.0	8.84
11.0	8.33	11.0	9.26	11.0	7.81	8.0	8.47
14.0	9.96	14.0	8.10	14.0	8.84	8.0	7.04
6.0	7.24	6.0	6.13	6.0	6.08	8.0	5.25
4.0	4.26	4.0	3.10	4.0	5.39	19.0	12.50
12.0	10.84	12.0	9.13	12.0	8.15	8.0	5.56
7.0	4.82	7.0	7.26	7.0	6.42	8.0	7.91
5.0	5.68	5.0	4.74	5.0	5.73	8.0	6.89

Figure 1: Anscombe's Quartet (source: Wikipedia)

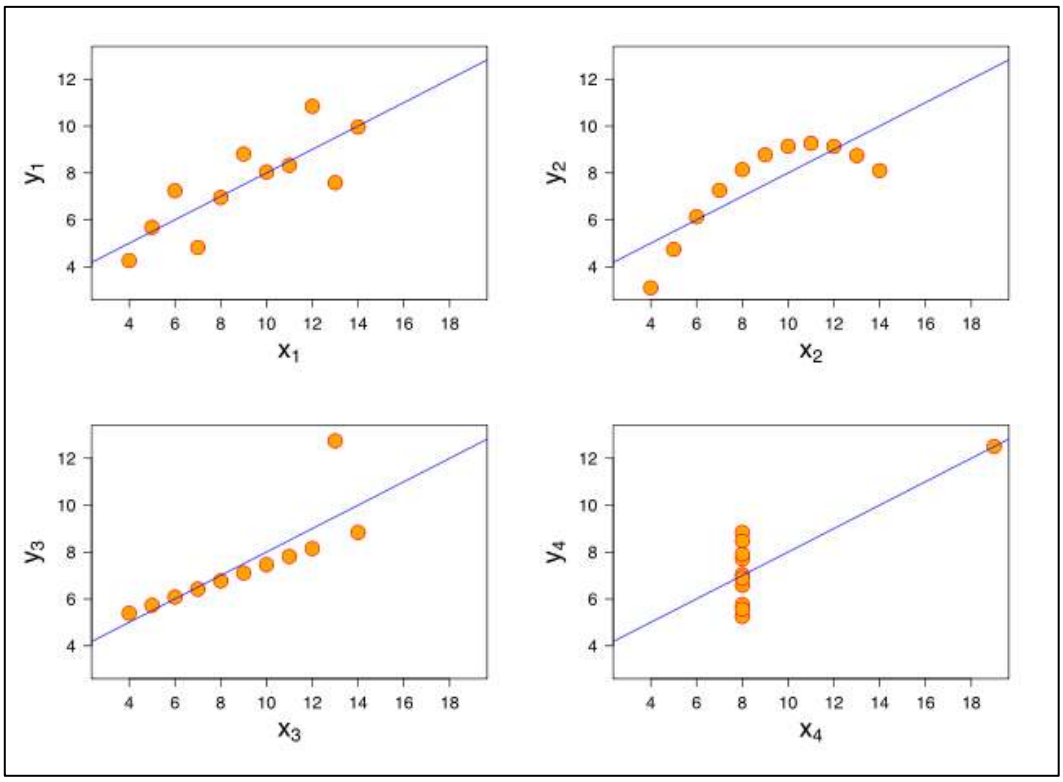


Figure 2: Anscombe's Quartet (source: Wikipedia)

Property	Value	Accuracy
Mean of *x*	9	exact
Sample variance of *x*	11	exact
Mean of *y*	7.50	to 2 decimal places
Sample variance of *y*	4.125	plus/minus 0.003
Correlation between *x* and *y*	0.816	to 3 decimal places
Linear regression line	$y = 3.00 + 0.500x$	to 2 and 3 decimal places, respectively

Figure 3: Used Property (source: Wikipedia)

We can deduce that, once our data reaches millions or billions of rows, it would be helpful for the analysis to have an expression power such as Power BI. This helpfulness is another reason why the Power BI tool is having great success. Users prefer Power BI because of its easy usage, great availability of visualizations, and capacity to integrate data from different sources.

Power BI is arguably one of the two or three most powerful and sophisticated data analysis programs. Research firm Gartner places it as the leader among the Magic Quadrant in business intelligence, transactional systems, data warehousing, and advanced analytics. At the time of this writing, the Microsoft data platform is first in all the reports regarding safety, performance, and capacity to do advanced analytics “in-place” (that is, inside the data contained in the database). By adding the in-memory and the capability to work in either on-premises systems or in the cloud, as well as in hybrid environments, we can obtain the full richness and the power of this tool.

Two licenses of Power BI are available:

- Power BI: Free
- Power BI Pro: Currently at $9.99 a month with annual subscription

Chapter 2 Service

Power BI is not a product we install, but rather a product that is carried out in the Microsoft Cloud that we can use in different interfaces.

The service allows us:

- Simple and quick data access
- Complete real-time vision of the business
- Search and discovery of the data (in order to create dashboards and reports)
- Data analysis from any device
- Sharing within the organization
- Company-wide displaying and analyzing of the data

Data sources

There are many types of data sources:

- SaaS solutions, e.g., Marketo, Salesforce, GitHub, Google Analytics, CRM Online
- On-premises data, e.g., SQL Server Analysis Services
- Organizational content packs, e.g., corporate data source or external data services
- Azure services, e.g., Azure SQL, Stream Analytics
- Excel files, e.g., workbook data/data models
- Power BI Desktop files, e.g., data from files, databases, Azure, and other sources

The organizational content packs stand out in the data sources. They offer a simple and intuitive way of sharing data within your company.

A program interface has been introduced in Power BI through a RESTful API that allows interaction directly with the service, building datasets and, if necessary, importing our Power BI visualizations inside our applications as web applications.

In order to try Power BI, you can register for the service free of charge. But be sure to pay attention and use an Office 365 email address, which must end with the domain microsoft.com (you can also use a company email account). Email accounts from many public servers, such as gmail.com, live.com, hotmail.com, and outlook.com do not allow the registration.

When you have registered, you can test the functionalities. It is possible to update the license at any time, simply by subscribing to Power BI Pro.

The supported browser and mobile applications

Because it is an online service, we can use Power BI directly from the browser, through a supported web browser or a mobile app. The supported browser versions are:

- Microsoft Edge

- Internet Explorer 11
- Chrome desktop, latest version
- Safari Mac, latest version
- Firefox desktop, latest version
- Power BI (works with older browser versions, but may have rendering problems)

Along with the browser, we can also use these mobile apps:

- Windows
- iOS (iPhone and iPad)
- Android
- Windows Phone 10

The supported functionalities are:

- Definition of the Favorites bar for the most important visualizations.
- In/out zoom of the visualizations
- Annotations of the visualizations and snapshot sharing.
- Alert configuration for receiving reports about the critical business KPI.

Building blocks

Power BI is composed of “tiles,” which are the main elements of Power BI. Called “building blocks,” they are:

- **Dataset**: the data collection that is the subject of analysis.
- **Report**: allow us to view our data in different ways, which can give us greater insights and allow us to draw deeper conclusions.
- **Dashboards**: allow us to monitor the most important data at a glance. We can select the visualizations/reports we want to publish to the dashboards through the pinning mechanism and combine them in order to offer a glance at the business.

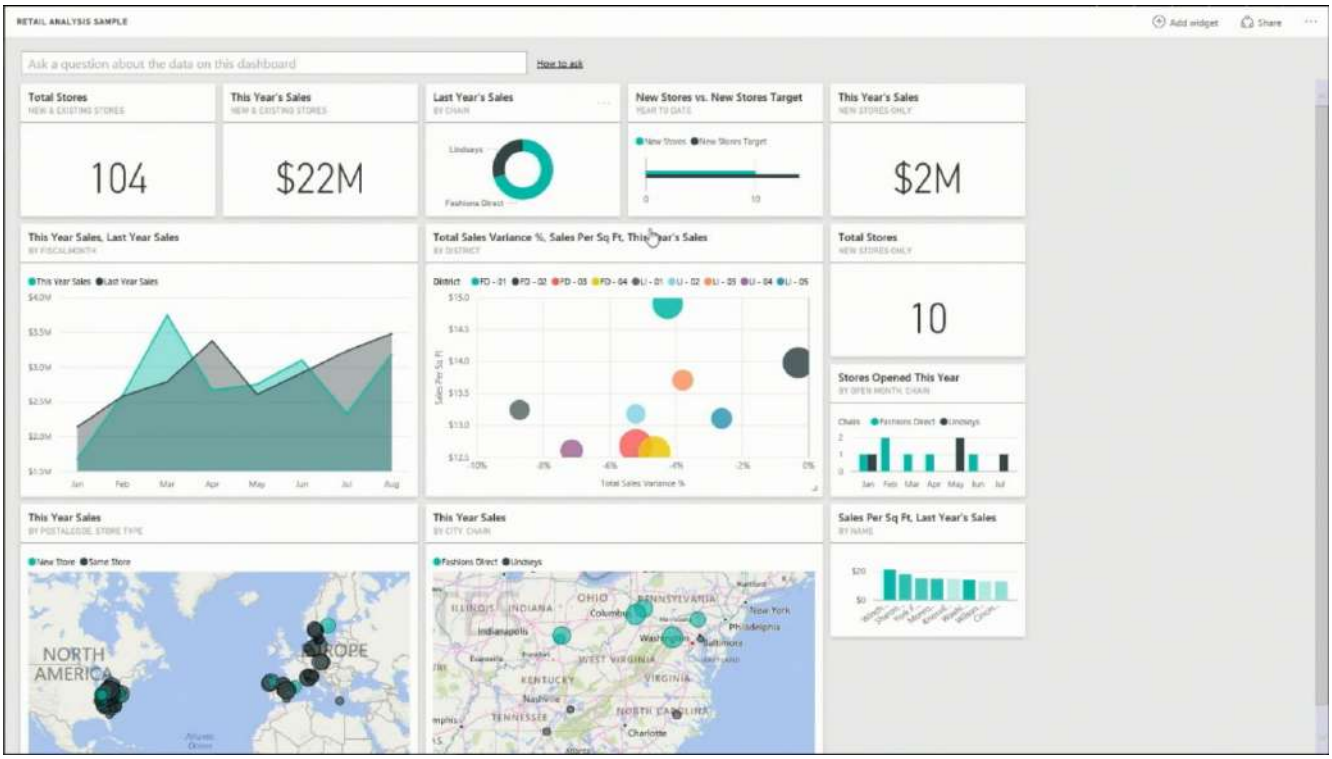


Figure 4: Dashboard Examples

It is possible to use the Navigation pane to create or select the elements, or building blocks, of interest.

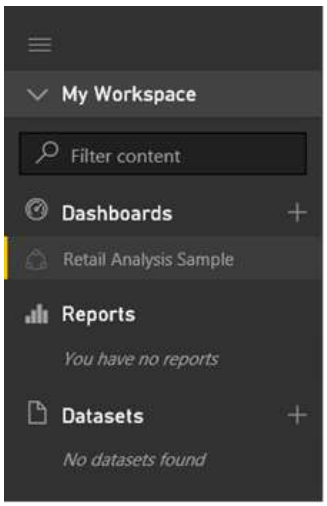


Figure 5: Building Blocks

Note that the dashboards show the tiles in a single canvas.

A tile derives from the pinning of a single display in a report or from a Q&A answer.

A dashboard can be based on one or more datasets.

The tiles can be resized, and they can be updated.

As Figure 6 shows, you can use Q&A to ask questions in the dashboard and in the natural language.

Figure 6: Q&A Field

In order to connect to a dataset, the dashboard must contain at least one tile.

Only cloud-based datasets associated with the current dashboard are supported. Q&A provides support for the formulation of questions, and Q&A also formats the questions with suggestions, autocomplete, and even corrections. Note that only questions in basic English are supported. Using meaningful nouns within the basic model works best.

A report is built starting from a dataset. The model is displayed, and we select the data from the dataset directly in the report.

Power BI Services is composed of the following areas seen in Figure 7.

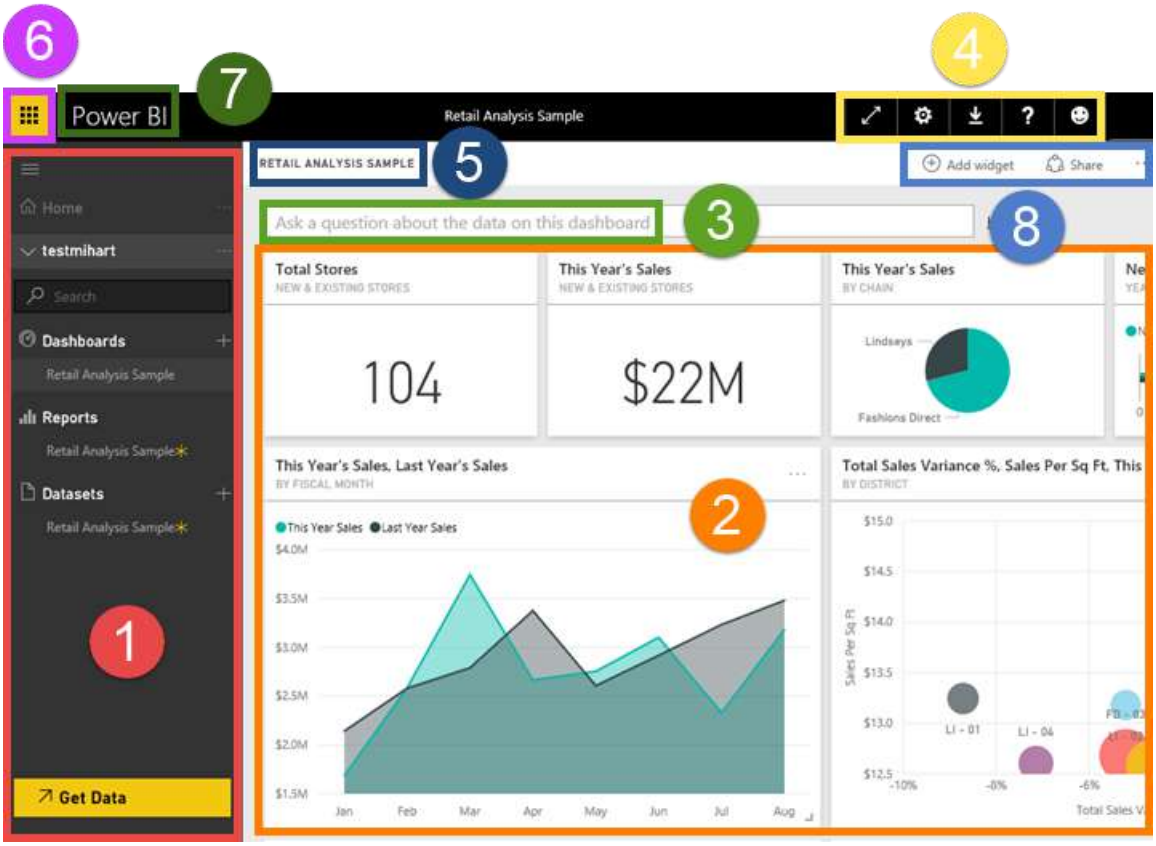


Figure 7: Power BI Service

- Navigation bar
- Dashboard with tiles

- Q&A question box
- Help and Feedback buttons
- Dashboard title
- Office 365 app launcher
- Power BI home buttons
- Additional dashboard actions

Service types

Power BI Services is, in fact, a particular service that we can access through a set of functions. It offers a number of functionalities and is free, so we refer to this version as the “freemium.” There is also a pay version of the same service, Power BI Pro, which does not have the limitations of the freemium version.

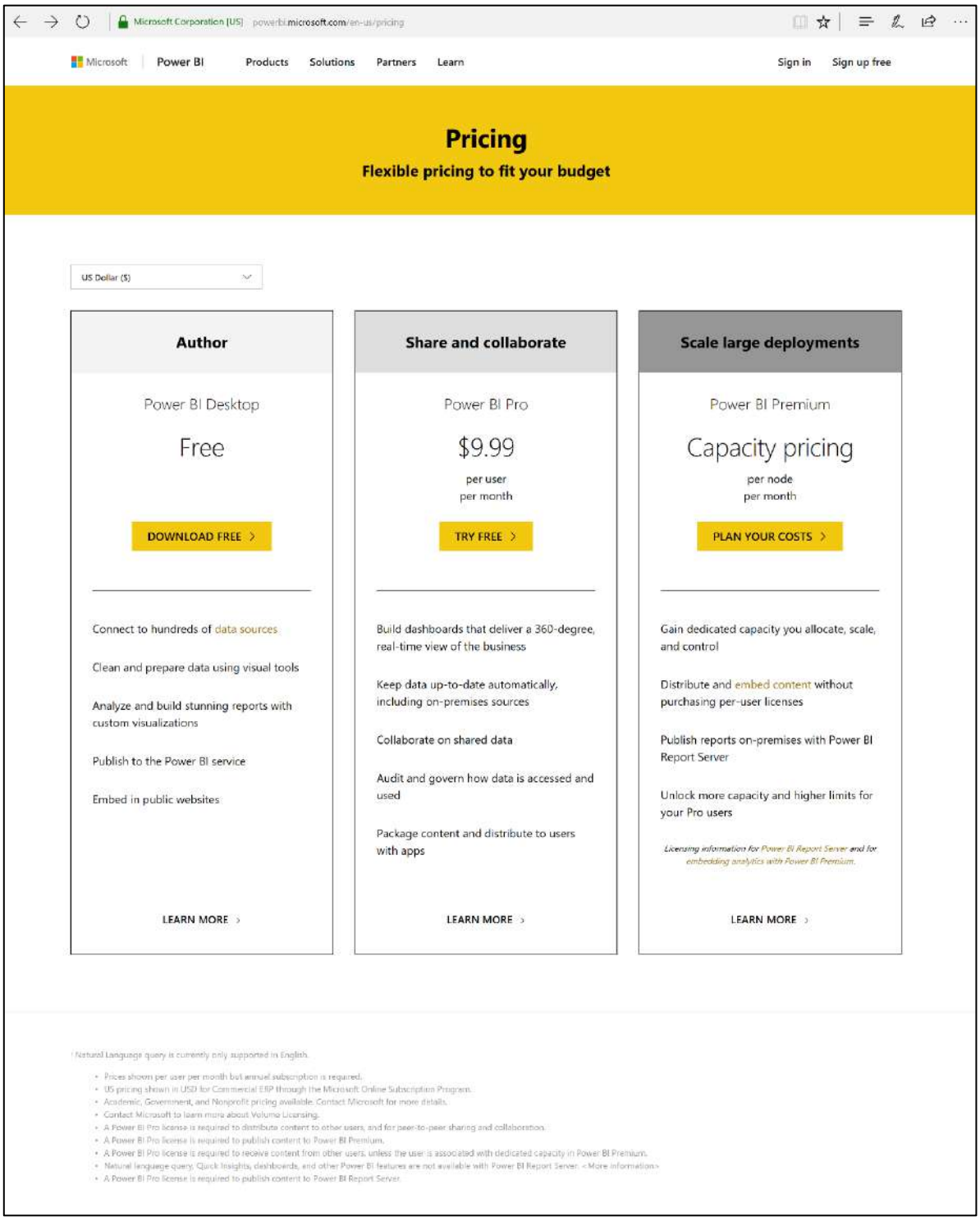


Figure 8: Power BI Licensing

Note that the “publish to the web” option is a free functionality in both versions. However, the data update and the cooperation of the two versions differ from each other—with the pay version, the data can be updated with very high frequency. Also, with the pay version we have 10 GB available for data storage and processing, while with the free version we have only 1 GB.

With the pay version, we can carry out planned updates more than eight times a day. With the free version, the updates can be carried out only once a day.

An example

Let's start from zero. We import a certain example of data that we can download from this website: Financial Sample workbook for Power BI. Note that the data can be loaded from:

- My Organization
- Services
- Files (directly from the system file)
- Databases

Uploading the data from the system file, a CSV in our example, Financial Sample workbook for Power BI, the data model is imported quickly and available within a few seconds.

The fields are automatically recognized by Power BI.

Figure 9 shows that if you drag the Sales field to the white page of the report, Power BI will display the data in the method it considers most appropriate at the time.

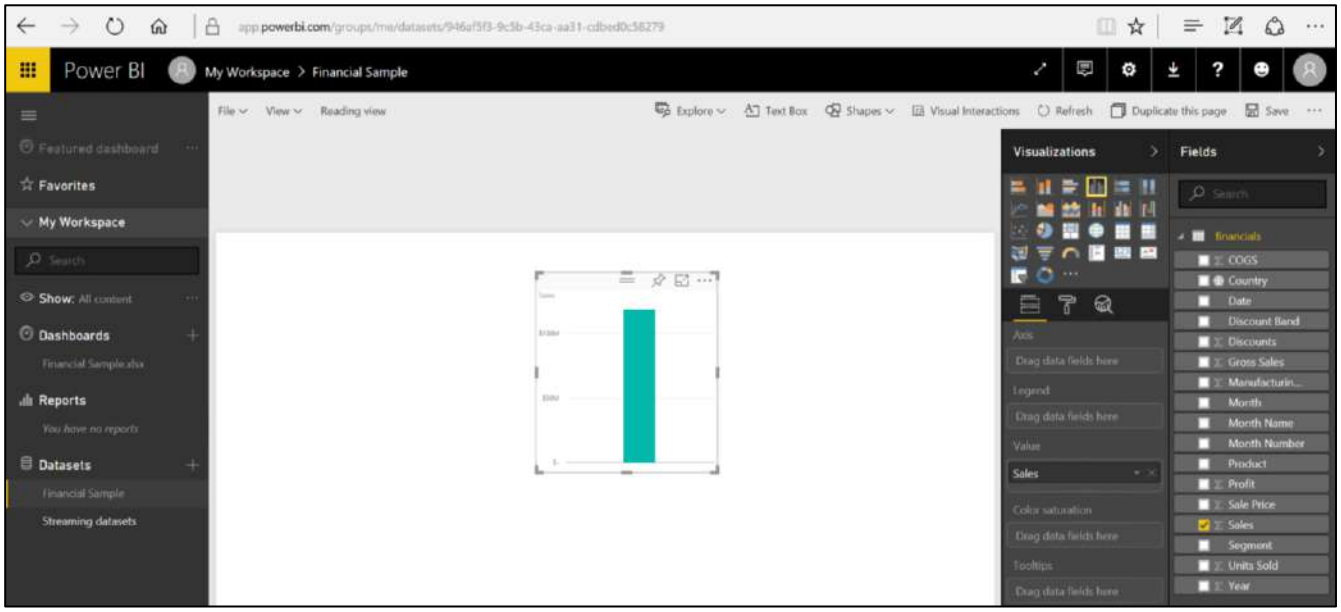


Figure 9: Power BI Displays Data

If you drag the Year field to the graph, as in Figure 10, you will immediately notice that Power BI reworks the visualization by adding the year.

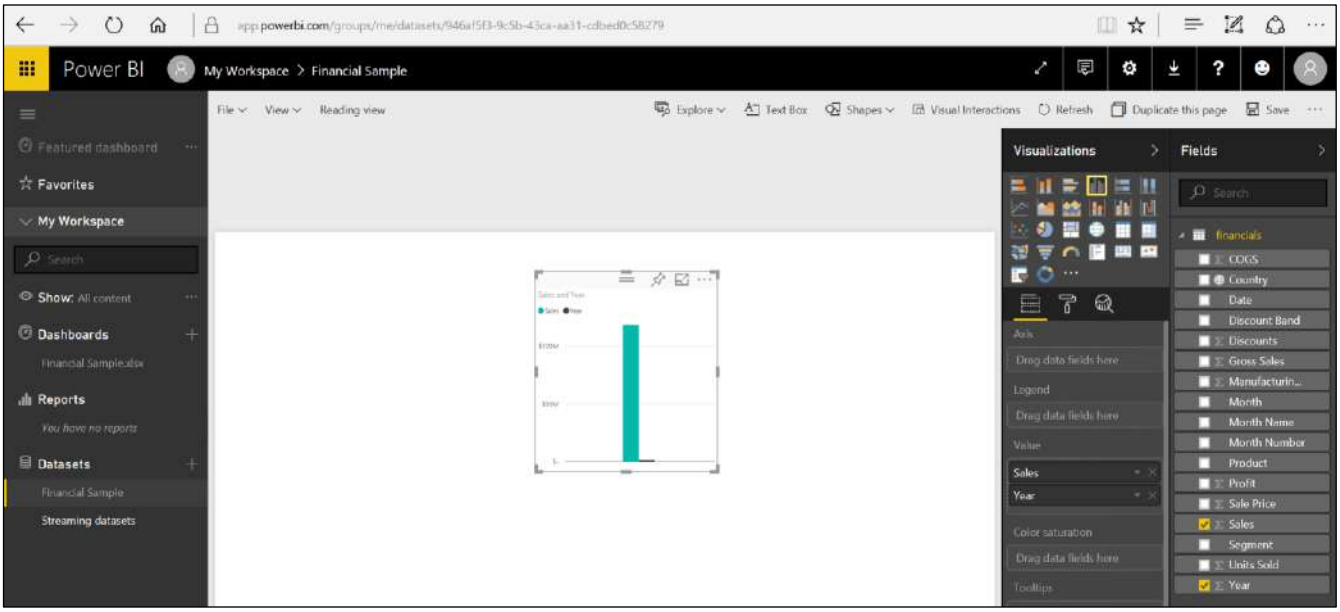


Figure 10: Power BI Displays Data

We can move the year to the abscissas field and see how Power BI now changes the visualization by taking the modification into consideration, as in Figure 11.

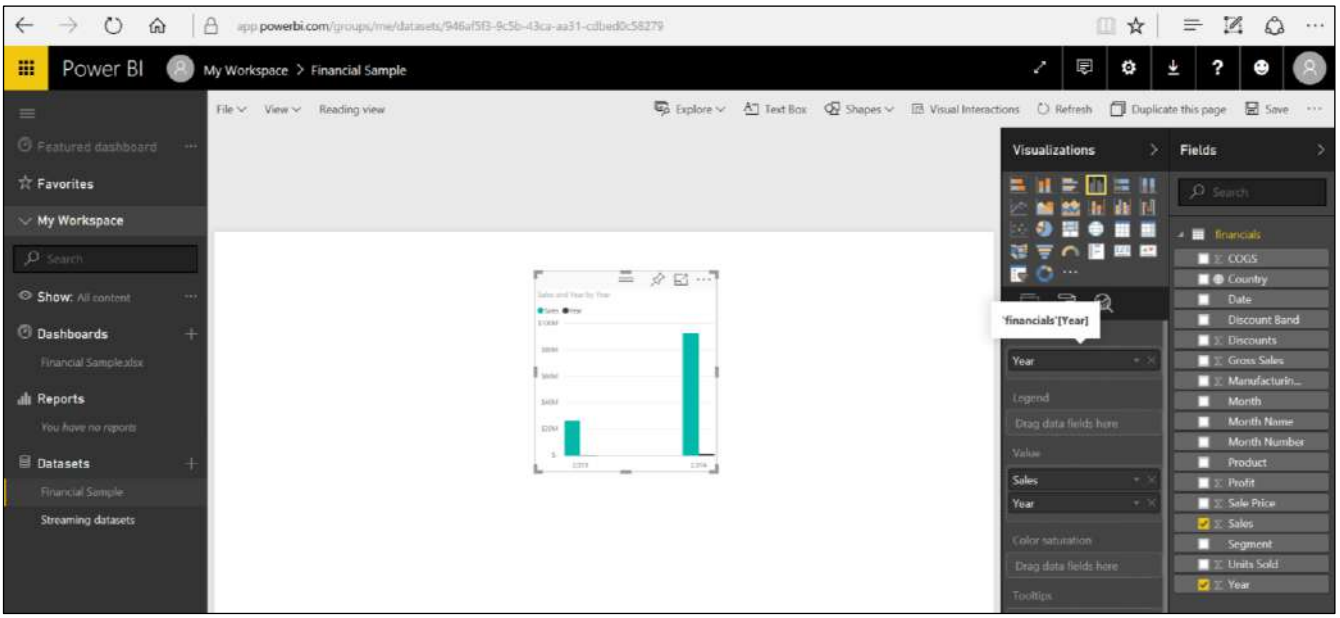


Figure 11: Power BI Displays Data

We can implement the drill down by placing the Month field under the Year field (in the x-axis section). The drill down is enabled by clicking the round icon with the “down” arrow. If we analyze the data by clicking the Year, the visualization changes by also showing the Months.

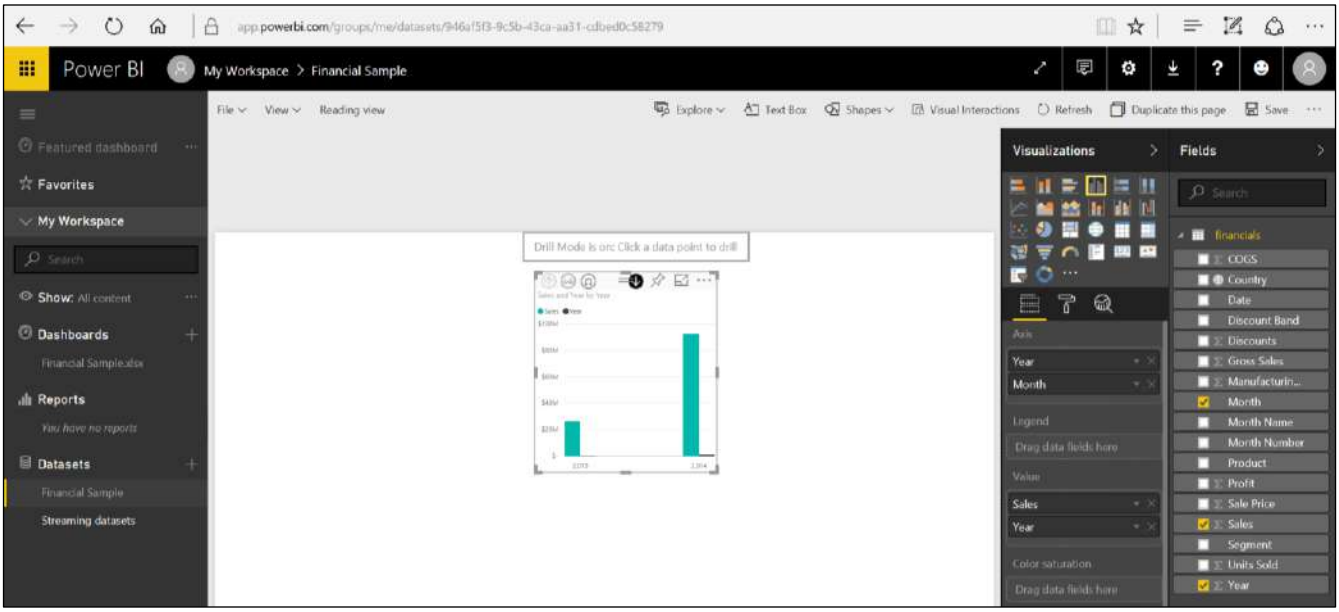


Figure 12: Power BI Displays Data—sequence

Note that by double-clicking on one particular year, we can go into detail.

If we want to go up with the visualization, we must click the “Drill Up” icon.

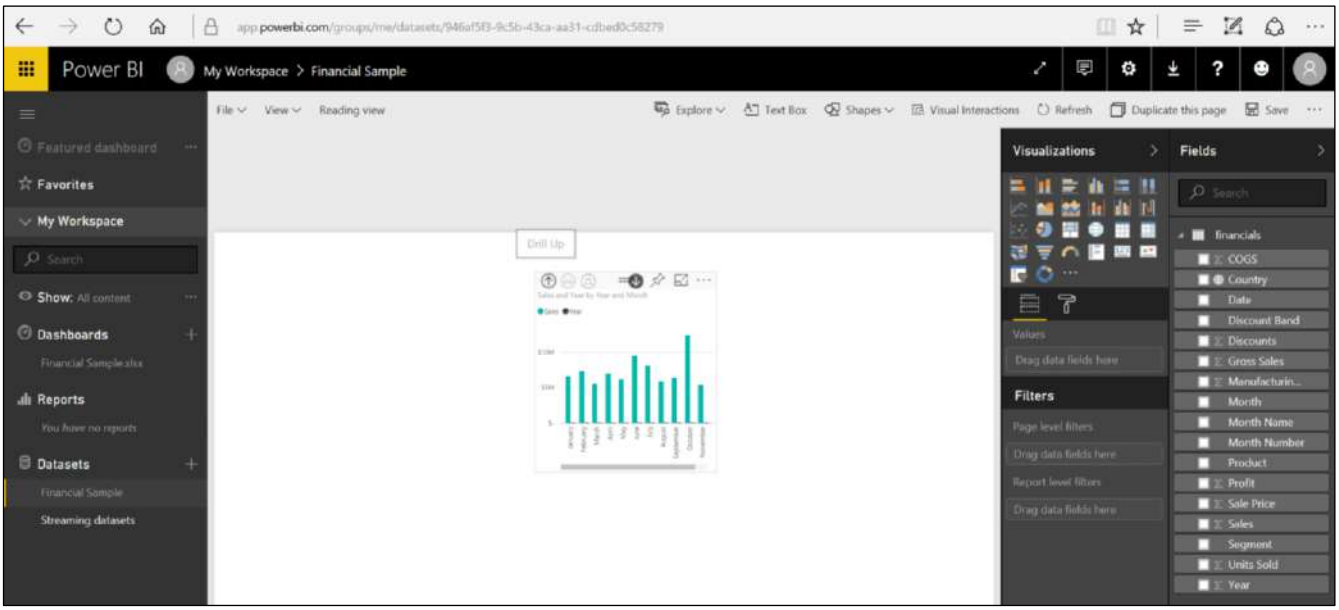


Figure 13: Power BI Displays Data

Afterward, we add another visualization—a slicer.

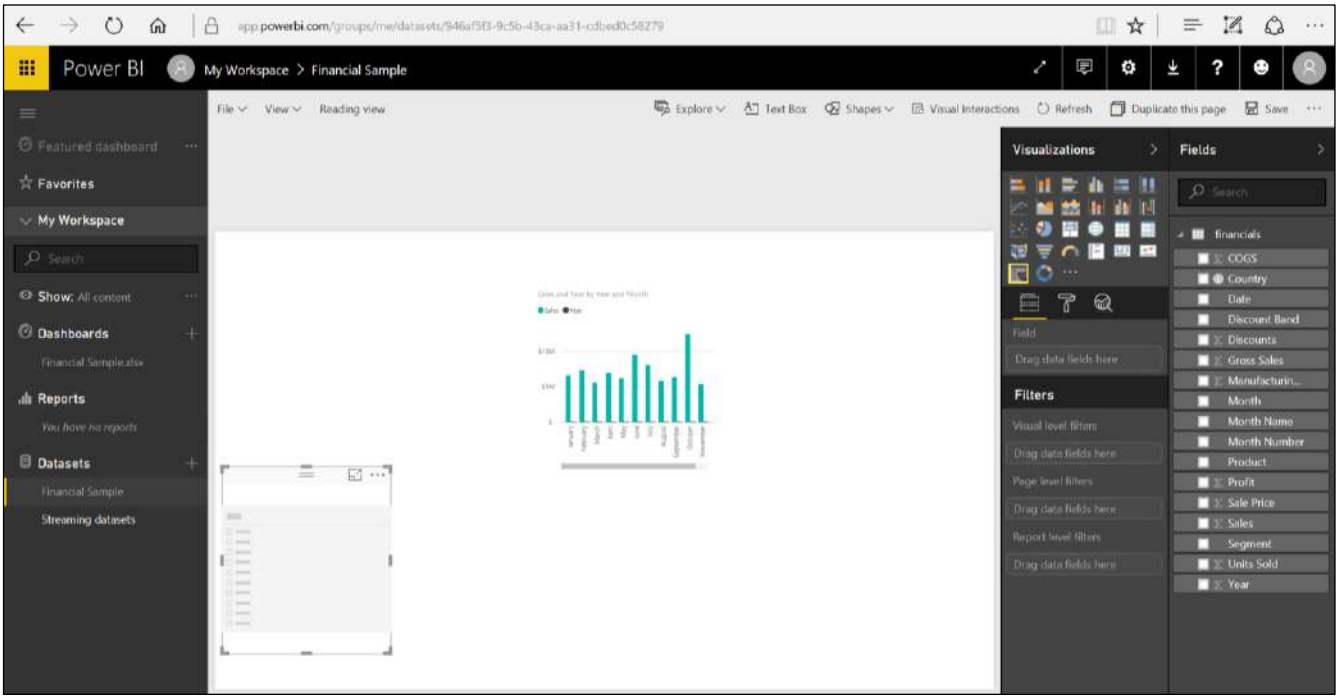


Figure 14: Power BI Displays Data—Slicer

Next, as we see in Figures 15 and 16, we select the Segment field and drag it into the slicer.

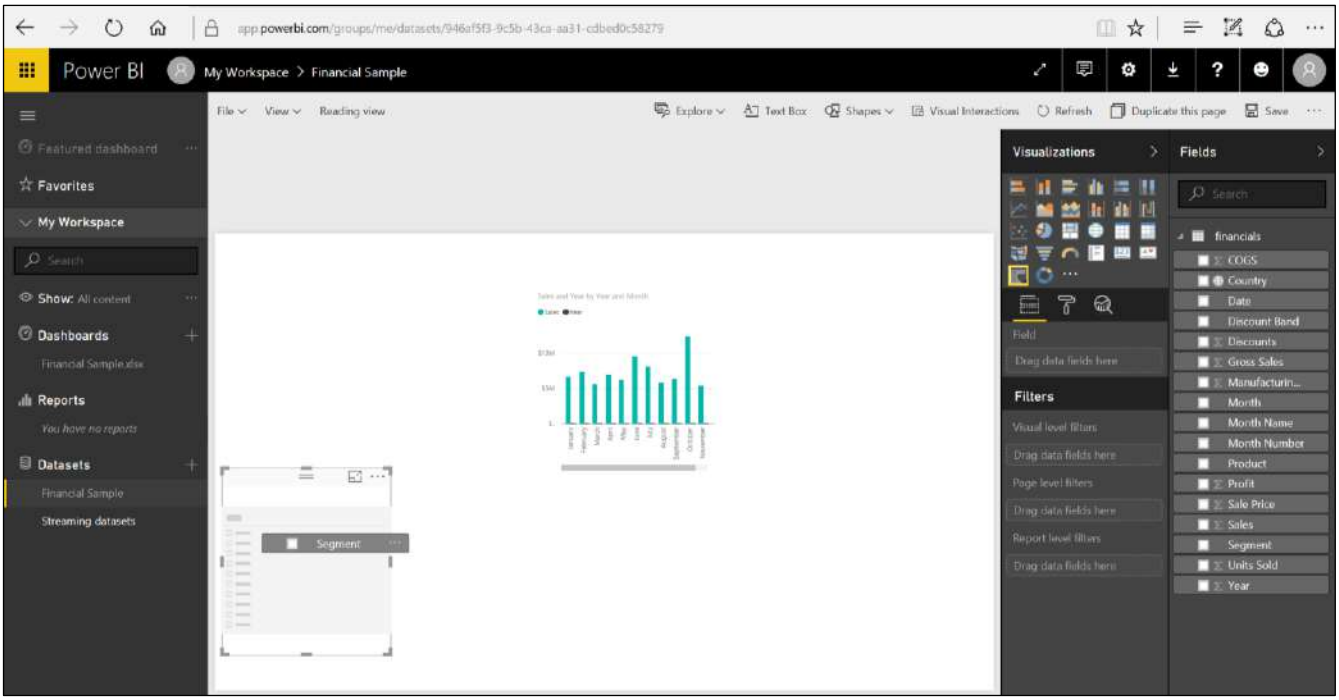


Figure 15: Power BI Displays Data—Use of Slicer

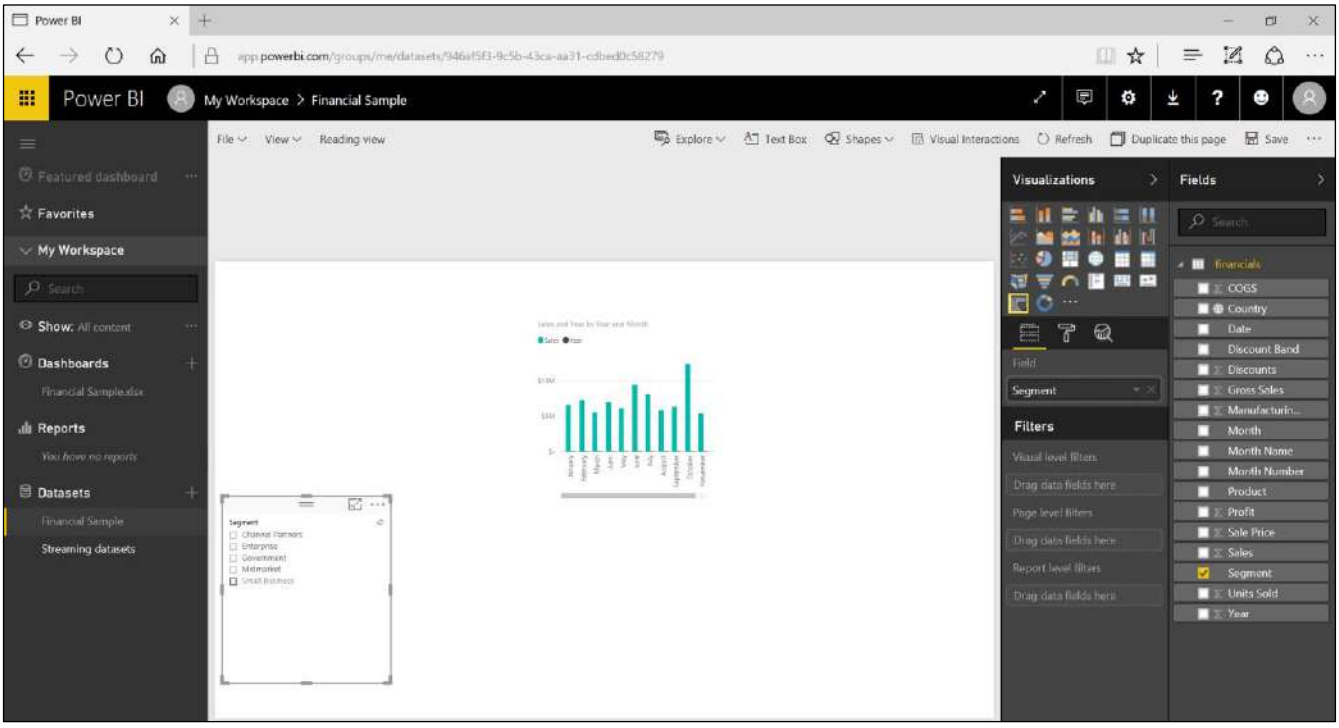


Figure 16: Power BI Displays Data—Use of Slicer

We can interact with the report by noticing how the selection of a value in the slicer affects the adjacent visualization.

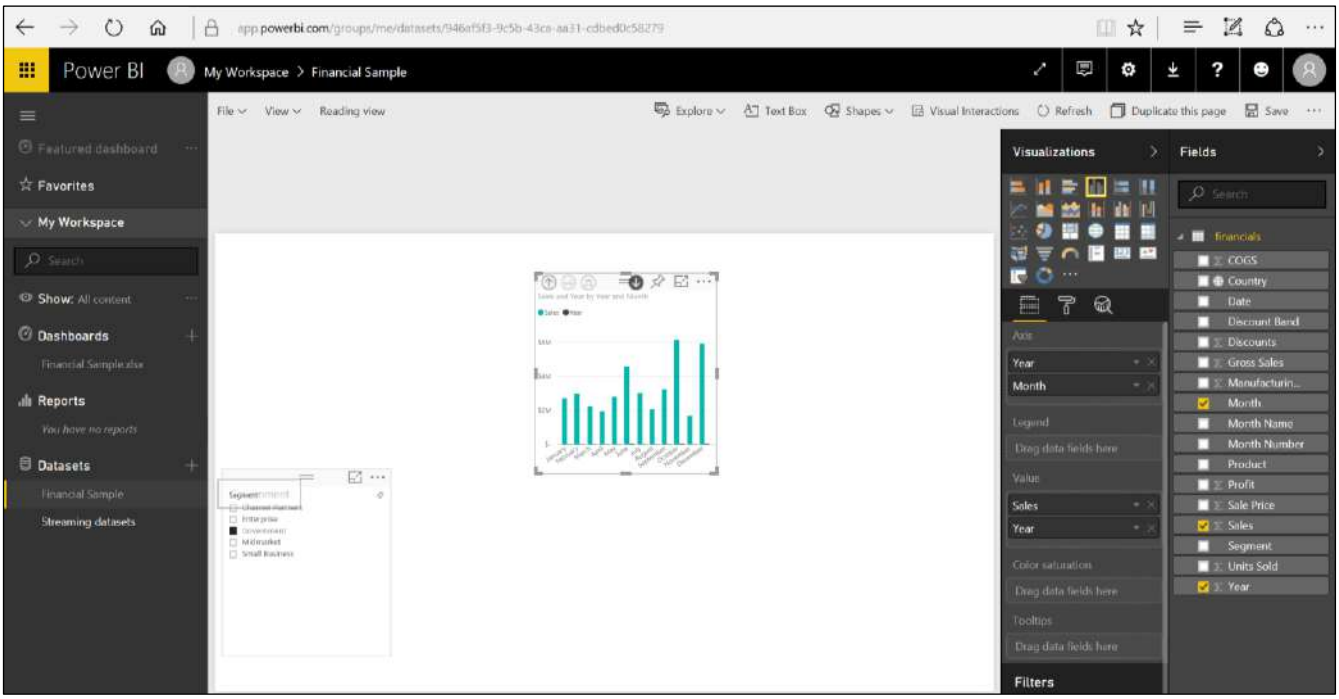
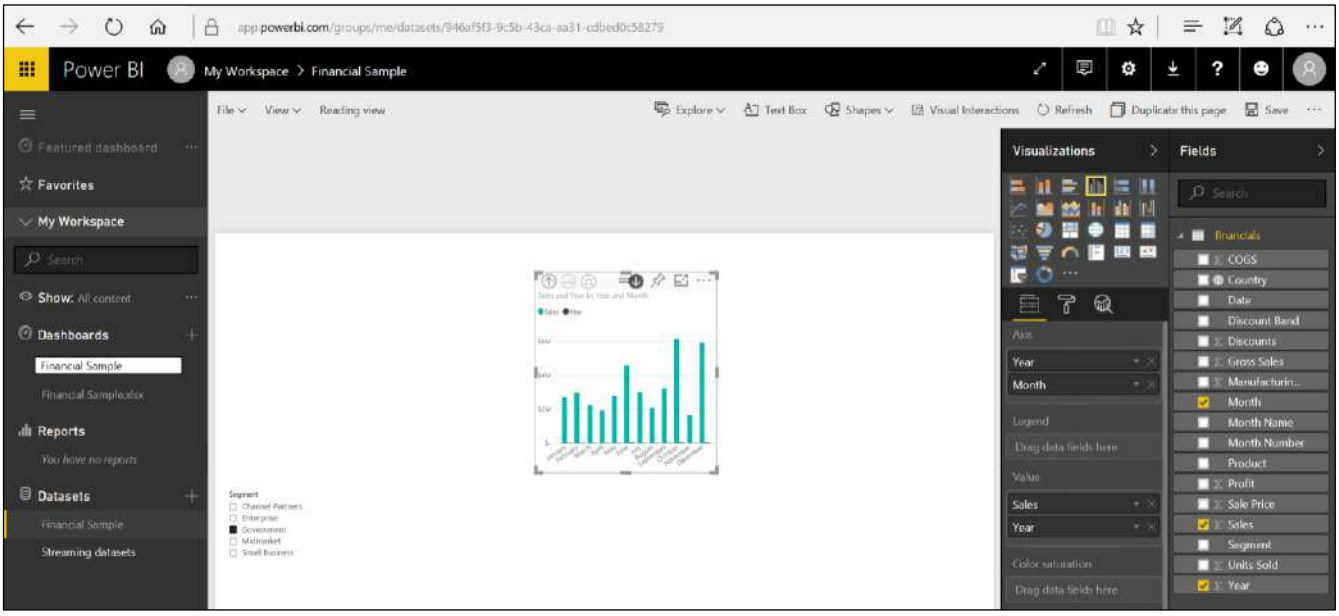
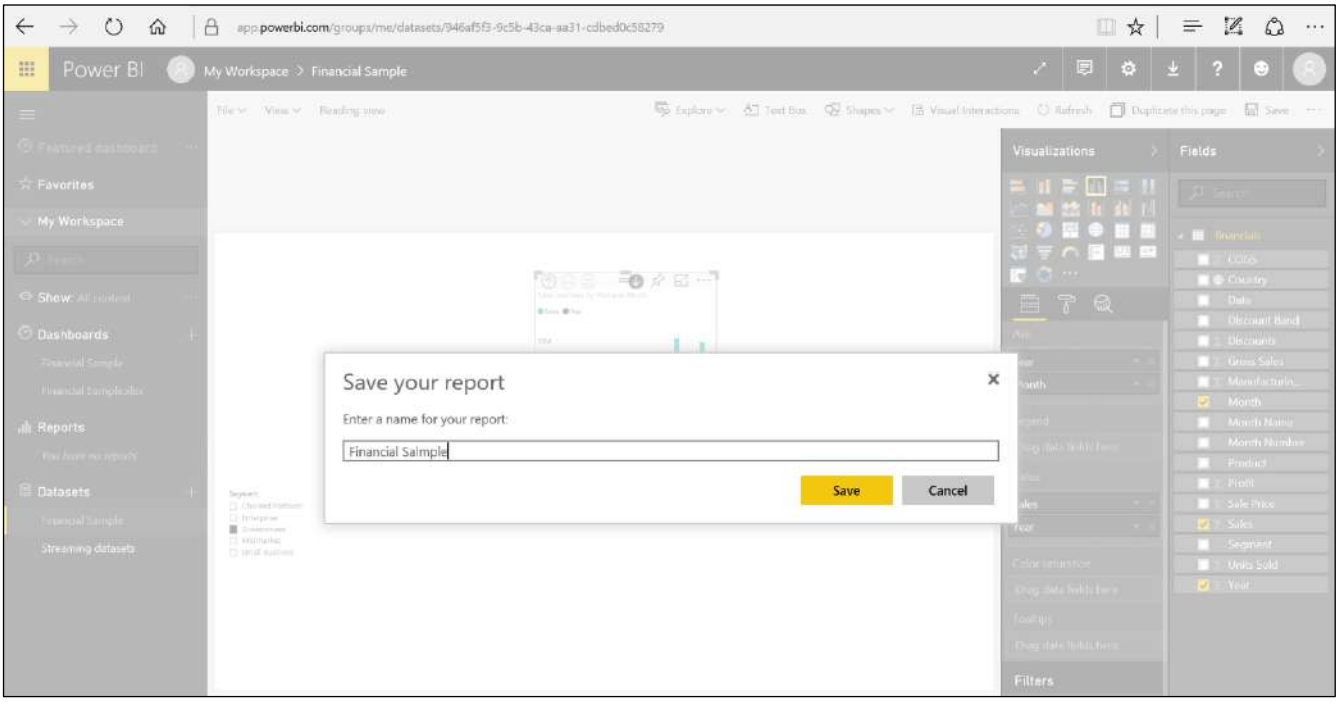


Figure 17: Power BI Displays Data—Use of Slicer

Now we create a new dashboard and save the report, as the three graphics in Figure 18 demonstrate.


app.powerbi.com/groups/me/datasets/946af5f3-9c5b-43ca-aa31-cdbed0c58279
Power BI
My Workspace > Financial Sample
File
View
Reading view
Explore
Text Box
Shapes
Visual Interactions
Refresh
Duplicate this page
Save
Featured dashboard
Favorites
My Workspace
Show: All content
Dashboards
Financial Sample
Financial Sample.xlsx
Reports
You have no reports
Datasets
Financial Sample
Streaming datasets
Visualizations
Fields
Axis
Year
Month
Legend
Drag data fields here
Value
Sales
Year
Color saturation
Drag data fields here
COGS
Country
Date
Discount Band
Discounts
Gross Sales
Manufacturin...
Month
Month Name
Month Number
Product
Profit
Sale Price
Sales
Segment
Units Sold
Year


app.powerbi.com/groups/me/datasets/946af5f3-9c5b-43ca-aa31-cdbed0c58279
Power BI
My Workspace > Financial Sample
Save your report
Enter a name for your report:
Financial Salmple
Save
Cancel
Visualizations
Fields
Filters

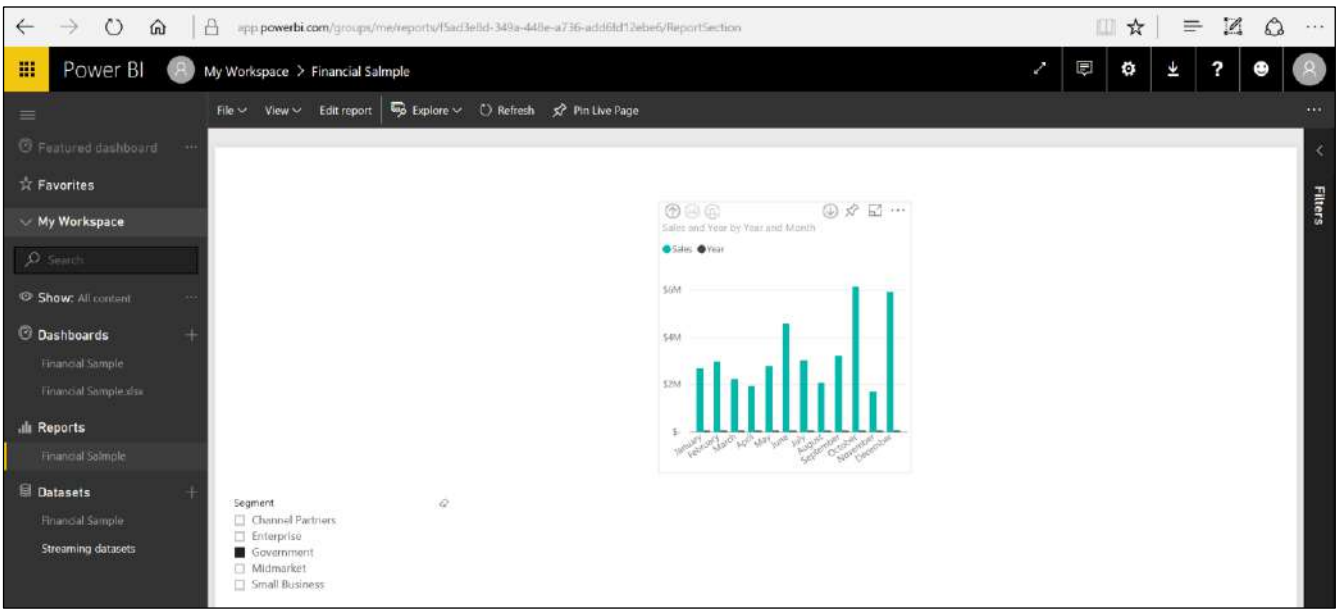
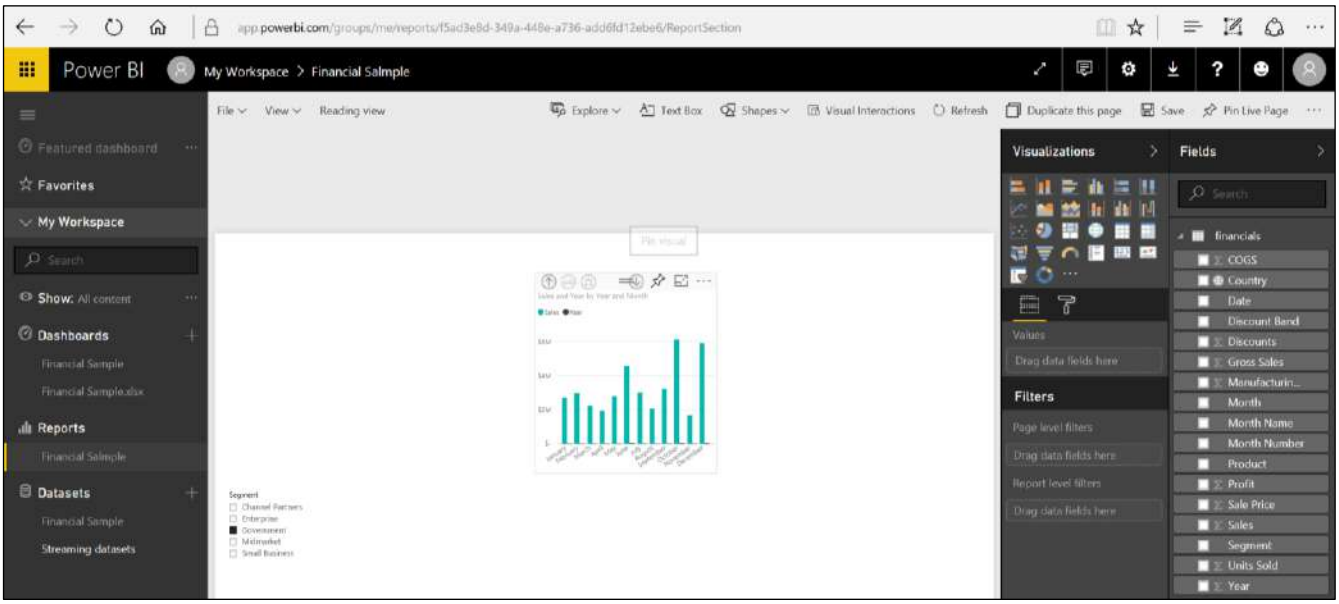


Figure 18: Power BI Display—Datase of Slicer—sequence

When we have saved the report, we can select the Reading View mode, which allows us to interact with the data by clicking the **Edit report** tab. By going back to the Design mode, we can change the information we want to modify by clicking the Pin icon.



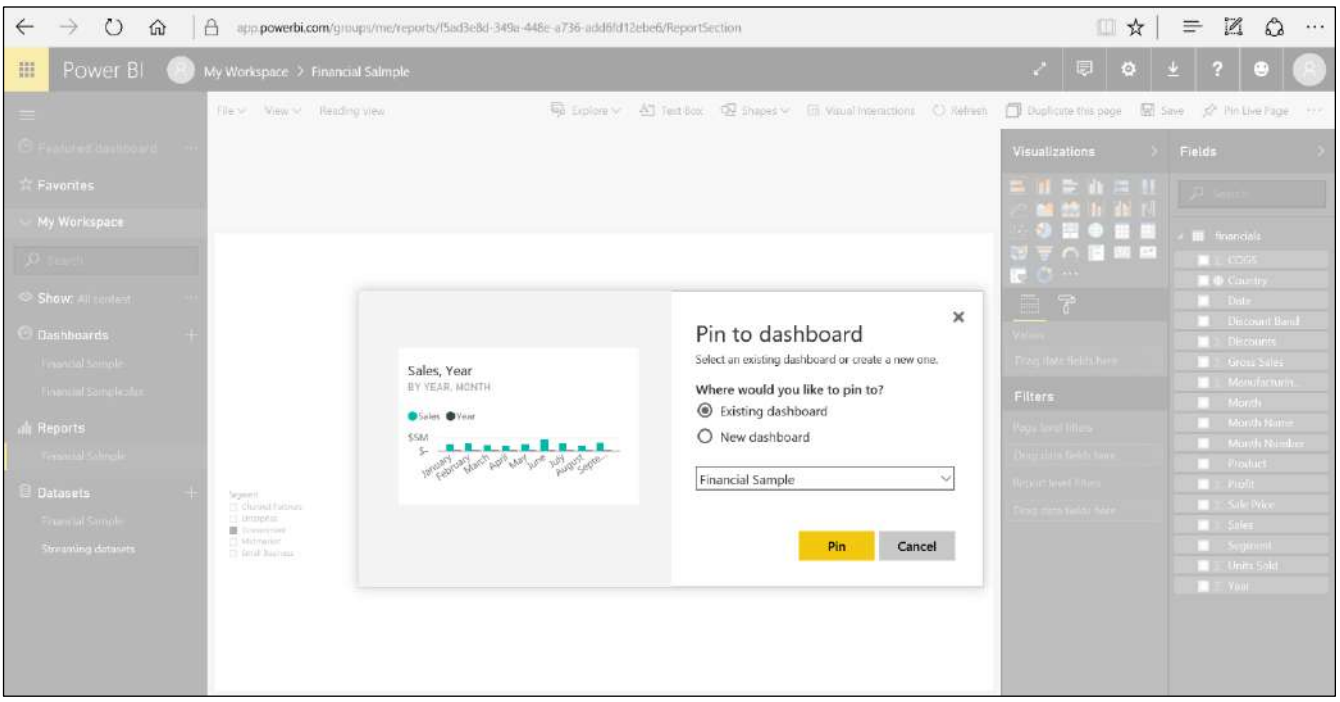


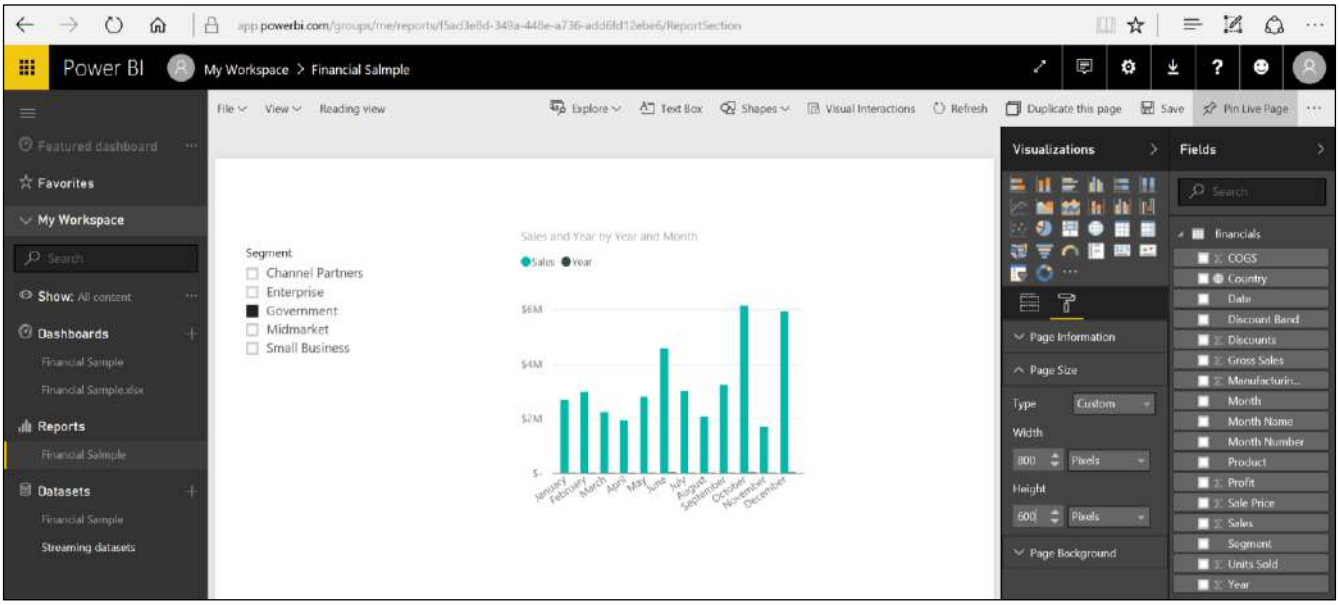
Figure 19: Pin to Dashboard—sequence

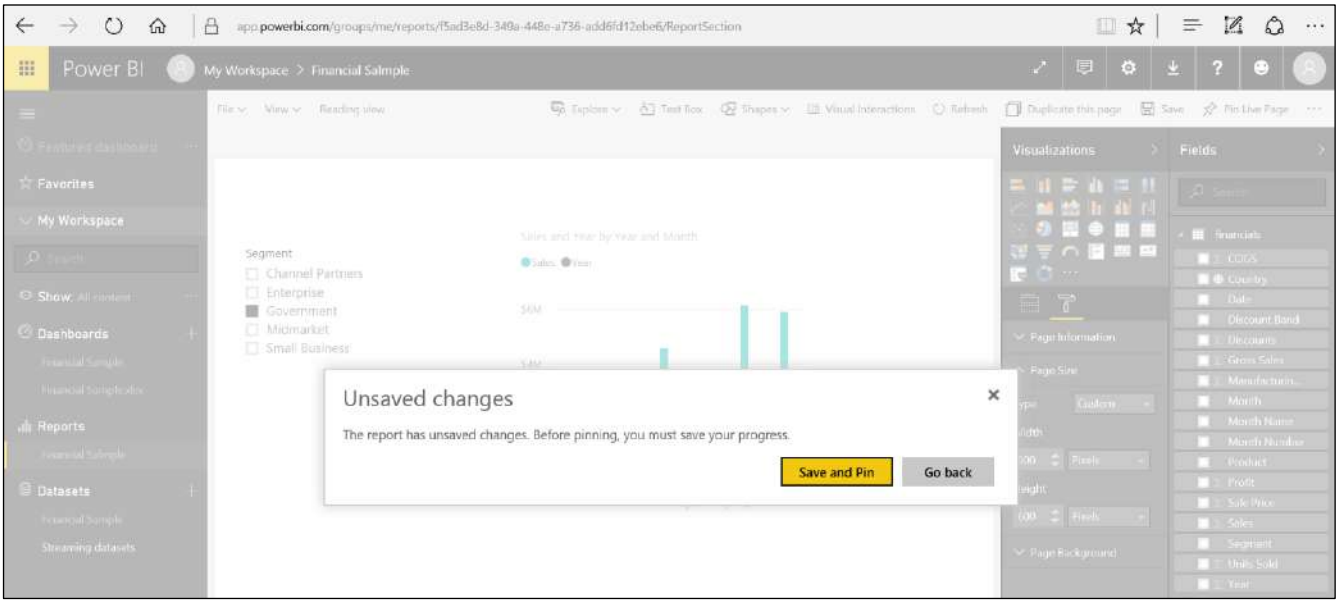
Next, we add the graph that has been pinned to the new dashboard.

Figure 20: Tile

When the operation has been completed, we will notice a tile, which is the graph we pinned into the dashboard. By clicking the tile, we open the original report in Reading View mode.

Another mode for pinning the report is accessed by going back to the same edit page, reducing it, then clicking **Pin Live Page** on the top right of the screen. The report will be pinned into the dashboard. This mode is important because it allows us to interact with the data through the slicer.


Power BI
My Workspace > Financial Salmple
File
View
Reading view
Explore
Text Box
Shapes
Visual Interactions
Refresh
Duplicate this page
Save
Pin Live Page
Featured dashboard
Favorites
My Workspace
Show: All content
Dashboards
Financial Sample
Financial Sample.xlsx
Reports
Financial Salmple
Datasets
Financial Sample
Streaming datasets
Segment
Channel Partners
Enterprise
Government
Midmarket
Small Business
Sales and Year by Year and Month
Sales
Year
$6M
$4M
$2M
$-
January
February
March
April
May
June
July
August
September
October
November
December
Visualizations
Fields
Page Information
Page Size
Type
Custom
Width
800
Pixels
Height
600
Pixels
Page Background
Financials
COGS
Country
Date
Discount Band
Discounts
Gross Sales
Manufacturin...
Month
Month Name
Month Number
Product
Profit
Sale Price
Sales
Segment
Units Sold
Year


Power BI
My Workspace > Financial Salmple
Unsaved changes
The report has unsaved changes. Before pinning, you must save your progress.
Save and Pin
Go back

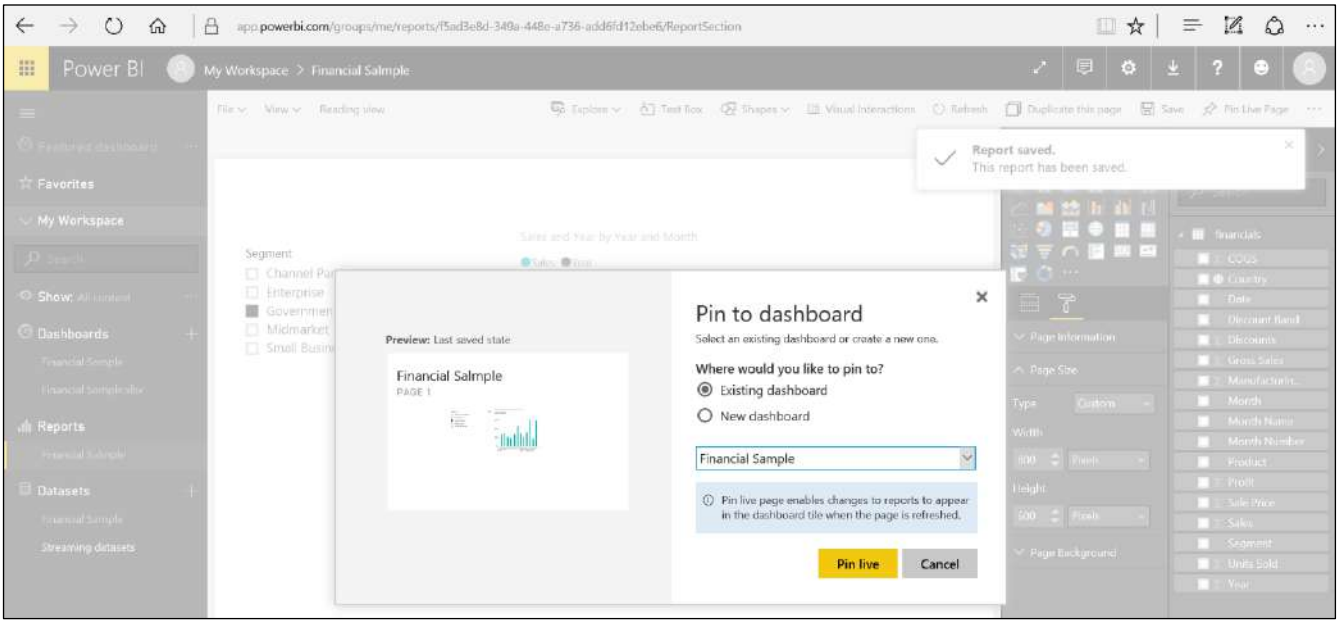


Figure 21: Pin Live Page—sequence

If we want to delete a tile from a dashboard, we click the ellipsis icon (…) that indicates whether a context menu is available. In the following visualization, we click on the Delete entry.

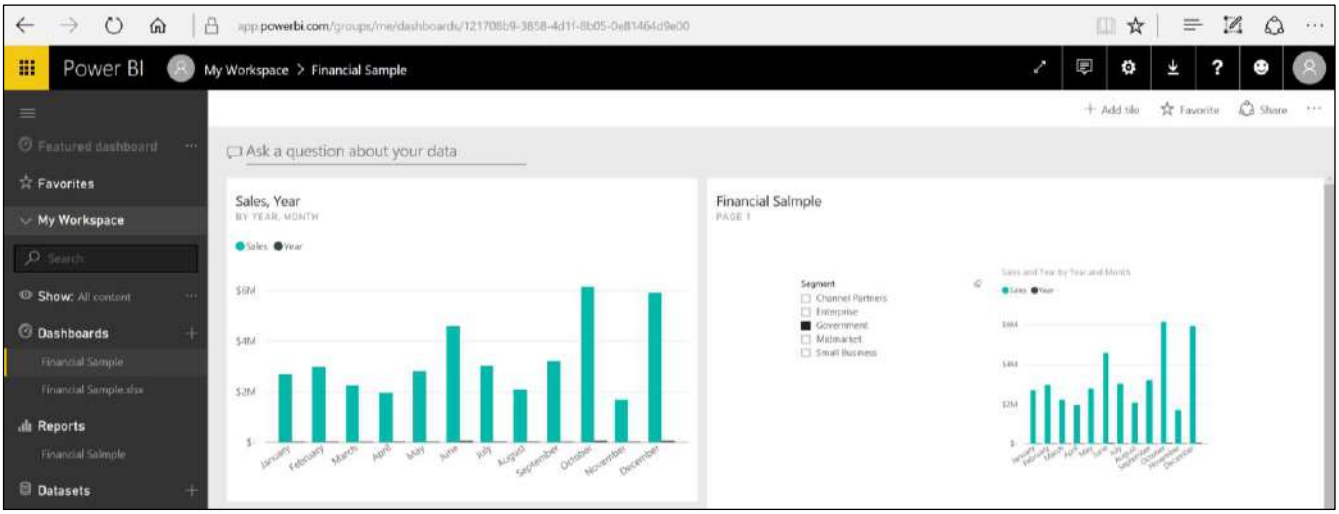


Figure 22: Dashboard

In the Context menu, we can also carry out other operations, such as Focus mode, which opens the graph to full screen, and Insights, which also opens the graph in full screen and offers significant values such as Value Sum and Summary charts.

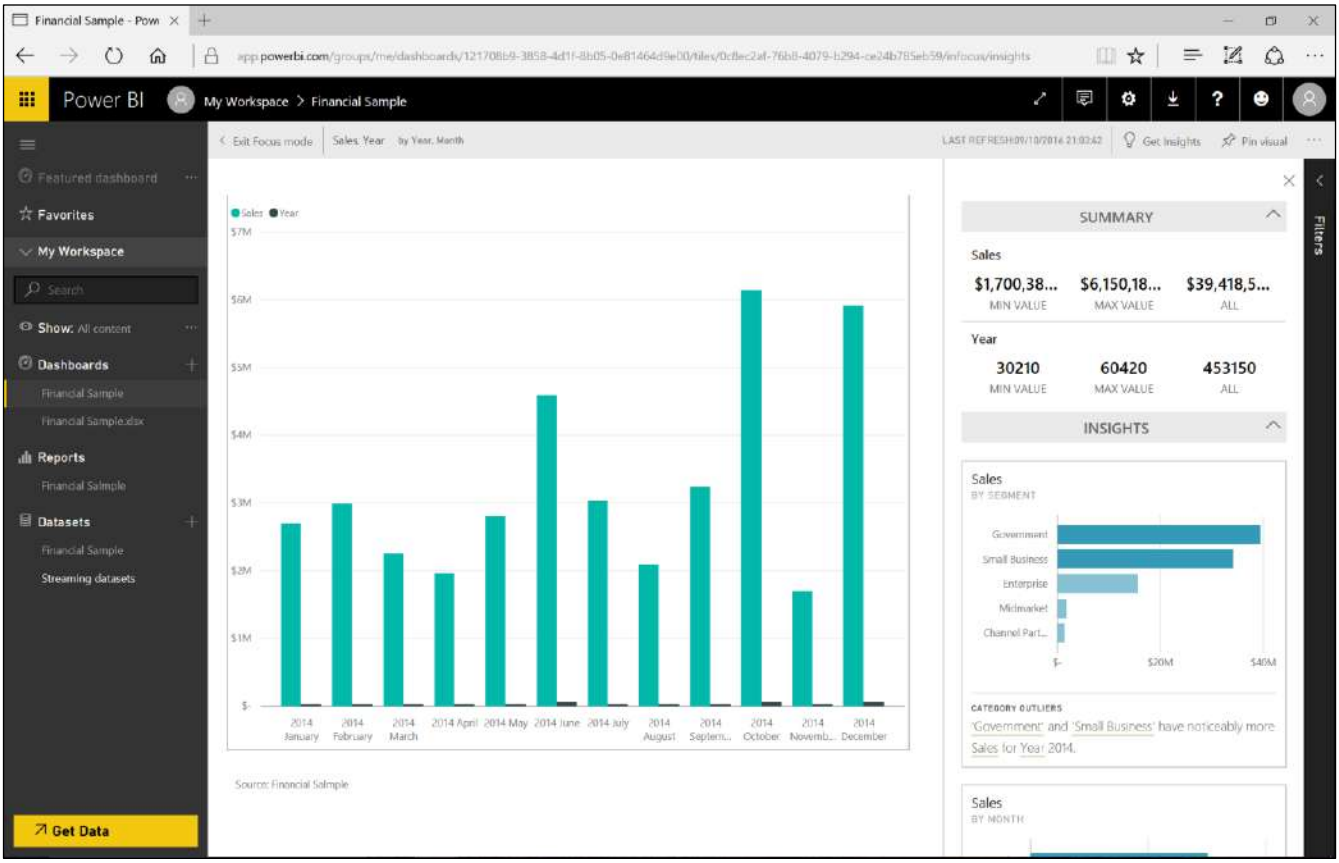


Figure 23: Insights

We can also use the “Pin visual” icon to pin the same tile into another dashboard. “Export data” allows us to export the data present in the tile (in CSV format).

With “tile details,” shown in Figure 24, another window is displayed in which we can edit the information of the tile.

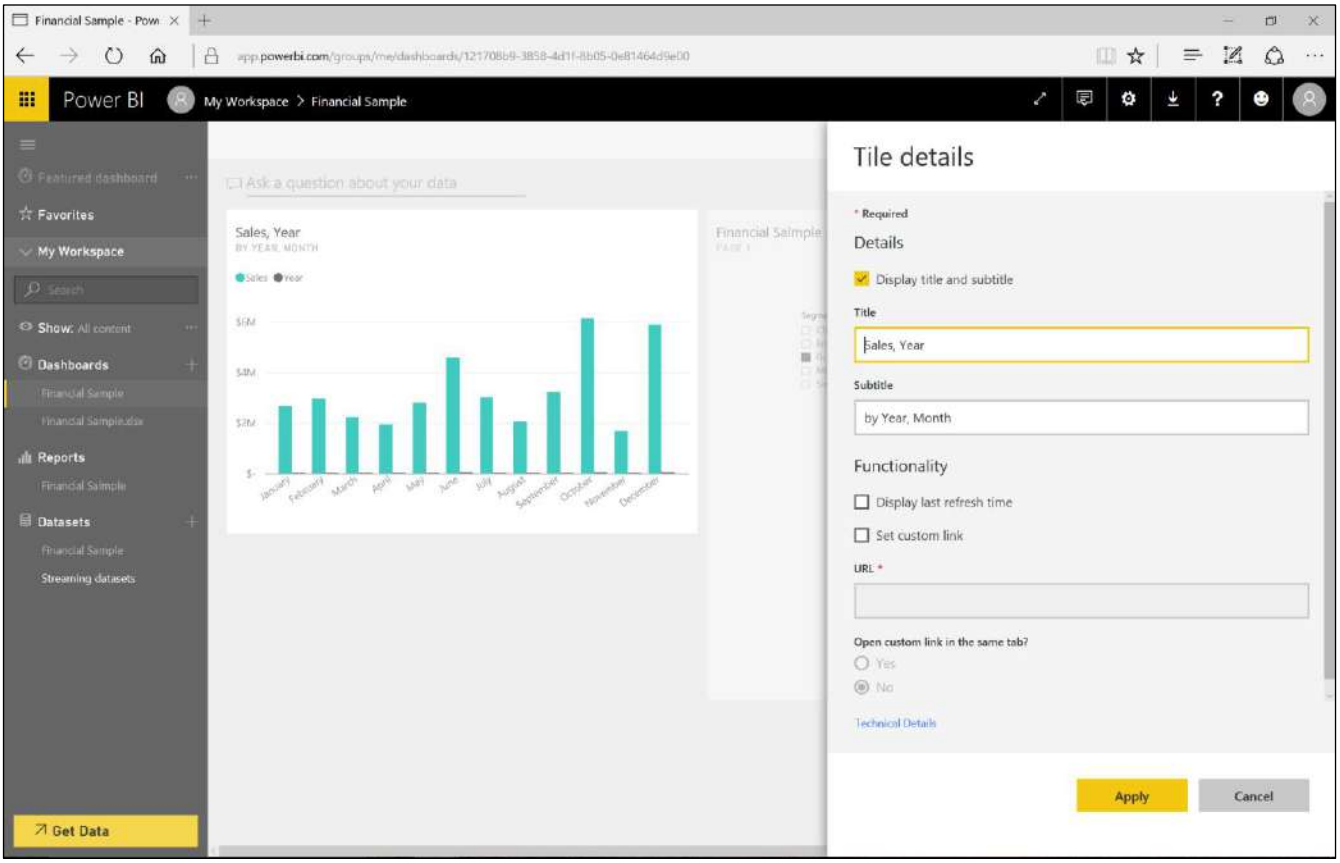


Figure 24: Tile Details

As we can see in the Edit page of the report in Figure 25, if we click a new page at the bottom left, we can add new pages to the report. If we explore the various fields, we'll notice that the Country field of the dataset is represented by a little globe. Power BI recognizes the data type and, if we drag it to the report, it will be identified as geographical data and so a map will be displayed. When we drag and drop the Sales field to the geographical map, the sales distribution across the map is displayed.

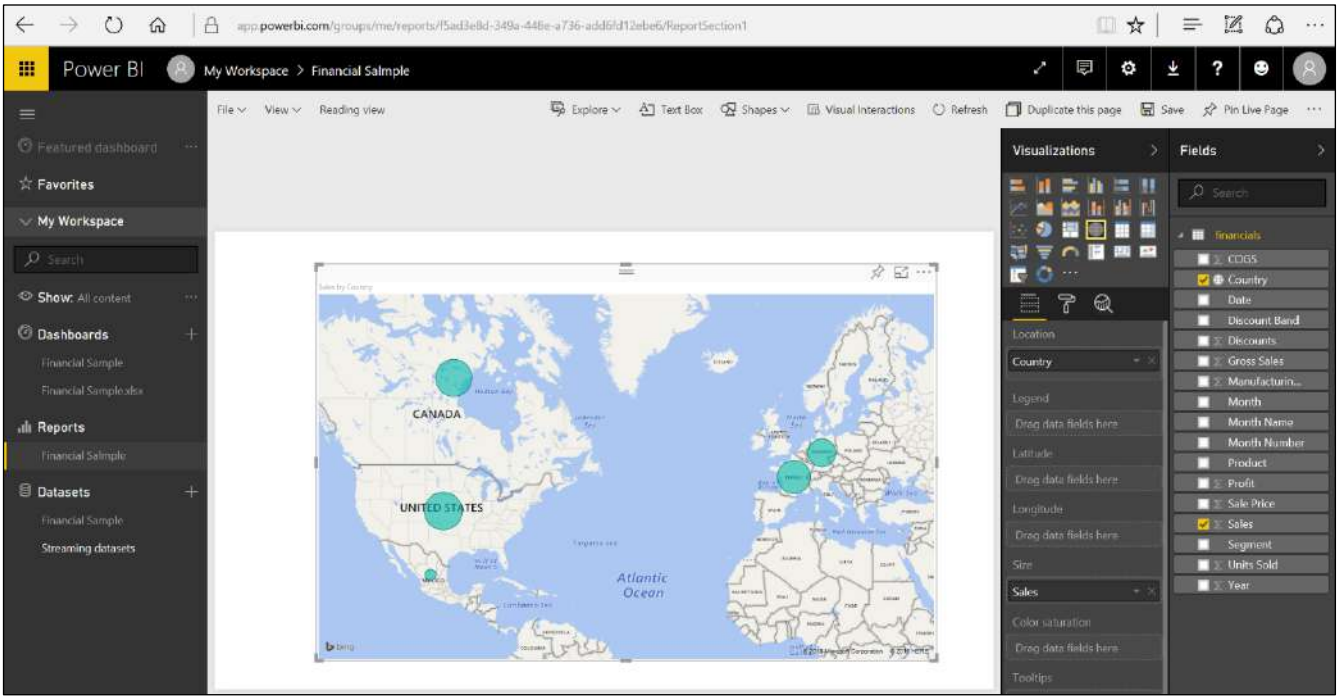


Figure 25: Power BI Visualizations

Figure 26 shows that we can add a new chart on the same page by inserting the Sales field, and in the same graph we can add the Month field in order to have a temporal base. By clicking the graph type, we can change the visualization to one that best meets our expectations.

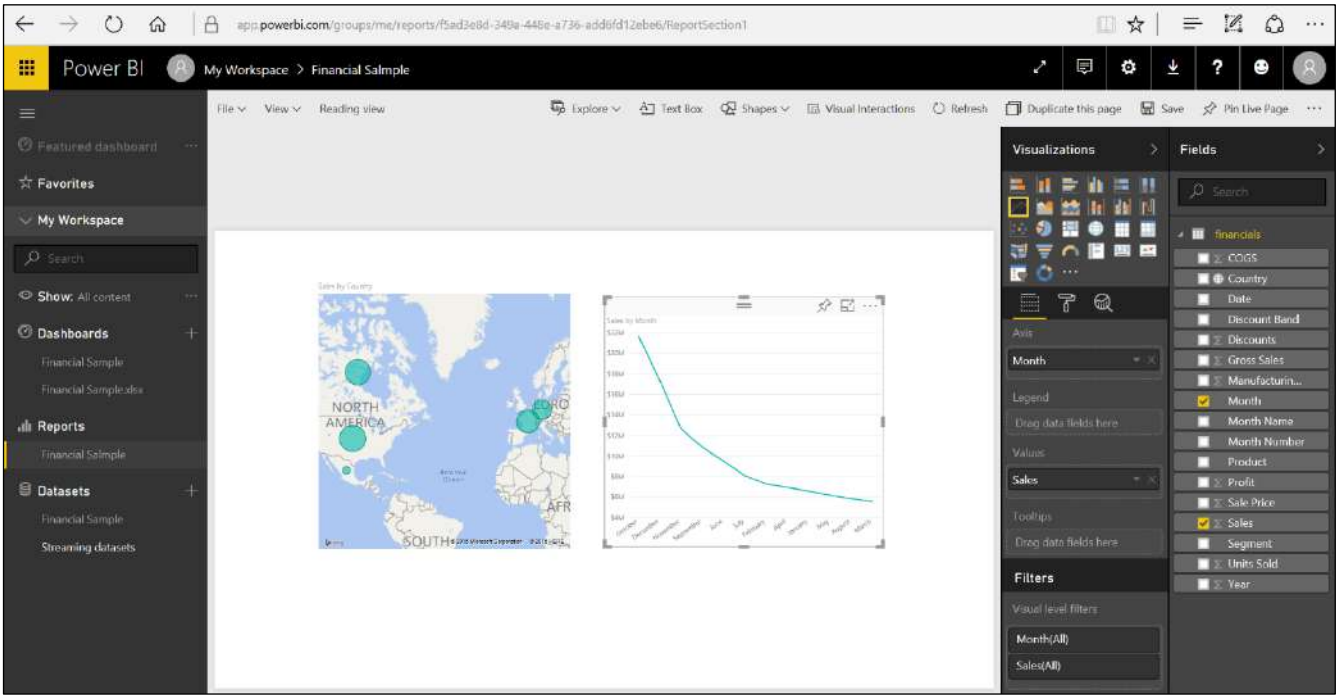


Figure 26: Power BI Visualizations

Visual gallery

If the visualizations are not sufficient, we can obtain new ones from the visual gallery at https://store.office.com/en-us/appshome.aspx?productgroup=PowerBI, where we can see numerous visualizations published by people who have implemented and shared them.

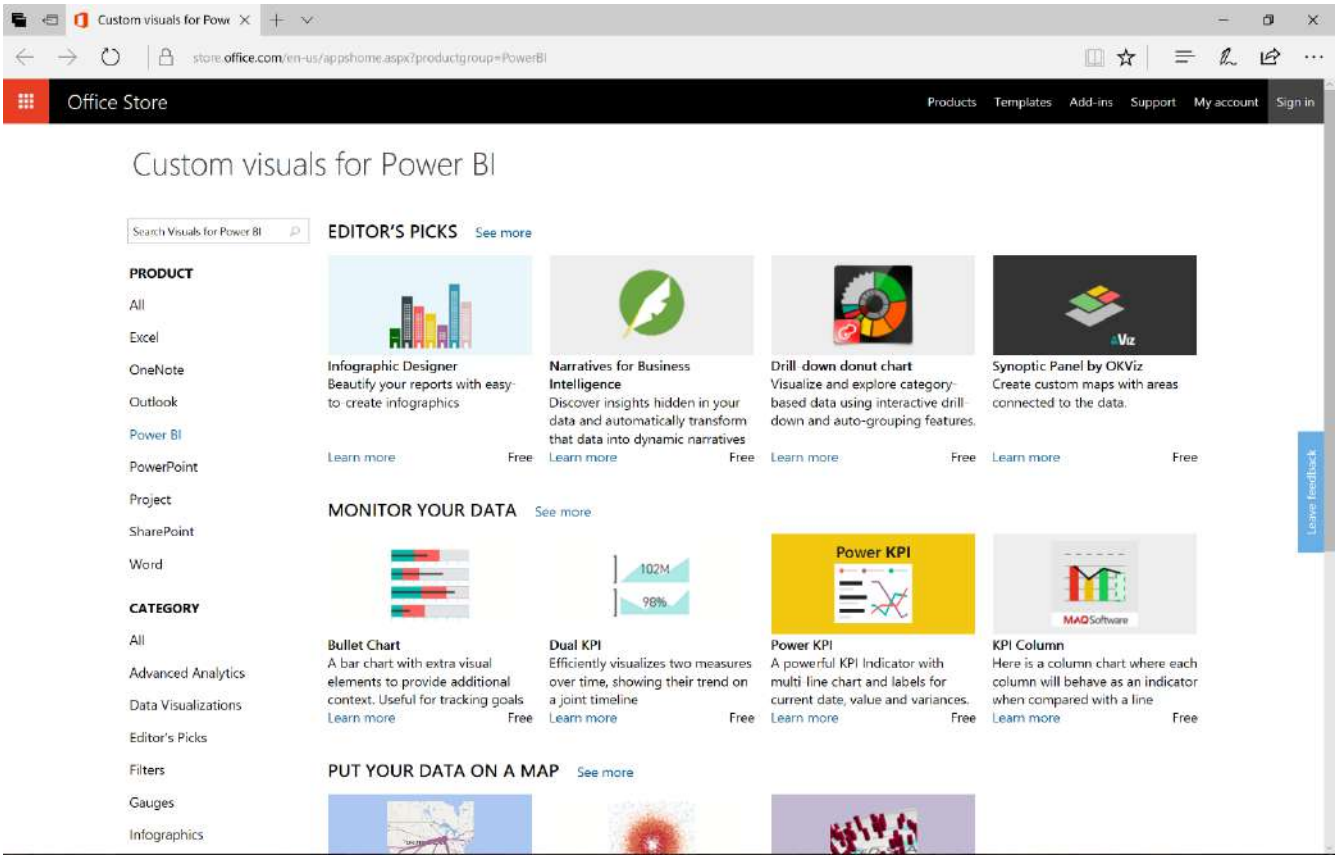


Figure 27: Power BI Visual Gallery

As Figure 28 shows, we can also download the Timeline. Note that the checks can be constantly updated, so it is advisable to always download them in order to be certain you have the updated version. To do this, click **Add**, then click **Select** to download Timeline Slicer.

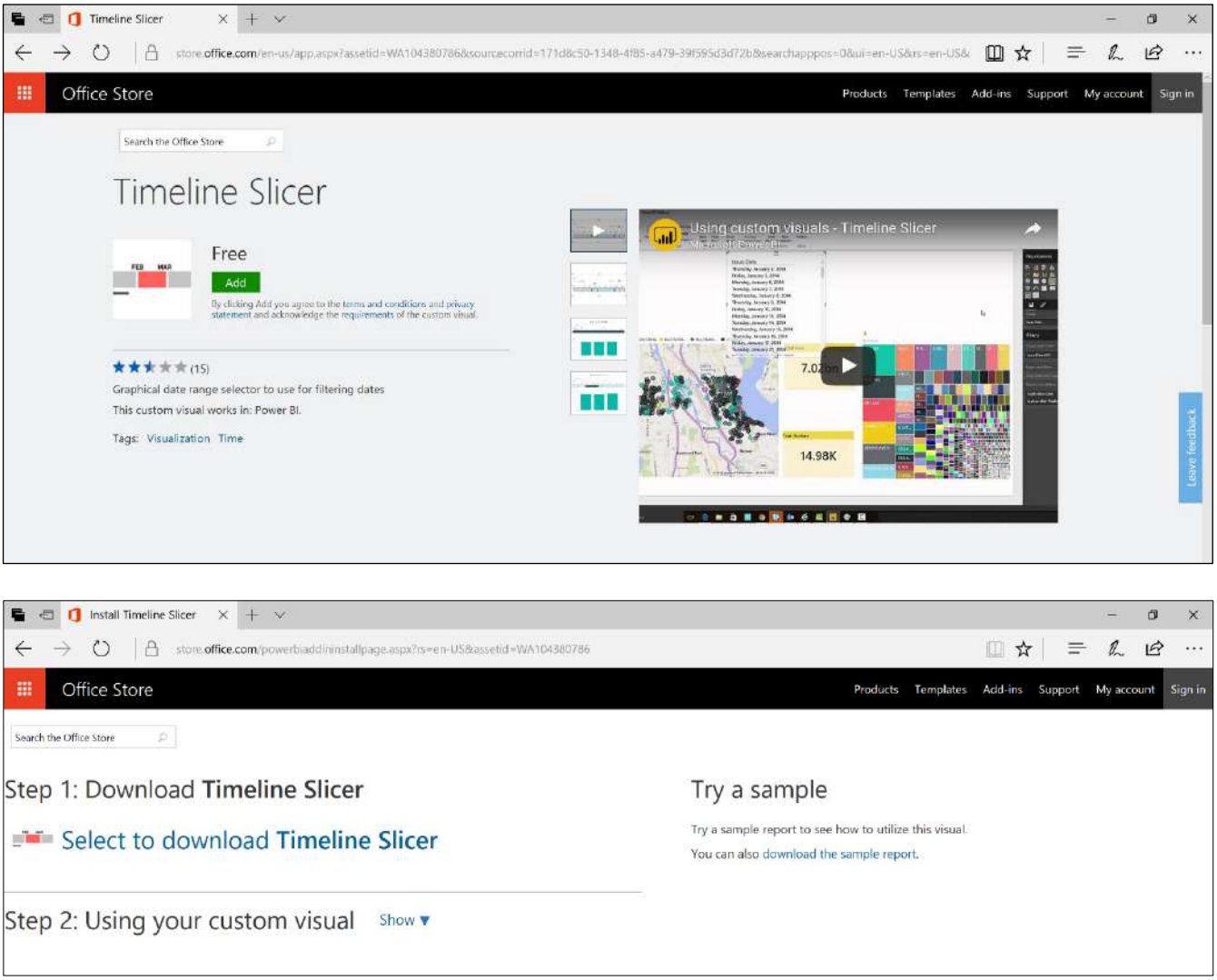


Figure 28: Power BI Timeline Slice—sequence

We import this new control by clicking the "..." icon in the visualizations section.

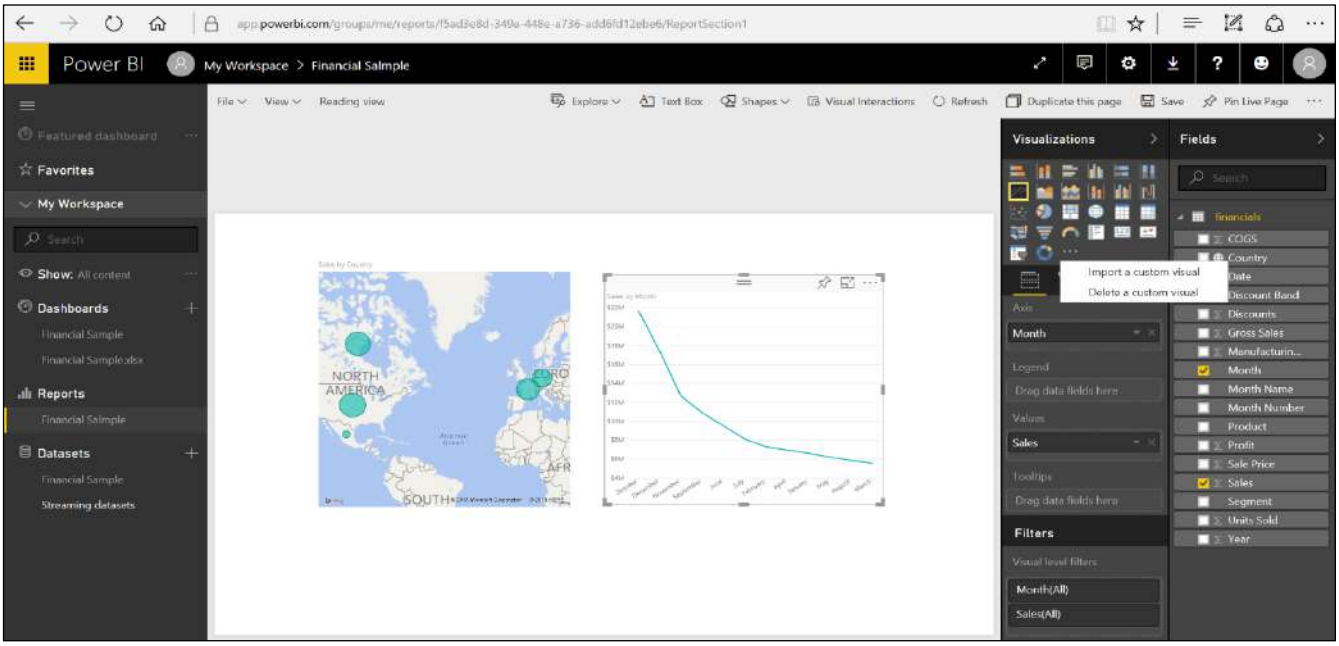


Figure 29: Power BI Visualizations

The Timeline visualization will be available on the display commands. By adding the Timeline to the report and later dragging the Data field there, we can implement a control that allows us to select the data based on temporal intervals. It is simple and intuitive. The other visualizations present in the report will display the data based on the selection carried out through the Timeline.

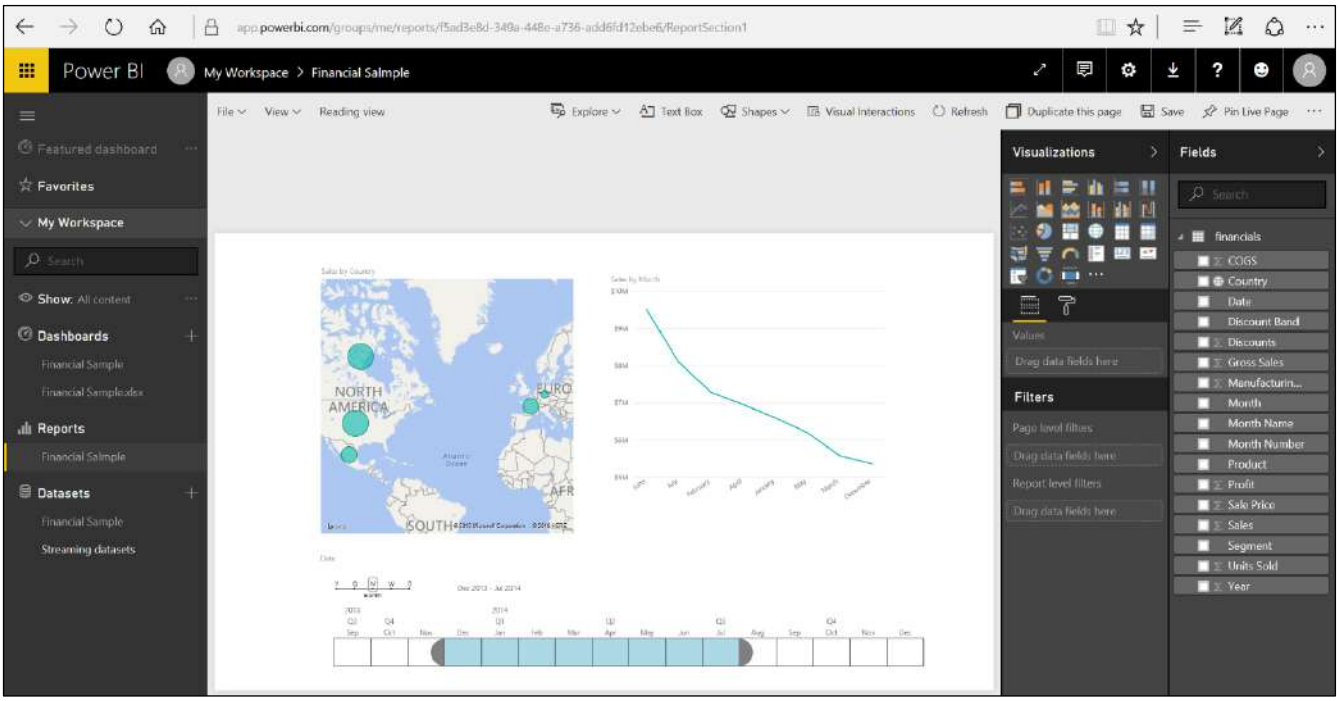


Figure 30: Power BI Visualizations—use of custom visuals

Note that because the custom visuals have been developed by third parties, they include a disclaimer asking us for usage confirmation.

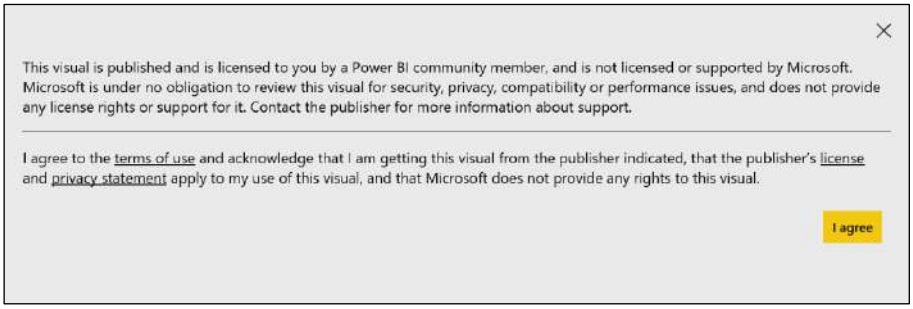


Figure 31: Custom Visual Terms of Use

Figure 32 demonstrates the download of a custom visual from the visual gallery. We can add a report on a new page by dragging the Country field onto it, and then, for example, the Sales field. Doing so will create a stage visualization.

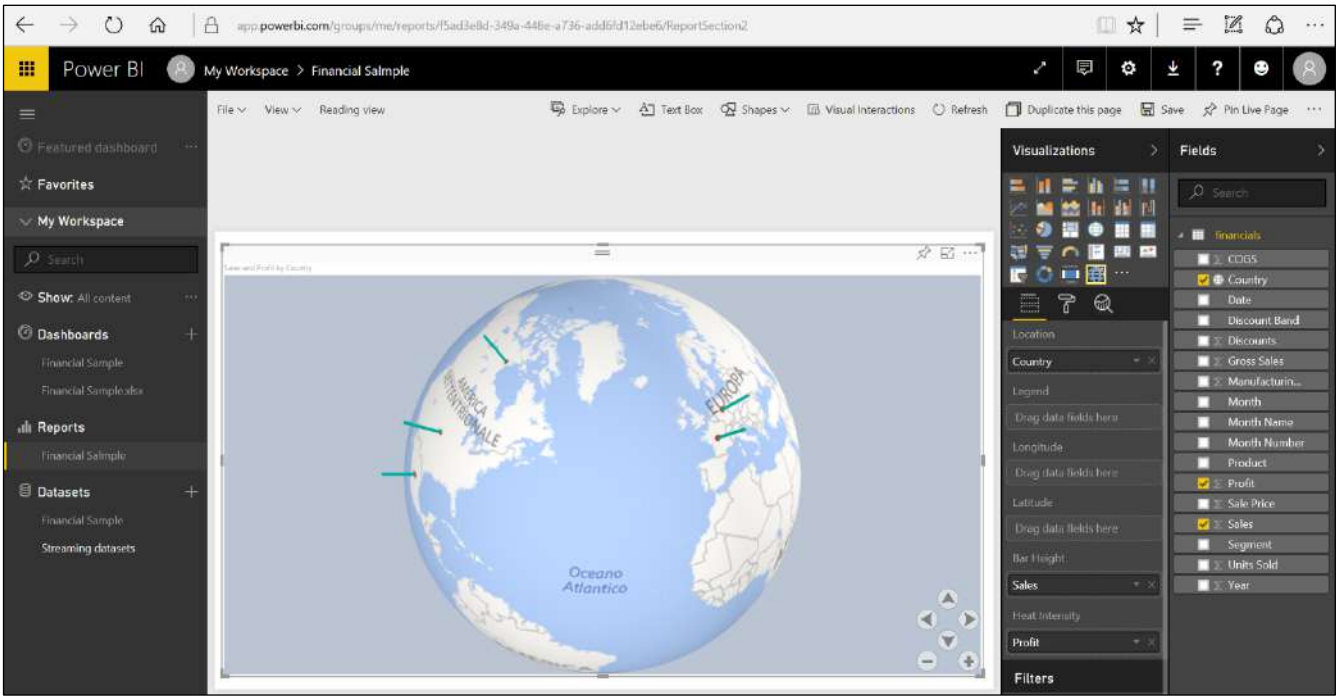


Figure 32: Use of Custom Visual

The report pages are independent. We can set certain filters on a page or report basis. On a page basis, the data is filtered on the page only, and therefore the visualizations will show only data that makes it through the filter. On a report basis, the visualizations present in all the report pages will show the data that makes it through that filter. In Figure 33, the Segment field has been selected and linked to a report-level filter. All the pages will interact based on the filtered data.

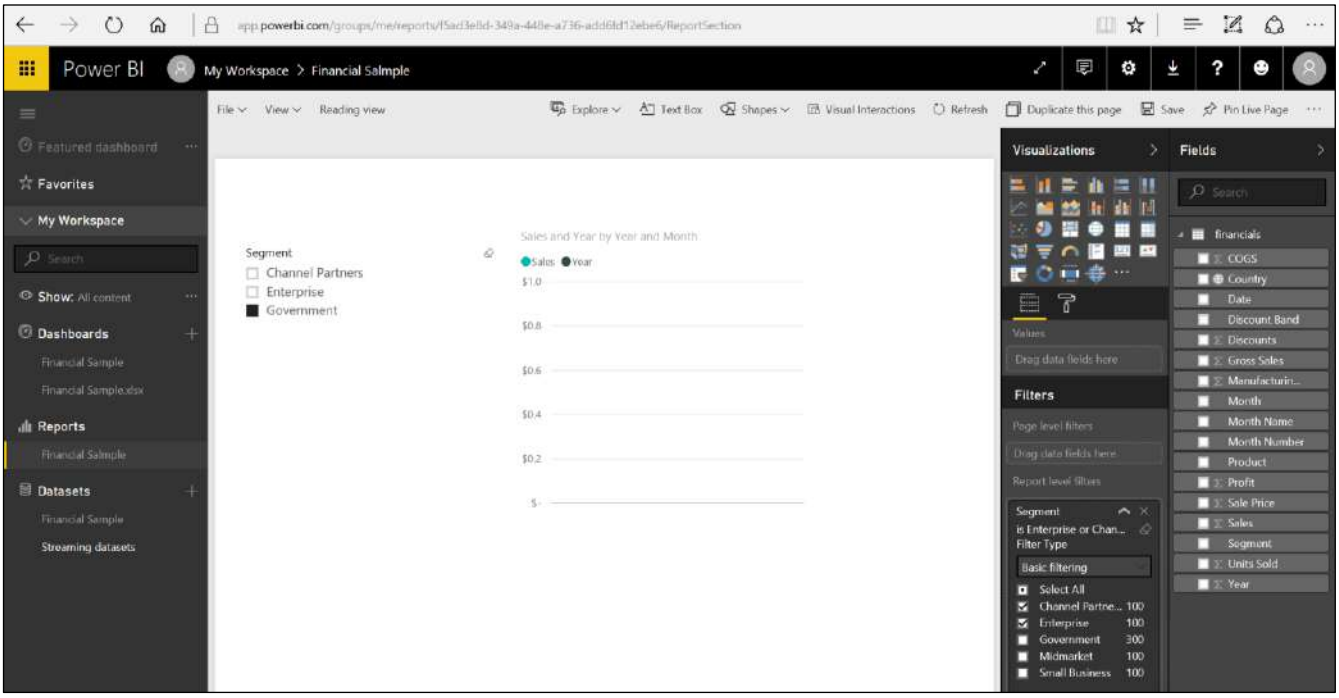
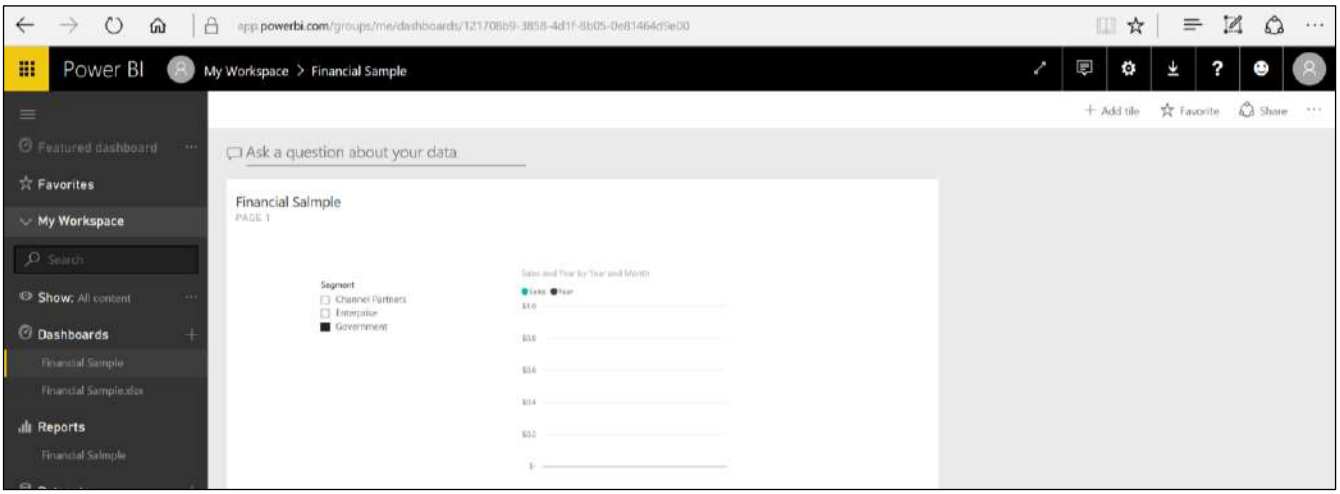


Figure 33: Use of Filters

Natural language capability

We can pose questions to Power BI based on the data model we are analyzing.

For example, in asking, “Which segment has the most revenue?” the term “the most” tells Power BI that we are asking for a system.



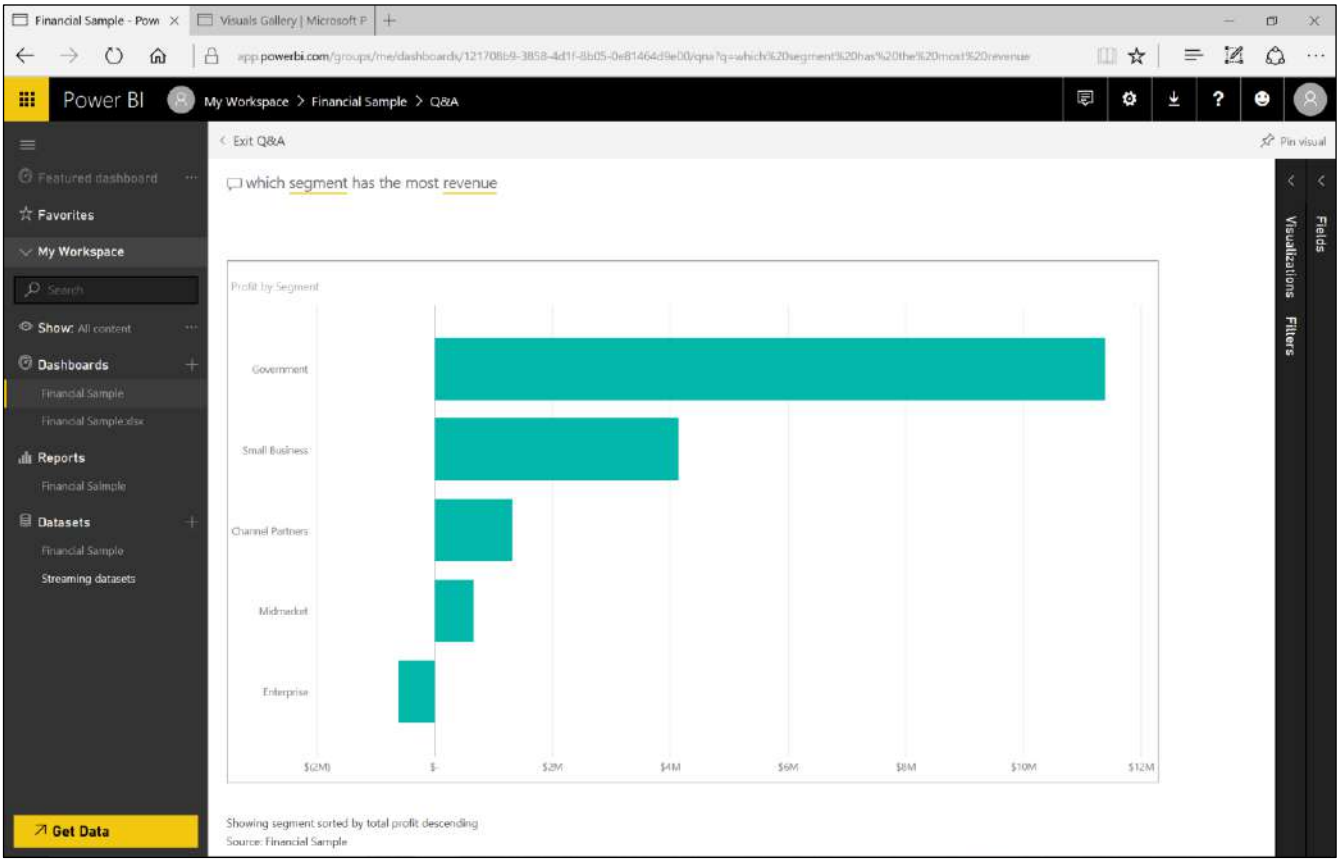


Figure 34: Use of Q&A—sequence

"What segment got the greatest number of profits?" will receive an answer through a visualization. If we are not satisfied with the graph form, we can change it by choosing the visualization that most satisfies us. We can carry out the pinning directly in the dashboard. And by clicking the tile that has just been inserted, we can go back to the original question. Notice that Revenue is not present in our data model, but Power BI has realized that that is related to the Profit field and shows us the profits.

Sharing and cooperation

Sharing and cooperation can be achieved through three different and complementary methods:

- Sharing dashboards and tiles directly with the users of our organization.
- Using Groups of Office 365: reporting and cooperating directly in a workspace dedicated to the group.
- Publishing an organizational content pack, which is shown in an internal gallery (all the users of the organization or of a group inside the organization are able to surf the store, the content packs are displayed, and it is possible to install the relevant packs with the same usage experience).

Each method requires that all users have a Power BI Pro license. And except for the dataset on-premises Analysis Services case, all the users will see the same shared data.

Table 1: Share Options Comparison

	Share	Office 365 Group	Org Content Pack
Purpose	Ad hoc	Cooperation	Large content delivery
Target audience	Individuals or members of an Office 365 distribution group. Typically suitable for users who strictly work together and need to display specific dashboards or tiles.	Members of an Office 365 group. Generally suitable for teams and collaborators engaged in a project or in a common scope.	The entire organization, members of a security group and/or Office 365 group, or singles. Suitable for all those who might be interested in a particular content.
It is applied to:	Specific dashboards or tiles	All the contents	All the contents
Discovery	Invitation email	Office 365 group	Get Data → Content Gallery
Permits	Read-only invisible datasets. The reports are in Reading View. Resharing permits can be granted.	Edit. All the members of the group have the same rights in the management of the group contents.	Mainly read-only. Members can unlock in order to personalize their copy. Personalized content packs that must be updated will serve a notice.

Some functions are more suitable to specific contexts than others. For example: the sharing of a dashboard is very useful when we must share it with a collaborator in a very short time. In this case, we must insert the account of the collaborator—we will receive a notice and will be able to display the dashboard—but we will not be able to modify the work. In the context of a collaboration group, we can utilize the collaboration method through a workspace, and all the users can modify reports and dashboards, cooperating as a team for the writing of a report. In short, the content pack is well suited for scenarios in which multiple people are using the data because we publish the package (which includes dashboard, report, and dataset) in a package and we publish it to a gallery that is visible to all the users of the organization (who are therefore free to install and modify it).

Cooperate

In order to cooperate, we must go into the dashboard and click **Share**.

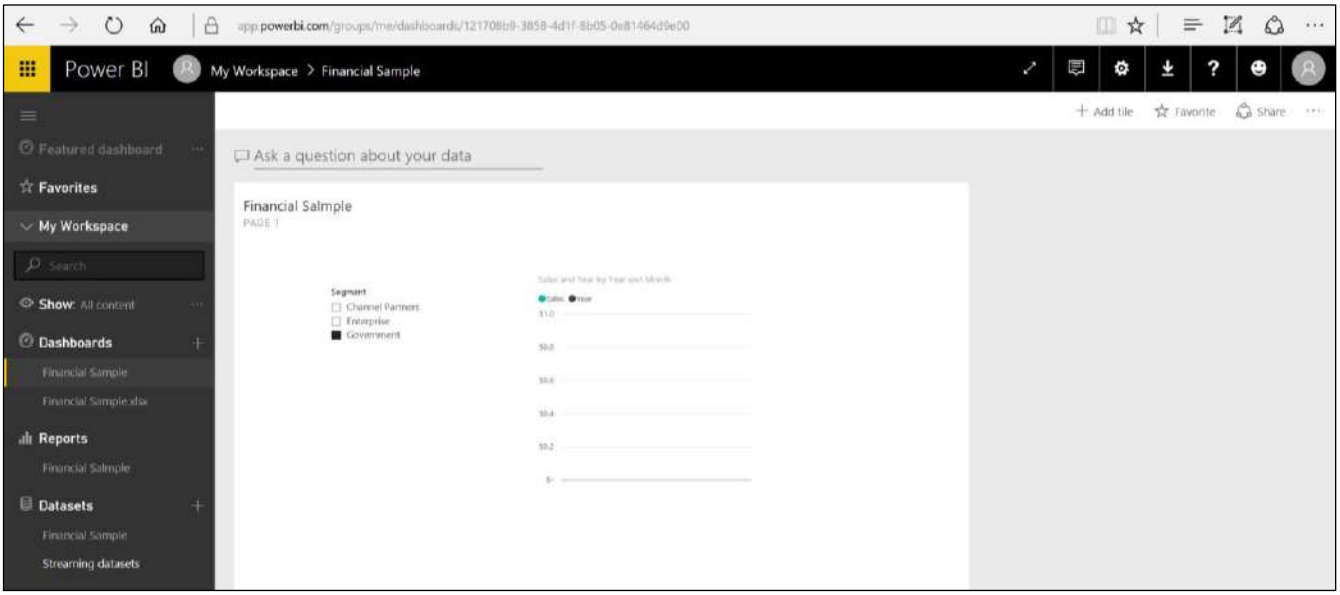


Figure 35: Dashboard

Next, we indicate the users we want to share the dashboard with and click **Share**.

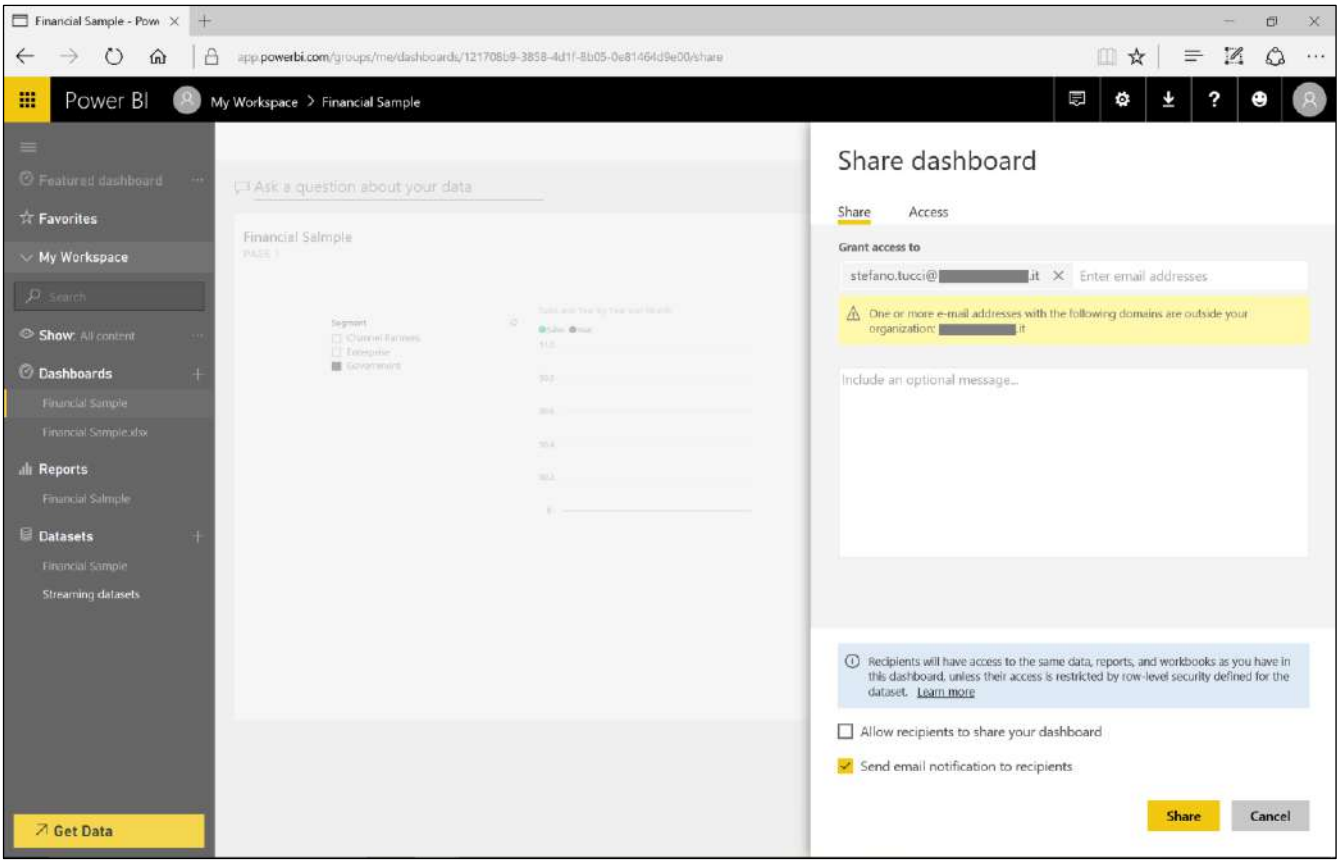


Figure 36: Share Dashboard

On the same sharing page, we will find the option to specify whether or not the user can share the dashboard with other users. We can also choose to send notifications via email (or not).

In the cooperation example (where Option 1 is no and Option 2 is yes), the user who receives the shared dashboard will not be able to modify it—only display it.

We can cancel the sharing with particular users by selecting **Share** in the Access section and then clicking **Cancel invite**.

Regarding cooperation through the Office 365 groups, on the upper left of the screen we will find the entry “My workspace,” indicating it is our personal workspace.

We can browse to check whether there are several group workspaces. If there are none, we can create a new one by specifying the name of the group, whether the group is private or public, and if all the members of the group are able to modify the contents within the workspace or only display them. Lastly, we include the users who belong to the workspace.

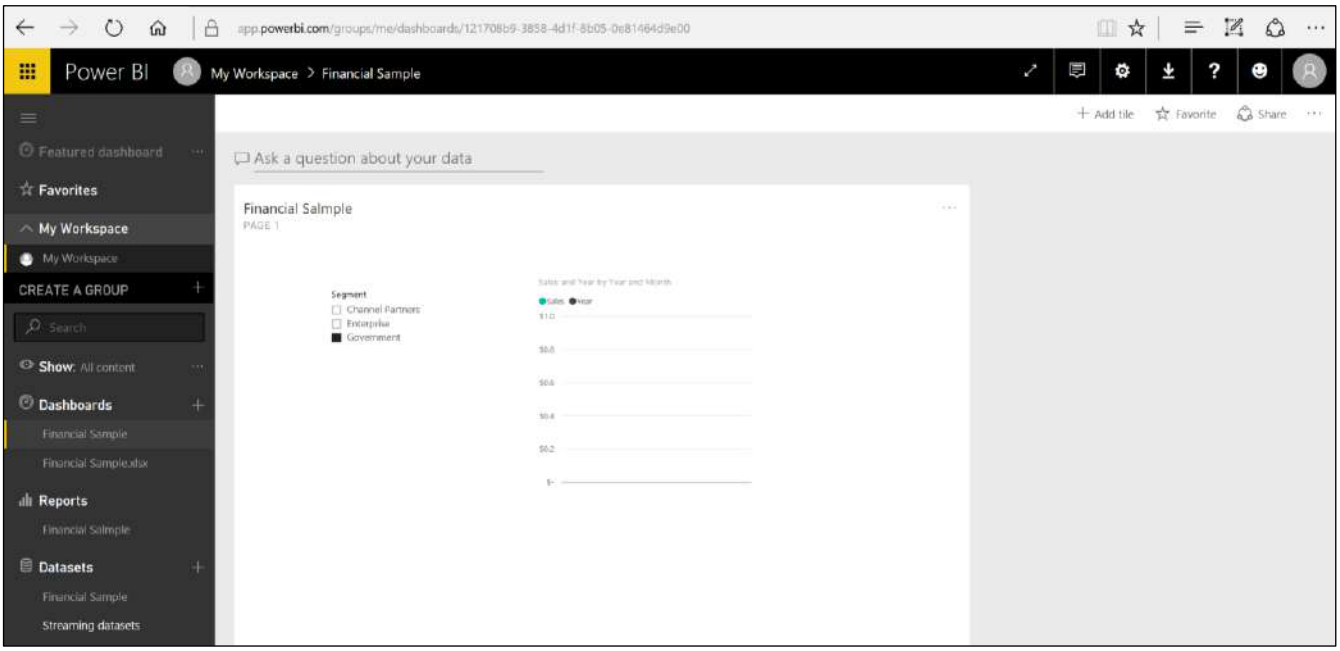


Figure 37: Dashboard

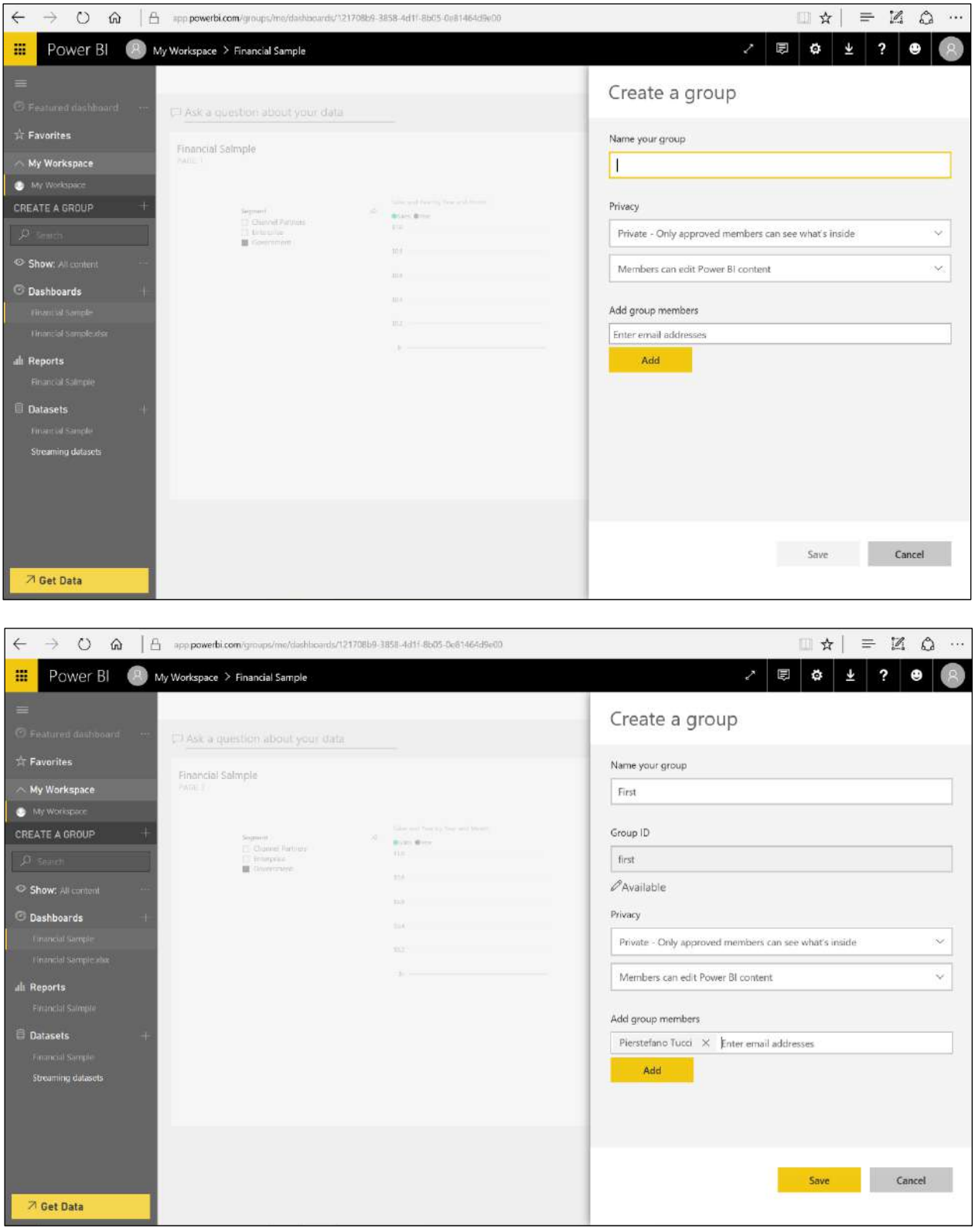


Figure 38: Share Dashboard to Cooperate—sequence

We can go back to sharing by selecting a group workspace. We get a new triple set of dashboard, report, and datasets that are independent from the personal ones. In this new workspace, we are able to cooperate with others who are part of the Office 365 group.

We can also share the report we write over the web.

Publishing a report on the web allows us to make the report visible to anyone, assuming we allow default anonymous access to the report.

From the report, we click **File**, then **Publish to web**.

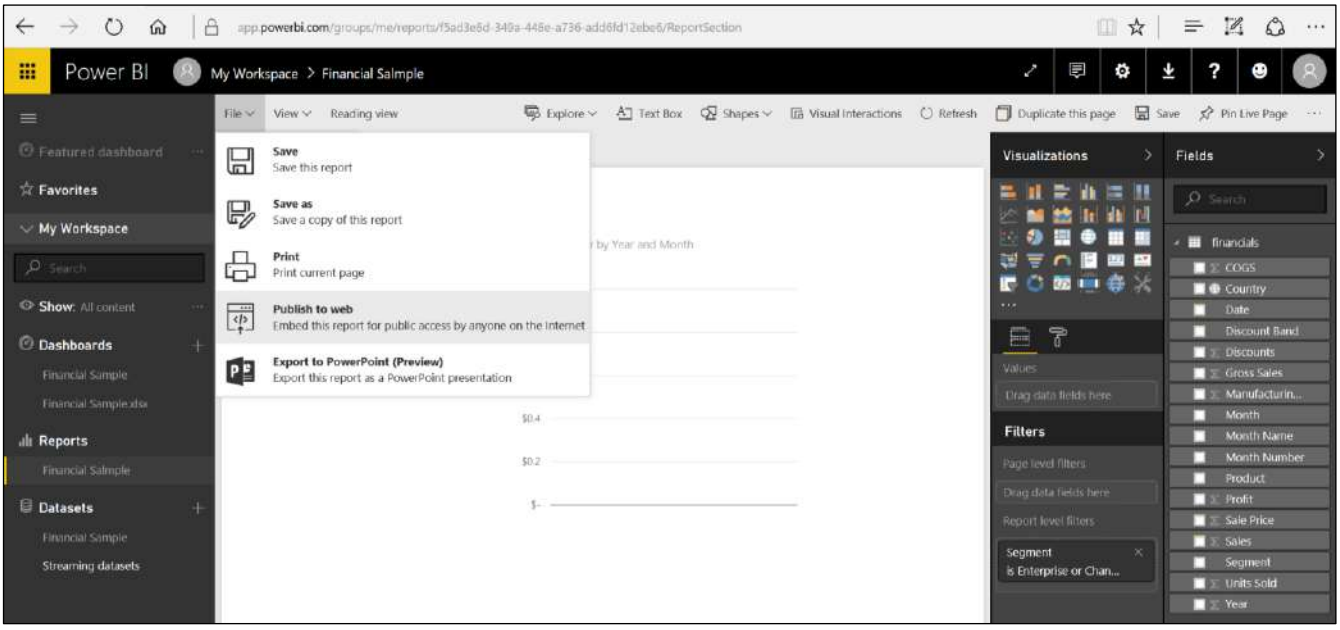


Figure 39: Publish to Web

Figure 40 shows the sequence of our next actions—selecting **Create embedded code**, then **Publish**.

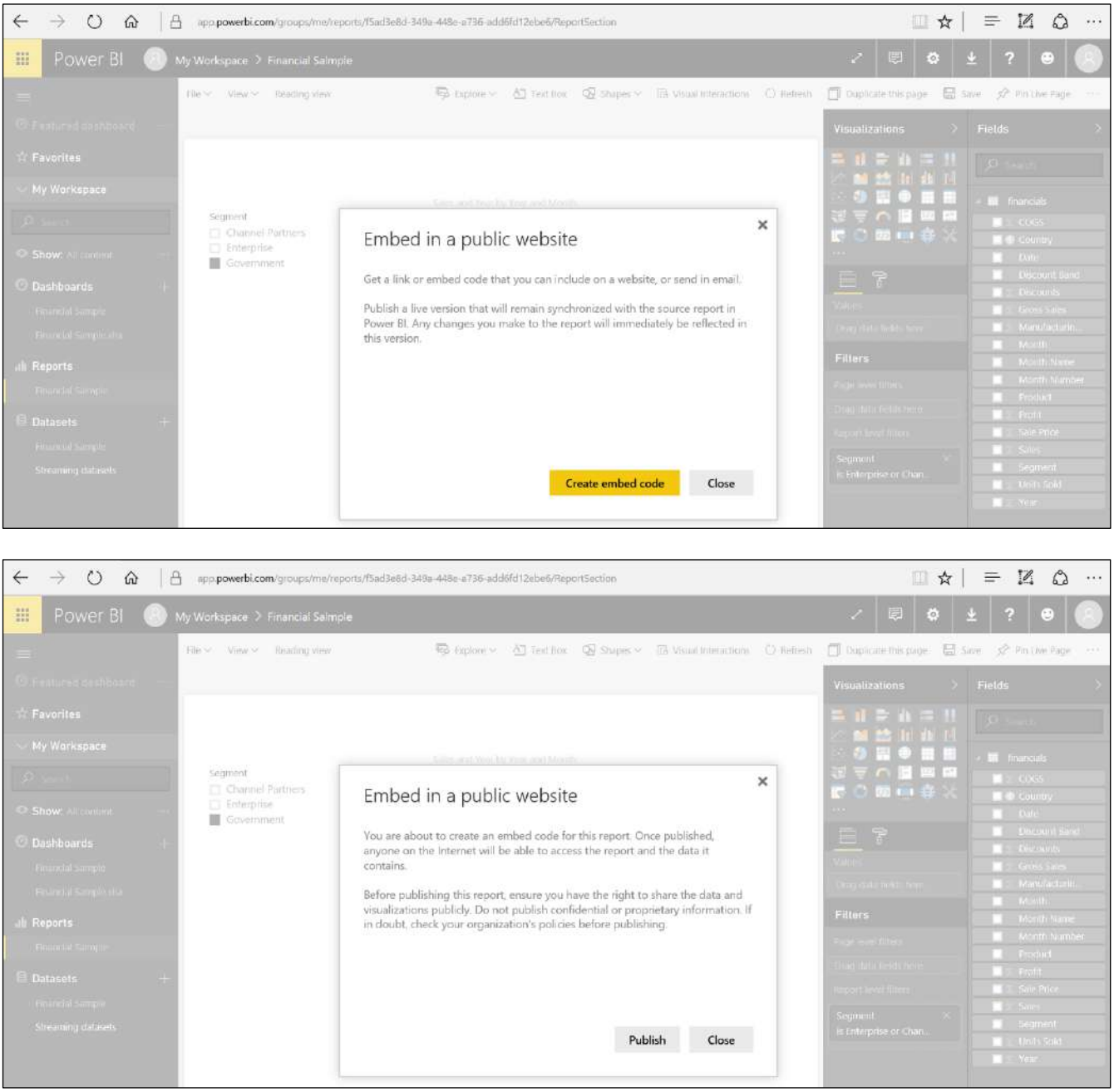


Figure 40: Publish to web—sequence

The new window (in Figure 41) will display several items: a link we can send by email; HTML we can paste into our blog or website; and HTML iframe tags we can integrate in our web application. Note that the size of the graph is in pixels and we can choose from three different dimensions.

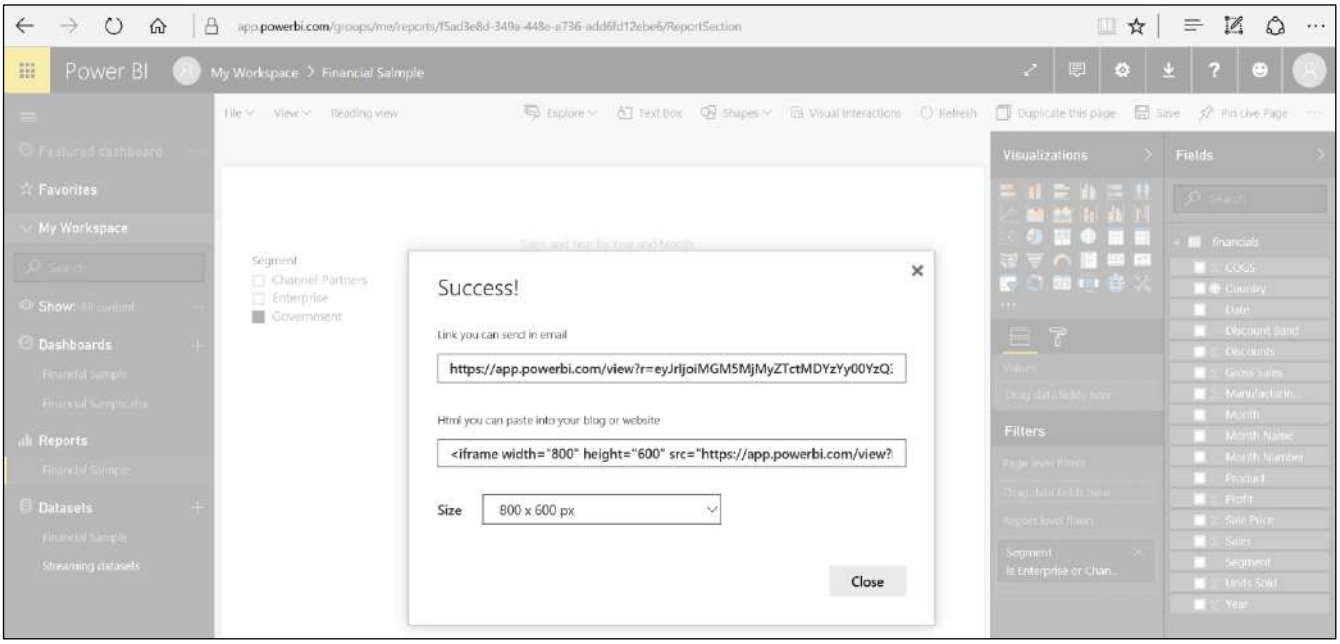


Figure 41: Publish to Web—finishing the publish step

Figure 42 depicts the available dimension choices.

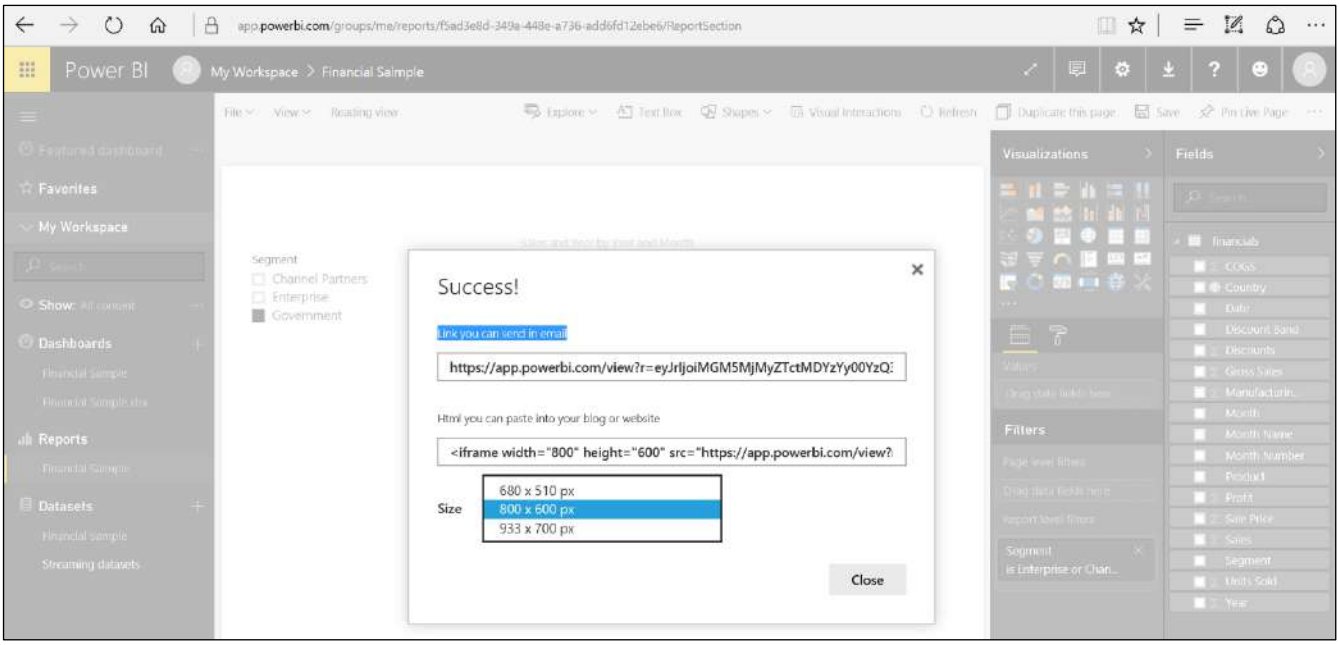


Figure 42: Publish to Web—finishing the publish step

If we use the link by specifying it in the URL of a browser, we will have direct access to our report and can browse it without authentication.

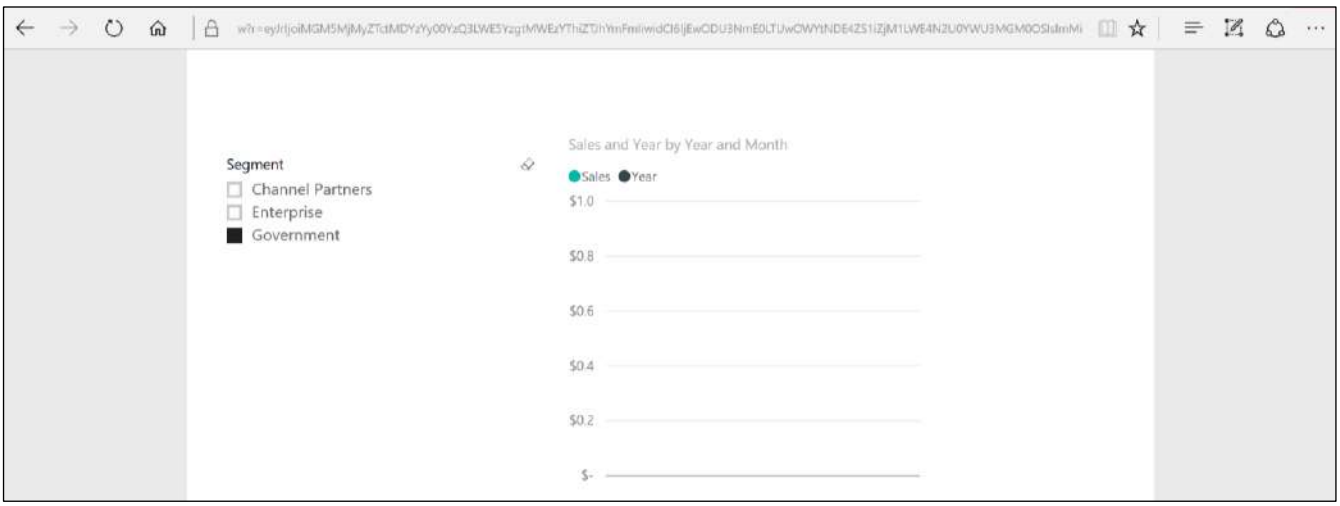


Figure 43: Publish to Web—result

Import data sources: focus

Logging on to the data sources through the web interface is very simple.

We choose the data source type we want to use by clicking **Get Data**.

In the Content pack, we can specify one of four different sources for the data:

- Organizational
- SaaS application (files included)
- Excel workbook
- Files produced through the Power BI Desktop app (databases and data services present in the cloud)

The Excel files can be imported directly from the browser and can be loaded from a local source, or we can access OneDrive for Business, OneDrive Personal, or SharePoint team sites.

The file updates in OneDrive/SharePoint are recognized automatically in the reports and dashboards, and the data will remain in the workbook. If the workbook connects to the external data sources, we can use Power BI for the refresh.

Supported file types:

- Power BI Desktop file (PBIX)
- Excel workbook
- CSV file

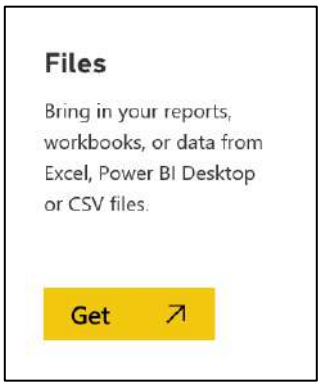


Figure 44: Get Data

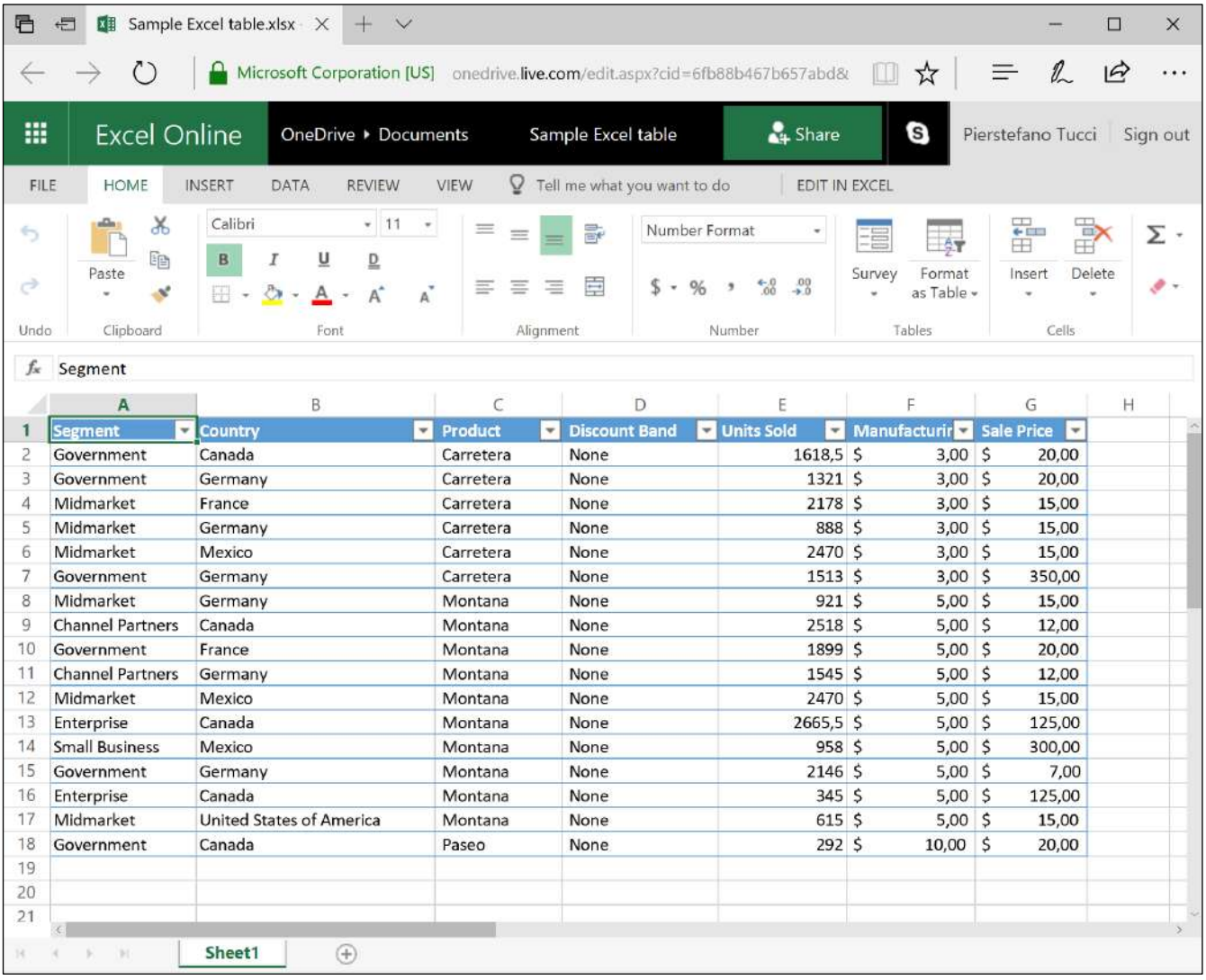
The maximum size for the uploaded files is 250 MB (240 MB for the data model present in the workbook and 10 MB for the remaining part of the data inside the workbook).

If the Excel files are in OneDrive Personal/For Business or in a SharePoint team site, in addition to the import, we can also connect to the Excel workbook, which will have its own life (where it lies). We can also use refresh functions to update our dataset once the data has been imported.

During the import of the Excel workbook, everything contained in the data model will be converted to a dataset. In this particular case:

- Contents of the data model take precedence over the contents at the worksheet level.
- Workbook data model will be converted to a Power BI dataset.
- Workbook Table will be converted to a Power BI dataset.
- Power View Sheets will be converted to pages of a Power BI.
- PivotTable, PivotChart, and custom fields are not recognized.

The data in the workbook will be imported only if it is defined as tables. Power BI will ignore scattered data in the workbook.



	A	B	C	D	E	F	G	H
1	Segment	Country	Product	Discount Band	Units Sold	Manufacturir	Sale Price	
2	Government	Canada	Carretera	None	1618,5	$ 3,00	$ 20,00	
3	Government	Germany	Carretera	None	1321	$ 3,00	$ 20,00	
4	Midmarket	France	Carretera	None	2178	$ 3,00	$ 15,00	
5	Midmarket	Germany	Carretera	None	888	$ 3,00	$ 15,00	
6	Midmarket	Mexico	Carretera	None	2470	$ 3,00	$ 15,00	
7	Government	Germany	Carretera	None	1513	$ 3,00	$ 350,00	
8	Midmarket	Germany	Montana	None	921	$ 5,00	$ 15,00	
9	Channel Partners	Canada	Montana	None	2518	$ 5,00	$ 12,00	
10	Government	France	Montana	None	1899	$ 5,00	$ 20,00	
11	Channel Partners	Germany	Montana	None	1545	$ 5,00	$ 12,00	
12	Midmarket	Mexico	Montana	None	2470	$ 5,00	$ 15,00	
13	Enterprise	Canada	Montana	None	2665,5	$ 5,00	$ 125,00	
14	Small Business	Mexico	Montana	None	958	$ 5,00	$ 300,00	
15	Government	Germany	Montana	None	2146	$ 5,00	$ 7,00	
16	Enterprise	Canada	Montana	None	345	$ 5,00	$ 125,00	
17	Midmarket	United States of America	Montana	None	615	$ 5,00	$ 15,00	
18	Government	Canada	Paseo	None	292	$ 10,00	$ 20,00	
19								
20								
21								

Figure 45: Excel Online—sample Excel table

Figure 46 shows how to look up an Excel sheet.

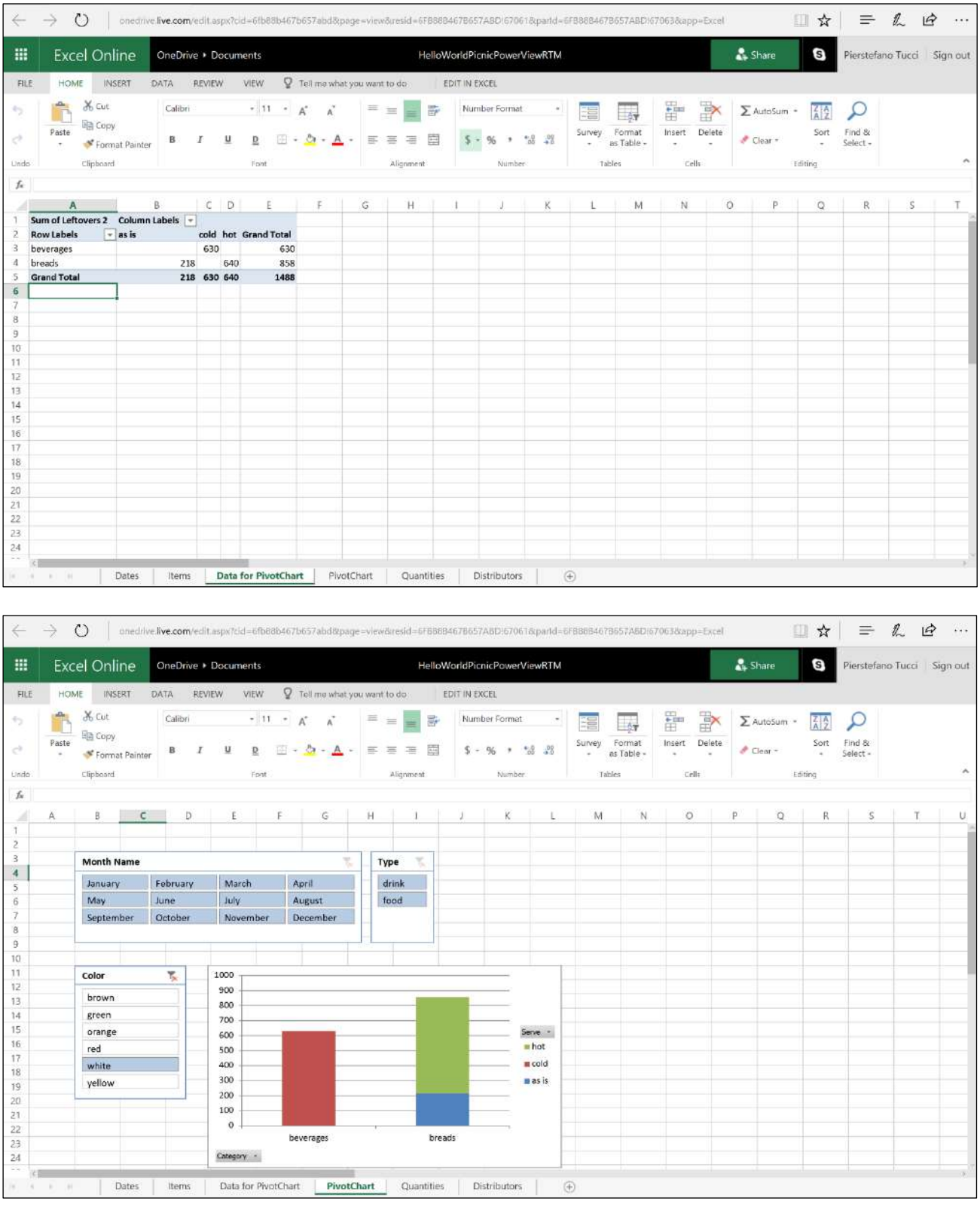


Figure 46: Excel Online—Pivot Chart—sequence

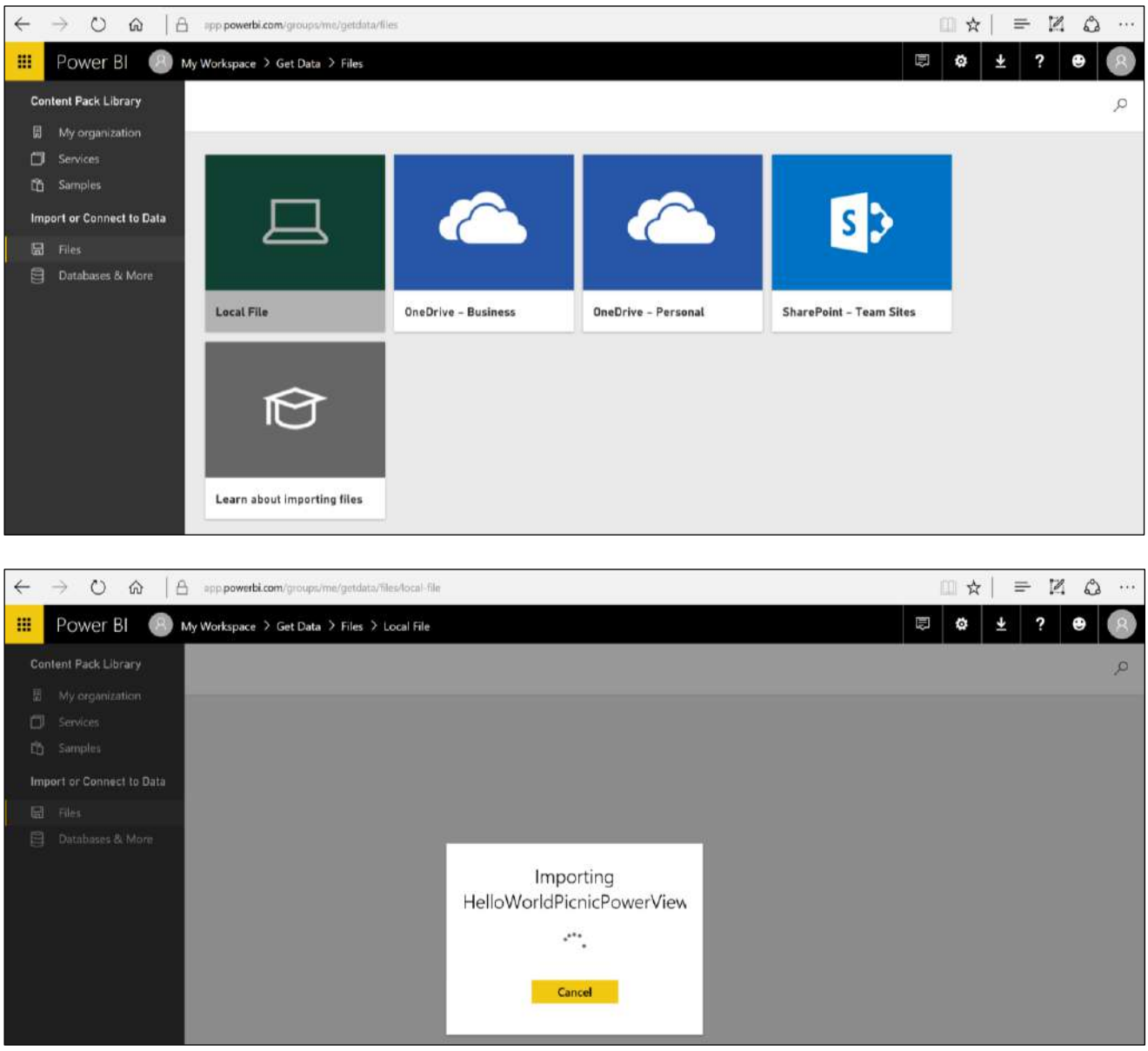


Figure 47: Importing Excel file—sequence

Note that the workbook is converted to various Power BI contents, which are created gradually. After importation of a single content, it will be present in the corresponding section (report, dataset).

By displaying the dataset, we can see how the table present in the PowerPivot data model has been converted to a Power BI dataset and that all the columns have been reported.

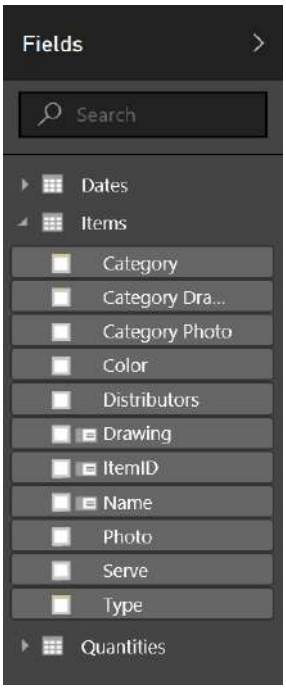


Figure 48: Fields from Excel File

A report will be created that contains a page (with the name of the imported Power View sheet) and the same graph we had on the Power View sheet in the workbook.

We can edit the content or create new graphs and use them together to create dashboards. The content will be identified as a Power BI graph.

Here is an example with OneDrive:

First, select an XLSX file.

Two options will be shown:

- Import Excel data into Power BI
- Connect, manage, and view Excel in Power BI

The connect option does not convert the workbook but rather embeds it into Power BI. We can see this in the Navigation pane on the left side, among the reports with the Excel icon.

By selecting it, we can now surf the Excel workbook in Excel Online.

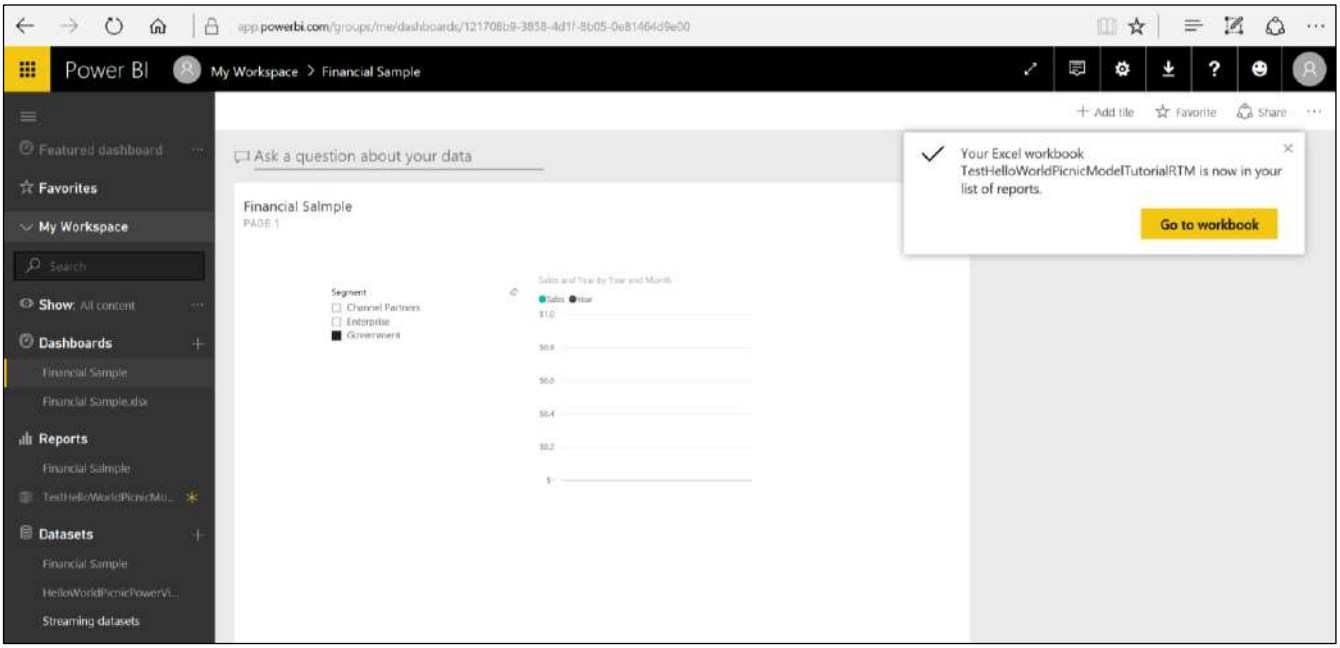


Figure 49: Finishing the Import

The value of this workbook connection with Excel in OneDrive or SharePoint lies in the possibility of setting up the data to refresh. As Figures 50 and 51 show, we can carry out the pinning of the contents by selecting a table or part of a table and doing the pinning in the dashboard.

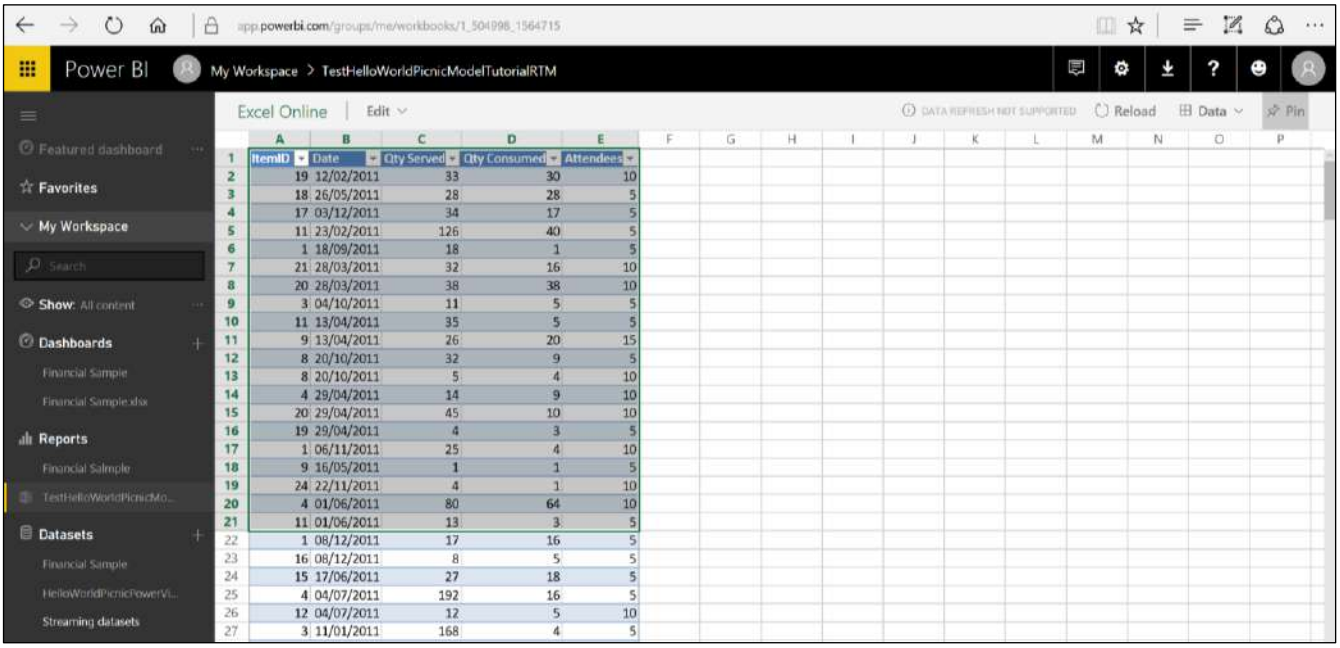


Figure 50: Excel Workbook on Power BI

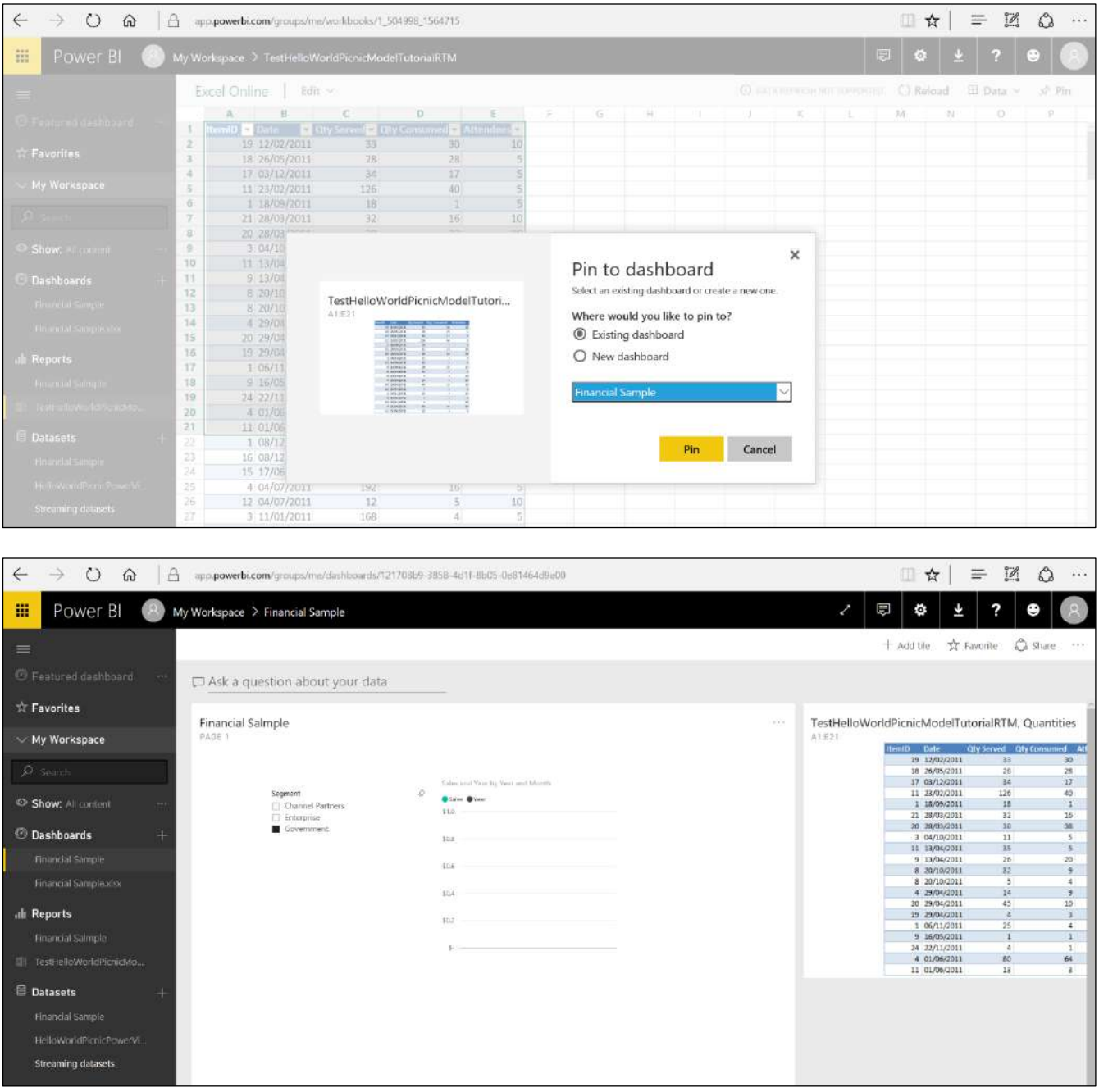


Figure 51: Pin to Dashboard—sequence

When we click the tile in the dashboard, we are redirected to the Excel file with Excel Online.

Chapter 3 Desktop

An overview of the Power BI site

The website www.powerbi.com is the starting point for learning about this product. There you will find:

- Products: a display of the product differences and the licensing and service policies.
- Solutions: examples of how Power BI has been used in certain industries and departments. The examples are interactive and available for download.
- Partners: a gallery dedicated to partners. You can search for and find partners who have developed solutions or have experience with your concerns.
- Learning: a set of resources for studying relevant topics; a community where we can compare our work or look for new information.

If we look toward the bottom of the website, we will find a section that offers an entire series of tools. Below that, there is a series of resources for developers available together with a series of examples, so that we can develop the application and embed to Power BI inside our websites, mobile applications, or dashboards. We will also find important examples and in-depth videos. The second resource is the YouTube channel of Power BI. Accessible in user/mspowerbi, the channel is very informative, and it is constantly updated directly by the product group.

Power BI Desktop

The Power BI Desktop app can be downloaded free of charge through Power BI.

Power BI Desktop a visual tool for reporting as well as for elaboration of the data. It is available in 32- or 64-bit, and it is optimized for use with the Power BI service. It completes consolidated Microsoft technologies, such as Excel, where an add-in is available: Power Query, PowerPivot, and Power View.

The application allows us to connect to different data sources, and it is typically updated on a monthly basis. It simplifies the building of data models used as the basis for the reporting design that will be published on Power BI.

Development method

The development method consists of these simple steps:

1. Query creation to filter, format, or improve the data.
2. Set up relations to create the basis of a data model.
3. Enhance the data model with calculation logic and formatting.
4. Explore the data in a new mode through the drag-and-drop canvas.
5. Choose an interactive design for the report from a wide range of data visualization types.
6. Publish the solution directly on the Power BI service.

The procedure is summarized in the following data flow:

Query creation → Relation set up → Data model definition → Report design → Publication on Power BI

The development method consists of a series of necessary steps and uses the creation of queries to work toward our data sources, filter the data, and determine the correct formats for the data as well as the information we want to build our reporting on. Once we have imported different data sources and determined the queries for those data sources, we can identify or establish relationships between the data. Possible relationship types are 1-to-1, or 1-to-N relations, just as in the relational (i.e., SQL) world. The data model can be enhanced by defining calculated columns or measures. We can explore the data model by using drag-and-drop functionalities, which is exactly what we do when we work with the online Power BI. We start from the dataset, drag the columns related to the tables (part of our dataset), and establish the most appropriate visualizations for building our reporting.

Steps 4 and 5 are actually the same steps, the same techniques we carry out online. The difference is that we carry them out on the app. In the end, the result of our work can be shared with our coworkers and other users in our organization by publishing reports and datasets directly on Power BI. Afterward, we would be able to build dashboards online.

A query is defined as a "data retrieval," and data source types include File, Database, and Azure, along with other services and data sources.

For creation of a query:

1. Specify the data source (file, database, Azure, etc.).
2. Specify a unique name for the query.
3. Define the behavior of the query using the "M" language (where advanced logic can optionally be passed to a program-defined function).
4. Call the query to load data into a table in the data model.

These steps are performed using Power Query.

In the app, the query creation uses a module that is easily linked to the experience of Power Query in Excel. Power Query was an add-in of Excel until the 2013 version, then it was integrated into the 2016 version. The same techniques are used—defining the data sources and carrying out the necessary elaborations, which are later described through a particular language, the "M" language, which is the informal name of Power Query Formula Language. Through this language, we can refine our data source and our table. The supported data sources are numerous, and they are the same supported in Power Query.

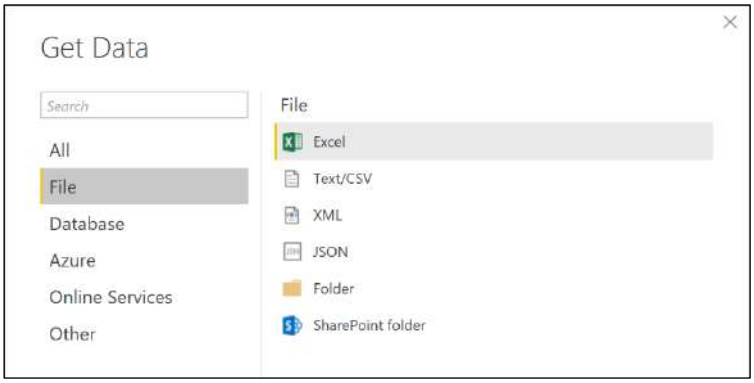


Figure 52: Power BI Desktop—Get Data

Figure 52 shows how to get data such as Excel, CSV, XML, text files, or other files.

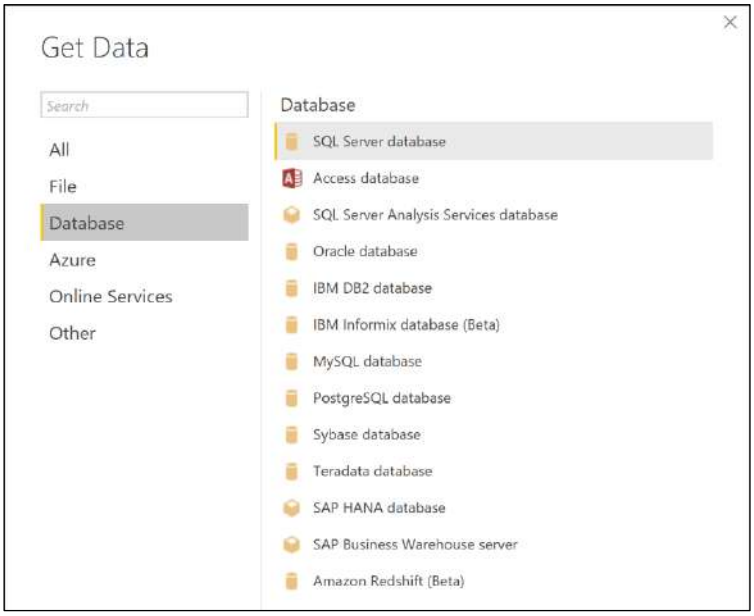


Figure 53: Power BI Desktop—Get Data

Figure 53 shows that sources such as SQL Server, Analysis Services, Oracle, and many others are added as the app updates are released.

It supports the online data services present on Azure such as SQL Azure Database and SQL Azure Data Warehouse.

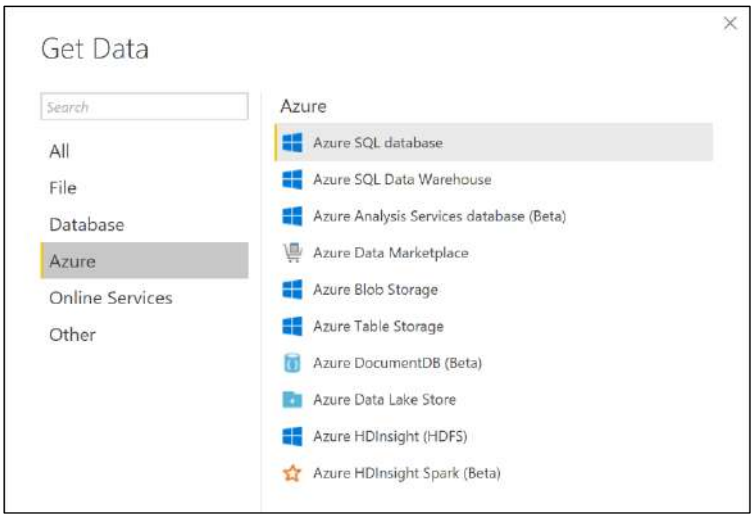


Figure 54: Power BI Desktop—Get Data

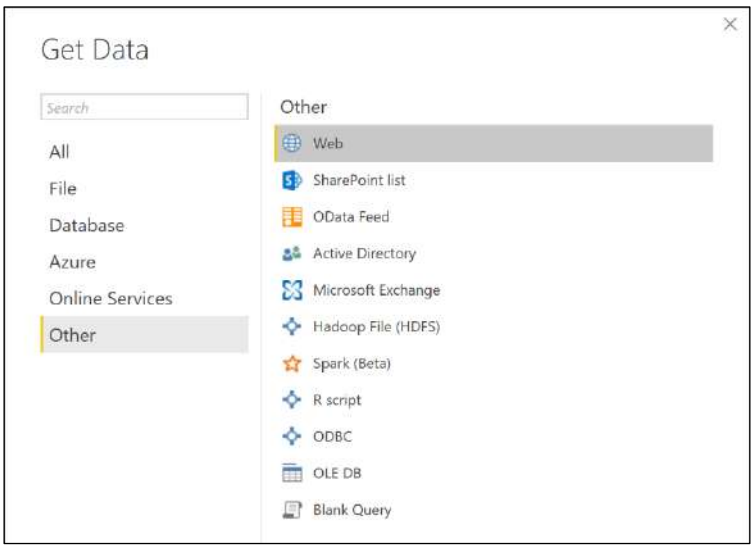


Figure 55: Power BI Desktop—Get Data

Supported data connectors allow us to access data sources ranging from Facebook to Google Analytics, Sales Force, ODBC, and so on.

As shown in Figure 56, we can open the query editor by right-clicking the loaded table.

Figure 56: Fields—contextual menu

In this editor, we can manage columns and formulas, we can filter, and we can load other data (Figure 57).

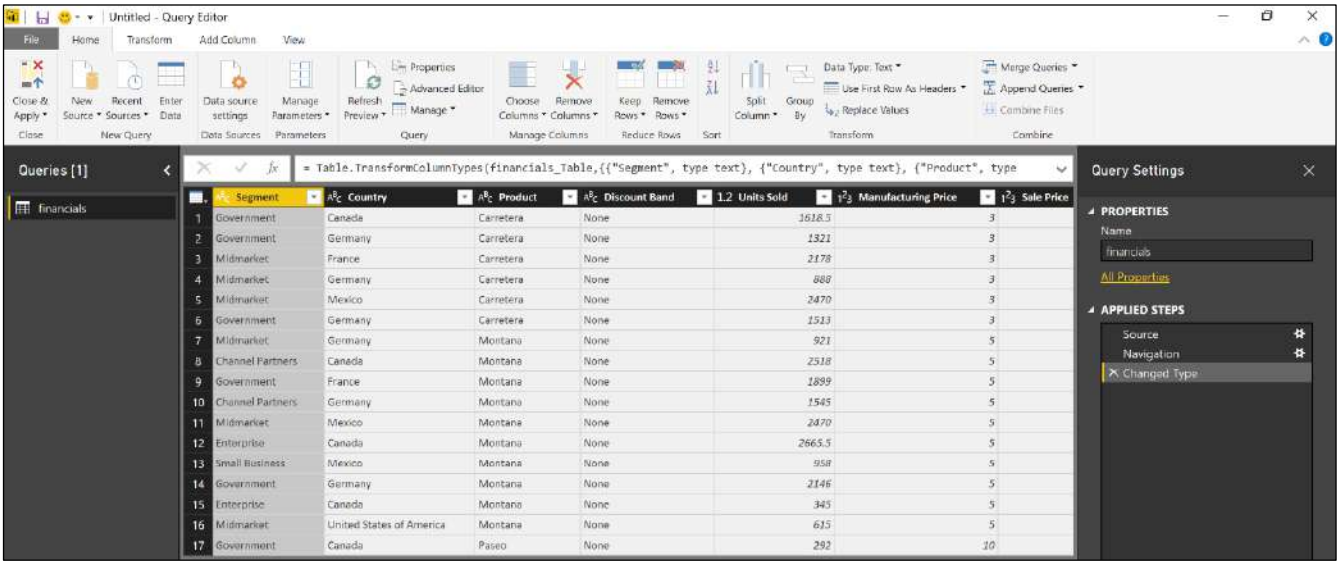


Figure 57: Query Editor

By clicking the columns, we can apply filters, order the data, or remove empty values (Figure 58).

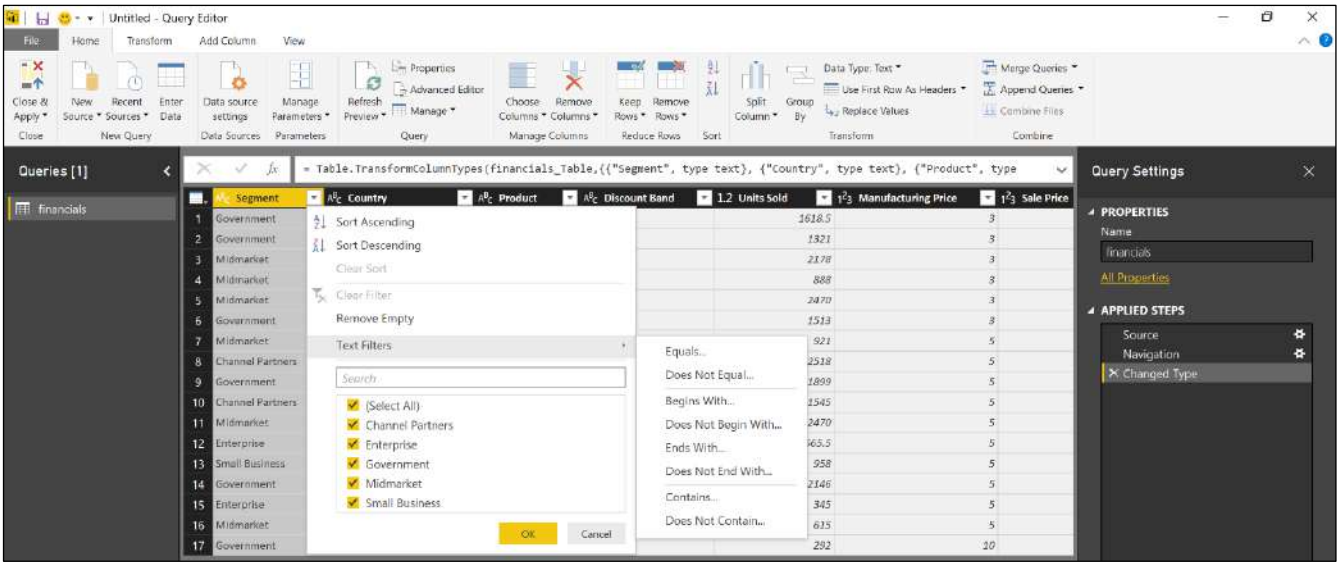


Figure 58: Query Editor—text filters

The creation of queries allows us to filter data, format, and refine them in order to obtain the desired result. The steps can be created easily by applying filters to the columns through the use of controls available from the ribbon or through the context menus of the query or columns. We can select a step and have a preview of the resulting data. The steps can also be removed, and the formula steps can be shown or edited in the formula bar.

As these sources are saved and we carry out the editing steps of our query, they are saved in a descriptive panel, as seen in Figure 59.

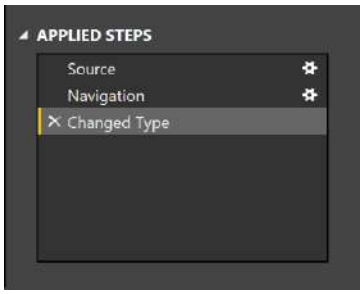


Figure 59: Applied Steps—descriptive panel

We can also delete or edit intermediate steps at any time. If, for example, we accidentally remove all the rows by leaving only the first 10, when we intended to remove all rows except for the first 20, we can cancel the remove operation and then correct the operation. All of the following steps will recognize the modification that has been carried out in the intermediate step.

Step definition of a query: the controls

Numerous controls are available on the ribbons of the query editor and in the context menus so that we can manage columns. For example, we can:

- Delete the rows and remove any mistakes.
- Transform the data.
- Split the data.
- Add columns by using a formula.

Because the query editor is WYSIWYG, we can easily test and do a rollback of the modifications.

We can also use different controls on the query that we want to define, and all that we do will be reflected in the data we display in tabular format. We display exactly the result of the steps as we carry them out. It is possible to manage columns, change data types, remove or add columns, split the tables (columns), and take many other actions.

New queries can be created through the merge of two queries (joined in a common column) or with the appending of two queries (union).

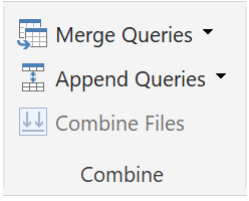


Figure 60: Query Editor Ribbon—combine section

As shown in Figure 61, the operations of merge and join include:

- Left Outer (all the data from the first table, matching with the second one)
- Right Outer (all the data from the second table, matching with the first one)
- Full Outer (all the rows from both the reported tables)
- Inner (only matching rows)

- Left Anti (rows only related to the data present in the first table)
- Right Anti (rows only related to the data present in the second table)

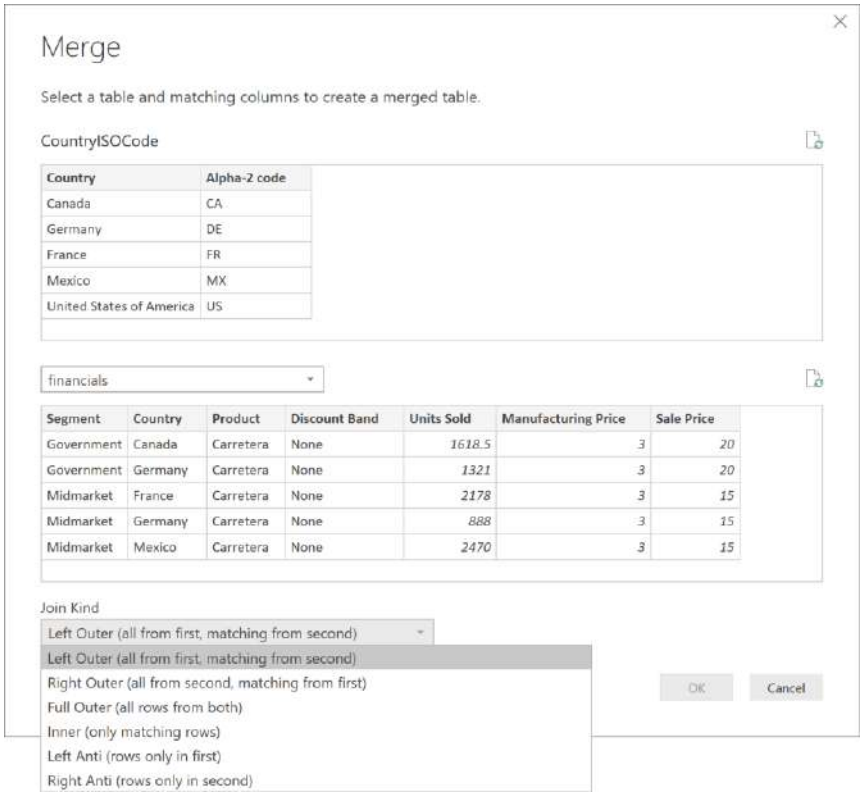


Figure 61: Query Editor—Merge options

We can define more than one query. Typically, numerous queries are used, and they can be connected to heterogeneous data sources. Once the queries are defined, they can be combined by carrying out the merge or the append. In the case of the merge, a join is defined. In fact, we connect two queries. In the case of the append, the results of two different queries are put together and connected. It is good to make sure that the tables are compact as well as solid and that they have the same column names so that the columns can be identified in both tables.

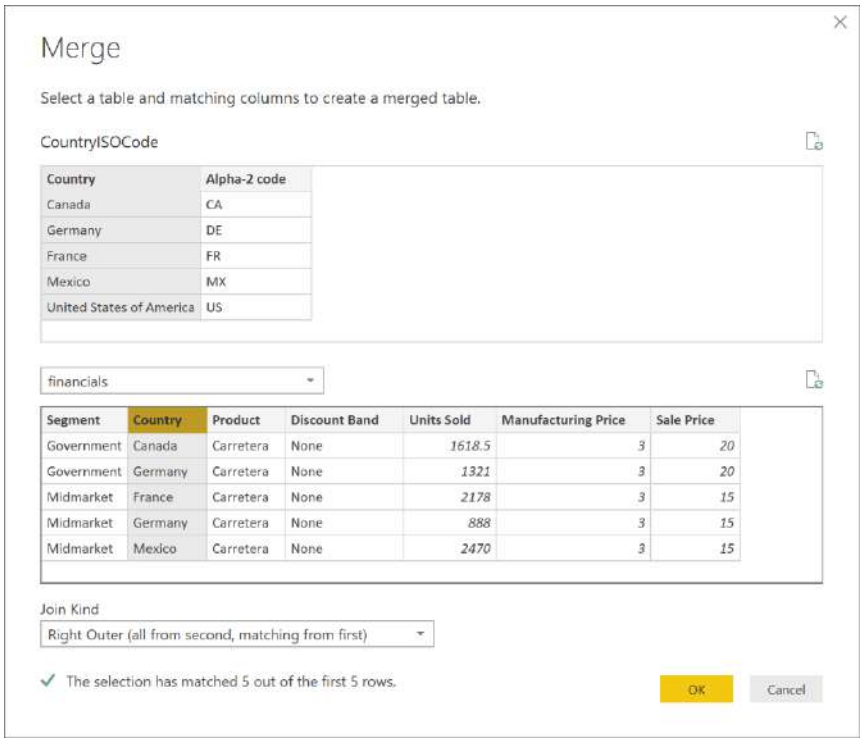


Figure 62: Query Editor—merge options

Figure 63 demonstrates that you can edit this merge via the Merged Queries icon.

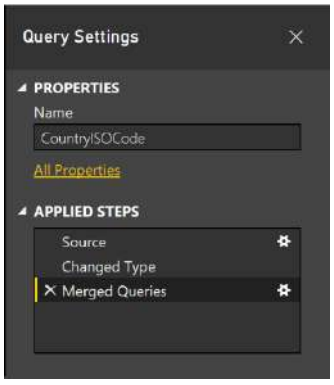


Figure 63: Query Settings—descriptive panel

Configure relations

By defining a relationship, we create an implied filter that operates across two or more tables. The filtering relationship is not the same as Foreign Key Constraint, which is used to ensure data integrity. The relationship can be defined between any two tables, independent of their data connection or data source type, but they must be based on a single column with matching data types. Self-referential relations are not supported.

In fact, with a merge between two queries, we set up a relationship on which we can carry out further steps. There are restrictions for which a relationship cannot be created in a single table (hence the so called self-referential relationships are not supported). However, there are no restrictions at the level of Foreign Key.

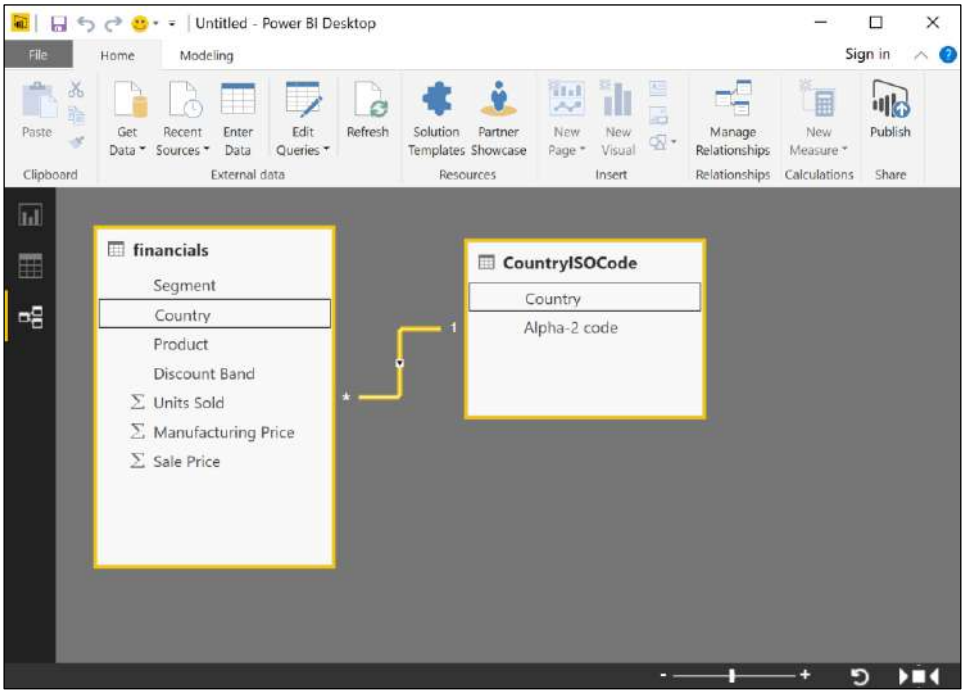


Figure 64: Power BI Desktop—relationship section

In this way, we define our relations by obtaining, for example, a model like the one represented in Figure 64, in which we have N relations that can be 1-to-1 or 1-to-N. We are, in fact, building our data model.

Data model definition

The data model tables can be extended with calculations or they can hide the calculated field that has already been created (in other words, they are made unavailable for reporting). We can set up these features of the data model columns:

- Data type
- Format
- Category (that is spatial or web URL)
- System (based on another column of the table)
- Visibility

Starting from the relationship, we have a data model. At this stage, once the data model has been defined, we can define the types that will be used and the formats, and we can decide if and which columns will be visible or not, define the systems among data, and so on.

Data model definition: calculations

We can also enhance the data model by using the Data Analysis Expression language (DAX). People familiar with the PowerPivot models will have a familiarity with the DAX language, which allows us to implement calculated columns and measures.

DAX consists of:

- Excel functions (about 80 functions)
- Table functions
- Aggregated functions
- Web-surfing functions of the relations
- Modification functions of the context
- Time intelligence functions

We must insert calculated columns or measures in whichever data model we use. We will hardly be able to do without it. All the support for the DAX language is also present in Power BI.

There are two different types of calculations, and both are defined by using DAX:

- Calculated column
- Measure

Calculated column

We use a formula to define a calculated column. Starting from other columns, we add a new column, which is the result of the formula we have previoiusly defined. A calculated column in a model uses memory space at runtime. We should keep this in mind to avoid Power BI out-of-memory errors.

There are defined calculated columns to add new columns to the tables. Each Column Value per row creates and saves the data in the data model. Note that they use up space in the data model. The Column Values are recalculated when a refresh occurs in the table and when the refresh is carried out on the dependencies of the formula.

Measure

The measures, unlike the calculated columns, do not use up storage space. The measures are defined by using a formula. They are made available to the dataset of Power BI, but they are calculated in real time, and for this reason they do not use up space. The formulas are defined by using the group functions, such as sum, count, distinct count, average, minimum, and maximum.

There are defined measures for adding group logic to the data model. The values are neither created nor saved in the data model. The formulas are enhanced in the query time mode.

Design report

Once the data model has been set up, we can proceed with the report design exactly as it occurs online by using the Power BI service through the browser interface. We can also build a report by using the Power BI app.

The reports can be designed based on the interface of the visible data model. It is also possible to add text boxes and pictures.

The design experience is very similar to the one available with the online service of Power BI.

Publication on Power BI

At the end, our work can be directly published online on Power BI, and it can be shared with other users. Starting with the reports and datasets we published, we can also define certain dashboards.

The Power BI Desktop file can be loaded on the Power BI service or published directly. In case of overwriting an existing dataset, we must take into account that if there are two or more datasets with the same name, one must be removed and the Power BI Desktop file must be renamed.

Remember, too, that when renaming the columns or the measures, the report or the dashboard tile can be damaged.

A work example with Power BI Desktop app

We start Power BI Desktop and import the data sources that are online.

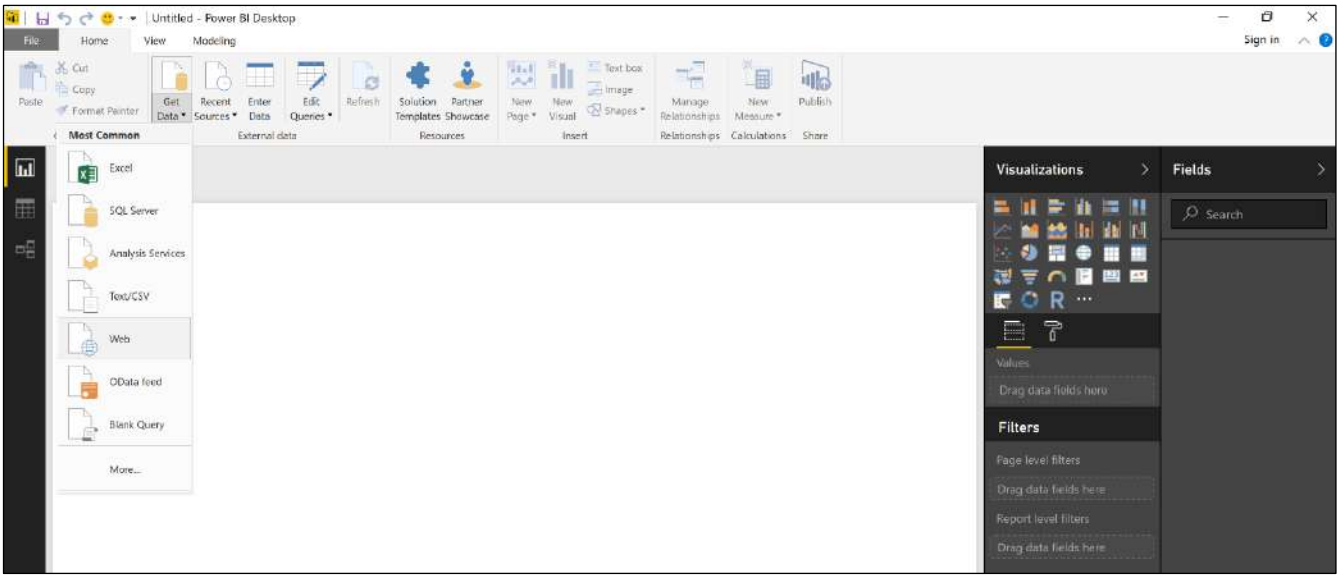


Figure 65: Power BI Desktop—Get Data

In particular, we will use the following data source:

http://www.bankrate.com/finance/retirement/best-places-retire-how-state-ranks.aspx

The link includes an article about retirement in the United States—specifically, which states are the best for retirement. It displays the states with different indexes, such as the cost-of-living, the crime rate, and so on. Let's use one of the article's tables for writing a report.

Note that the use of the full name of a state can be a problem in the graphical representations. Generally, the use of standard abbreviations as recorded by Wikipedia is preferred. In particular, we will use ANSI abbreviations.

The reference website is:

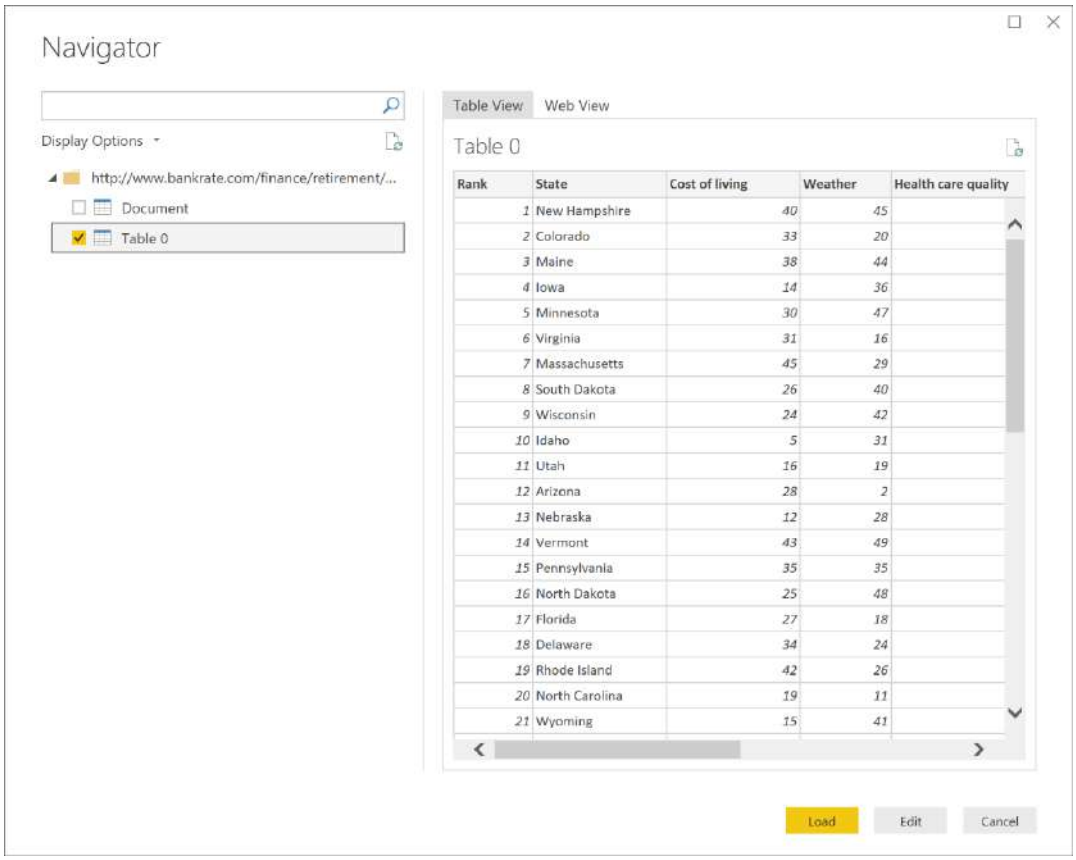
https://en.wikipedia.org/wiki/List_of_U.S._state_abbreviations

Consequently, we import the data through the Get Data functionality.

If we click on the More entry, we can display all the source data currently supported. We choose the entry **Web**, click **Connect** in the URL, specify the page concerning the state statistics, and click **OK**.

Figure 66: Power BI Desktop—Get Data, From Web

As we see in Figure 67, two types of content are recognized on the page (Document and Table). By searching, we can see the contents and select the table containing the data we are interested in.



Rank	State	Cost of living	Weather	Health care quality
1	New Hampshire	40	45	
2	Colorado	33	20	
3	Maine	38	44	
4	Iowa	14	36	
5	Minnesota	30	47	
6	Virginia	31	16	
7	Massachusetts	45	29	
8	South Dakota	26	40	
9	Wisconsin	24	42	
10	Idaho	5	31	
11	Utah	16	19	
12	Arizona	28	2	
13	Nebraska	12	28	
14	Vermont	43	49	
15	Pennsylvania	35	35	
16	North Dakota	25	48	
17	Florida	27	18	
18	Delaware	34	24	
19	Rhode Island	42	26	
20	North Carolina	19	11	
21	Wyoming	15	41	

Figure 67: Power BI Desktop—Get Data, Navigator

Note that Figure 68 shows how to click Edit in order to edit as well as to work on the data of the table.

The column named Rank is not relevant, and we can remove it (right-click).

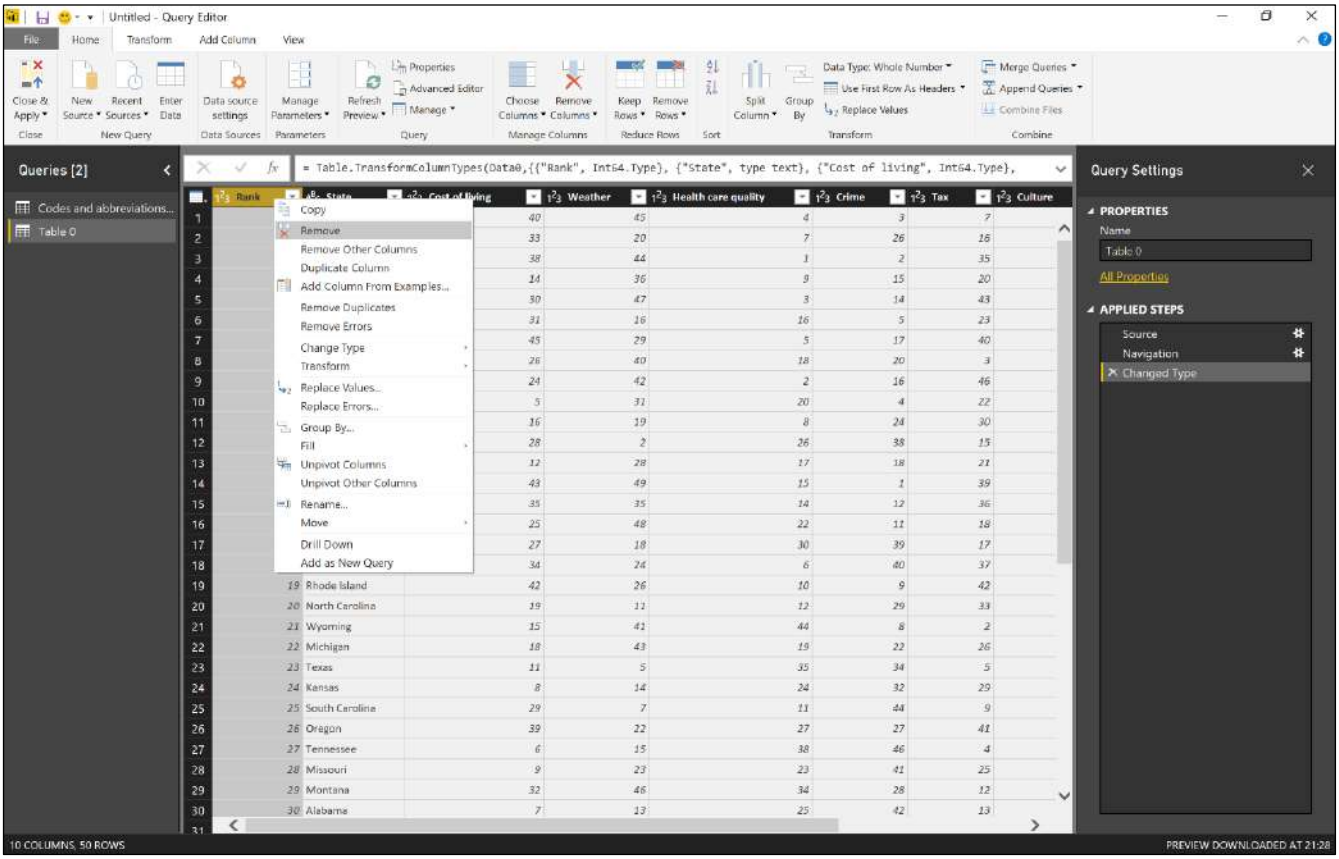


Figure 68: Query Editor

If a field has been identified as a numerical whole number, we must correct the string with empty spaces via the Replace Values interface, as in Figure 69.

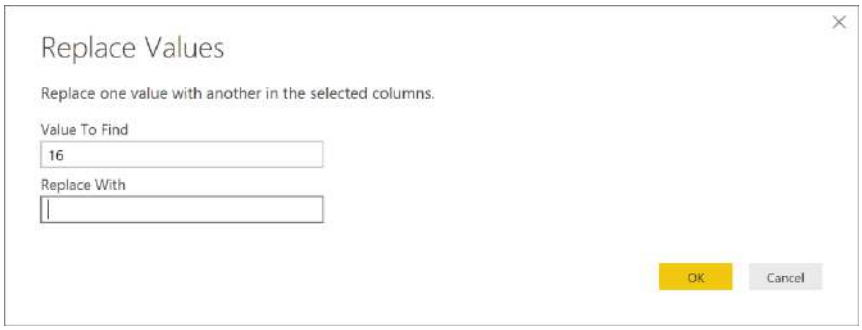


Figure 69: Query Editor—replace with

Note that when all the target values have been replaced, it is possible to convert them to whole numbers (no decimal points) by right-clicking and selecting from the context menu.

Figure 70: Query Editor—the formula bar

Figure 71 shows how to use Advanced Editor on the ribbon in the query section in order to get a summary of all the reproducible steps.

Figure 71: Query Editor—Advanced Editor

Next, save the query by naming it.

Figure 72: Query Editor—Query Settings

The query is imported automatically and is visible on the right panel, as in Figure 73.

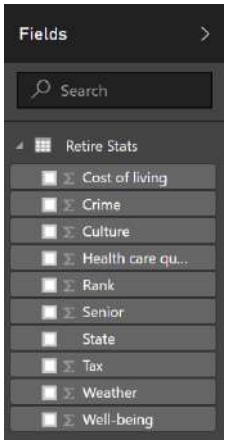


Figure 73: Report—Fields

We also import the data concerning the abbreviations present in Wikipedia. We use the “Get data” functionality, then we click **Web** and specify the address in which the information is contained.

In Figure 74, you can see the address we want—the one that contains the table with state abbreviations—and we edit.

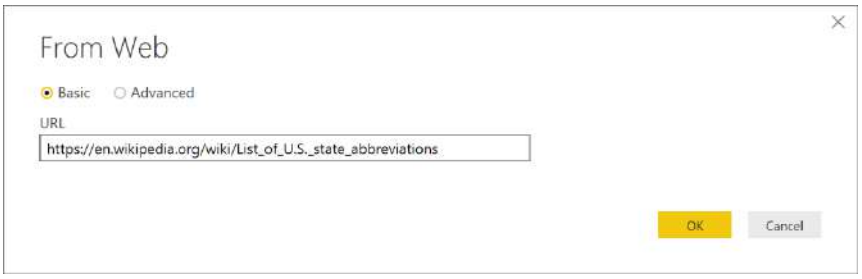


Figure 74: Power BI Desktop—Get Data, From Web

Next, click **OK**.

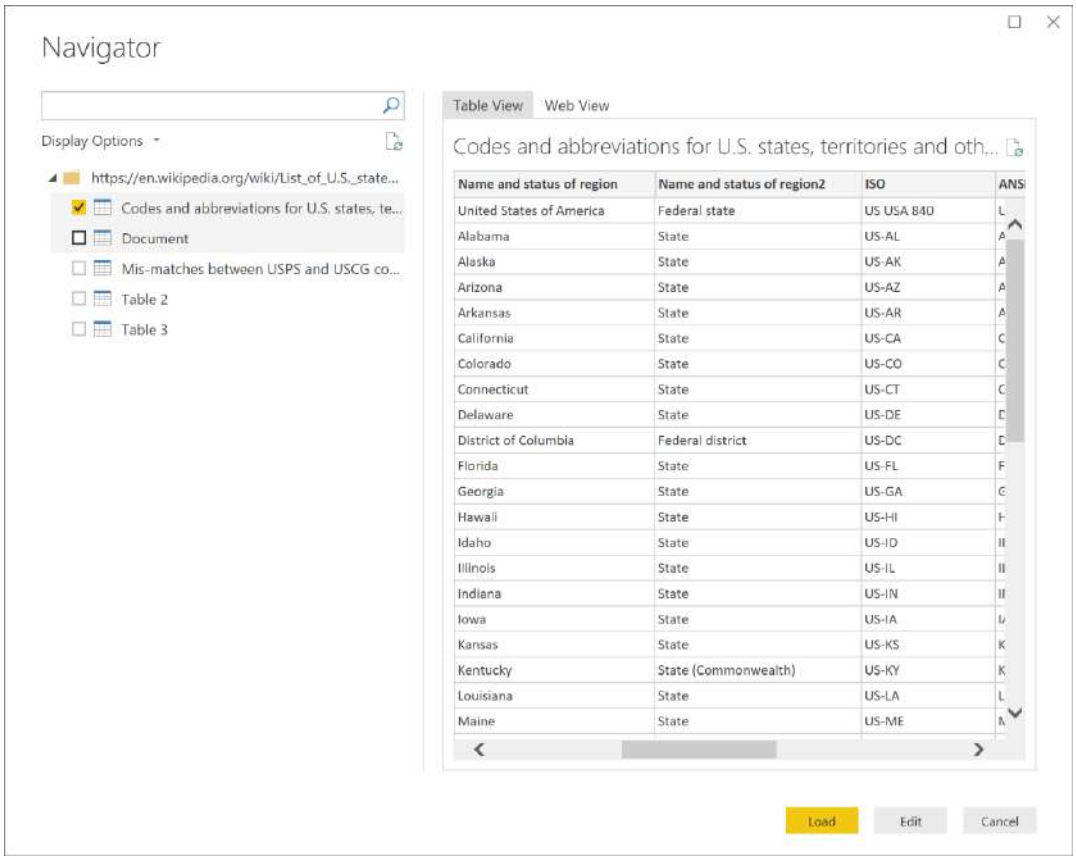


Figure 75: Power BI Desktop—Get Data, Navigator

In Figure 75, we click **Load**.

We remove rows via the row removing functionality present on the ribbon: Reduce Rows, Remove Rows, and Remove Top Rows.

Figure 76: Query Editor—Remove Top Rows form

Next, we specify the number of columns to be removed (two, in this case).

We are only interested in the states present in the statistics, therefore all the other areas won't matter here. By using the Remove Bottom Rows button in Figure 77, we will remove the 25 rows from number 54 to 78.

Figure 77: Query Editor—Remove Bottom Rows form

Remember, we are only interested in the abbreviations with the ANSI two-letter codes, so it makes sense to select the two columns we are interested in and remove all the others.

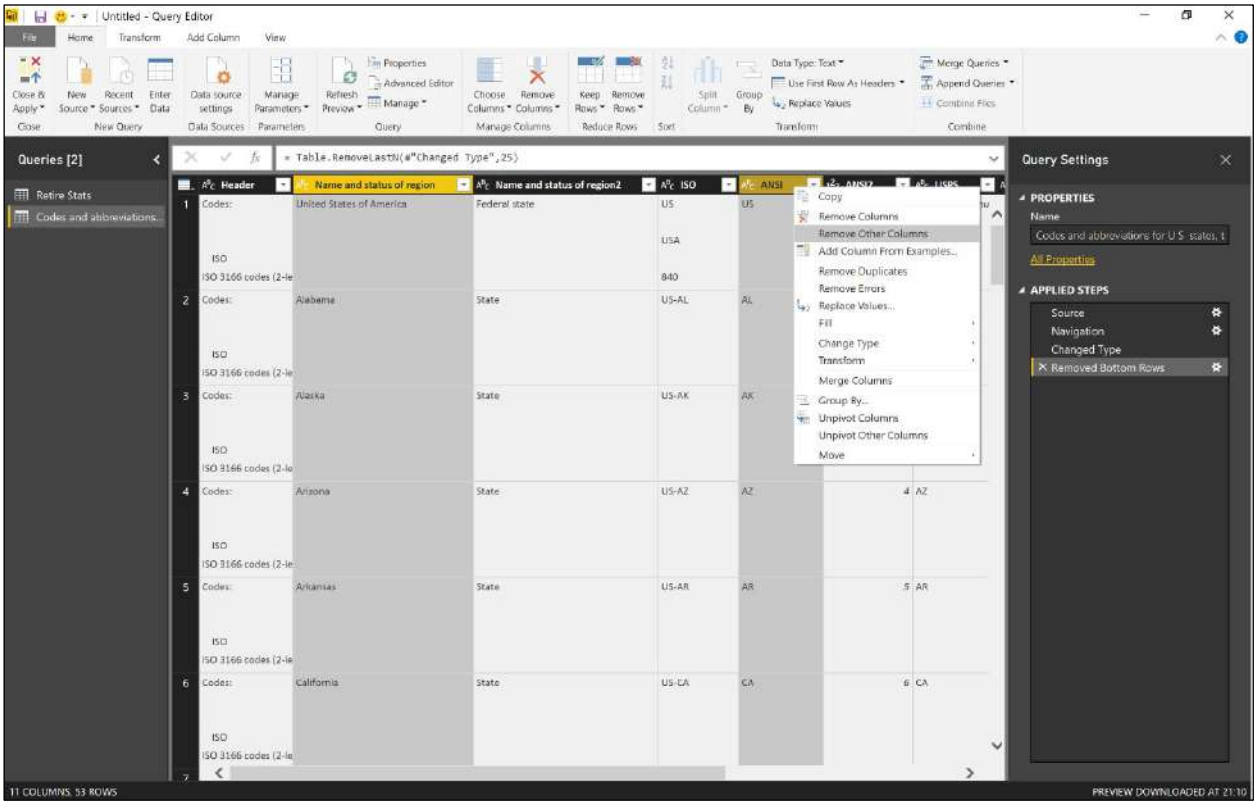


Figure 78: Query Editor—column options

As a result, we have the table we will use to build the data model.

We can rename the table and save it with **Close and Apply**, which is the first entry on the top left of the ribbon.

Both tables are now ready to be reported.

Figure 79: Report—Fields

We use the “Manage relationships” function present on the ribbon Relationships in order to obtain the relationship.

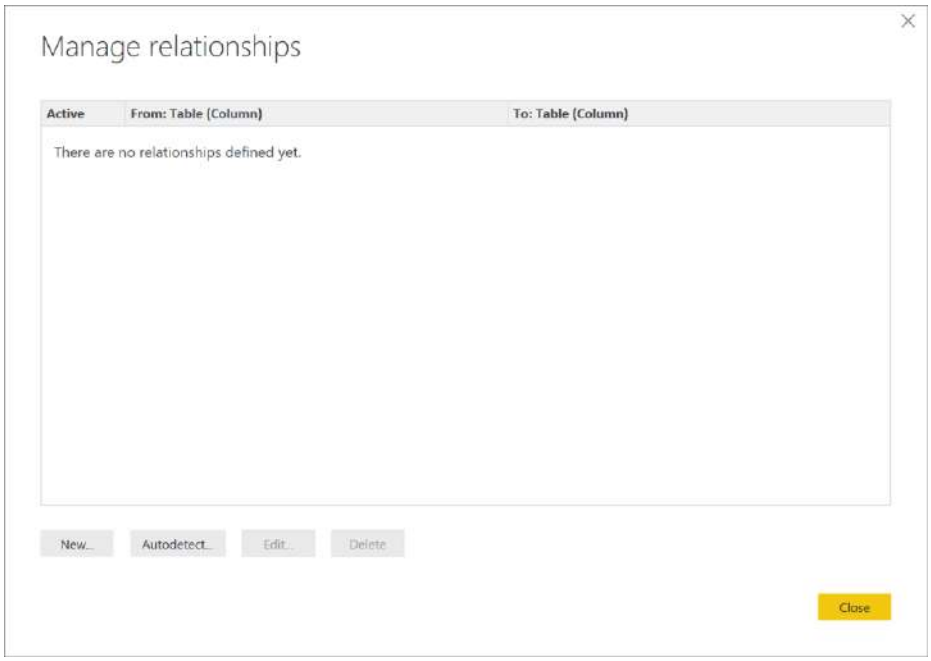


Figure 80: Relationships—Manage relationships

We can also try to implement Autodetect in order to carry out the automatic survey, which is based on the names of the columns.

Figure 81: Relationships—Autodetect

Figure 81 shows a case that will not work, which demonstrates why it is wise to create the report in manual mode.

Next, we click the entry **New** and identify the first table. As shown in Figure 82, the other table is automatically selected.

We use the name of the state to create a relationship of 1-to-1. Note that the direction of the filter can work both ways.

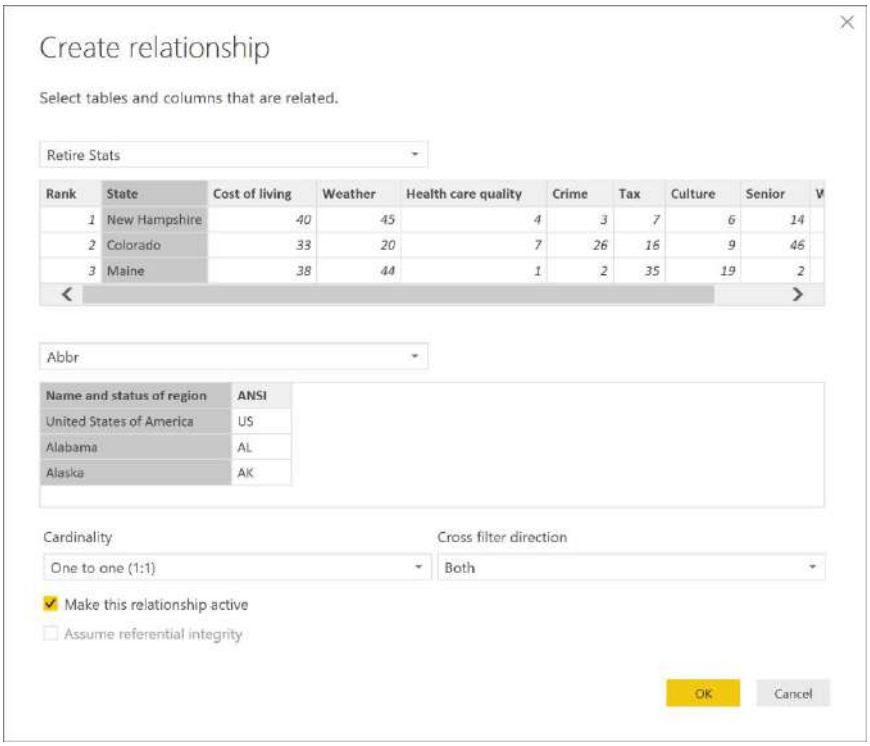


Figure 82: Relationships—Create relationship

In light of the modifications, Figure 83 displays the graph of the relationships.

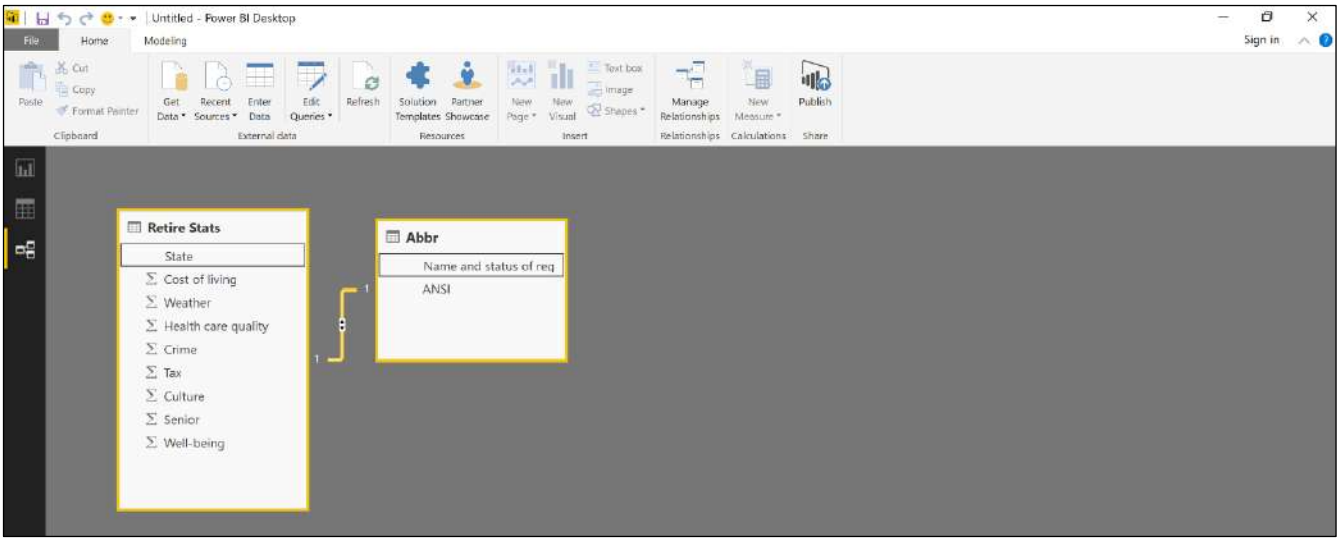


Figure 83: Relationships

Now the data model is set up. We can display the data of our queries at any time (see Figure 84).

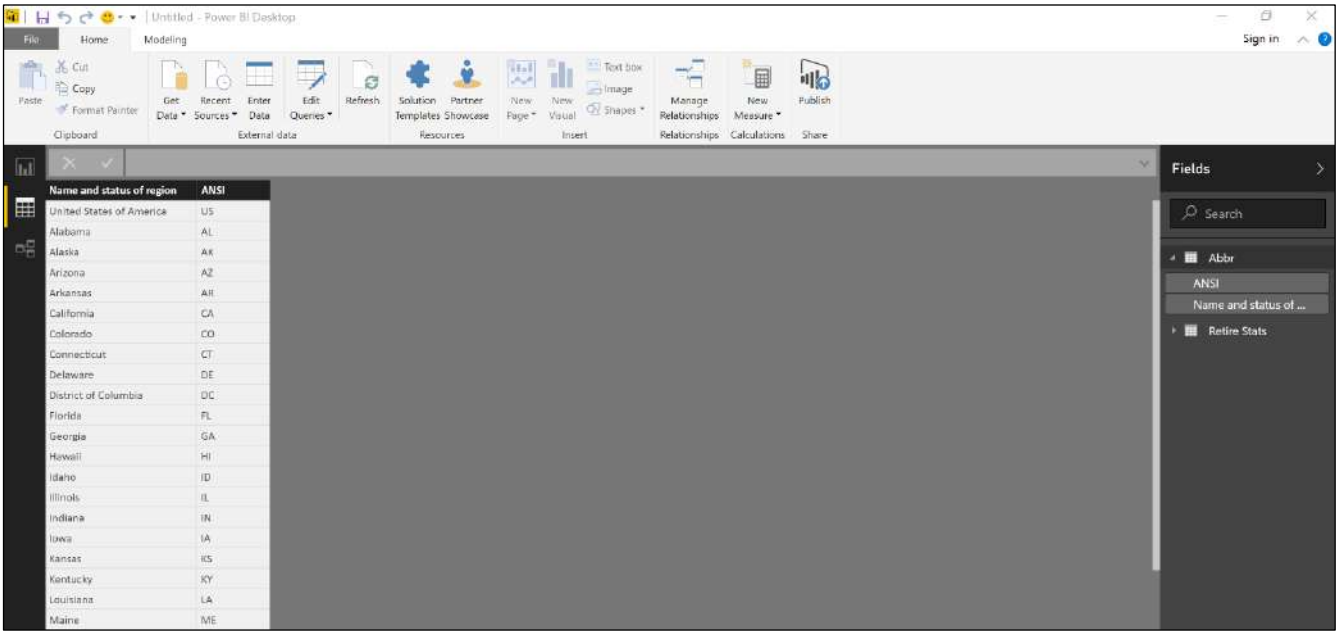


Figure 84: Data Section

We create the report exactly as it takes place online.

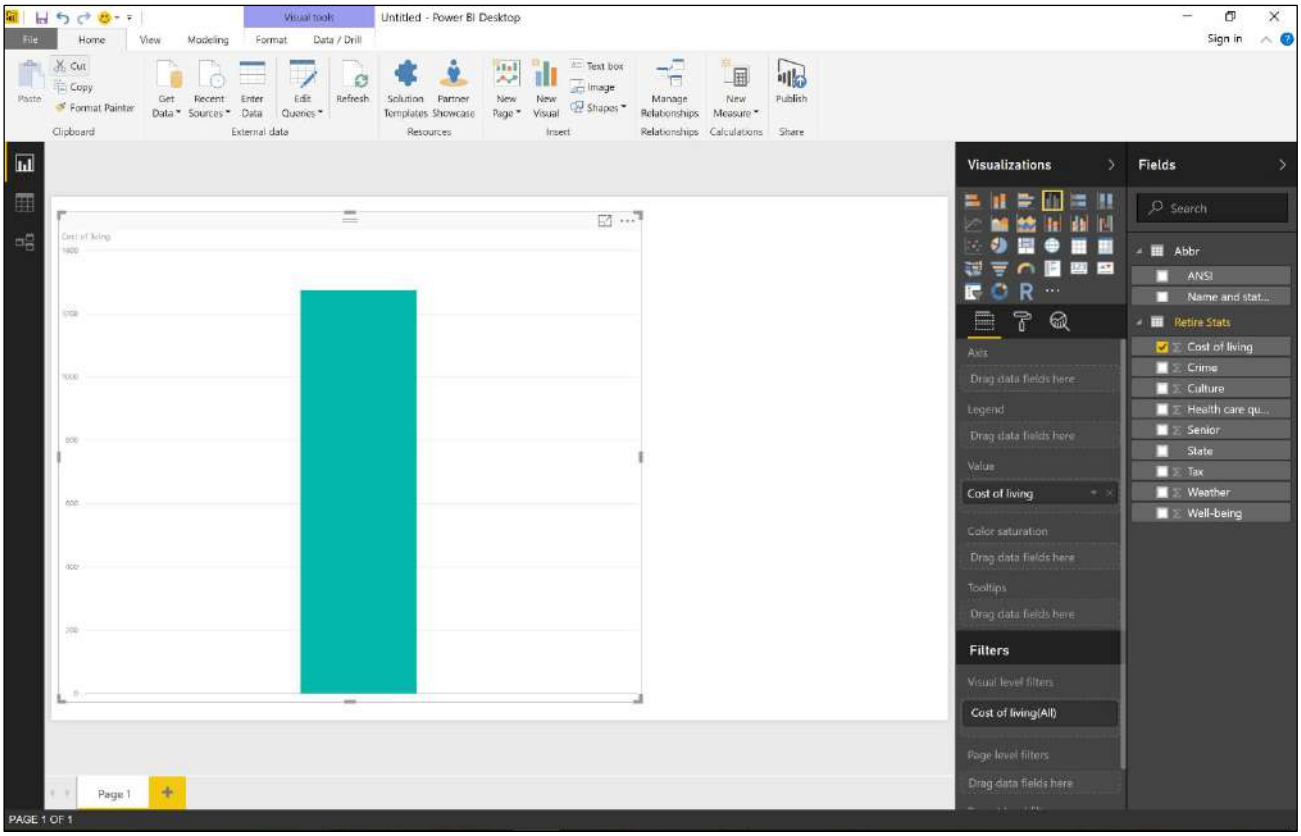


Figure 85: Visualizations

Figure 86 shows that we can create a graph about the cost of living. Note again that instead of using the name of the state, we use the two-letter ANSI code.

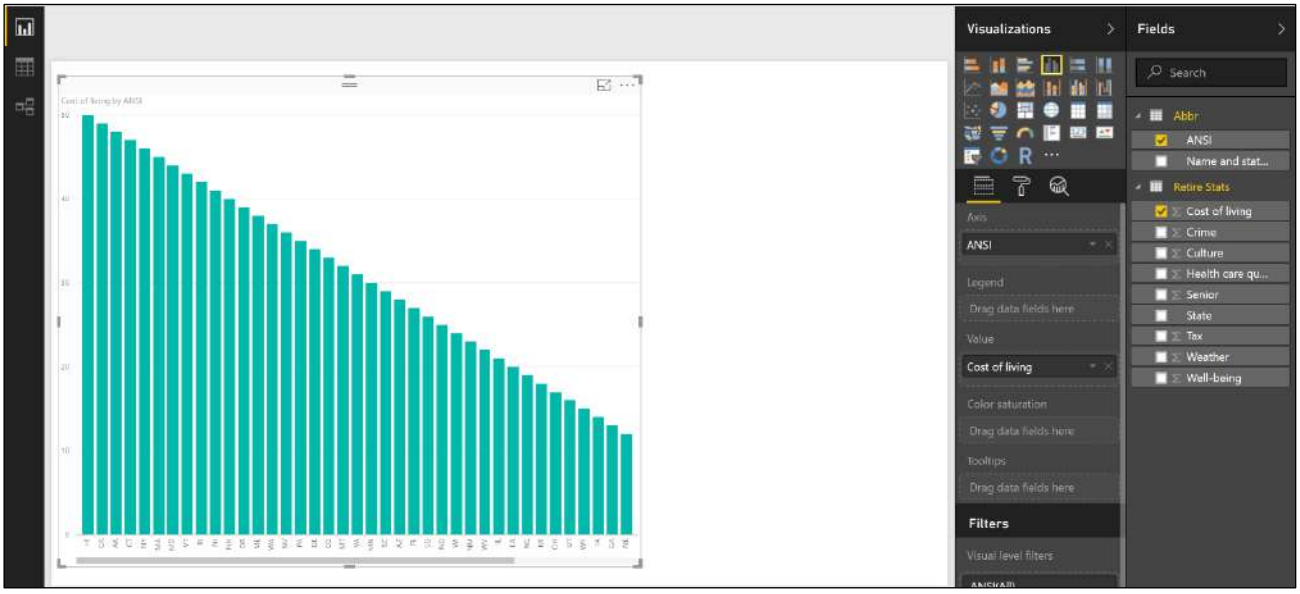


Figure 86: Visualizations

Figure 87 shows how we might change our graph by using a more suitable visualization.

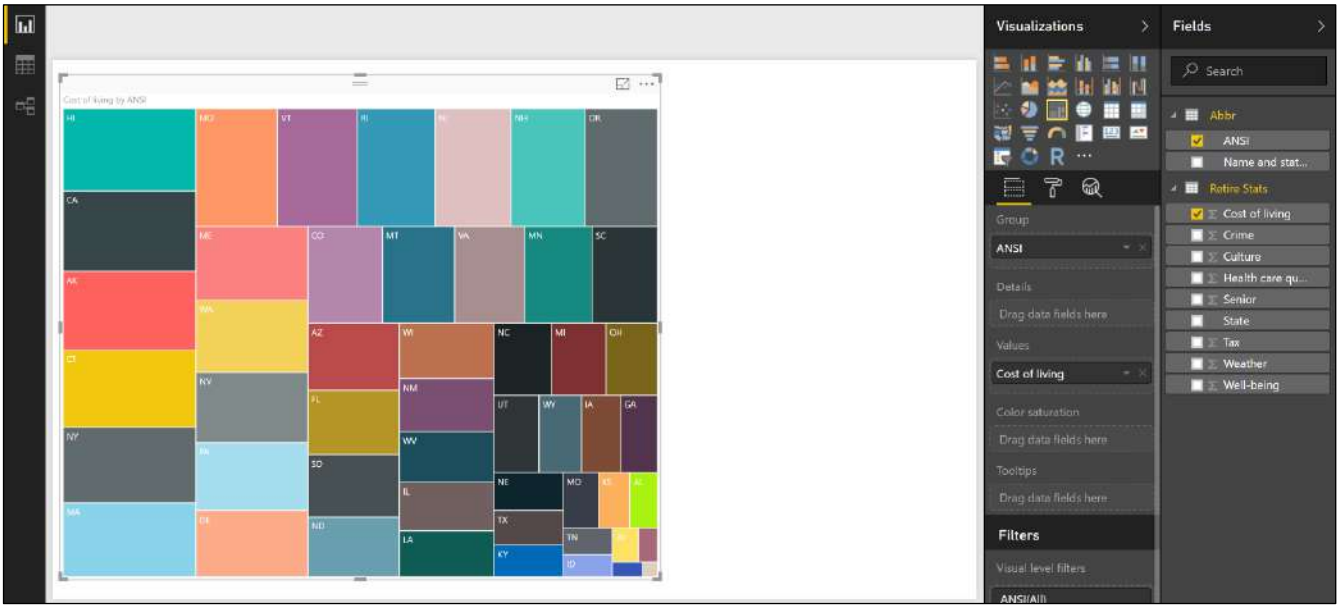


Figure 87: Visualizations

Next, we save the result of our work in PBIX format.

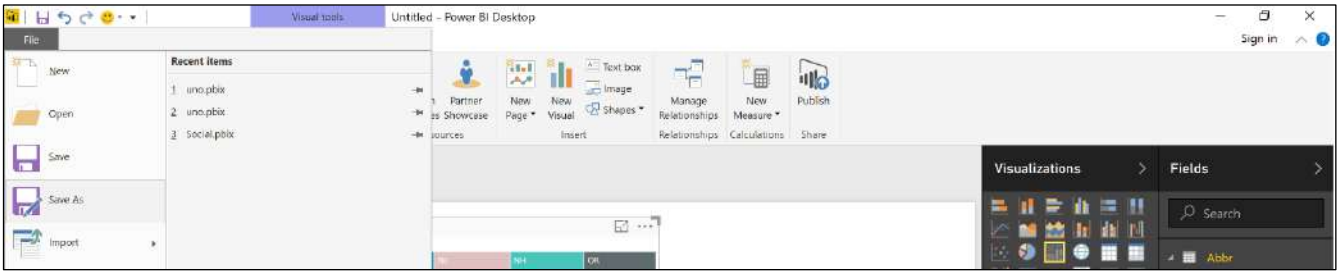


Figure 88: Save Power BI File

Now we proceed with the publication.

Click the entry **Publish** in the Share section of the ribbon (Figure 89).

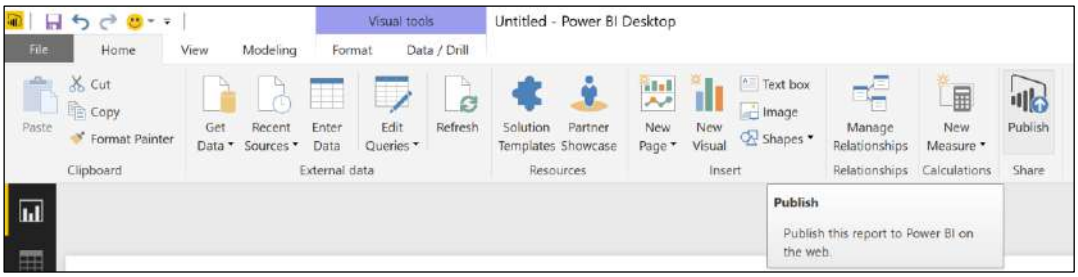


Figure 89: Publish

Now go to “Sign in.”

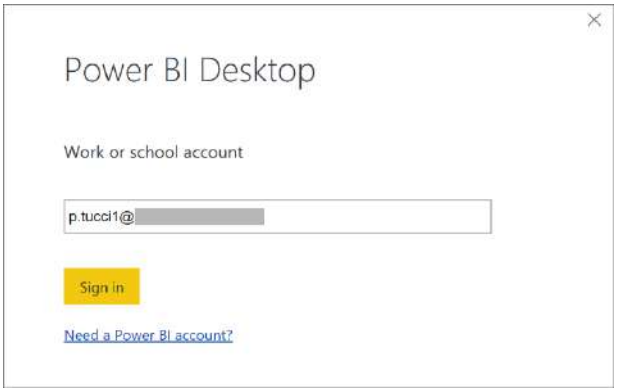


Figure 90: Publish—Sign in

Figure 91 depicts how to authenticate—by supplying our access credentials to Power BI.

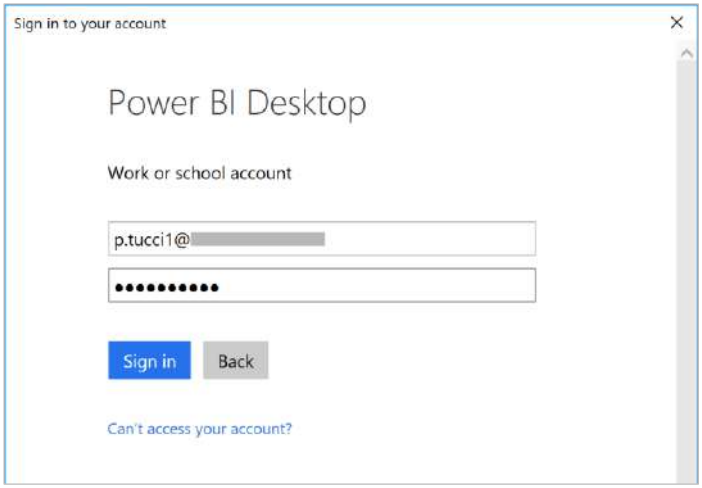


Figure 91: Publish—Sign in

At this point, the publication takes place.

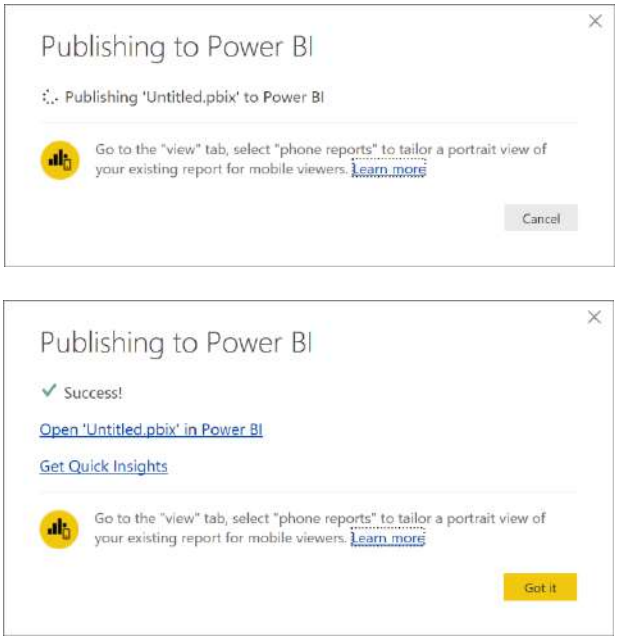


Figure 92: Finishing Publishing—sequence

Now we display Power BI through the browser to see the effect of our last operation.

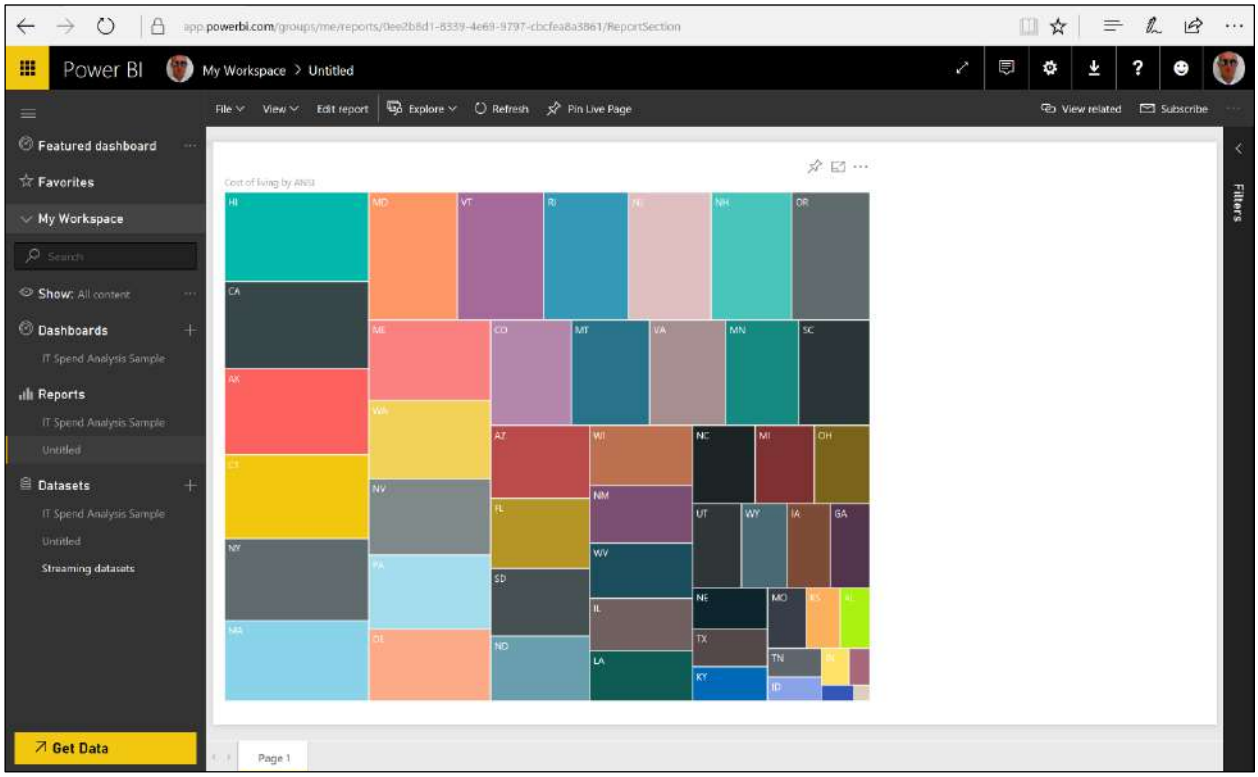


Figure 93: Publish Result

As we can see on the left panel, we have options of Datasets and Reports. The dataset is exactly the one we have built with the data model using the app. And the report containing the visualization we chose is also present.

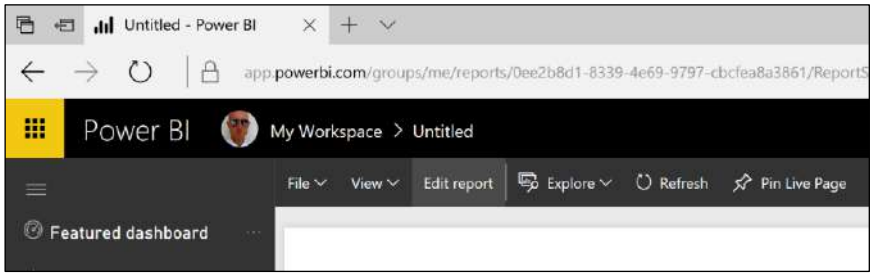


Figure 94: Power BI Service—Edit report

We can create a dashboard at any time.

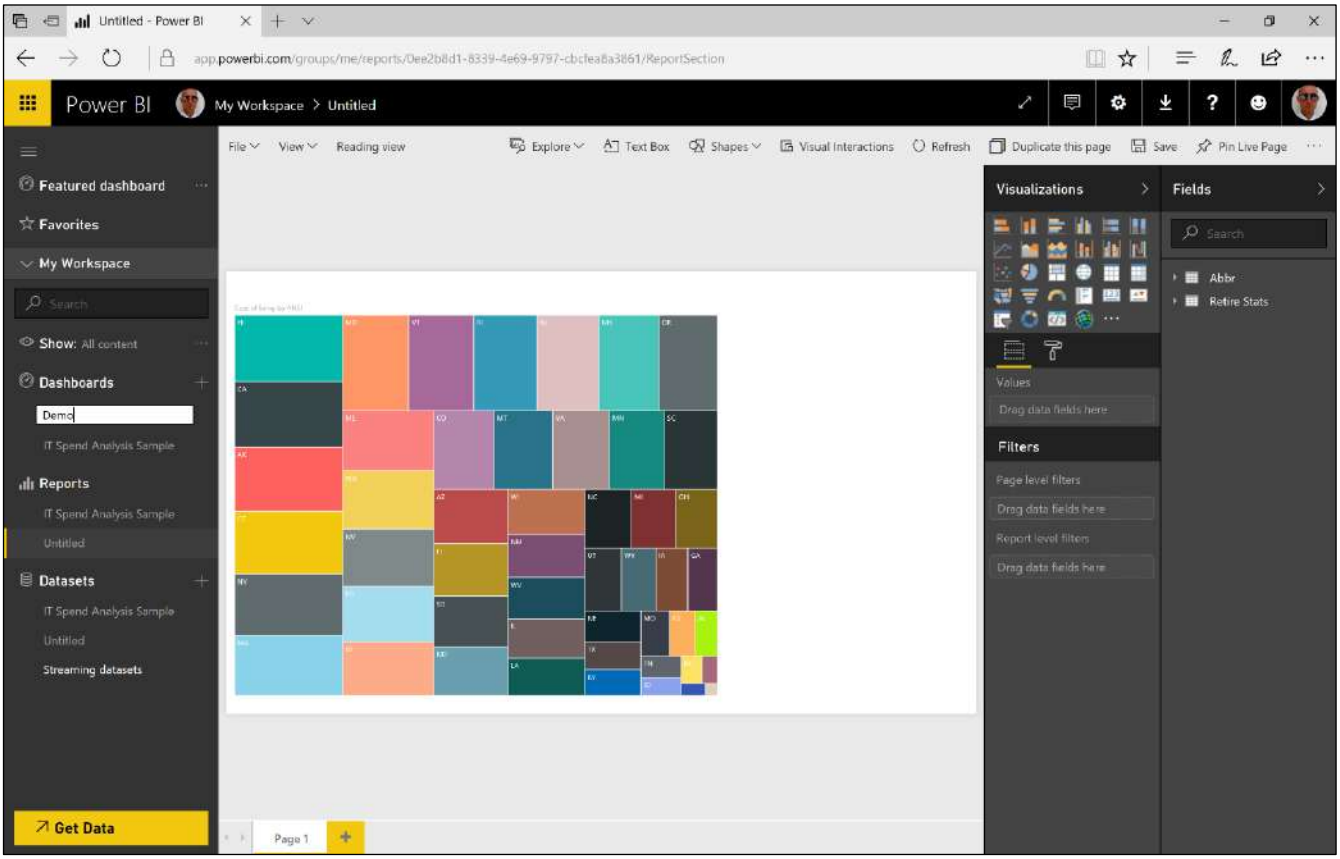
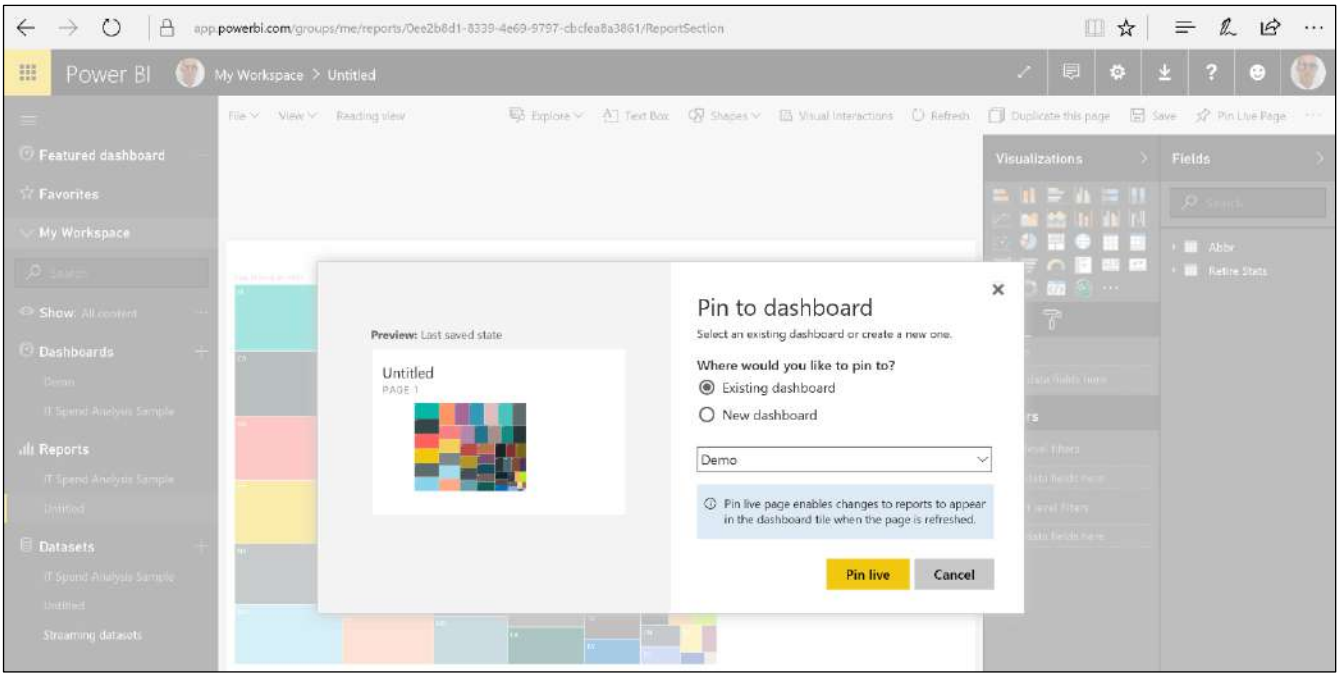


Figure 95: Power BI Service—create a dashboard

Figure 96 is a reminder that, as usual, we need to pin the visualization to the dashboard.



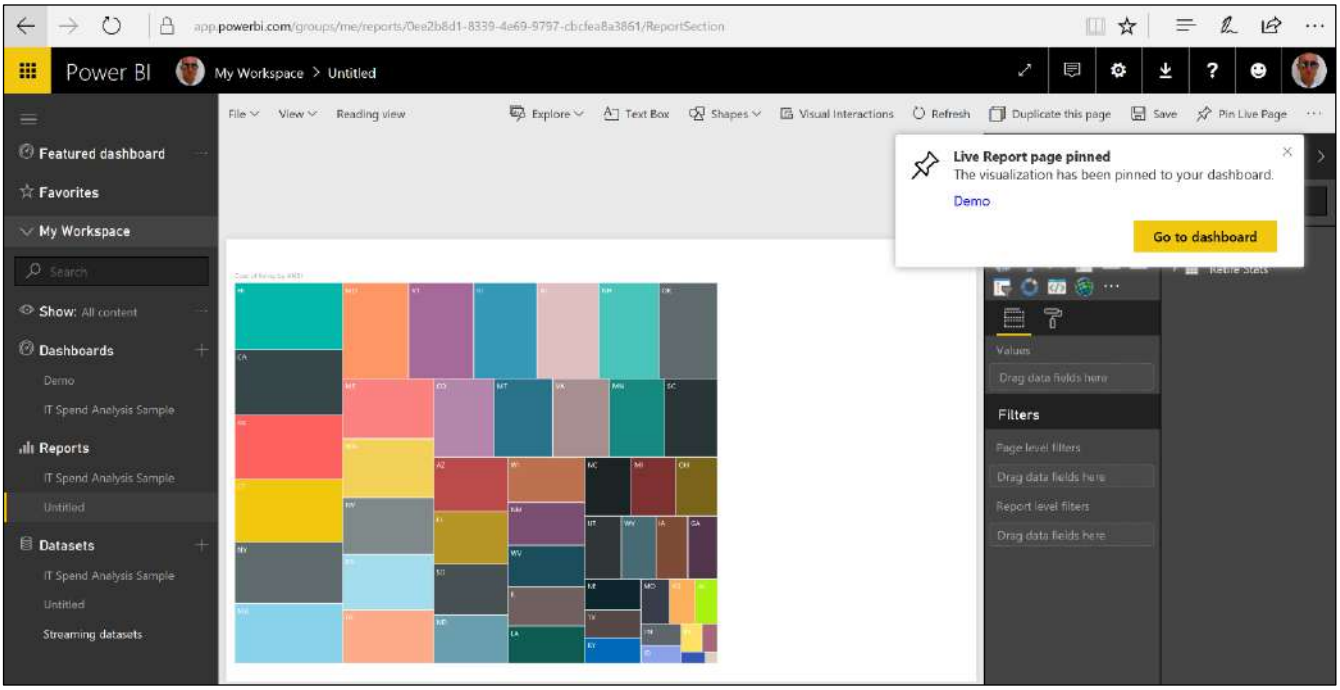


Figure 96: Pin to Dashboard—sequence

Our report, which has been built on-premises through the app, can be edited in order to enhance its information. Any modifications will remain available online.

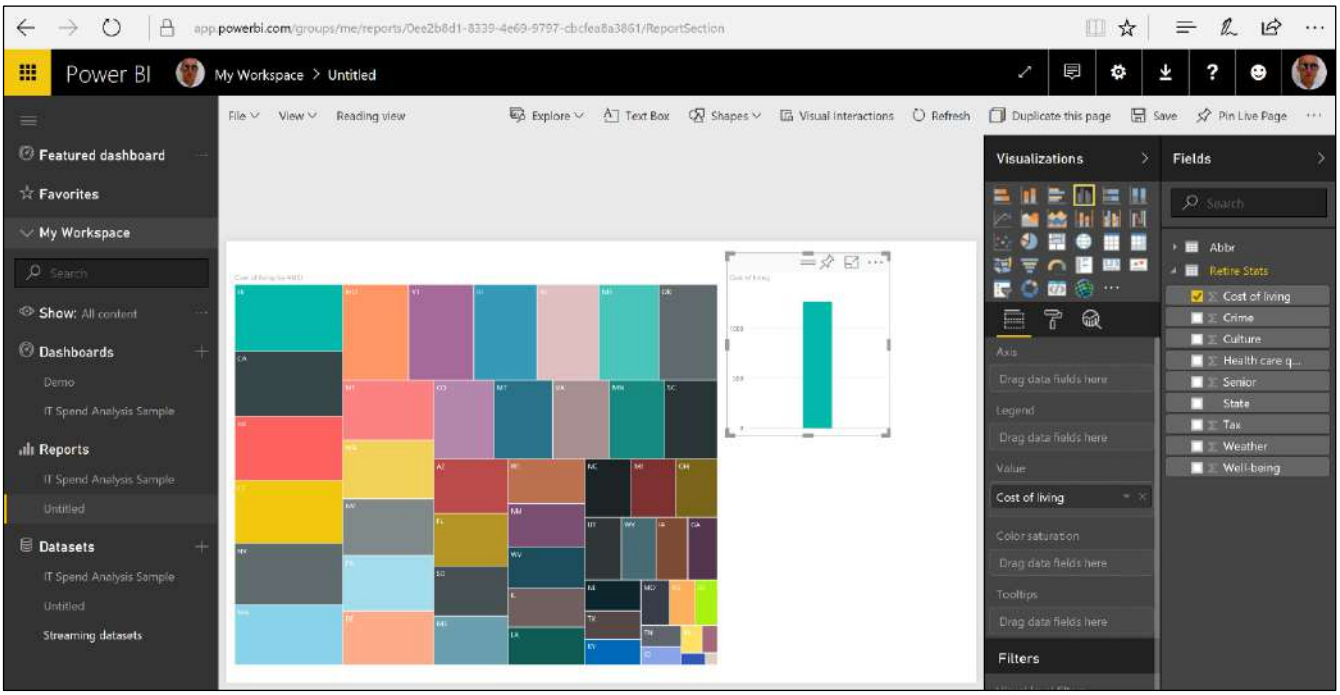


Figure 97: New Chart

Chapter 4 Mobile Apps

Power BI is available in apps for the following operating systems: Windows 8, Windows 10, Windows Phone, iPhone, and Android. The app versions allow us to connect to the Power BI service and explore the dashboards and reports available.

Windows 8 and Windows 10

Look for Microsoft Power BI in the Windows store.

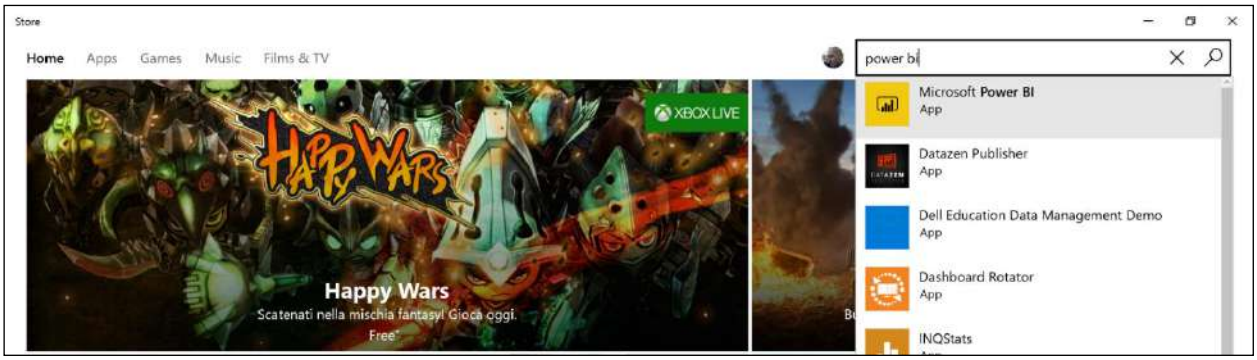


Figure 98: Windows Store

Install the app by clicking **Install**.

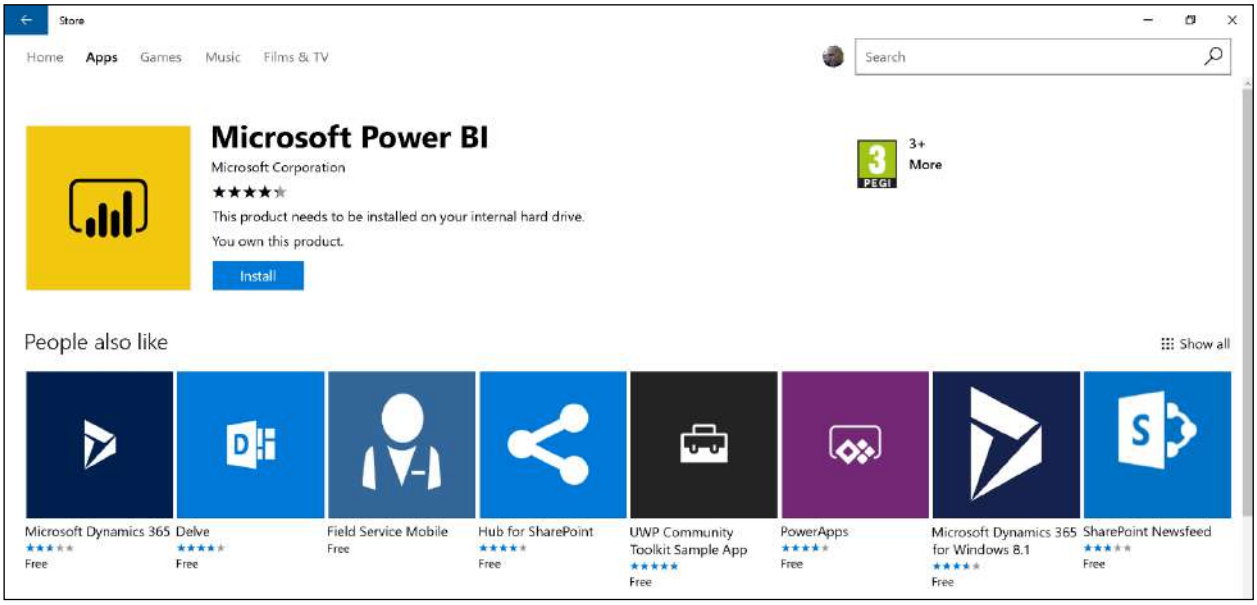


Figure 99: Power BI App Manifest

Run the application and click **Get Started**.

Figure 100: Power BI App

The first section leads to a selection of samples, as shown in Figure 101.

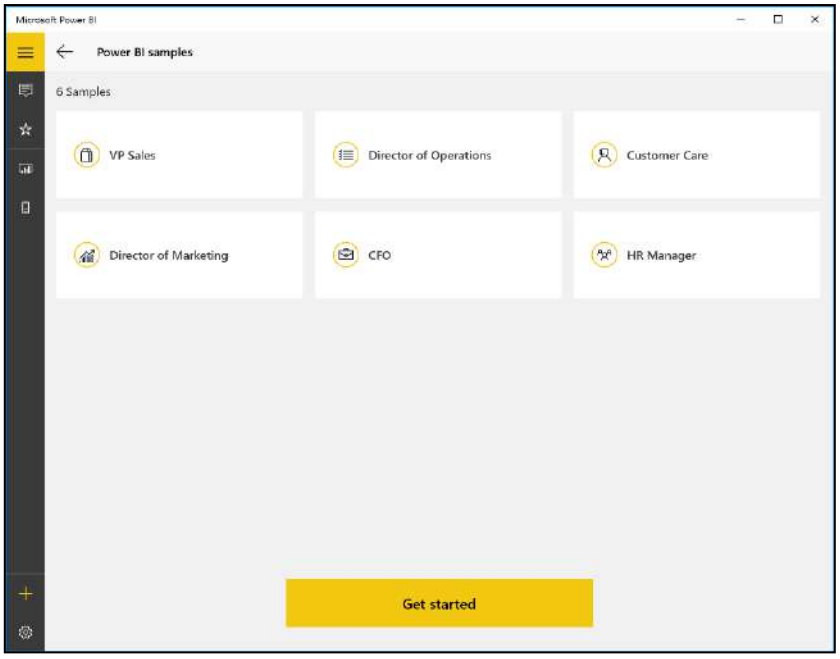


Figure 101: Power BI App

By proceeding with Get Started, we access the connection request. You can connect to the Power BI service or to a reporting sevice of SQL Server 2016. In our example, we will connect to the Power BI service.

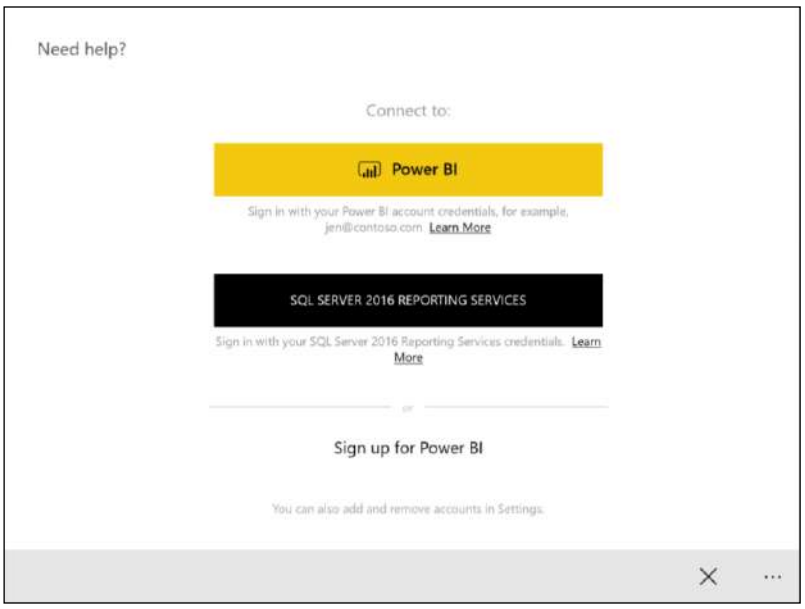


Figure 102: Power BI App

Supply the access credentials to the Power BI service, as shown in Figure 103.

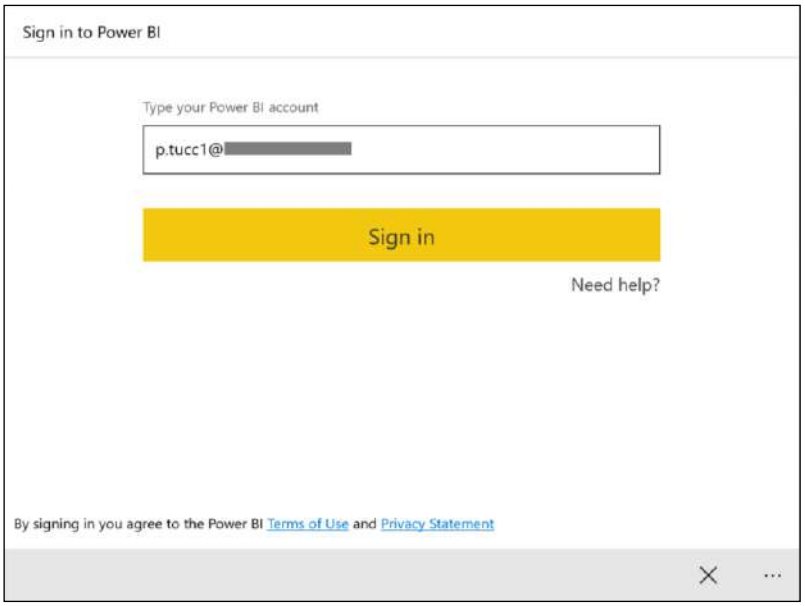


Figure 103: Power BI App

At this point, you will be redirected.

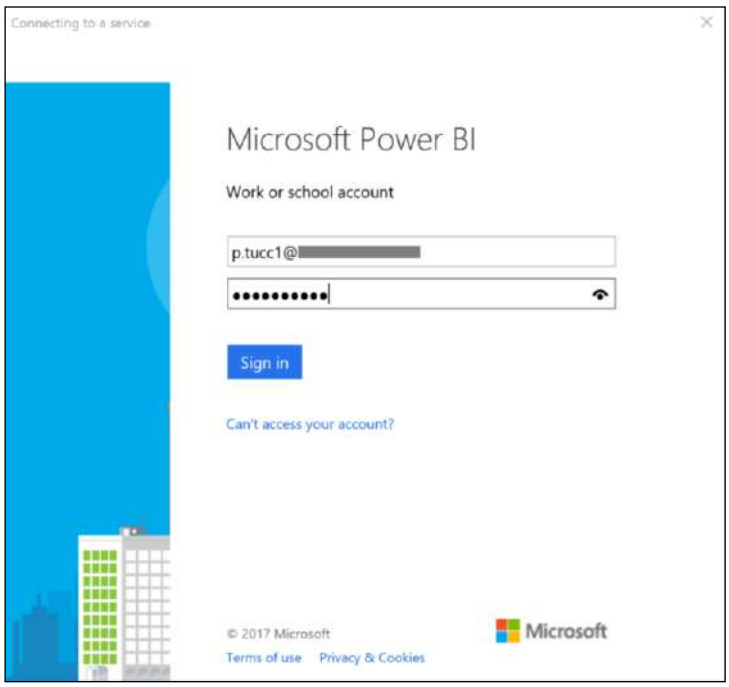


Figure 104: Power BI App

Figure 105 shows us that the authentication has been successful.

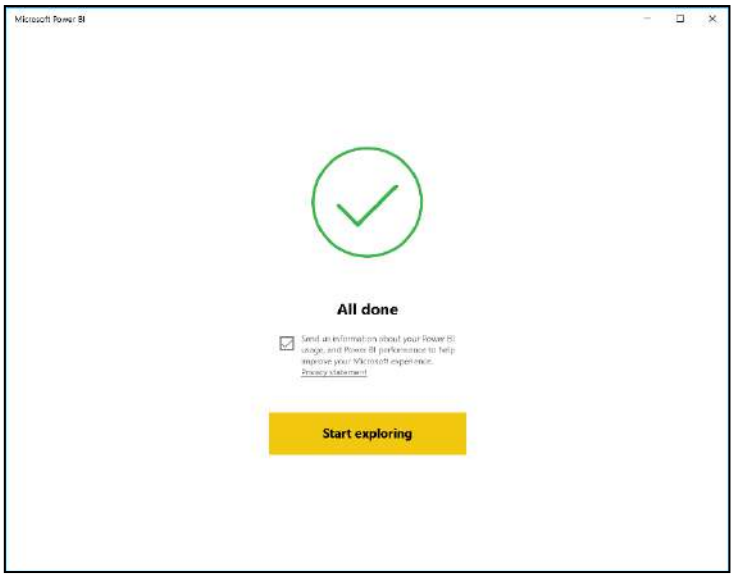


Figure 105: Power BI App

Figure 106 shows that we can explore the reports and dashboards present in our Power BI service.

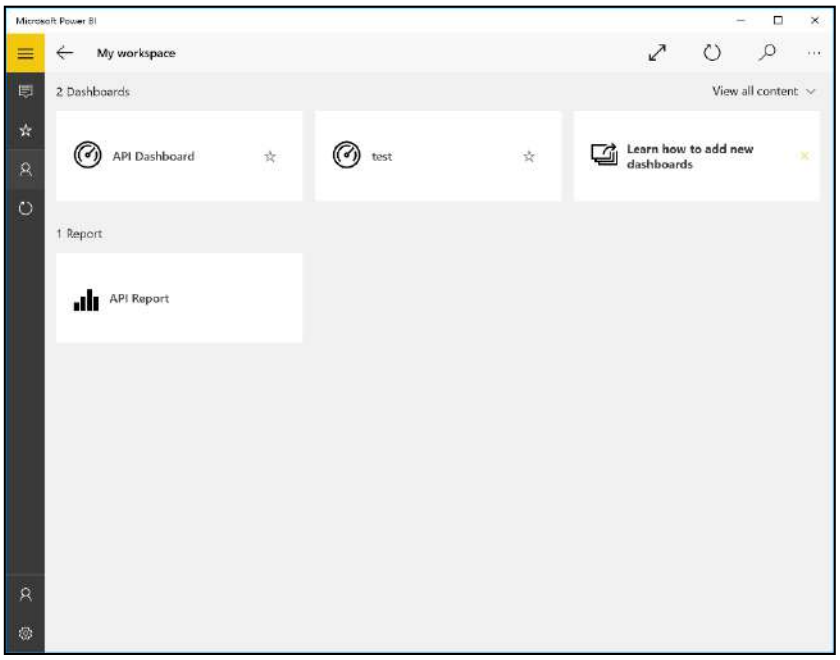


Figure 106: Power BI App

By selecting a dashboard, we can activate the contextual menu to invite people and share our dashboard, add it to the favorite dashboard list, or insert the dashboard into the start panel.

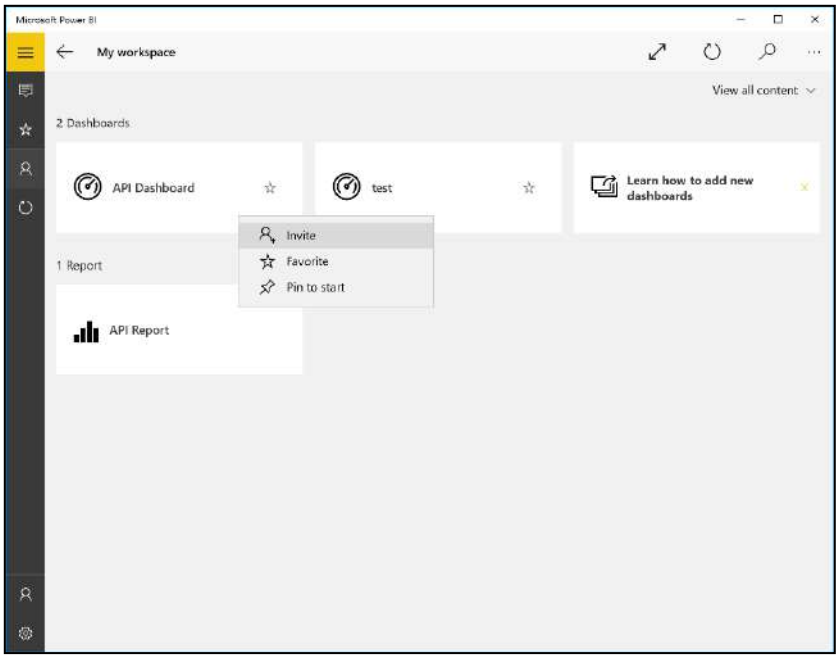


Figure 107: Power BI App

When we invite users who are external to the visualization of our dashboard, a context window appears in which we must specify the user or users with whom we want to share the dashboard. We must insert a contextual message while sending the invitation and specify if the recipient will be able, in turn, to share the dashboard with other users.

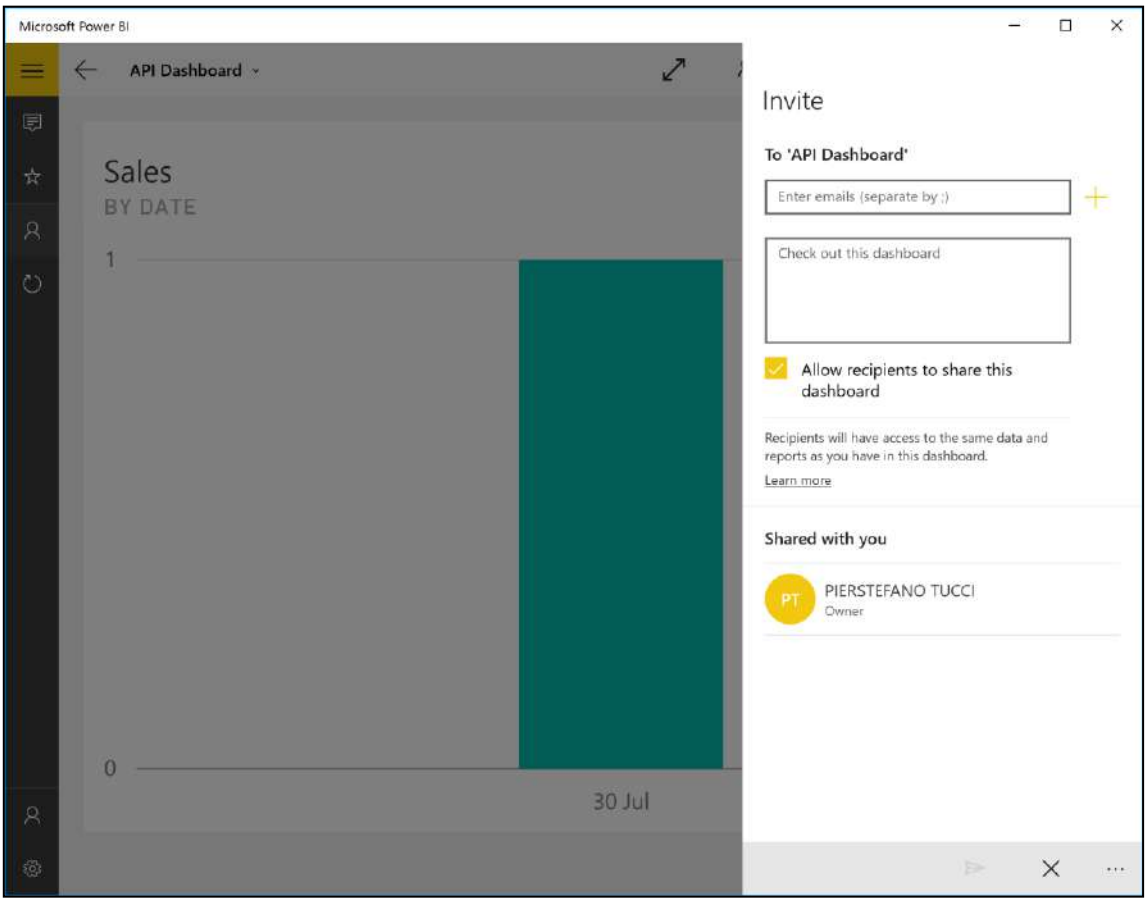


Figure 108: Power BI App—invite options

If you select a dashboard or a report, a series of icons will appear on the right side that will allow you to enlarge the chart/information you are consulting. You are now able to:

- Invite people.
- Add the dashboard or report to the favorites list.
- Insert the dashboard into the start panel.
- Update the dashboard or report you are displaying.
- Search for the dashboard or report in your Power BI service.

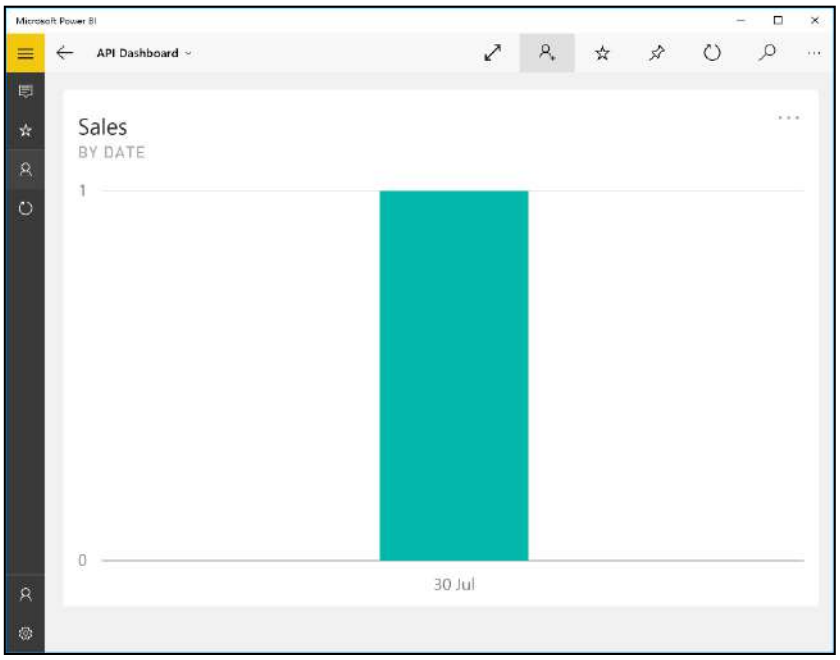


Figure 109: Power BI App

Figure 110 shows a research sample.

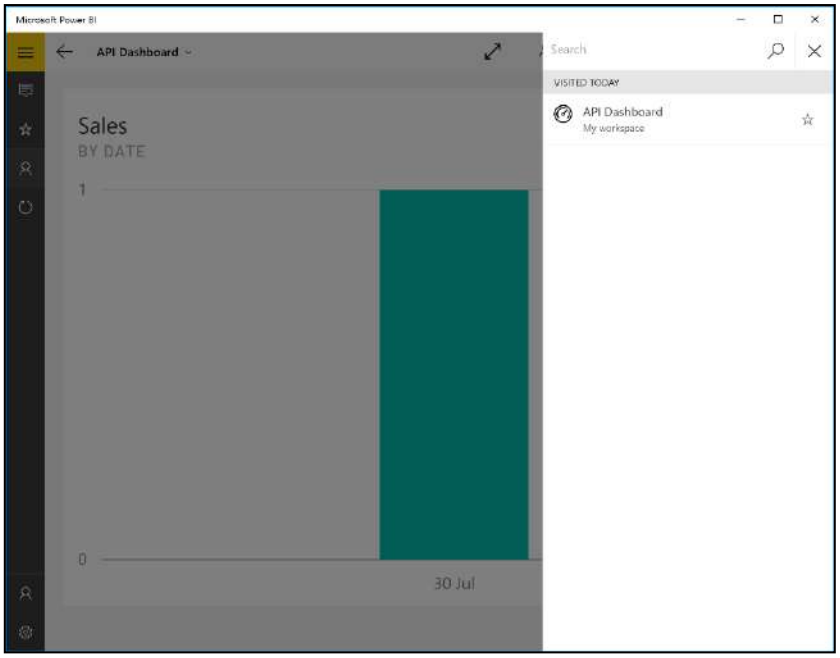


Figure 110: Power BI App

Next, go back to your dashboard and click the chart. The menu on the top right will change to a new set of icons. Enlarge to full screen, open **Report**, **Share**, **Search**.

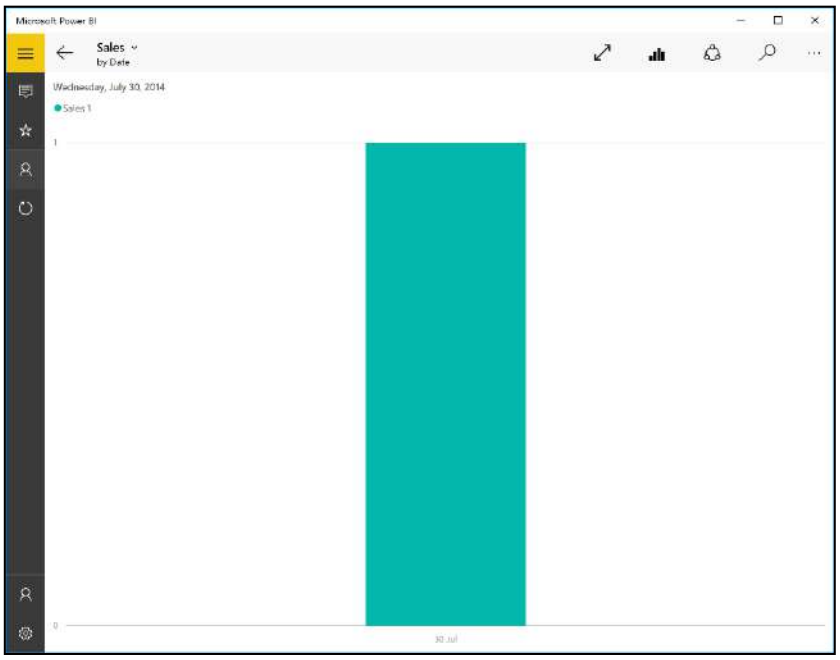


Figure 111: Power BI App

If you click Open Report, you can also base the selection of your data on filters, set up customized filters, and obtain new results.

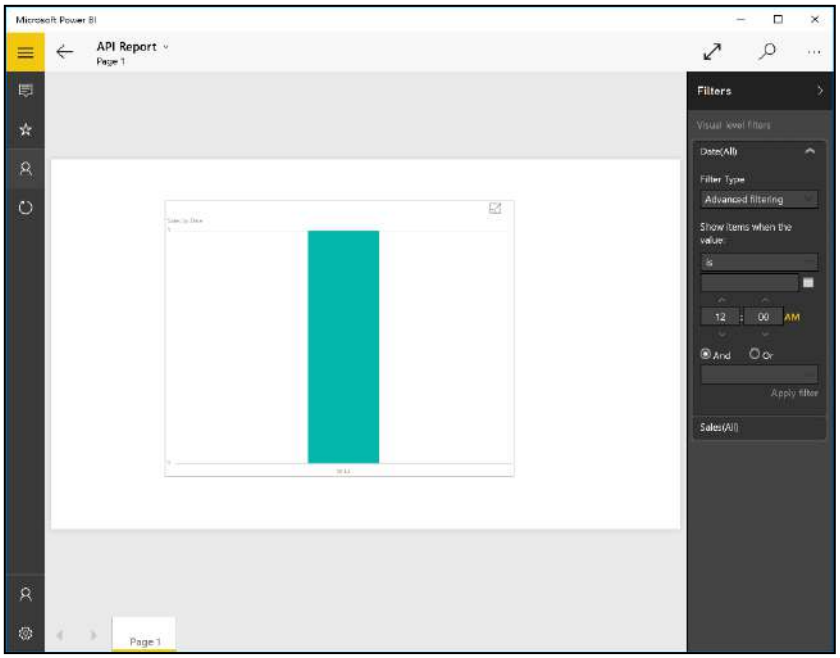


Figure 112: Power BI App

App for Android

From Google Play, you can install the application, authenticate to Power BI, and begin using the contents right away.

On the top right of the menu, you can add notes or explore the report.

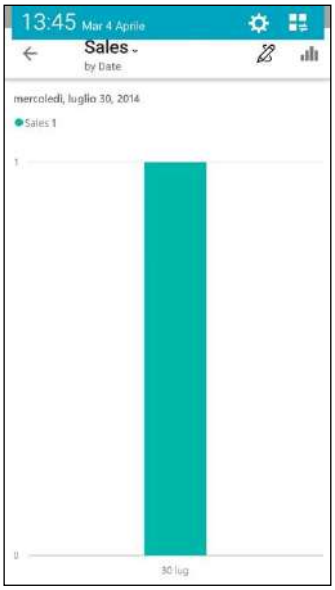


Figure 113: Power BI Android App

If you want to edit and add some notes, underneath the report you'll see the panel where you can insert your text, draw on the report, add emoticons, delete your notes, or share everything with other users.

Figure 114: Power BI Android App

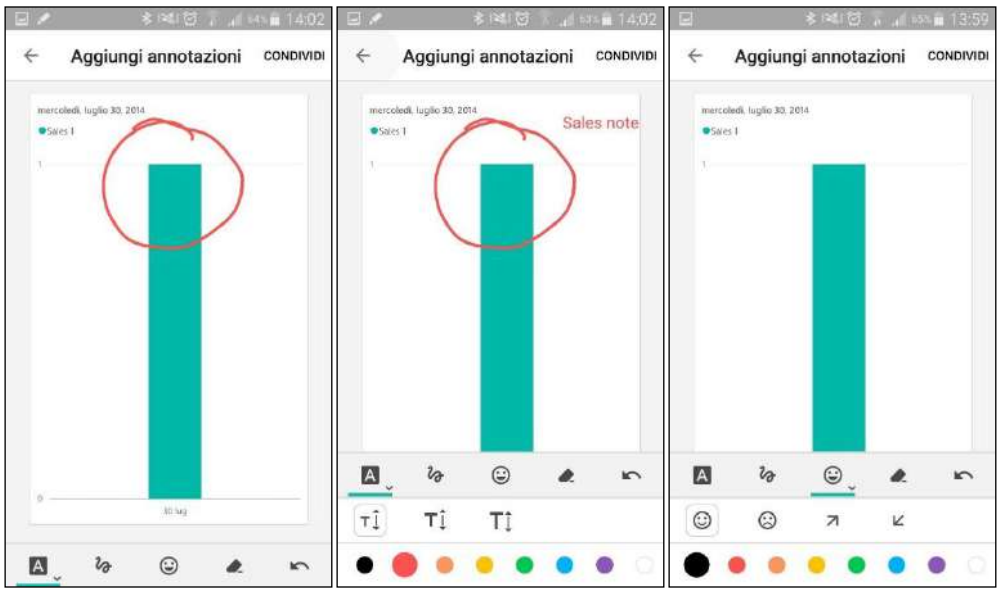


Figure 115: Power BI Android App—sequence

App for iPhone

From the store, install the application, authenticate to Power BI, and begin using the contents right away.

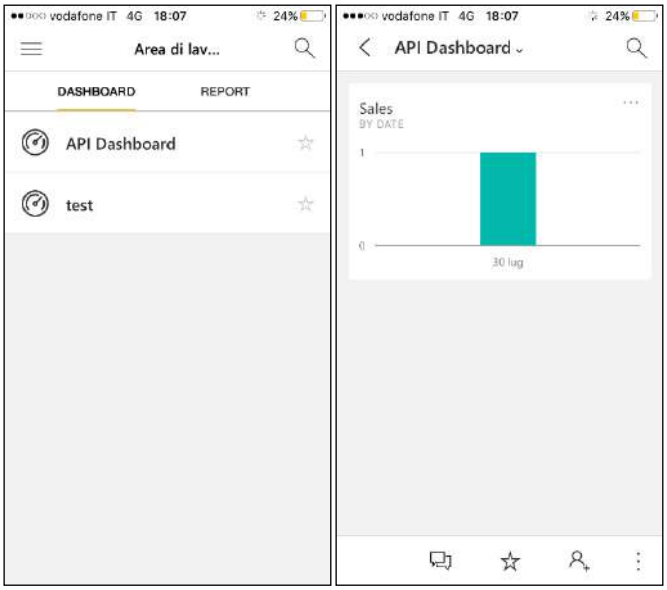


Figure 116: Power BI iPhone App—sequence

The procedure for the iPhone is the same as for the Android. If you go to edit the report and you want to add notes, you can insert text underneath the report where the panel is displayed. You can also draw on the report, add emoticons, delete your notes, or share everything with other users.

Figure 117: Power BI Android app

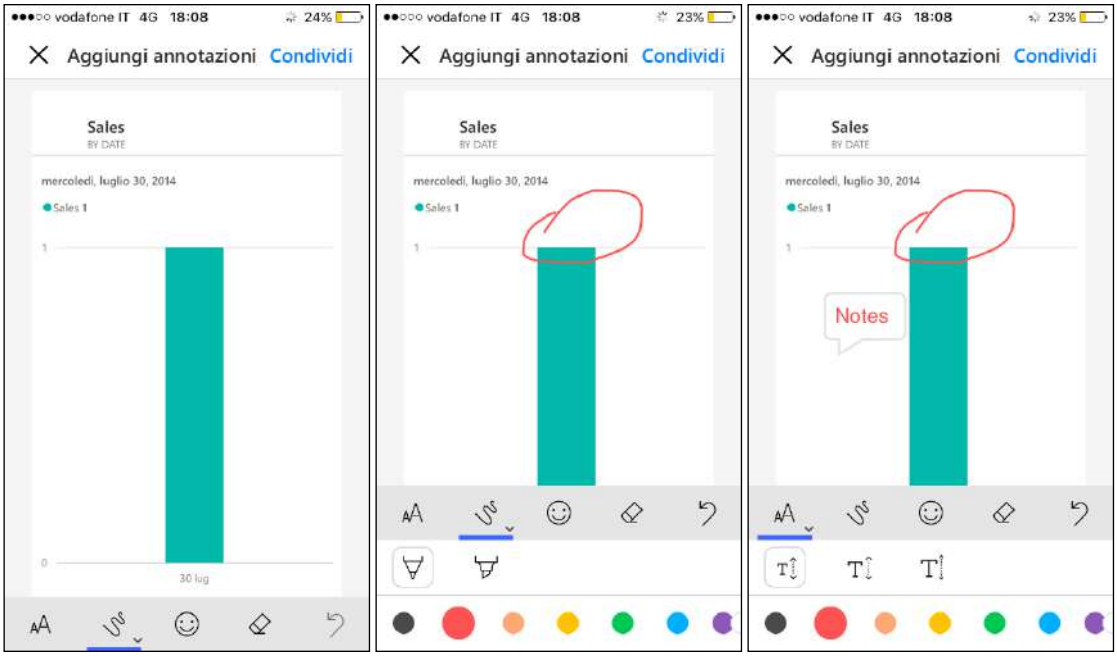


Figure 118: Power BI iPhone App—sequence

App for Windows Phone

From the store, install the application.

Figure 119: Power BI Windows 10 Mobile

Next, click **Get Started**, as in Figure 120.

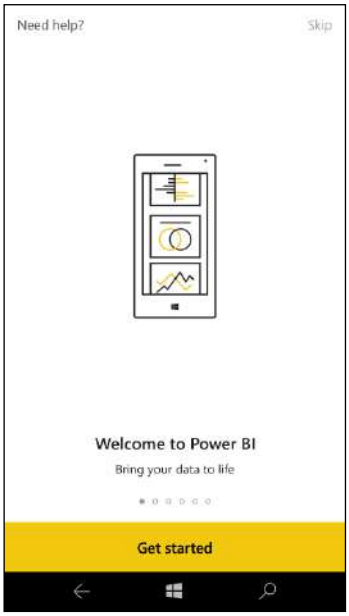


Figure 120: Power BI Windows 10 Mobile

Next, the connection type is requested. You can connect to the Power BI service or to a Reporting Sevice of SQL Server 2016. In our example, we will connect to the Power BI service, as in Figure 121.

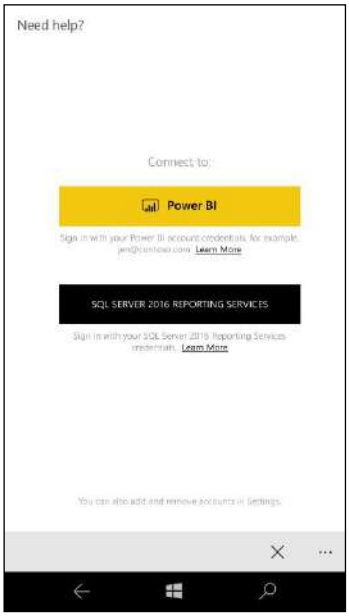


Figure 121: Power BI Windows 10 Mobile

Note that you will be asked to supply credentials in order to access the Power BI service.

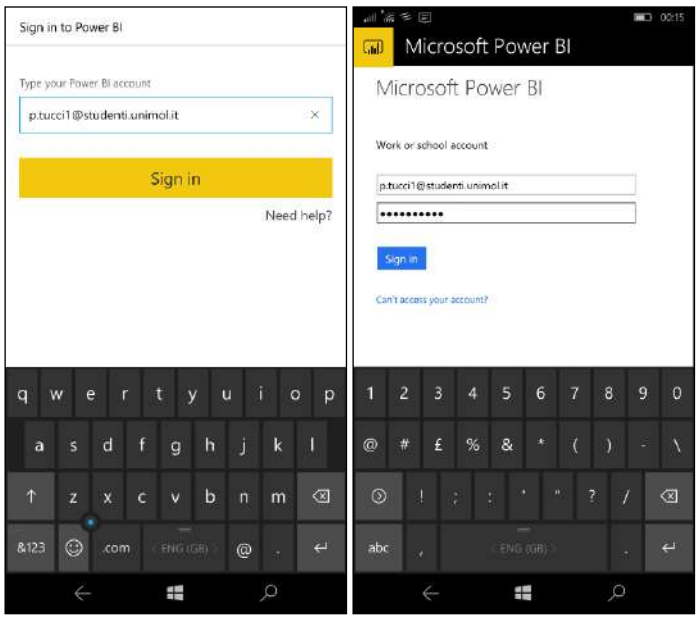


Figure 122: Power BI Windows 10 Mobile Sign In—sequence

At this point, the authentication has been successful and you can click the “Start exploring” bar.

If you prefer, you can also accept the boxed offer that reads: “Send us information about your Power BI usage and Power BI performance to help improve your Microsoft experience.”

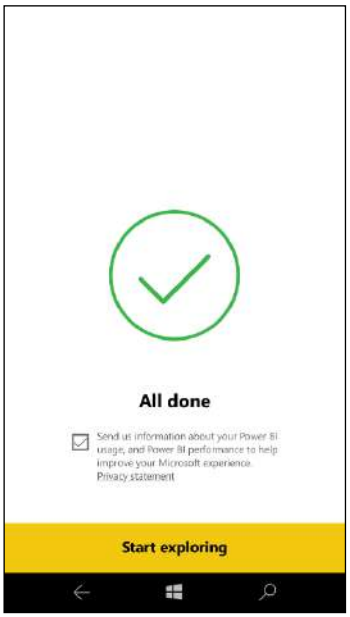


Figure 123: Power BI Windows 10 Mobile

Now you can begin using the dashboards or reports present on your service.

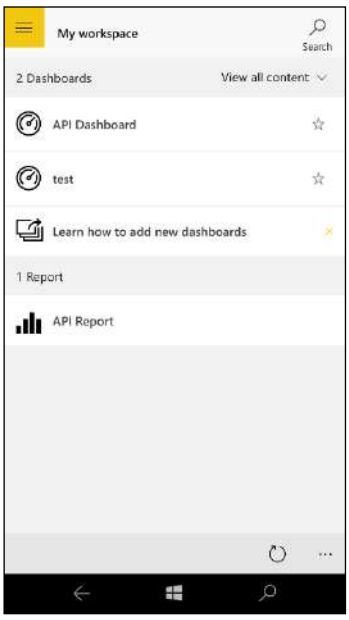


Figure 124: Power BI Windows 10 Mobile

On the top right, you'll see the search icon, which allows us to research dashboards and reports present on the service we are connected to.

Figure 125: Power BI Windows 10 Mobile

If you select a dashboard, you can invite other users to use the content, add the report to the favorites list, insert the dashboard into the start panel, or update the report.

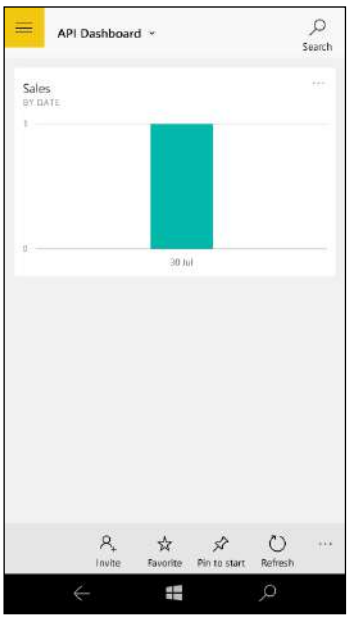


Figure 126: Power BI Windows 10 Mobile

If you invite other users to use the dashboard, you'll need to insert the email addresses of the users you want to include.

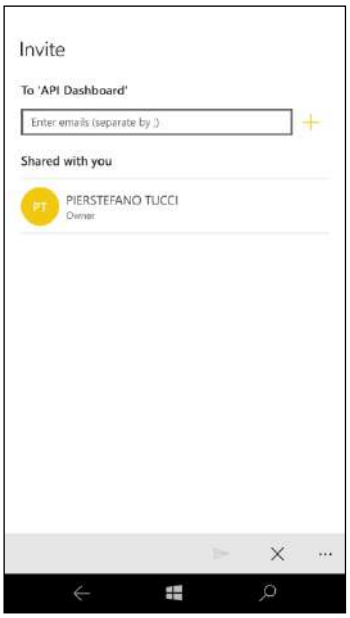


Figure 127: Power BI Windows 10 Mobile—invite users

If you select a report, you can share it or open it.

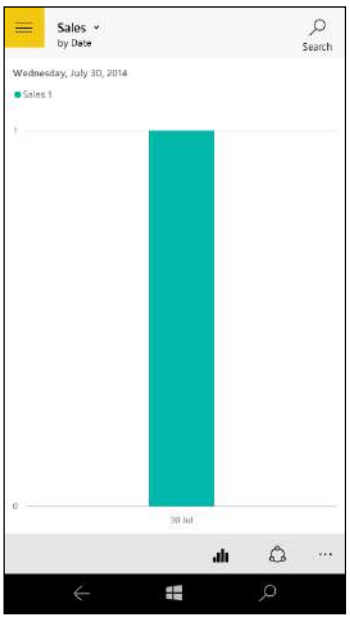


Figure 128: Power BI Windows 10 Mobile

In addition to the social profiles connected to the phone, the sharing option also offers the applications suitable to accept that information.

Figure 129: Power BI Windows 10 Mobile—Share

In our sample, we have chosen to share the report via email. The attachment inserted into the email by default is nothing but the snapshot image of the report we decided to share.

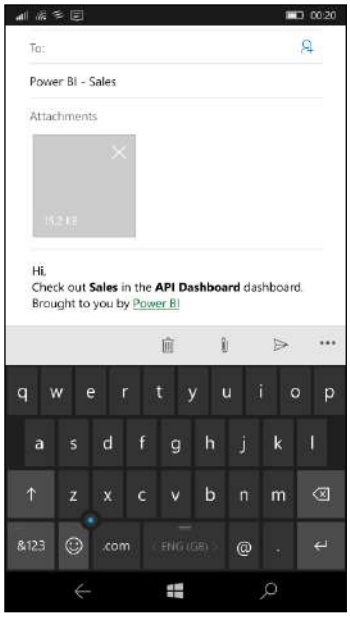


Figure 130: Power BI Windows 10 Mobile—share with email

Chapter 5 Solution Template

Now let's examine the latest sharing mode, which is the most innovative mode and offers us the ability to create content packs.

Figure 131: Power BI Settings Menu

In order to create a new content pack, first we go to the top right, click the Setting icon, and select **Create content pack**, as in Figures 131 and 132.

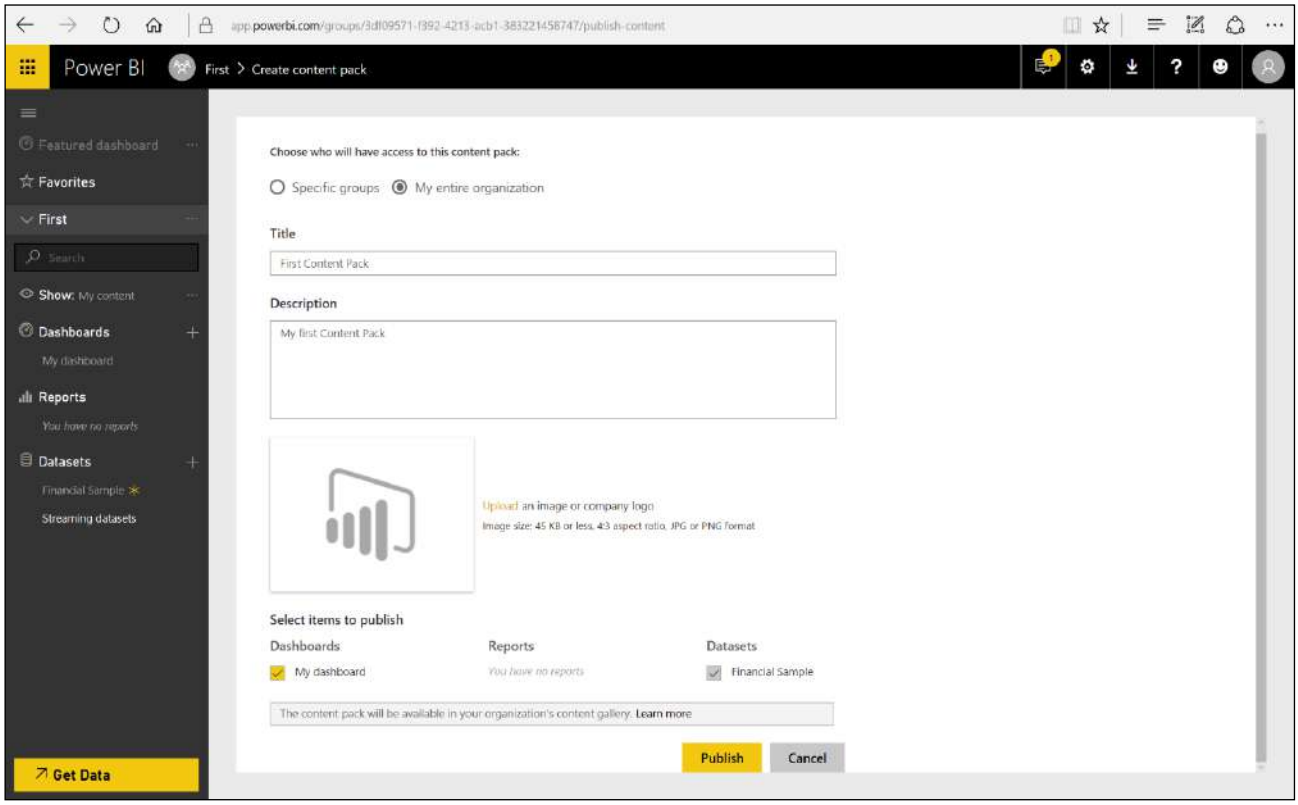


Figure 132: Create Content Pack

We can quickly decide if we want to share this with our entire organization or only with a specific group. In fact, we can specify:

- The user/users or the group/groups.
- The name of the content pack.
- A description of the content pack.
- The icon matched to the content pack.
- Any contents we want to share.

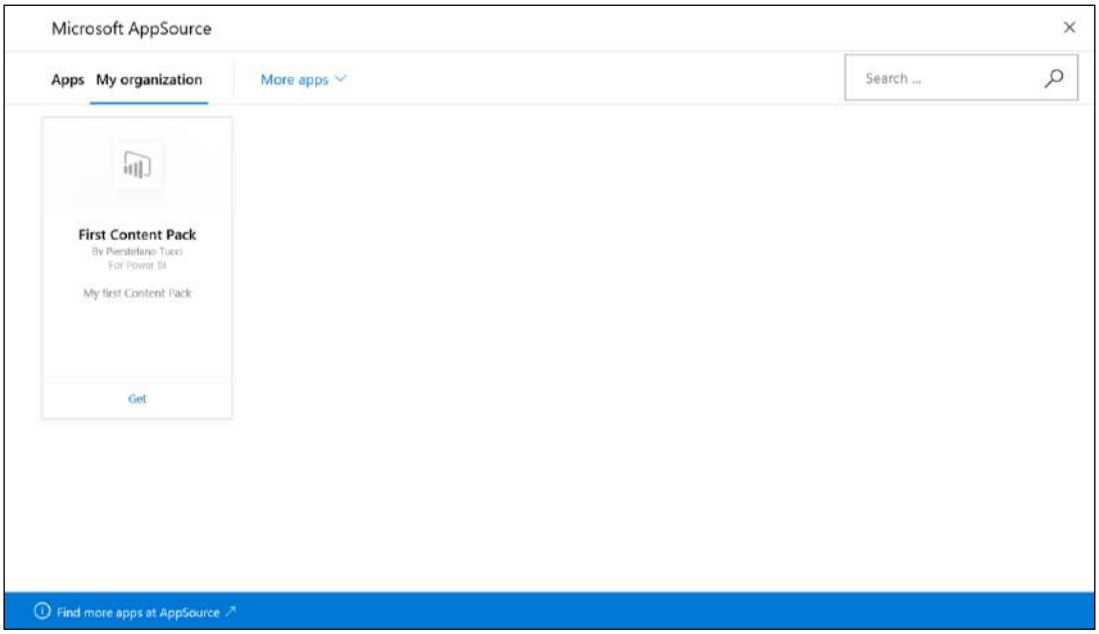
With any items we want to share, dependent reports and datasets can be selected, too. We can also add other datasets and reports without having problems related to interdependence restriction.

When we have finished the configuration, we can carry out the publication by clicking **Publish**.

Next, we can enter the Get Data as user, via the icon in Figure 133.

Figure 133: Get Data

As the graphics in Figure 134 show, we can next select **My organization**, where we can see all the previously published content packs. When we select the content pack, the owner is indicated. By clicking **Get**, we add that to the workspace.



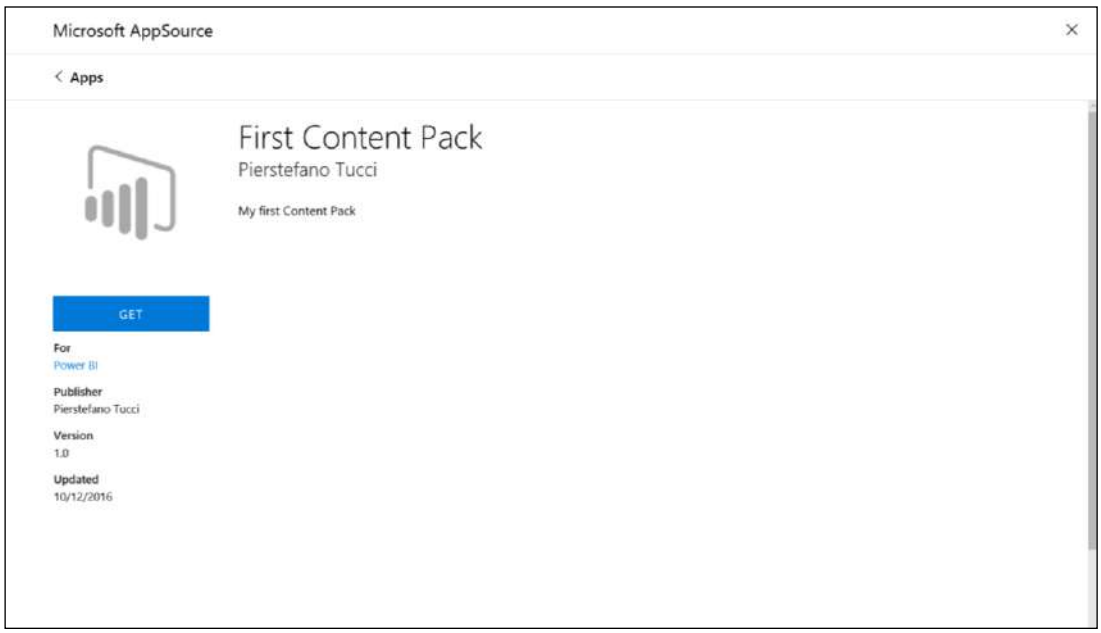


Figure 134: Microsoft AppSource—sequence

Next, we go to the top right, click the Settings icon, and select **View content pack** in order to delete a content pack. Next, we click the **Delete** entry corresponding to the content pack we want to delete.

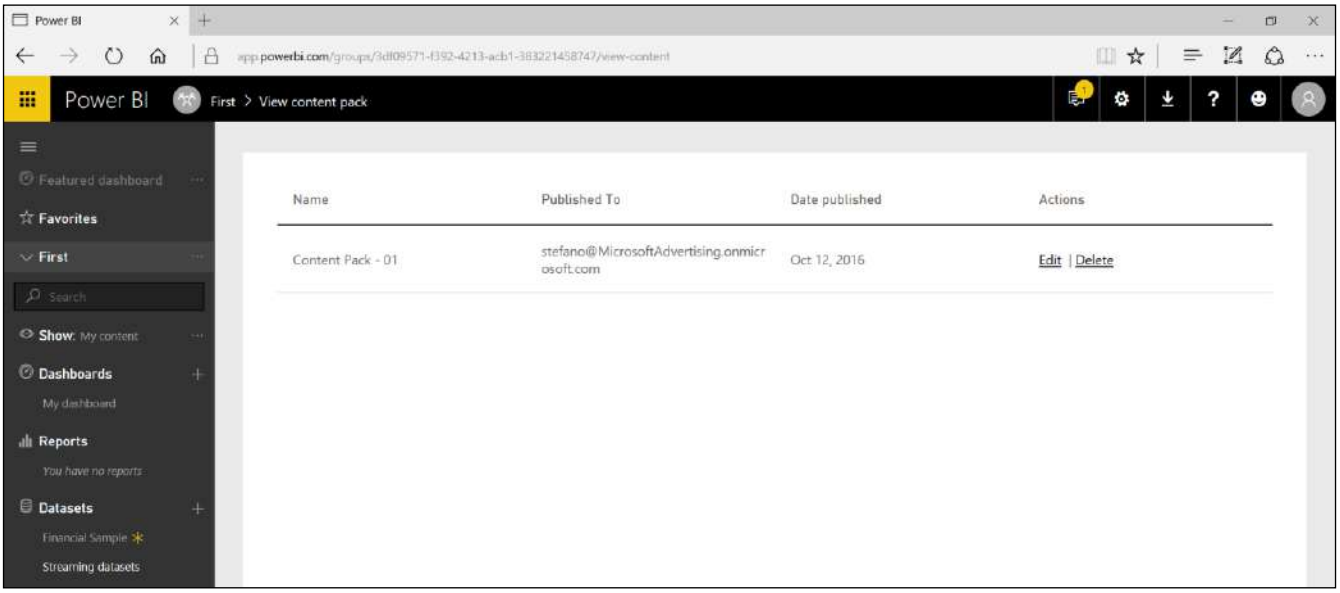


Figure 135: View Content Pack

Content pack

The content pack is one of the three sharing modes that allow us to gather together report and dataset dashboards as well as to share them within our organization.

The Software as a Services (SaaS) can be embedded by Power BI through content called connectors, which are made available directly on the Power BI portal and are simply content packs created specifically for this application. Once one of the connectors has been used and the connection data, such as the authentication account, has been provided, we can log onto source data like CRM Online, Google Analytics, and Salesforce. We can also use connectors to link to source data in the cloud, such as Azure SQL Database, Azure SQL Data Warehouse, and Spark on Azure HDInsight. (Note that Apache Spark is a generic engine for large-scale data storage with Azure HDInsight).

Notes on content packs:

- The connections require an existing subscription.
- Each content pack provides prebuilt dashboards and reports.
- They are updated nearly every month.

Database and data services are connectible, and a number of Azure (cloud-based) services offer the possibility of having a **direct connection**:

- Queries are sent to Azure, while the users export data and reports (the dashboard tiles are updated every 15 minutes).
- This functionality is for users who already have experience with databases and with the entities they connect to.
- It is available only with the Power BI Pro license.

Using Azure Online services such as Azure Stream Analytics, for example, leads us to interesting scenarios, like the possibility of creating RealTime dashboards with Azure Stream Analytics.

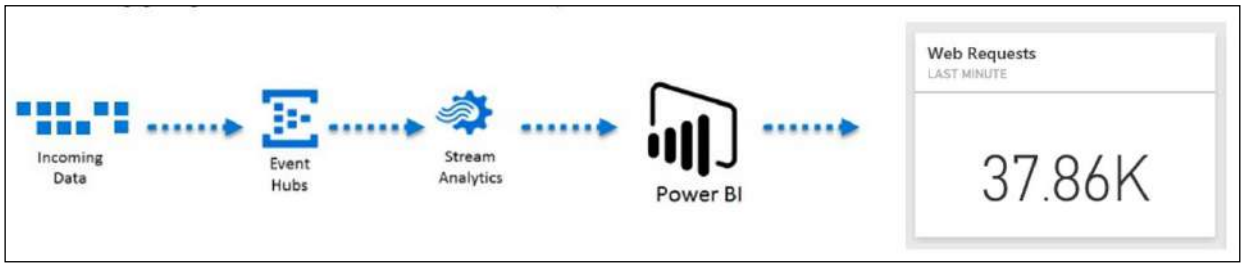


Figure 136: Stream Analytics Flow

In the Big Data world, we can collect a considerable quantity of data coming, for example, from sensors—in such a case, we have data flows that can be gathered through the Azure Event Hubs service. Afterward, we can carry out real-time analysis of the data by using the Azure Stream Analytics service. Finally, we can display the data by using Power BI and integrate it directly with a data source such as Azure Stream Analytics.

For example, we can imagine the data coming from a web application, perhaps an eCommerce website. In that case, we can collect data related to the web surfing of the user and we can analyze the temporal windows and display a summary of the analysis that can be the number of web requests received in the last minute. We can identify a temporal window because Azure Stream Analytics allows us to do real-time analysis. And because Power BI offers a visualization that is updated in real time, there is no refresh needed in this case.

We need to use connectors for the SaaS applications in order to connect Power BI to the online sources. So, we refer to the Get Data and we go to Services, where we can display the number of available connectors. We can use Microsoft Dynamics CRM Online and click **Connect**. We will be asked for the data services URL of Dynamics and also an authentication account. The result we receive will be very similar to the following reports because a report is created automatically.

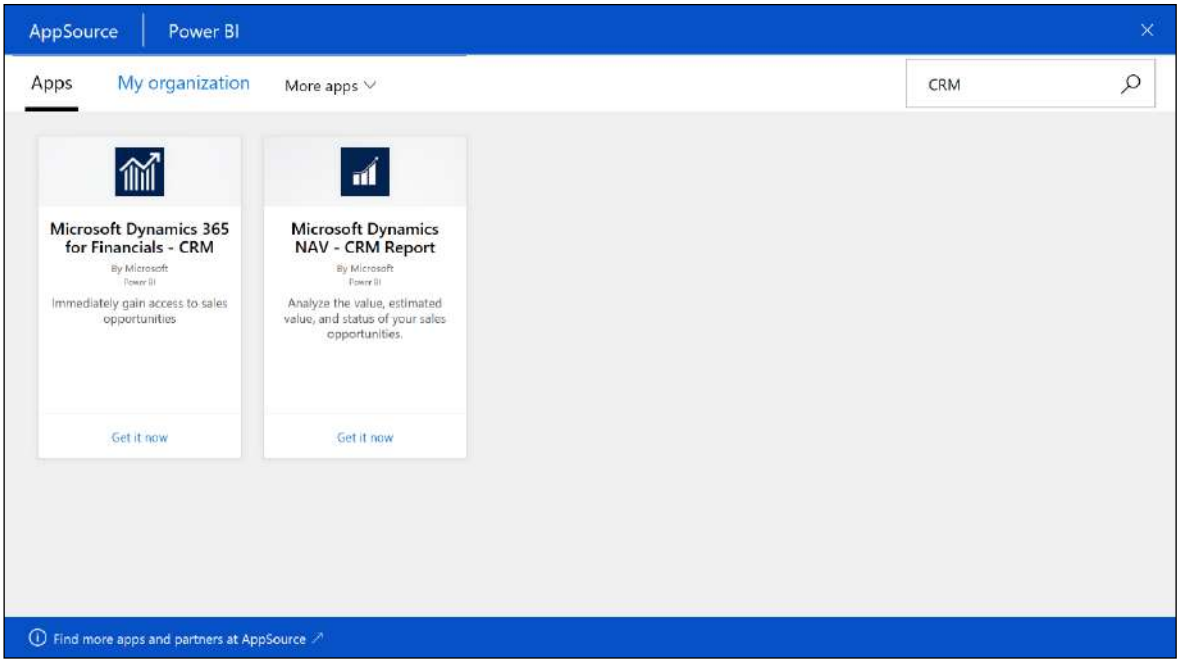


Figure 137: Microsoft AppSource

There are also several pages available for us in the report. The data comes from a CRM Online test. At the same time, we have a dataset available to implement a further report starting from our CRM Online.

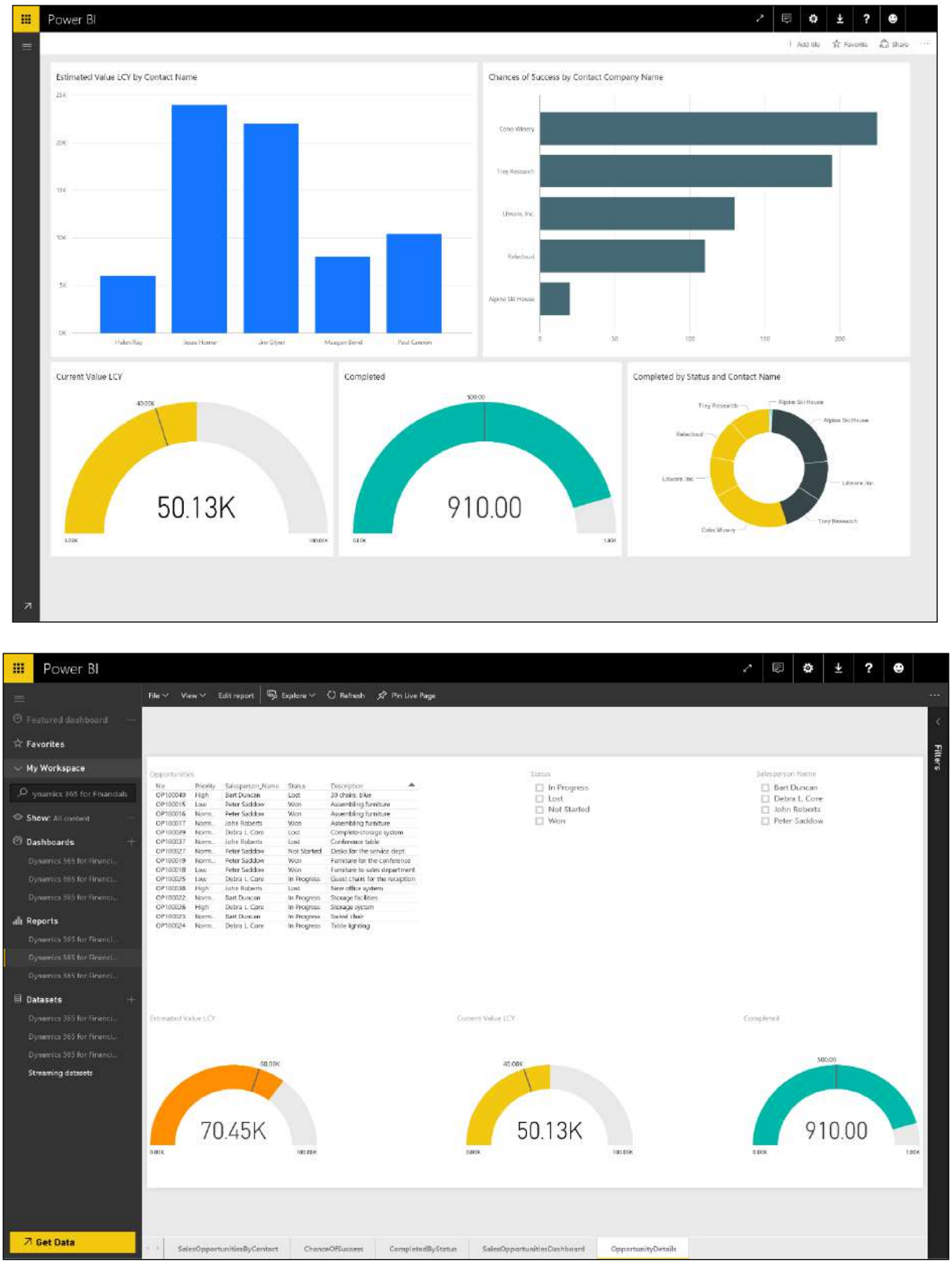


Figure 138: Dashboards—sequence

We can also use database online data sources—in particular, Azure SQL Database, Azure SQL Data Warehouse, SQL Server Analysis Services, and Spark on Azure HDInsight. In this case, we need only select **Azure SQL Database**, click **Connect**, then provide the server address, the name of the database, and information about the authentication. Automatically, we have a data model available that is updated about every 15 minutes.

Data refresh and schedule

There are data sources that need a refresh in order to keep the data updated in Power BI. For instance, an Excel workbook that connects to SQL Server needs to be updated through the refresh function. When the workbook is loaded on Power BI, we must carry out the refresh of the original data source, SQL Server, which is on-premises instead. We can also run the automatic refresh of the interval and decide then if we need to update our data sources in a scheduled or manual way. If we choose scheduled, we must decide if the update should occur on a daily or hourly basis.

Power BI allows the update of the online as well as on-premises data for the dataset loaded from:

- Power BI Desktop file
- Excel workbook (where Power Query or PowerPivot are used to connect to the data)

The data update can be on-demand or scheduled. If scheduled, the maximum frequency is:

- Power BI free license: daily
- Power BI Pro license: hourly

Figure 139: Refresh Frequency

We need to keep the credentials in order to keep the data updated in Power BI. Moreover, we must provide the access credentials for the data sources included in the dataset. The credentials are saved securely so that Power BI can connect on behalf of the user.

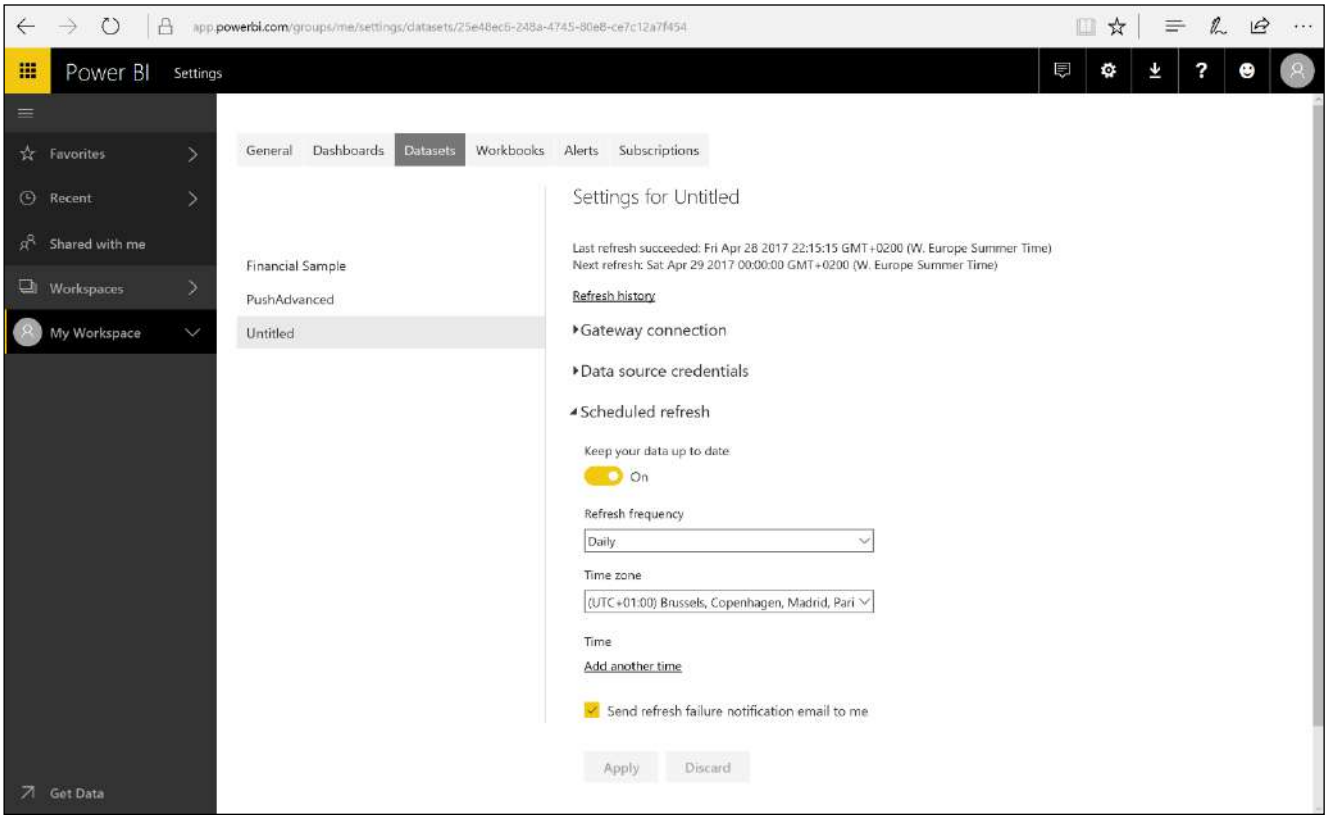


Figure 140: Datasets—Scheduled refresh

When we schedule the data refresh, Power BI should keep the credentials used to carry out the query of the data sources. We will be asked to provide the credentials, which will be kept securely in the cloud.

Now let's look at an example of how to implement an automatic refresh for the Excel workbook.

Typically, when working on-premises we can update the data. If we use an Excel workbook with a data source on the SQL Server, the database is AdventureWorks, and if we modify some data, the modification should also be mentioned in the reports and in the dashboard of Power BI.

Q	R	S	T	U	V	W	X
CreditCardID	CreditCardApprovalCode	CurrencyRateID	SubTotal	TaxAmt	Freight	TotalDue	Comme
16281	105041Vi84182		20565.6206	1971.5149	616.0984	23153.2339	
5618	115213Vi29411		1294.2529	124.2483	38.8276	1457.3288	
1346	85274Vi6854	4	32726.4786	3153.7696	985.553	36865.8012	
10456	125295Vi53935	4	28832.5289	2775.1646	867.2389	32474.9324	
4322	45303Vi22691		419.4589	40.2681	12.5838	472.3108	
806	95555Vi4081		24432.6088	2344.9921	732.81	27510.4109	

Figure 141: Excel Workbook

In Power BI Service, we add a new report page with a new visualization about the SubTotal column sum.

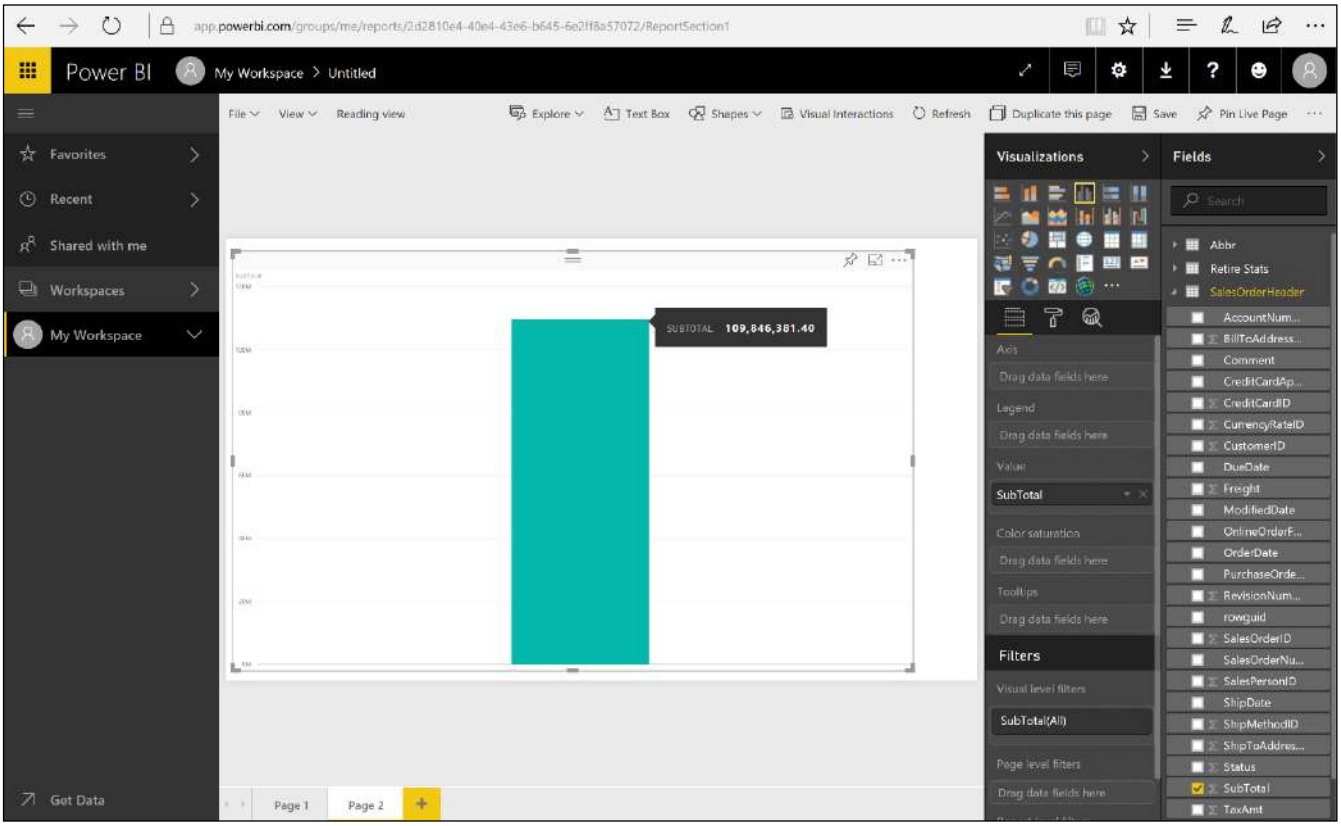


Figure 142: Power BI Service Report

Next, let's make a modification that cannot go unnoticed in our report. We'll update the SubTotal field from 20565.6206 to 205656.206. Now let's open the Excel workbook and run a data refresh.

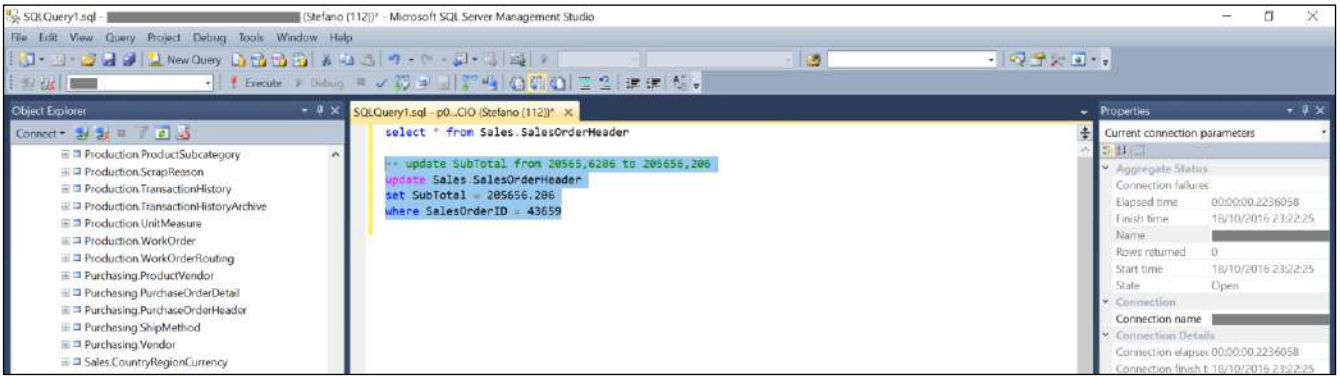


Figure 143: Update Query in Microsoft SQL Server Management Studio

After waiting, we should see that everything works correctly.

Figure 144 shows an update to the dataset of the Excel workbook, achieved by running the manual refresh.

	Q	R	S	T	U	V	W	X
	CreditCardID	CreditCardApprovalCode	CurrencyRateID	SubTotal	TaxAmt	Freight	TotalDue	Comme
5	16281	105041Vi84182		205656.206	1971.5149	616.0984	23153.2339	
5	5618	115213Vi29411		1294.2529	124.2483	38.8276	1457.3288	
5	1346	85274Vi6854	4	32726.4786	3153.7696	985.553	36865.8012	
5	10456	125295Vi53935	4	28832.5289	2775.1646	867.2389	32474.9324	
5	4322	45303Vi22691		419.4589	40.2681	12.5838	472.3108	
5	806	95555Vi4081		24432.6088	2344.9921	732.81	27510.4109	

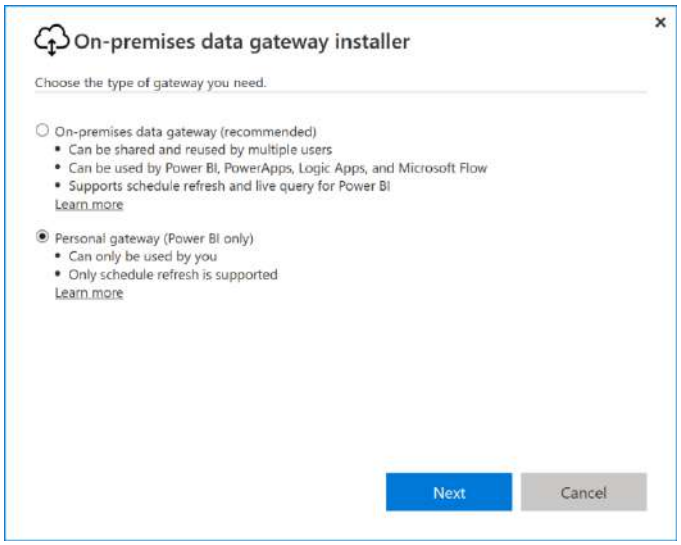
Figure 144: Excel Workbook

Power BI Gateway

After considering the previous example, we need to configure the Personal Gateway in order to refresh the Excel data. In the case of Excel workbooks, which have no connections to external sources, the error message will not appear and the data will be updated without any problems.

To install the Personal Gateway, we need to download the program by accessing the gateway pages at https://powerbi.microsoft.com/en-us/gateway and install it. Once the installation is finished, we can set up the Personal Gateway.

Figure 145 shows the steps for installing and configuring our Personal Gateway.



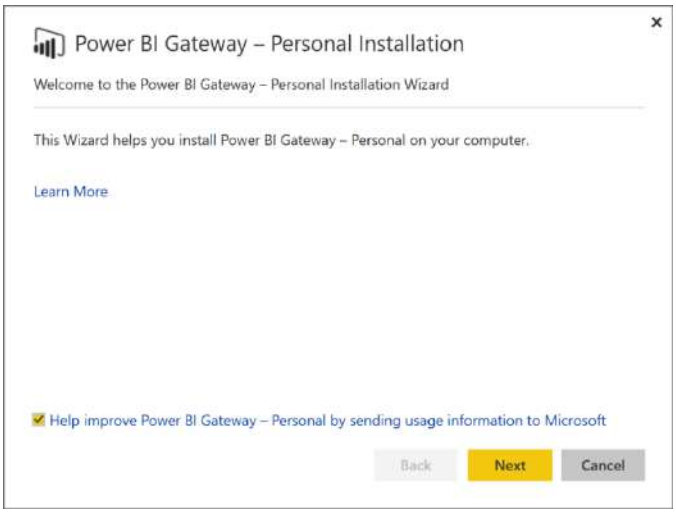


Figure 145: Personal Gateway Installation—sequence

We start with the Personal Installation Wizard. We'll see some advertising, then we can click **Next**.

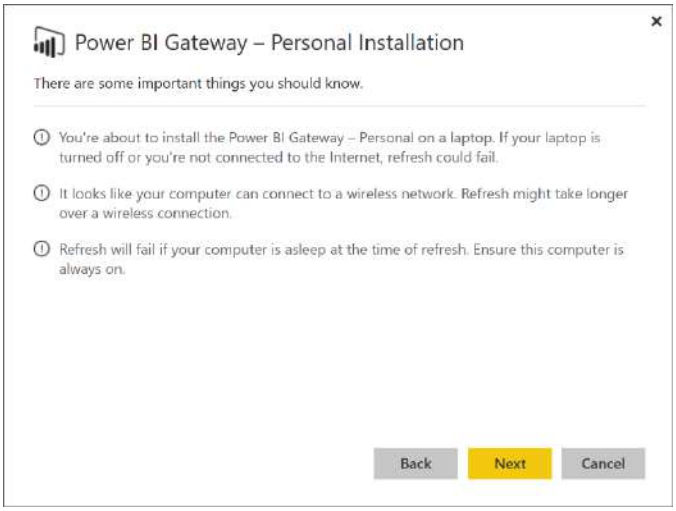


Figure 146: Personal Gateway Installation

As Figure 147 shows, you'll be asked to accept the license agreement. Do so.

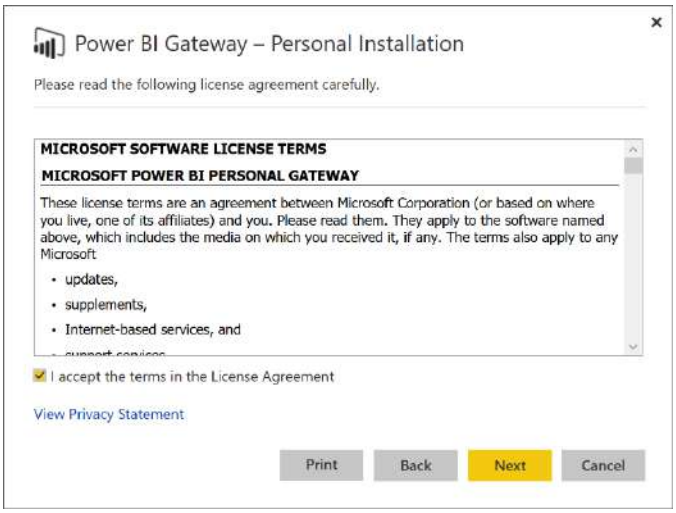


Figure 147: Personal Gateway Installation

Next, let's select the Installation folder.

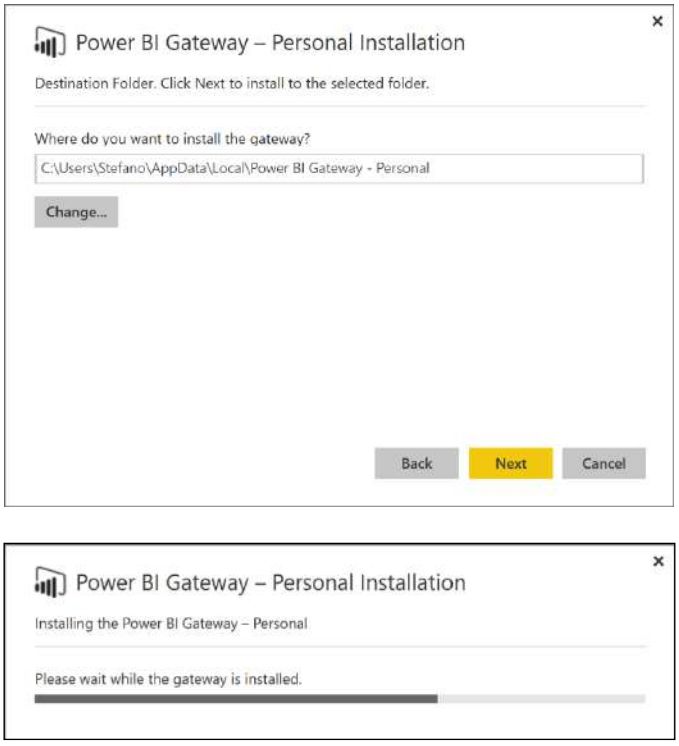


Figure 148: Personal Gateway Installation—sequence

Next, we sign in and provide our credentials.

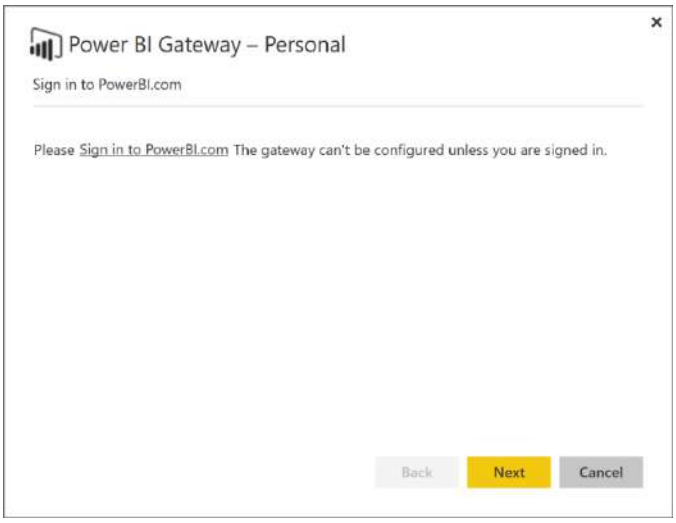


Figure 149: Personal Gateway - Sign in

Figure 150 shows the sign-in process.

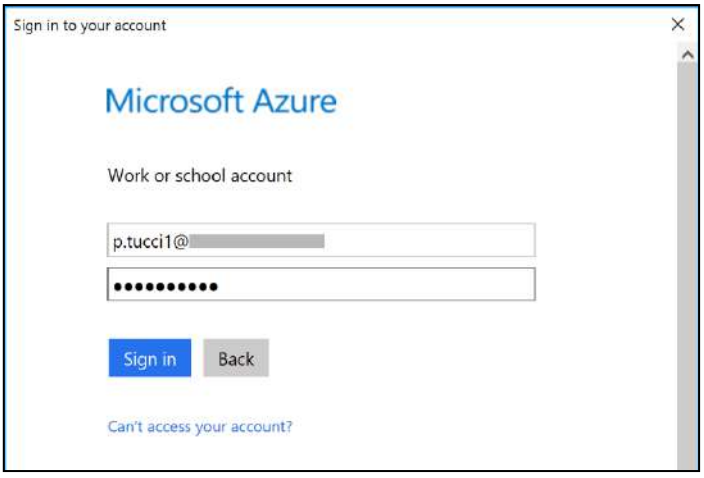


Figure 150: Sign In

This is how the Personal Gateway connects to the dedicated Azure Services Bus channel in order to establish communication with Power BI. When the authentication has occurred successfully, the Personal Gateway is online.

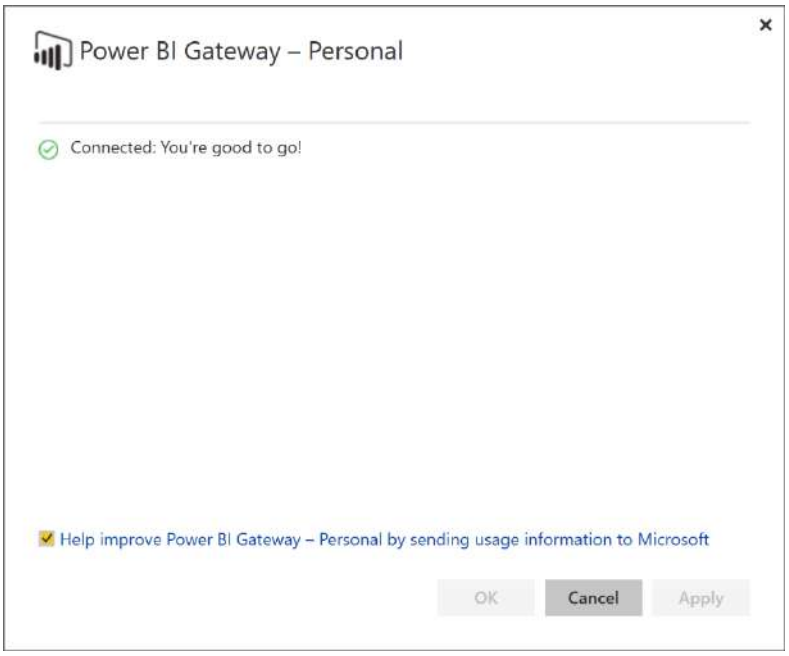


Figure 151: Personal Gateway—connected status

We also need to provide the credentials that will be used by Power BI to carry out the refresh. We can check the status on Power BI Service.

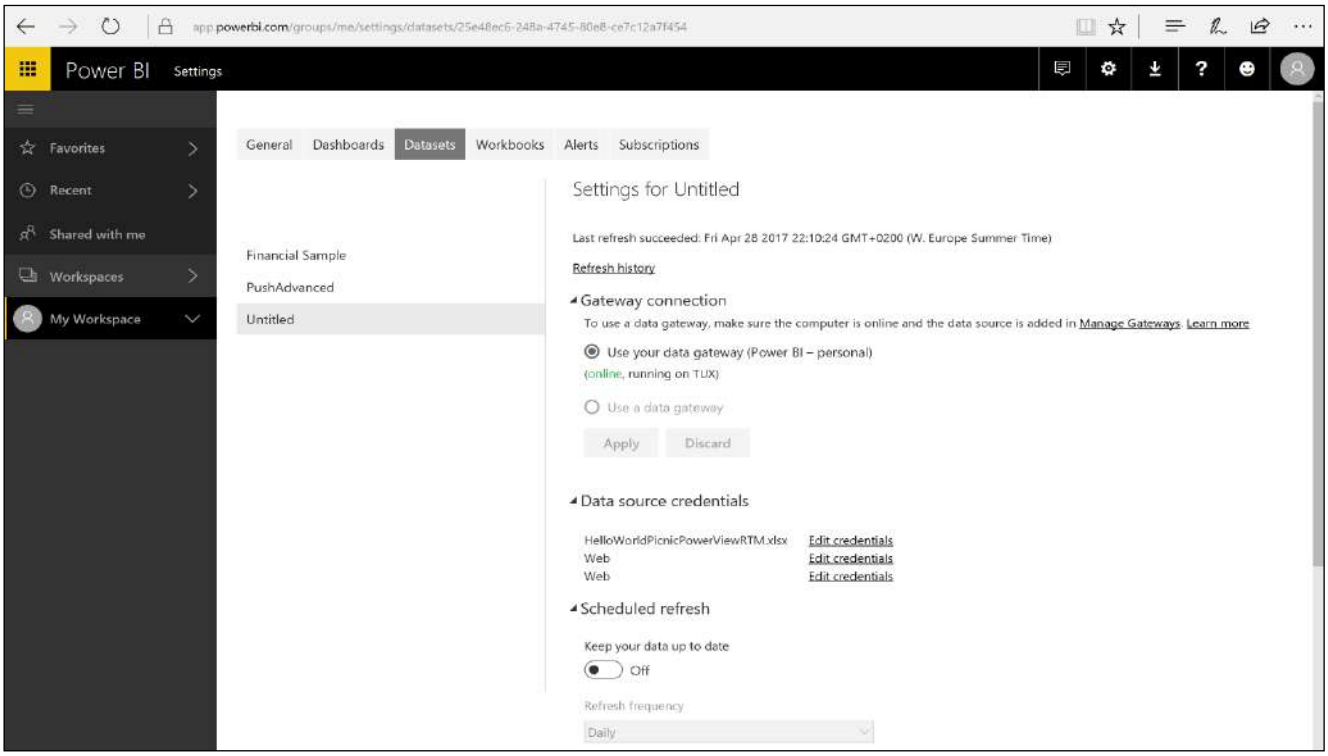


Figure 152: Gateway Connection Settings

In this case, we use the Windows Authentication, which means it is not necessary to provide the credentials because the Personal Gateway service credentials will be used. At this stage, the refresh is set up and it can be scheduled.

Note that we can add different timetables because we are using a Power BI Pro License; conversely, if we use a Power BI Free license, we can update only once per day.

We can go back to the dataset, click the data source, which contains the connection to the external SQL data, and run “Refresh now.”

Therefore, we can use the Personal Gateway to complete a data update even if the data sources are on-premises.

Next, after the dataset refresh, we can go to the report and see data updates. The SubTotal is displayed, as in Figure 153.

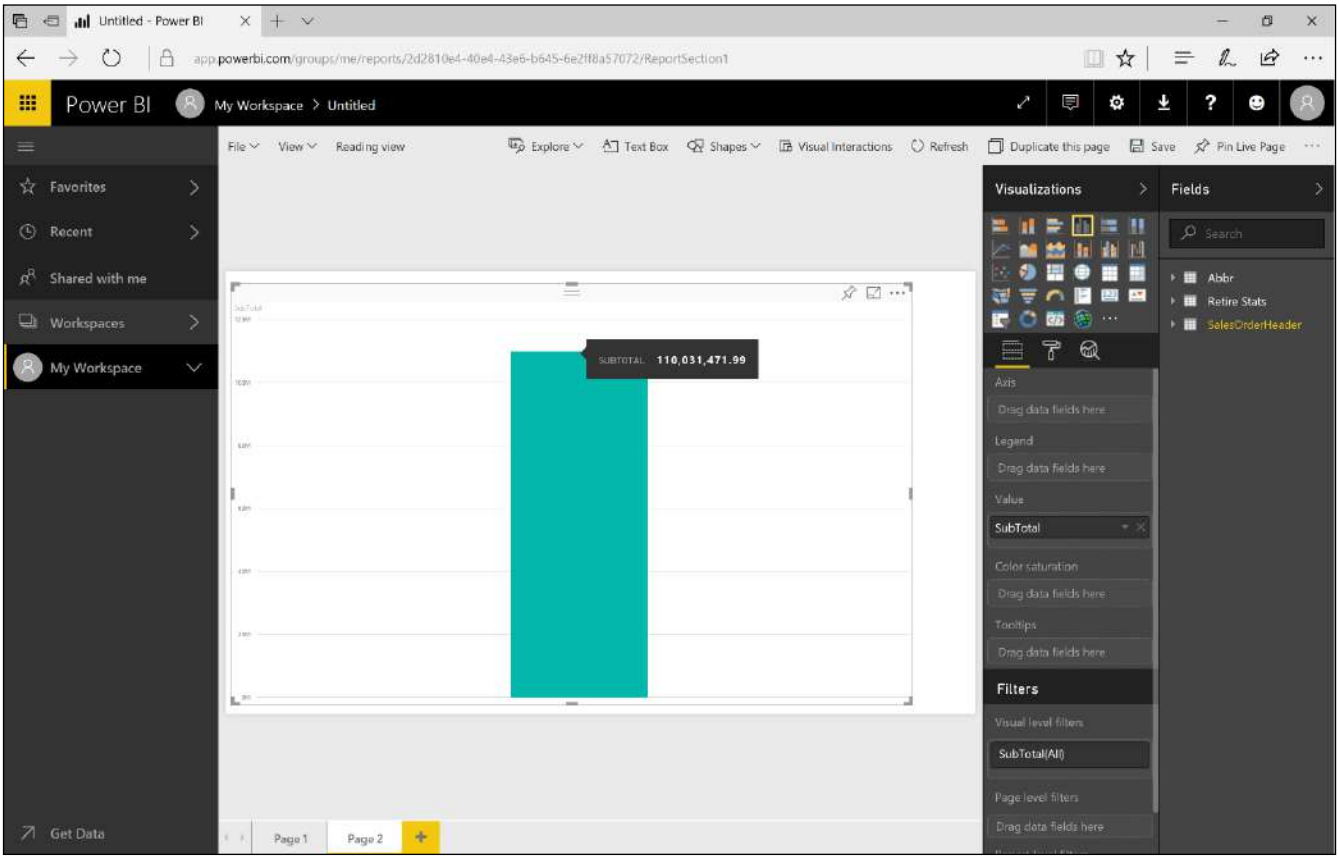


Figure 153: Power BI Service Report

Chapter 6 Developer

Power BI is not a closed environment—we can customize it as well as embed it in applications.

Developers can use two modes to extend and personalize Power BI:

- Power BI REST API
- Custom visual, which allows us to develop visuals, and in the visual gallery we can see all the visuals developed by other people. We can also develop a new one of our own and decide whether or not to share it with other users (by publishing it to the gallery).

It is possible to embed Power BI in other applications/documents by using Power BI Tile in Office documents.

Power BI REST API

We can connect through the API Power BI to push the Power BI data via direct programming. The reference site for Power BI developers is dev.powerbi.com, where you can find examples and documentation about the continuously updated APIs. You can start by clicking the entry **Try the API** and working through the interactive operating system.

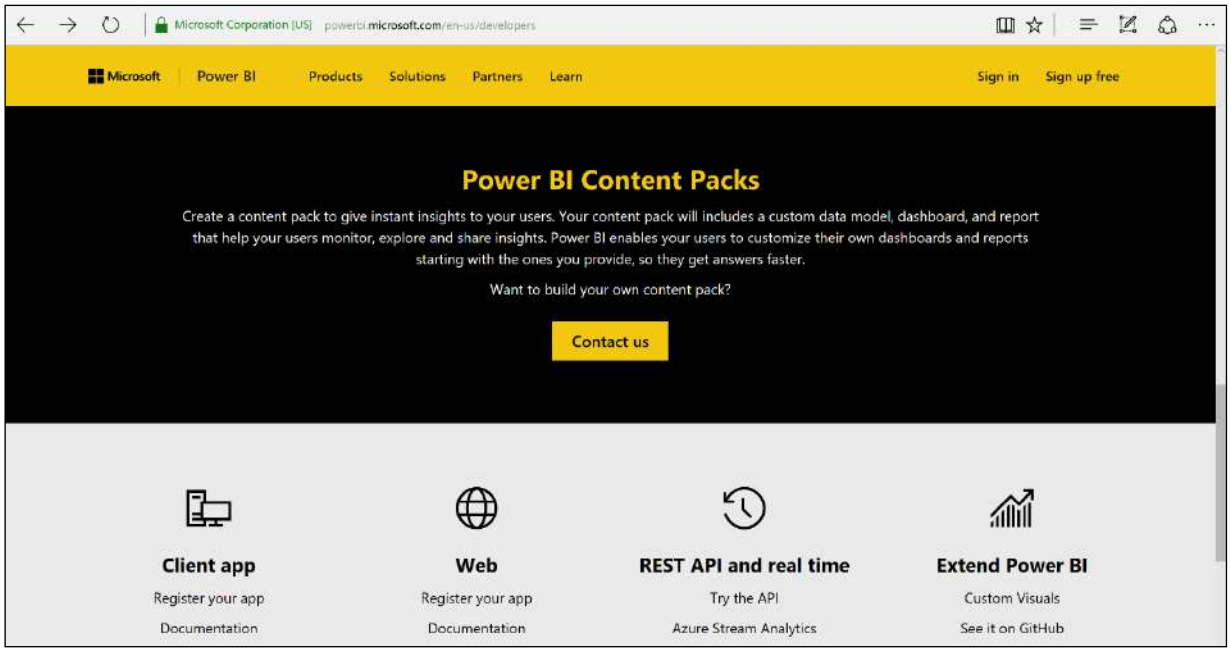


Figure 154: Power BI Developer Site

Try it by going to the console, carrying out calls to the services, and looking at what happens.

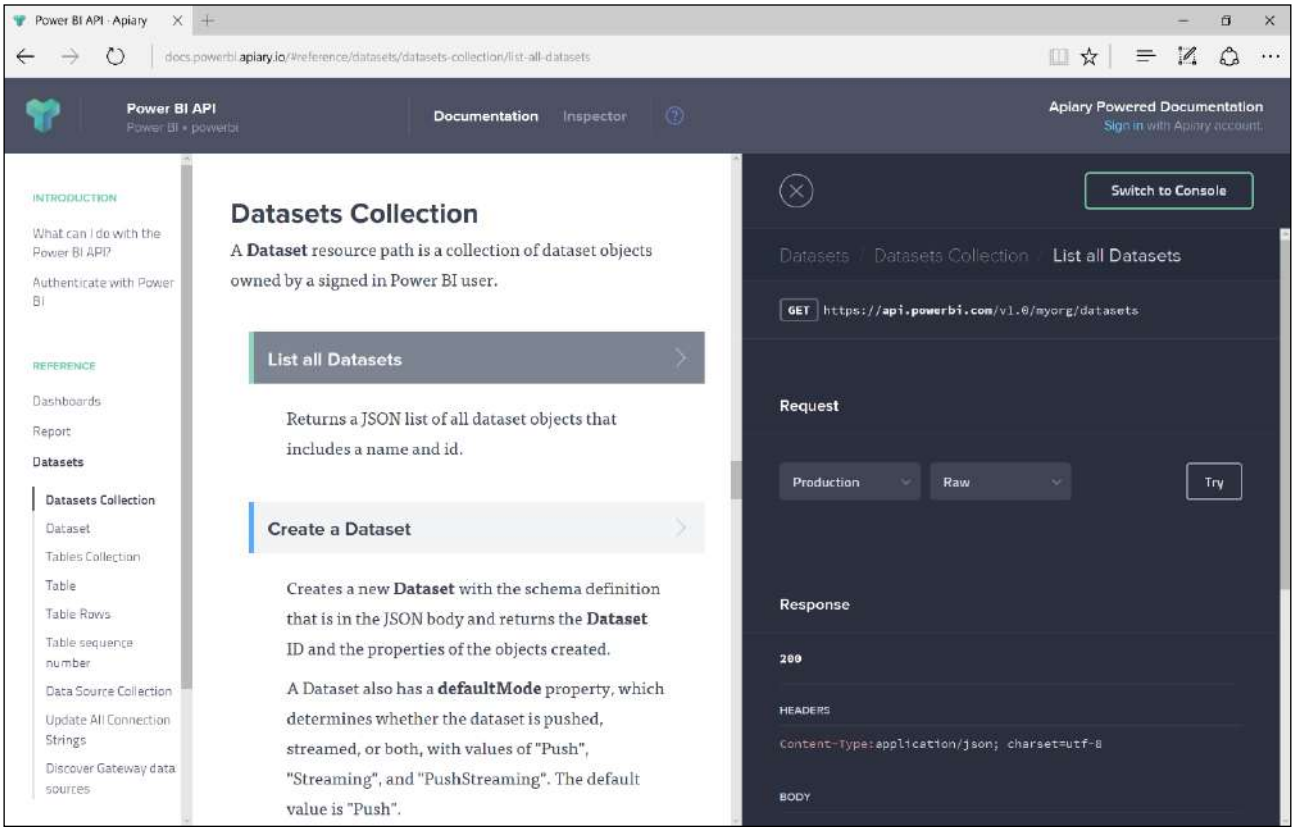


Figure 155: Power BI Console Portal

On the left side of the screen, you'll find references to the connections to the various APIs, and under the datasets you will find the calls to the APIs that can be evoked.

Our first action is to use our API to query a dataset currently saved in Power BI. By selecting Datasets and Datasets Collection, we'll find several APIs that can be called. Next, we'll obtain a list of all the datasets. On the right side of the screen, you'll see the web service that can be evoked, the URL you want to call, the details of the request type (necessary for the request transmission), and the format you will obtain from the call return. By clicking **Try**, the request gets carried out, but remember that you must authenticate and authorize Power BI.

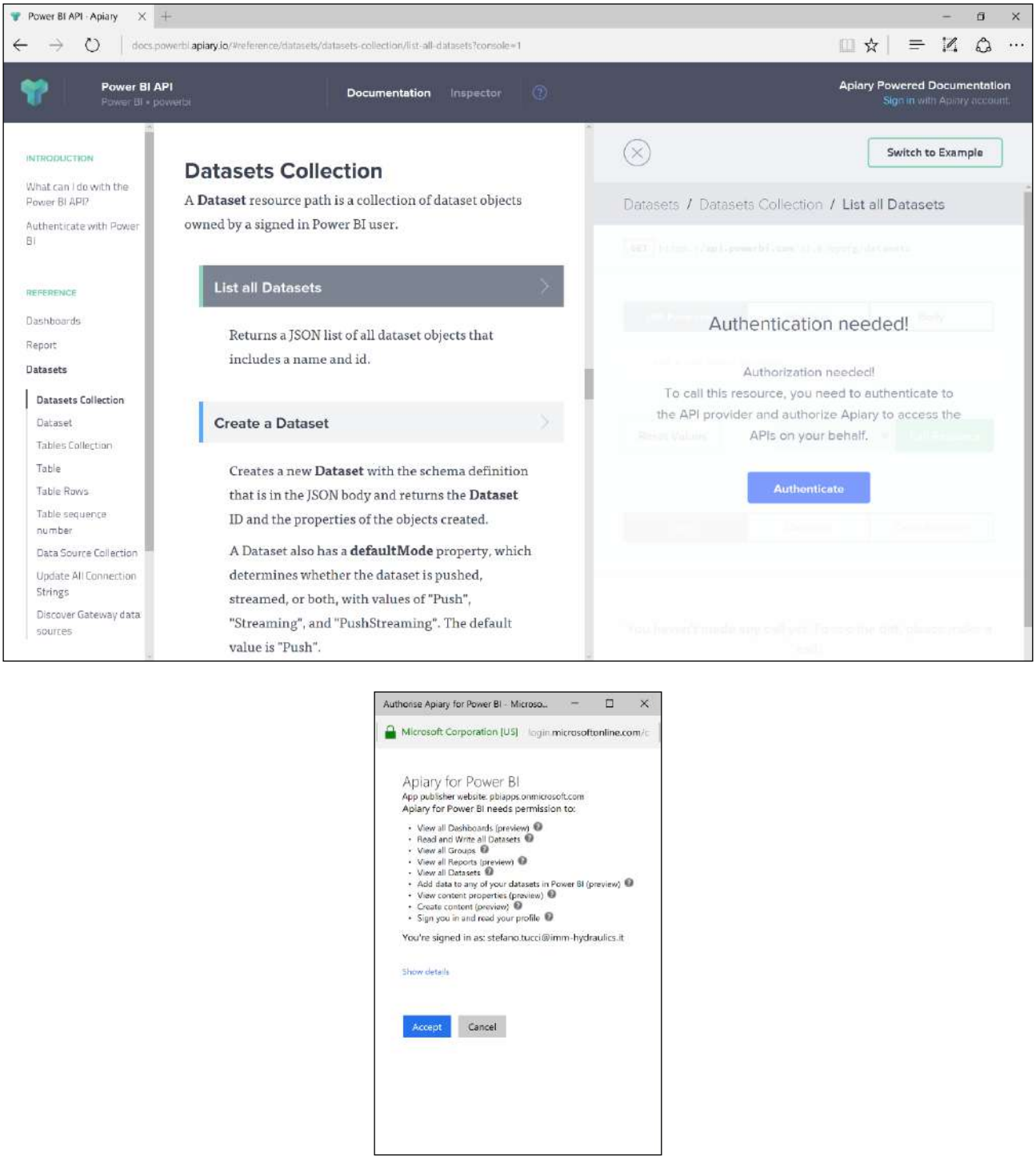


Figure 156: Power BI Console Portal, Authentication—sequence

At this point, choose your account and accept the conditions.

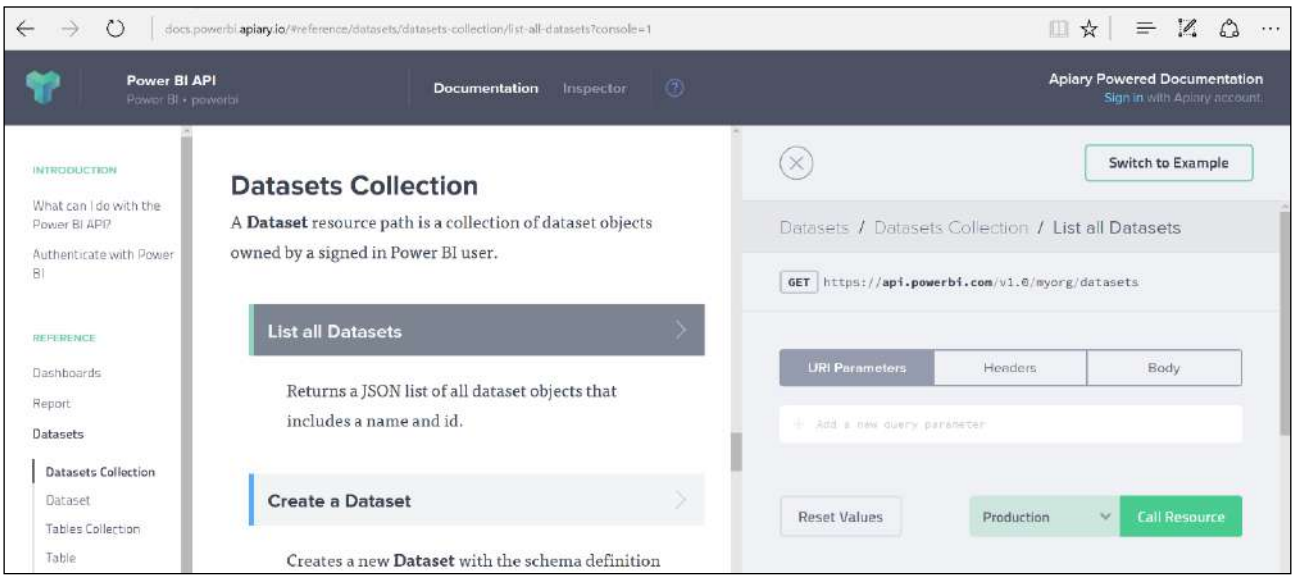


Figure 157: Power BI Console Portal—datasets collection

By clicking on the entry **Call Resource**, we allow the API to call the desired resource. So this first call simply lists the data sets available.

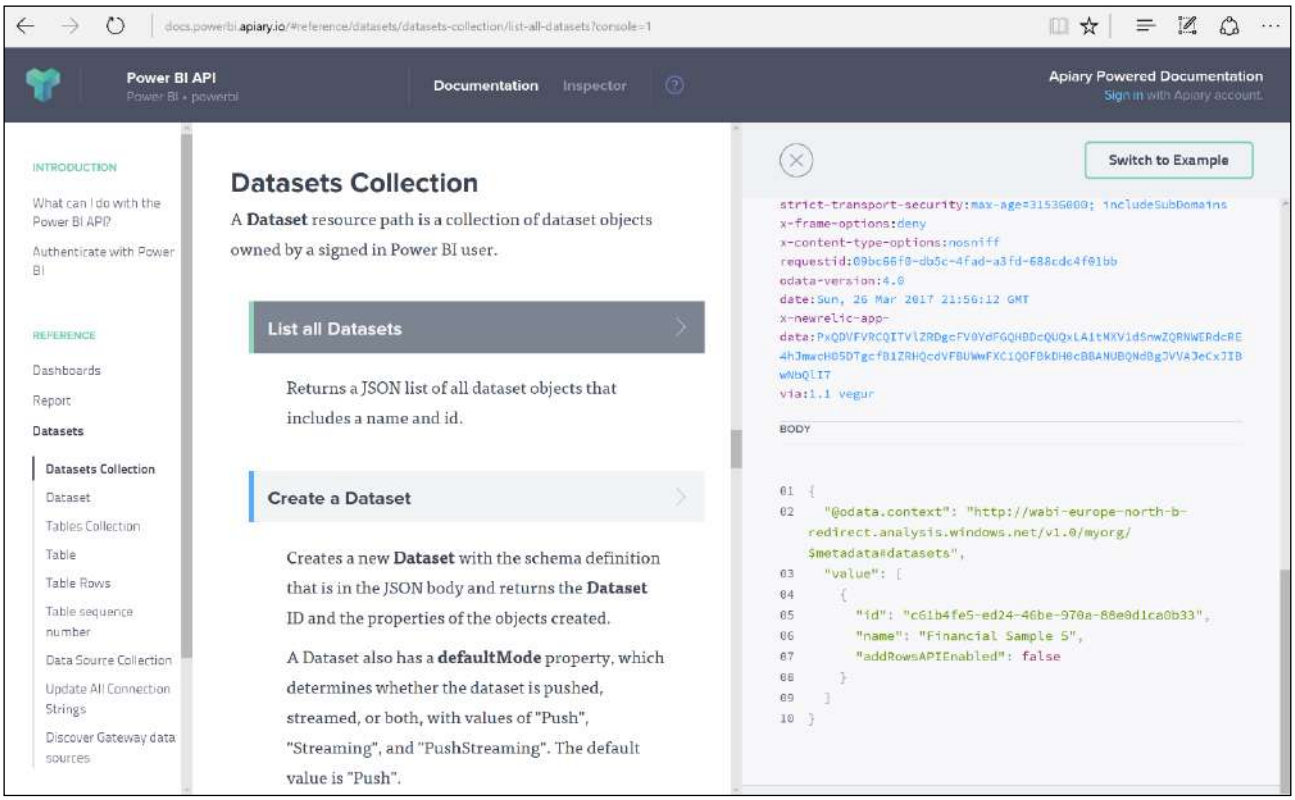


Figure 158: Power BI Console Portal—datasets collection

The result of the call will be displayed in the underlying section. On the left side of the screen, you'll see the possible calls you can carry out, while on the right side you'll see the interactive console used to evoke the calls. The answer to the call is quick, typically returned in less than one second.

Next, we go back to the data collection to create a dataset through the dedicated API. From the left side, we click on the entry **Create a Dataset**. The console changes the information. We can compose a new dataset from the Body section.

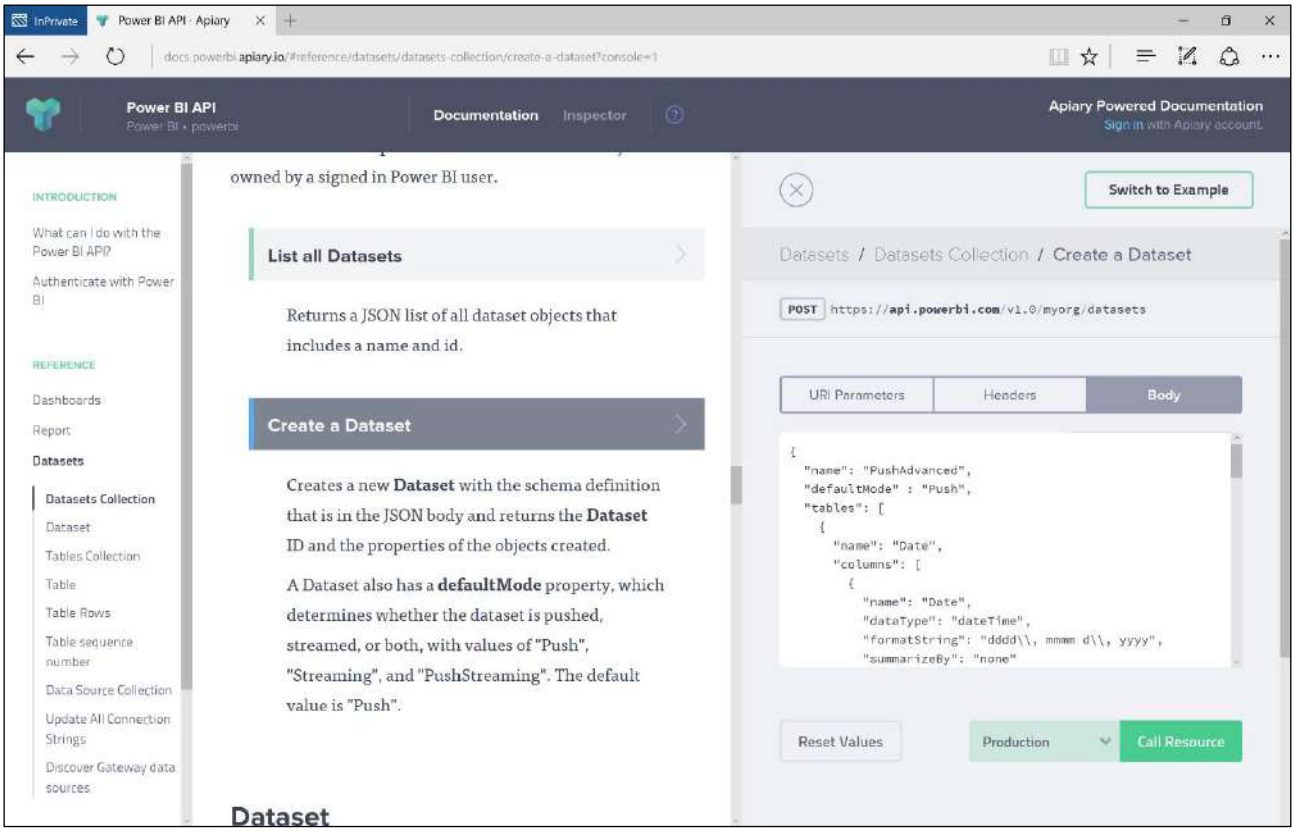


Figure 159: Power BI Console Portal—Create a Dataset

The entry Call Resource will carry out the request. Go back to Power BI and you will see the newly created dataset.

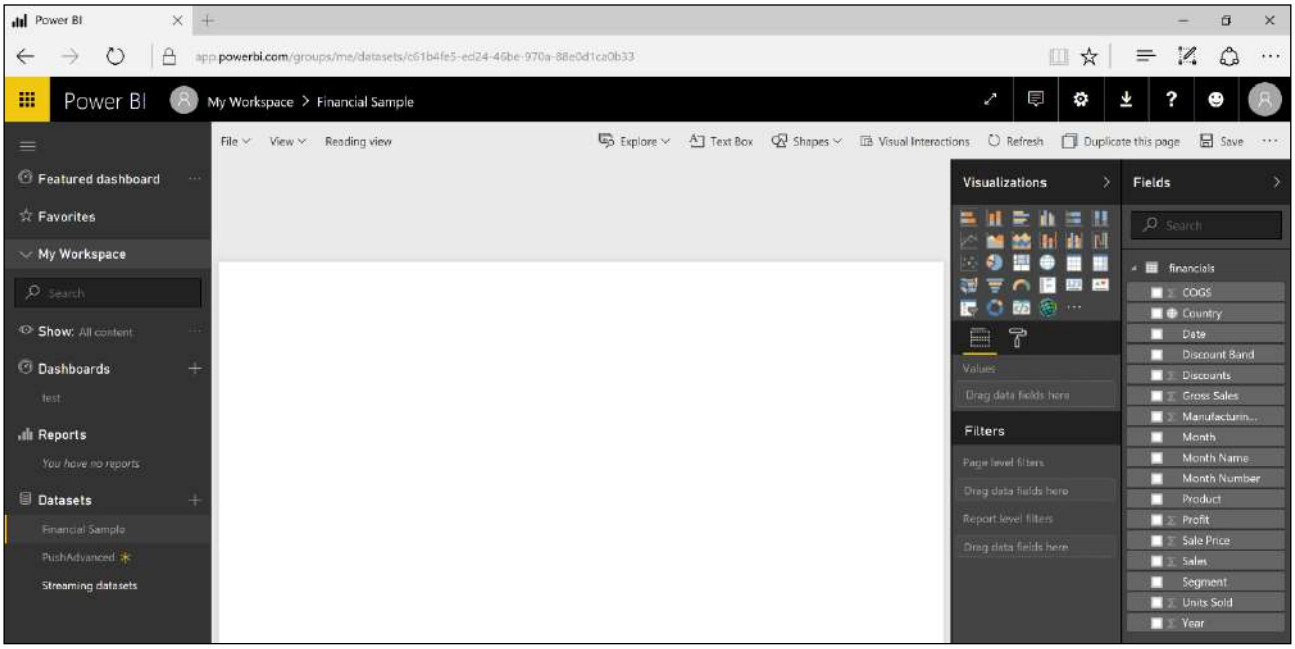


Figure 160: Power BI Service—datasets update

Now you'll notice the presence of the new dataset and you can start using it, but note the data is not present. Only the structure of the data has been defined.

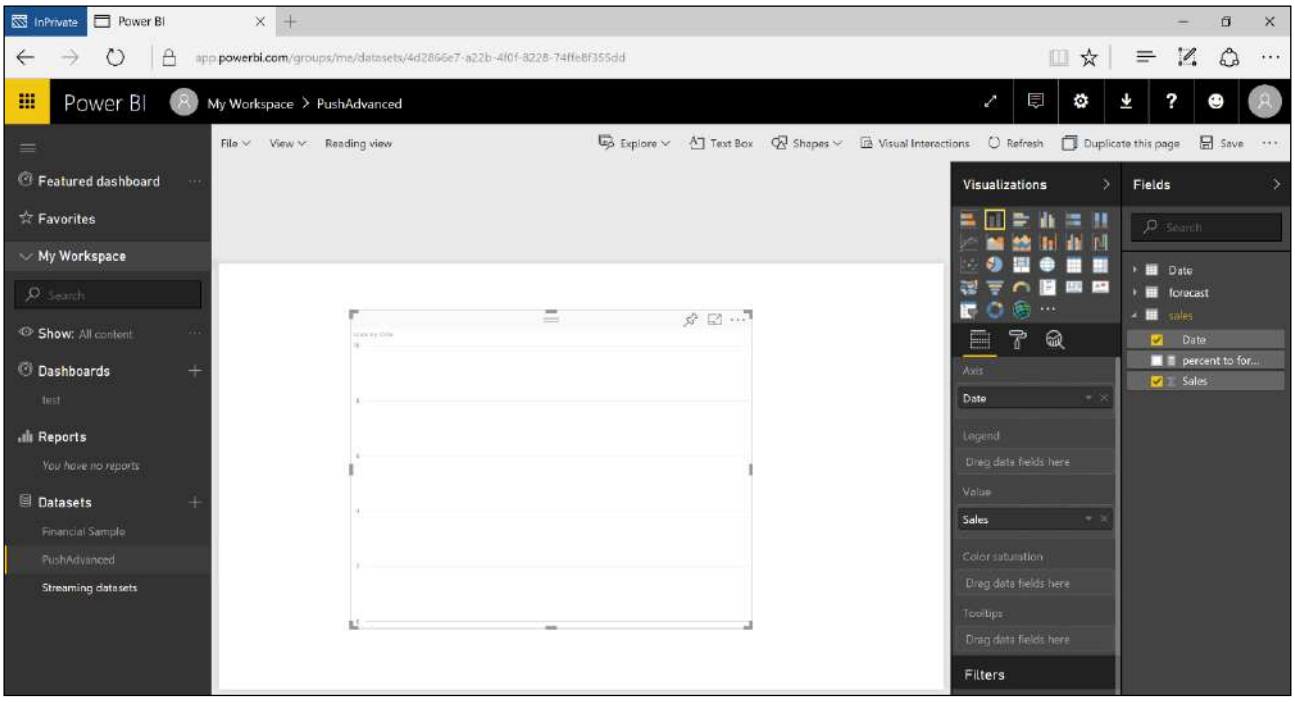


Figure 161: Power BI Service Report

Before populating the data, we should save the report and the dashboard.

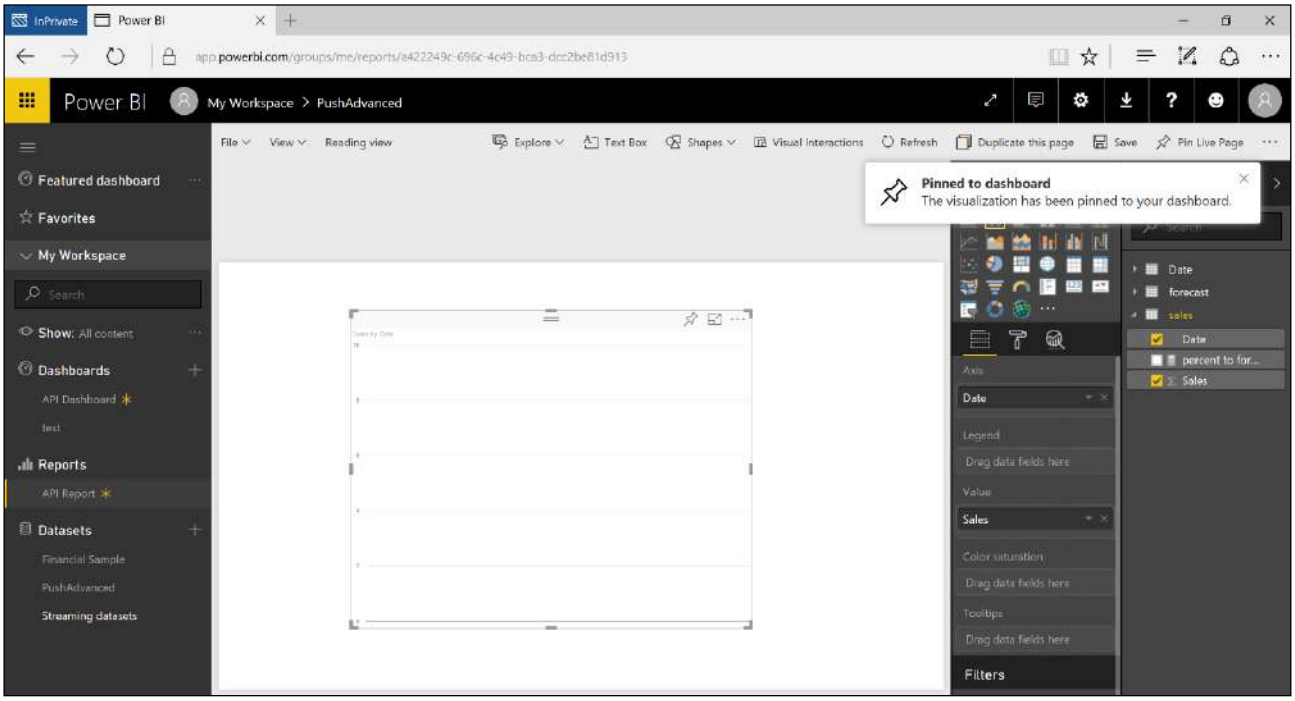


Figure 162: Power BI Service Report

Go back to the console to insert data and select the entry **Add Rows to a Table in a Dataset**. In the parameters, in the section "URI Parameter," specify the ID of the dataset and the name of the table to be populated (in our case, "sales").

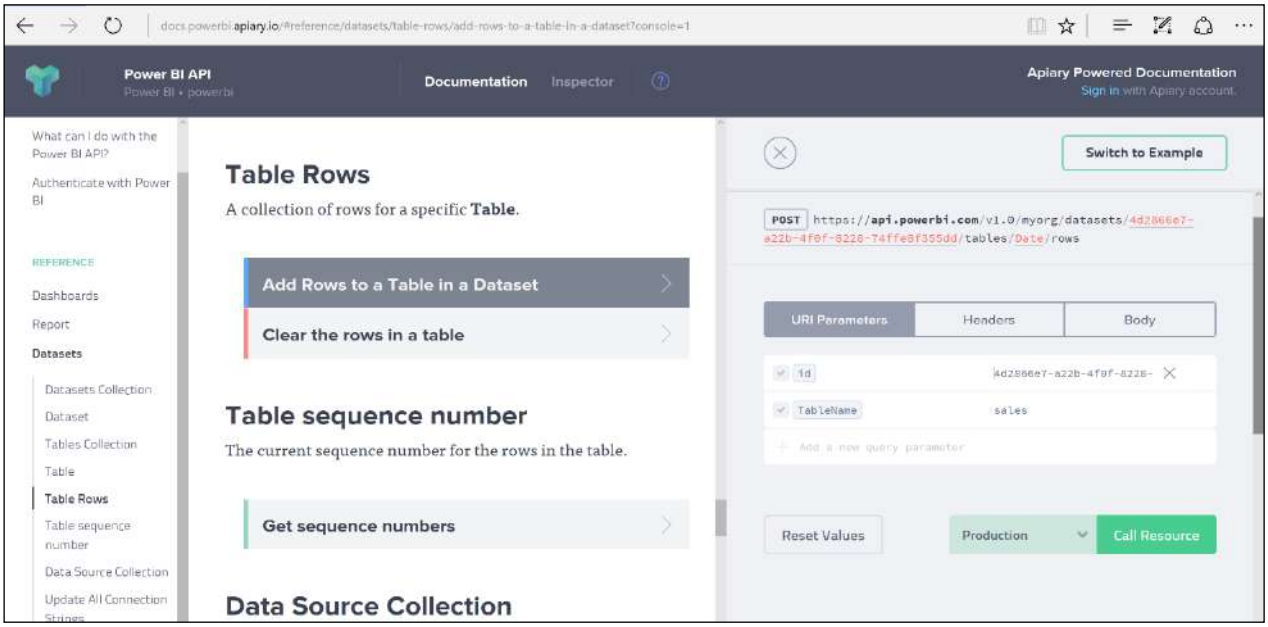


Figure 163: Power BI Console Portal—Table Rows, Add Rows

In the Body section, specify the data you want to insert, as shown in the example in Figure 164, and click **Call Resource**:

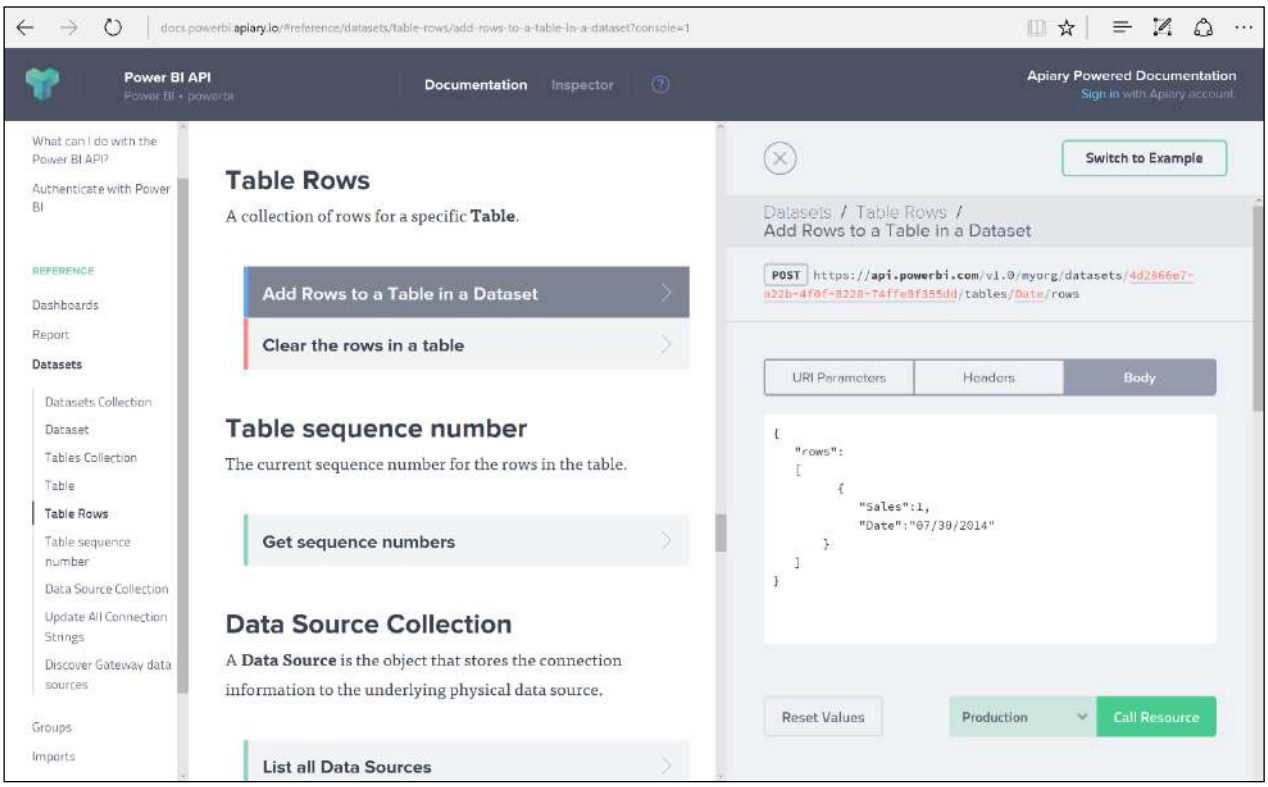


Figure 164: Power BI Console Portal—Table Rows, Add Rows

From this moment forward, the data will be persistent in our dashboard. You can keep that as is, or you can change the content. If so, you will notice the changes in the dashboard.

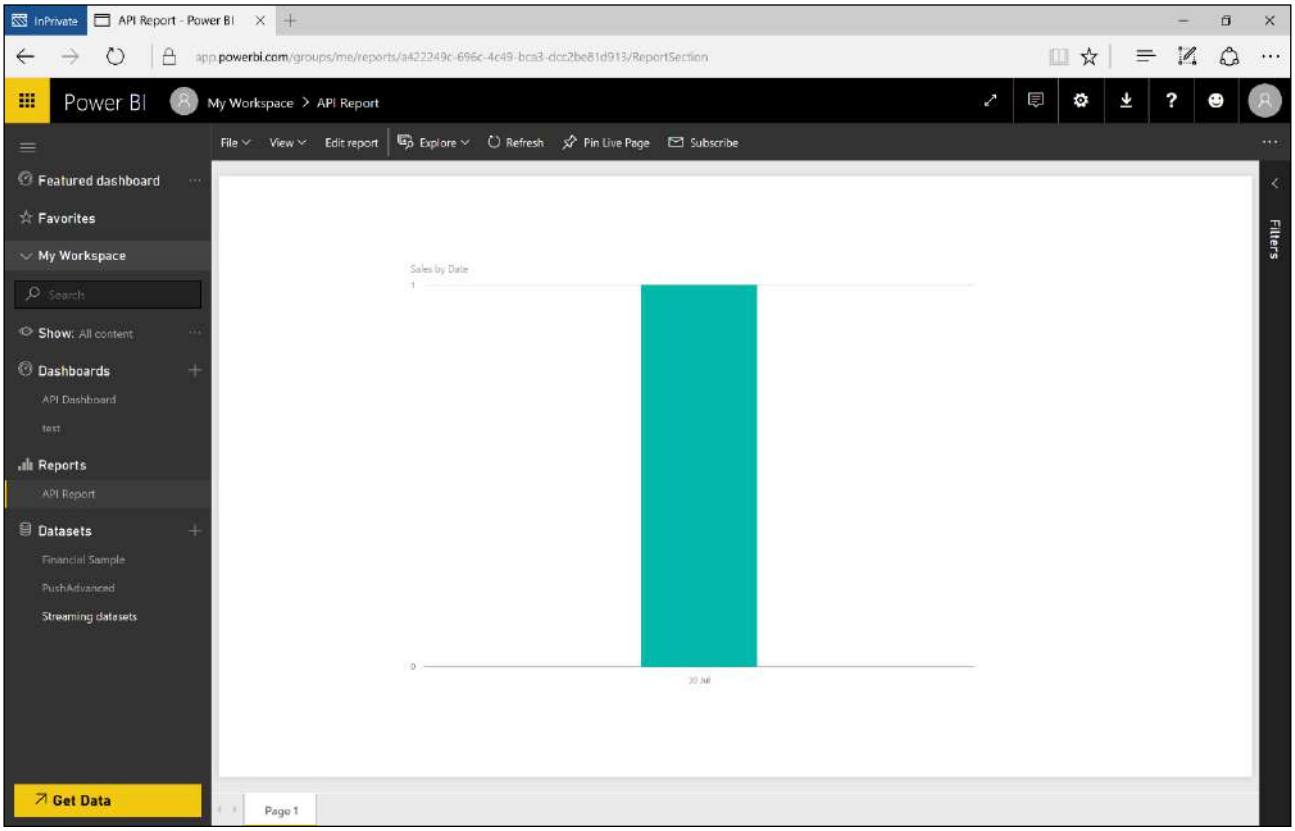


Figure 165: Power BI Service—report update

The same API functions can be invoked from our applications to create a dashboard that displays data in real time.

Power BI custom visual

Developers can easily create customized visual elements in Power BI for use in their own dashboards and reports. For starters, a code for all the graphic effects on GitHub is available. There are points of reference for visualization along with a test suite and tools for helping create high-quality, customized visual elements for Power BI. All this is available as an open-source project on GitHub at https://github.com/Microsoft/PowerBI-Visuals.

To create a visual object, you first need to set up the development environment. Then you need to install the NodeJS 4.0+ Required (5.0 recommended). Next, use the code in Code Listing 1.

Code Listing 1: NodeJS Command

```
npm install -g powerbi-visuals-tools
```

To confirm that the installation is the successful, you can execute the command in Code Listing 2.

Code Listing 2: NodeJS Command

```
pbiviz
```

You will obtain the following result, shown in Figure 166.

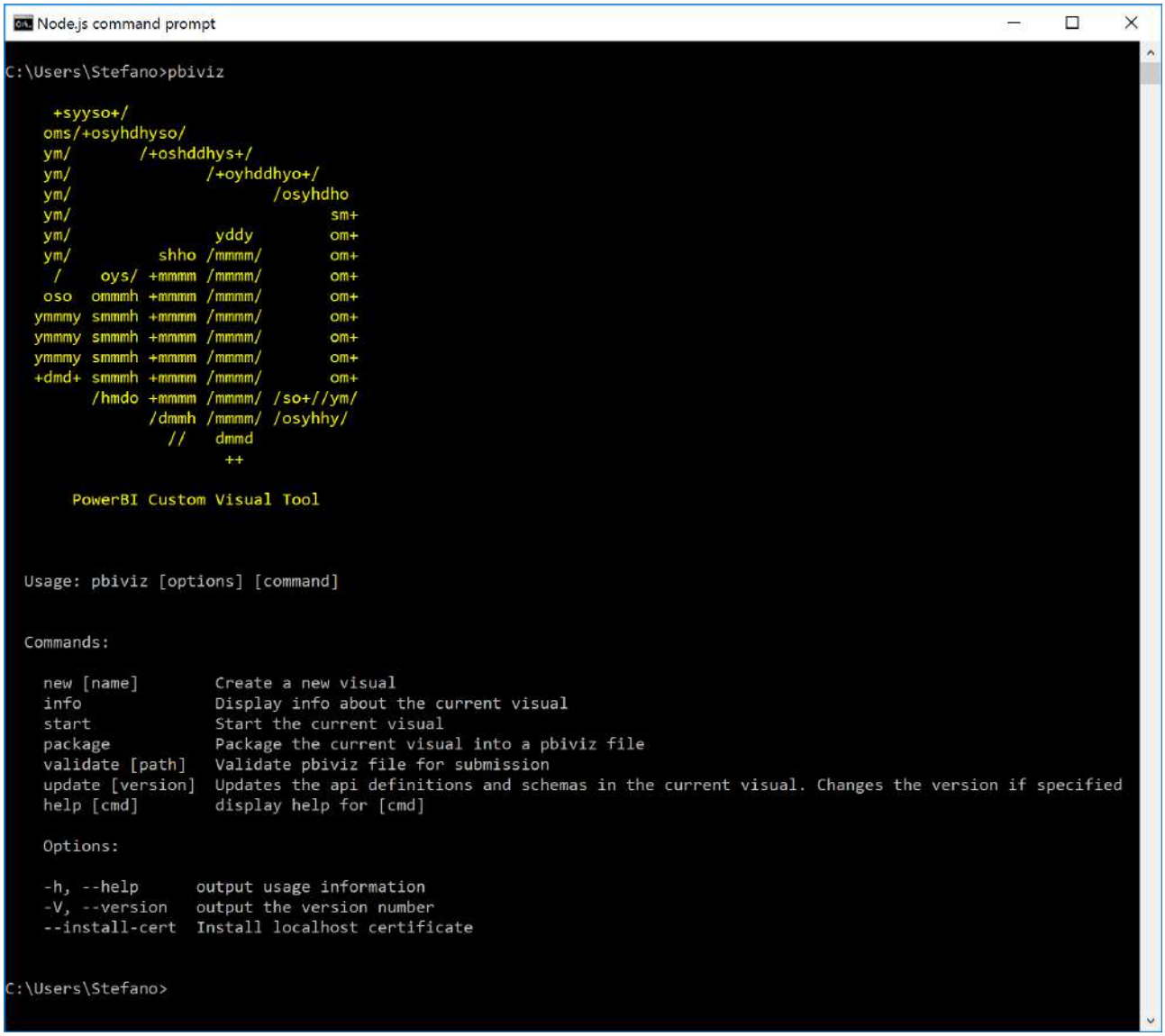


Figure 166: Pbiviz Command Execution

In order to enable the live preview of the visual objects, we must install an SSL certificate used in HTTPS. This will allow the display and the uploading of objects via the browser, and it will work for users of Windows or OSX.

Code Listing 3: NodeJS Command

```
pbiviz --install-cert
```

In the procedure for Windows users, the following window, as in Figure 167, will appear:

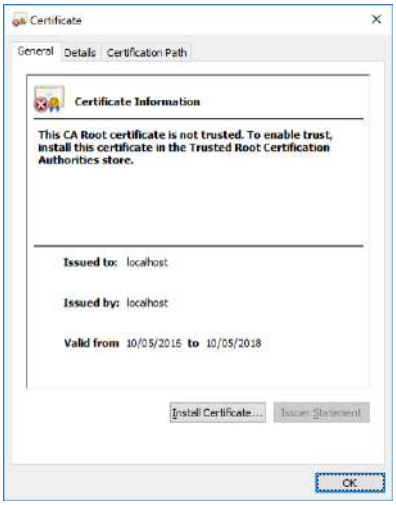


Figure 167: Certificate Information

Next, click **Install Certificate**, also as shown in Figure 167.

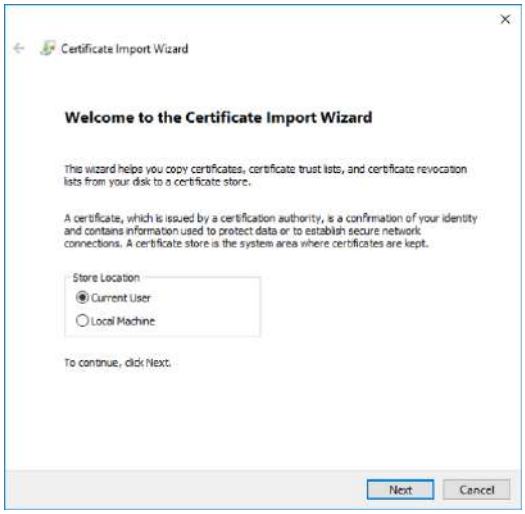


Figure 168: Certificate Import Wizard

Now we select **Current User** and click **Next**.

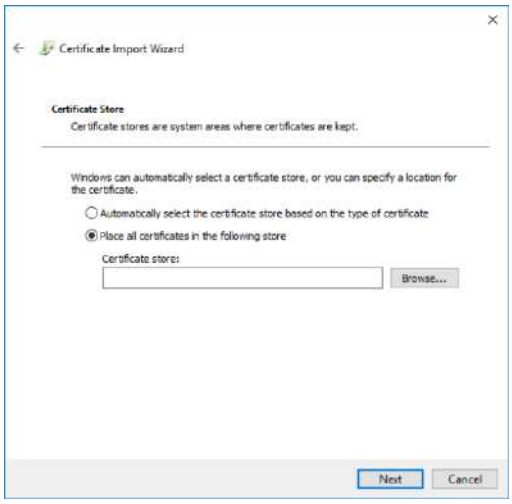


Figure 169: Certificate Import Wizard

Next, we select **Place all certificates in the following store** and click **Browse...** next to that line.

Figure 170: Certificate Import Wizard

Figure 170 shows the next steps—we select **Trusted Root Certification Authorities** and click **OK**. Then we go back to the previous screen, where we select **Next**.

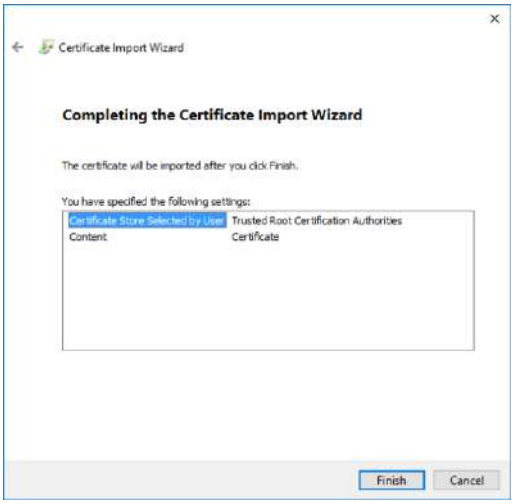


Figure 171: Certificate Import Wizard

We complete the installation by clicking **Finish**.

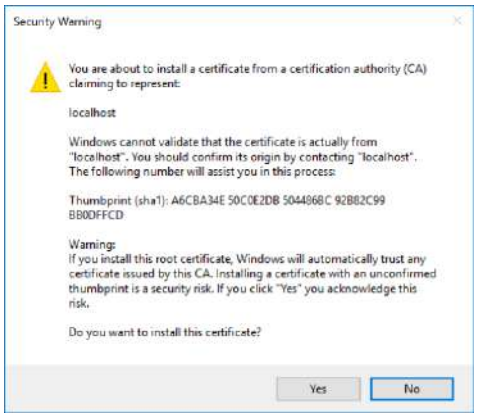


Figure 172: Security Warning

And now we verify the installation by selecting **Yes**.

In the procedure for Mac OS X users, the window in Figure 173 will appear.

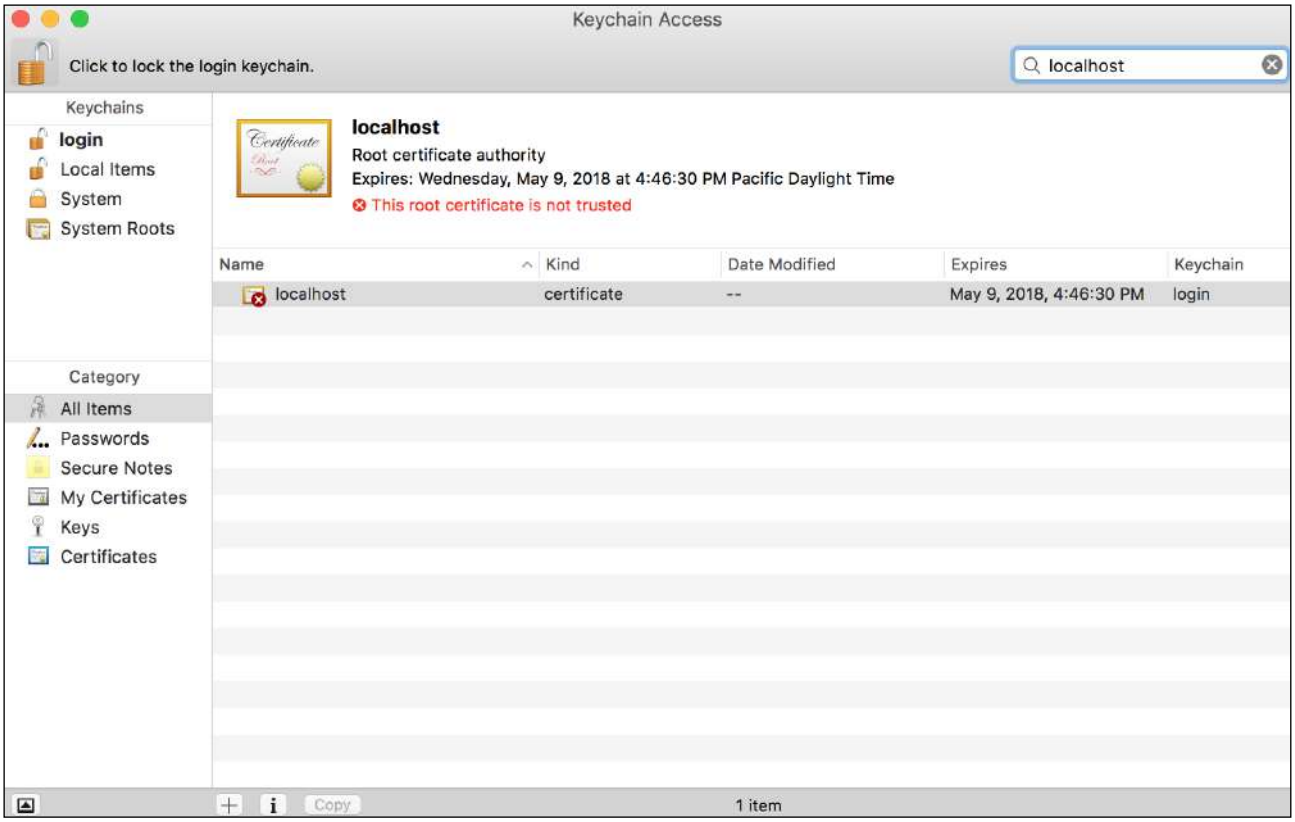


Figure 173: MAC Keychain Access

If the lock on the screen is indeed locked, click it to unlock.

Next, enter "localhost" in the search box.

Now select the new certificate.

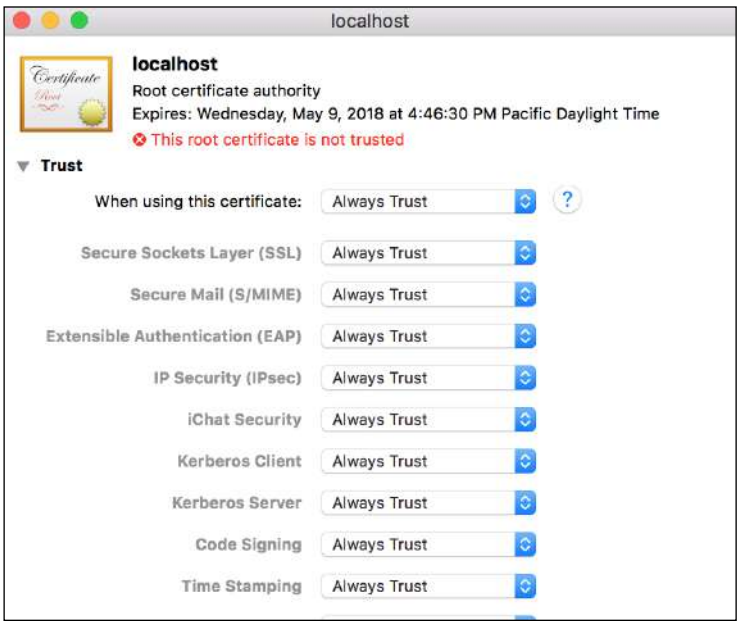


Figure 174: MAC Root Certificate Authority

Note that here we must select Always Trust.

Next, we close the window.

Figure 175: Credential Request

Figure 175 shows the next interface, in which we enter our username and password, then click **Update Settings**.

At this point, we have completed the installation.

When creating our first visual, we follow the prompt of the Node.js controls, as in Code Listing 4.

Code Listing 4: NodeJS Command

```
pbiviz new Name of visual
```

Rename “Name of visual” with a name that suits your needs. When the visual has been generated, you can change the name in the file pbiviz.json.

Figure 176: Pbiviz Create New Visual

Next, enter the newly created folder and execute the command in Code Listing 5.

Code Listing 5: NodeJS Command

```
pbiviz start
```

You will obtain the following result, as shown in Figure 177.

Figure 177: Pbiviz Execution

The following step enables the mode developer visual for testing—from the PowerBI service, we go to **Settings**.

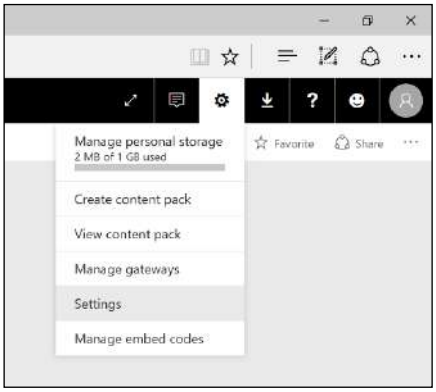


Figure 178: Power BI Service Settings

Next, click the entry **Developer**.

Now we select **Enable developer visual for testing**.

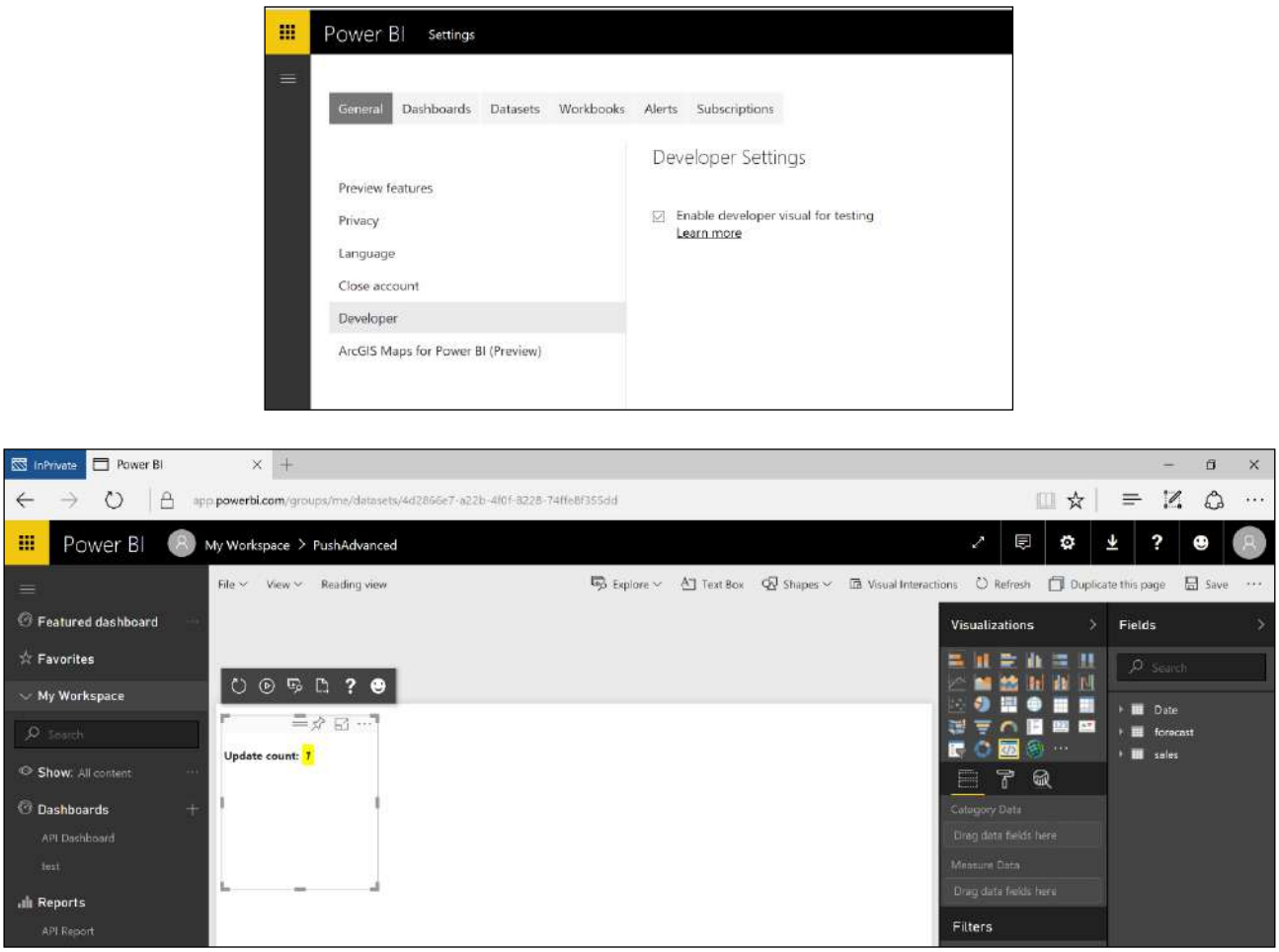


Figure 179: Power BI Service—report and new visual—sequence

From left to right, you'll see the following Toolbar icons:

- Refresh Visual—you can manually select the visual if the autoreload is off.
- Toggle Autoreload—when the visual is on, it will be updated automatically every time the file is saved.
- Show Dataviews—shows the visual's underlying dataview for debugging.
- Get Help—links to the documentation.
- Send Feedback—for sharing experiences.

In order to create the distribution packet, follow the Node.js prompt to the folder containing the visual and execute the command in Code Listing 6.

Code Listing 6: NodeJS Command

```
pbiviz package
```

You will obtain the result shown in Figure 180.

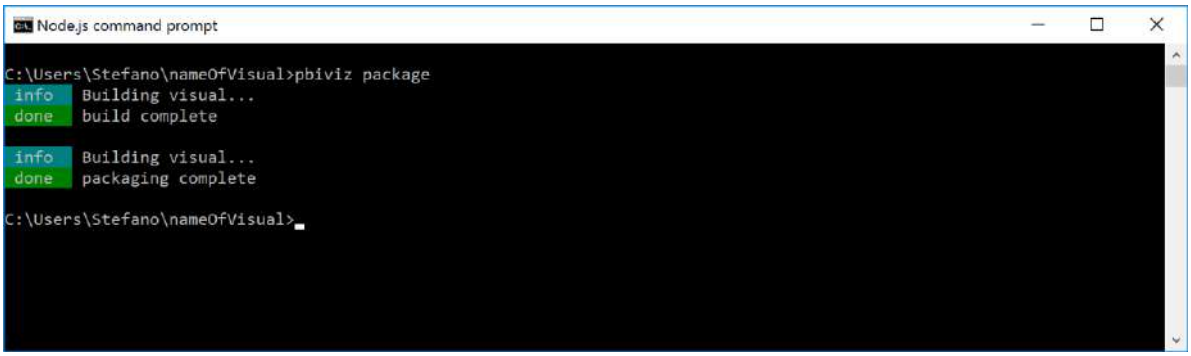


Figure 180: Pbiviz Create New Package

By navigating to the subfolder “dist,” you will notice the .pbiviz file, useful for importing as a custom visual within the visualization objects.

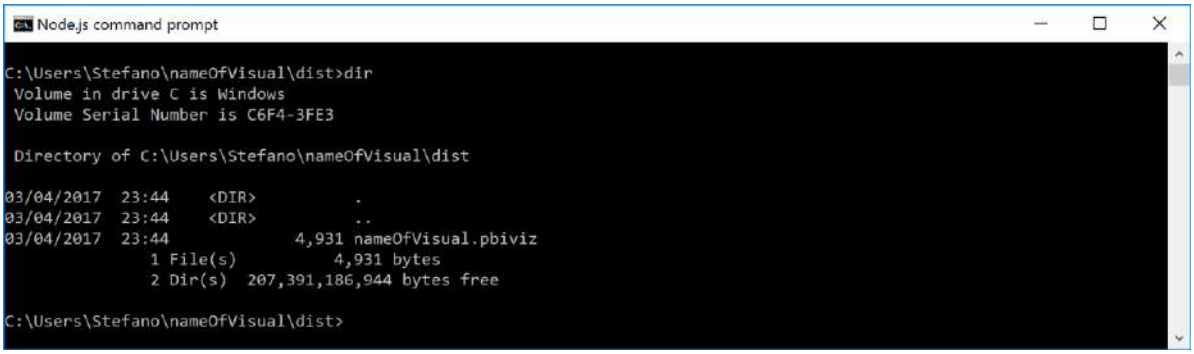


Figure 181: Package Path

Chapter 7 Power BI Embedded

Power BI Tiles

Power BI can be embedded in the Office suite (for example, PowerPoint).

In PowerPoint, it is sufficient to insert an add-in from the ribbon via the Insert, Store within the Add-ins section. We must look for Power BI Tiles in the store and install it; if it is installed, we must recall it in the Add-ins section.

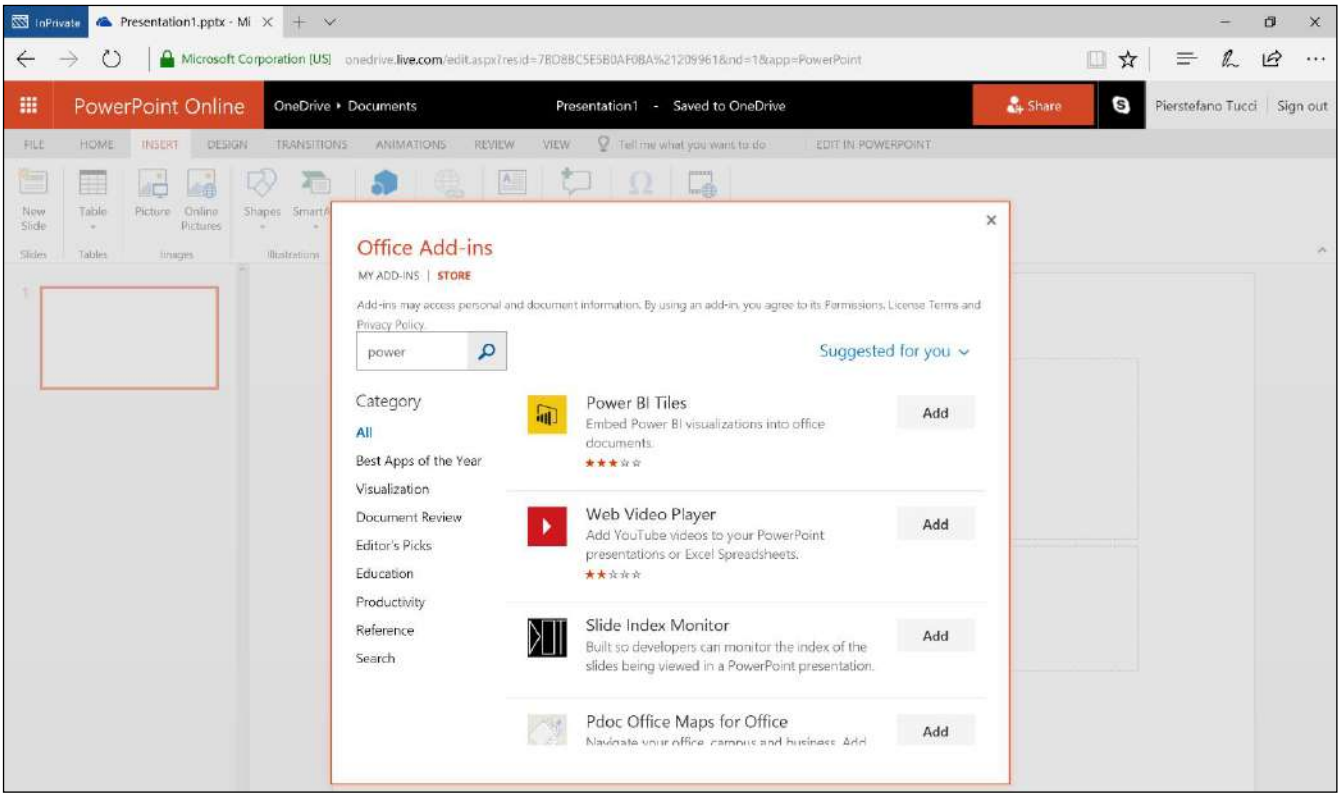


Figure 182: PowerPoint Online—Office Add-ins

Now we select the data using the From Power BI button.

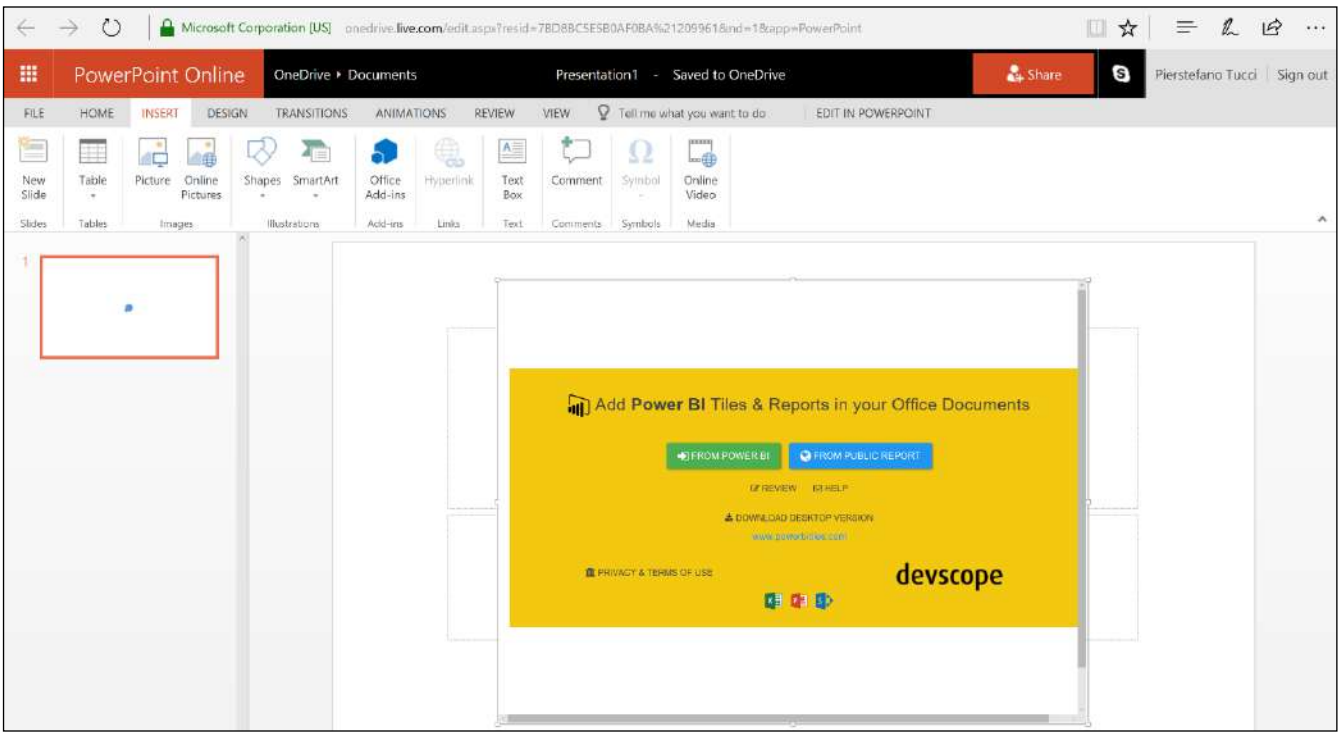


Figure 183: PowerPoint Online—Power BI Add-in

We log on automatically to our Power BI in the Personal Workspace.

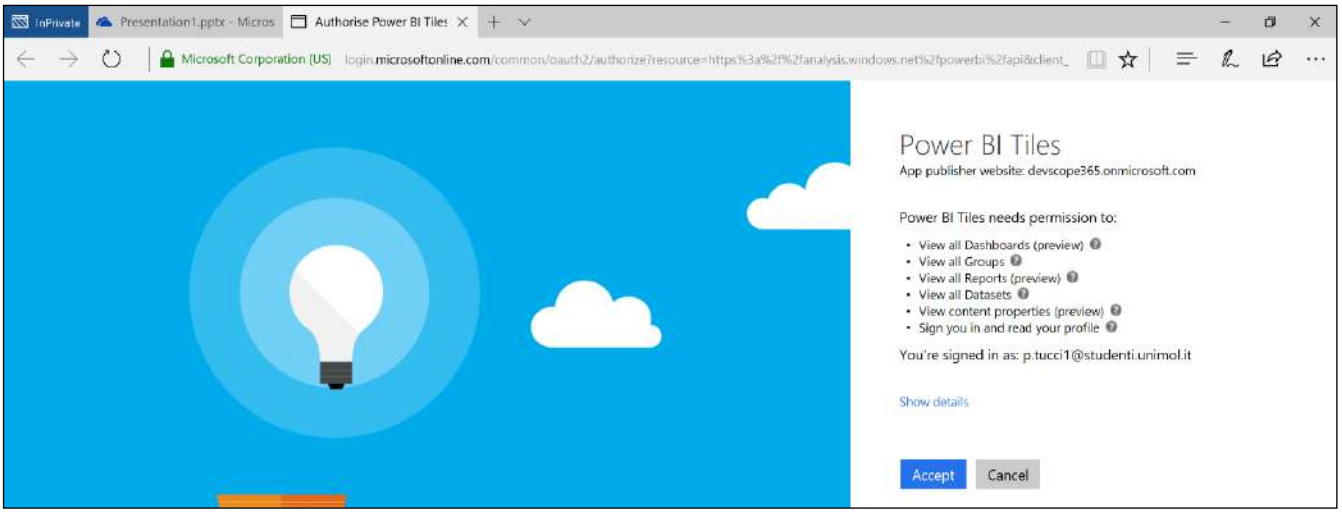


Figure 184: PowerPoint Online—Power BI Add-in, authentication

Next, we select a dashboard.

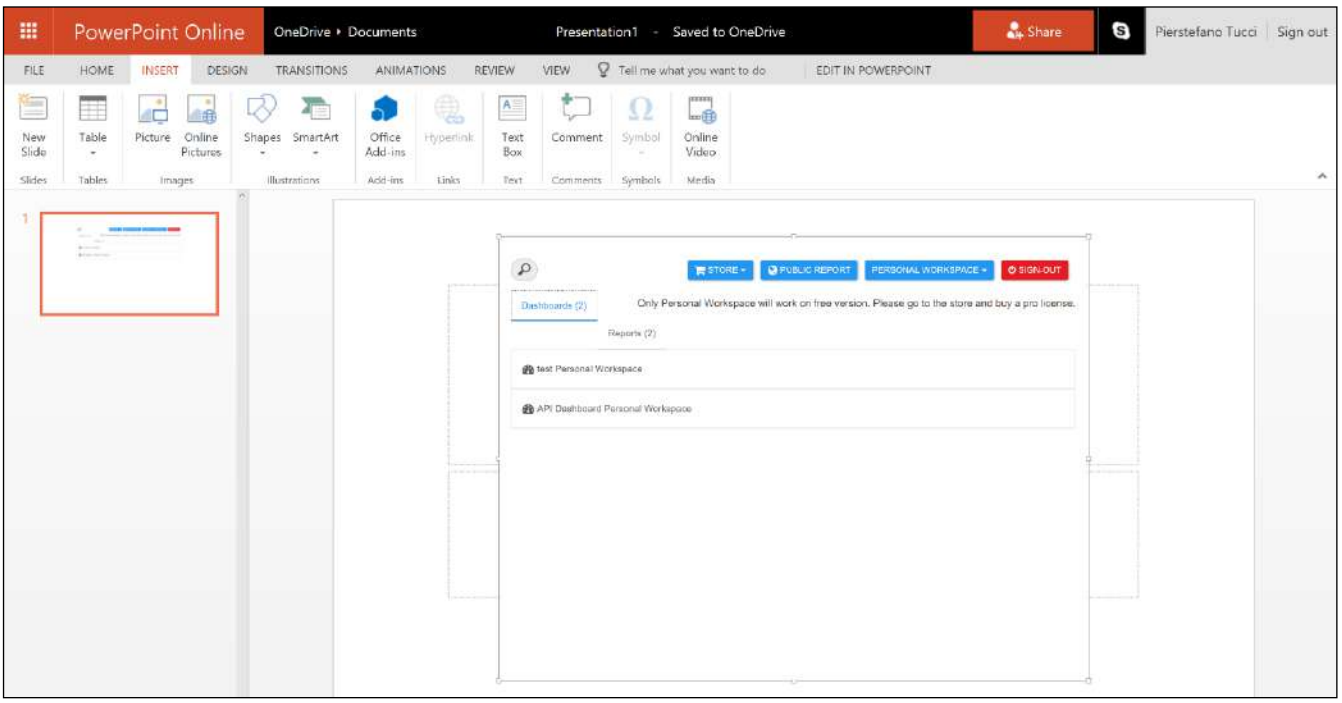


Figure 185: Power Point Online—Power BI Add-in

The selectable reports are displayed according to the selected dashboard.

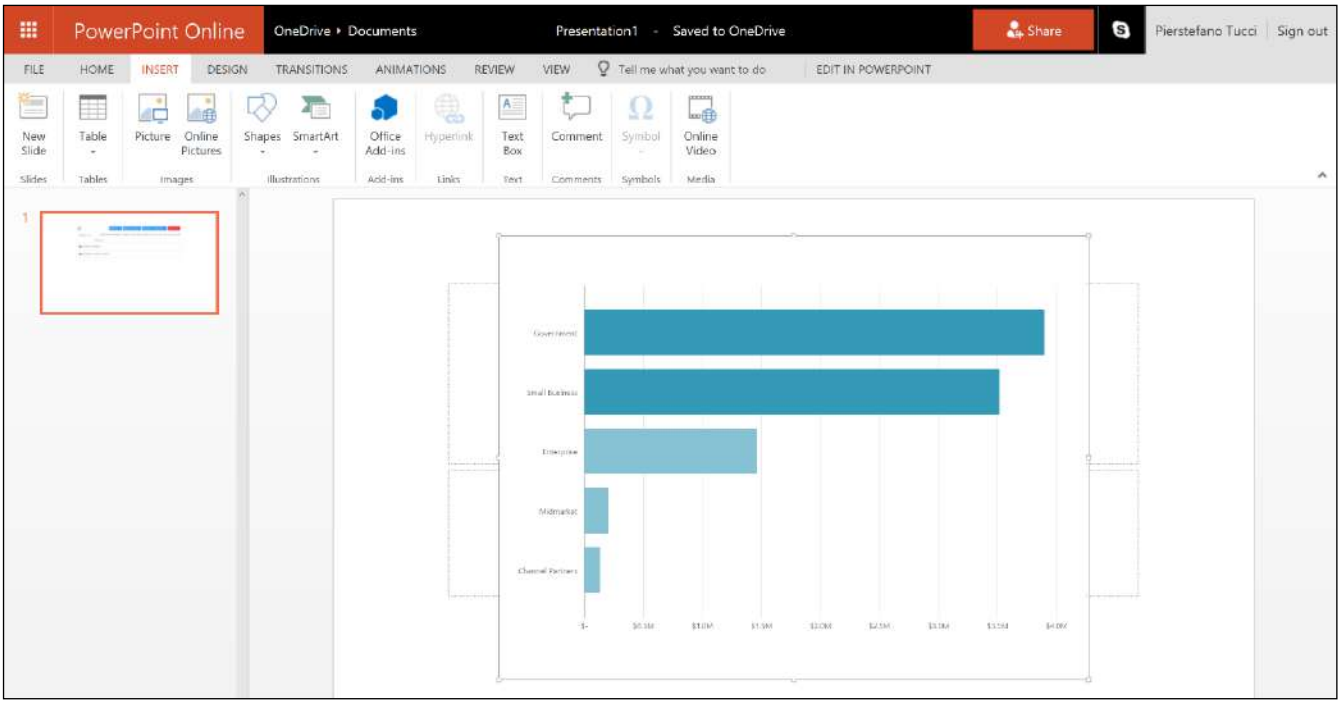


Figure 186: PowerPoint Online—Power BI Add-in

We can also run a tile refresh. Power BI allows you to update data and information dynamically in a way that's somewhat analogous to the way online speakers join and leave a group conversation.

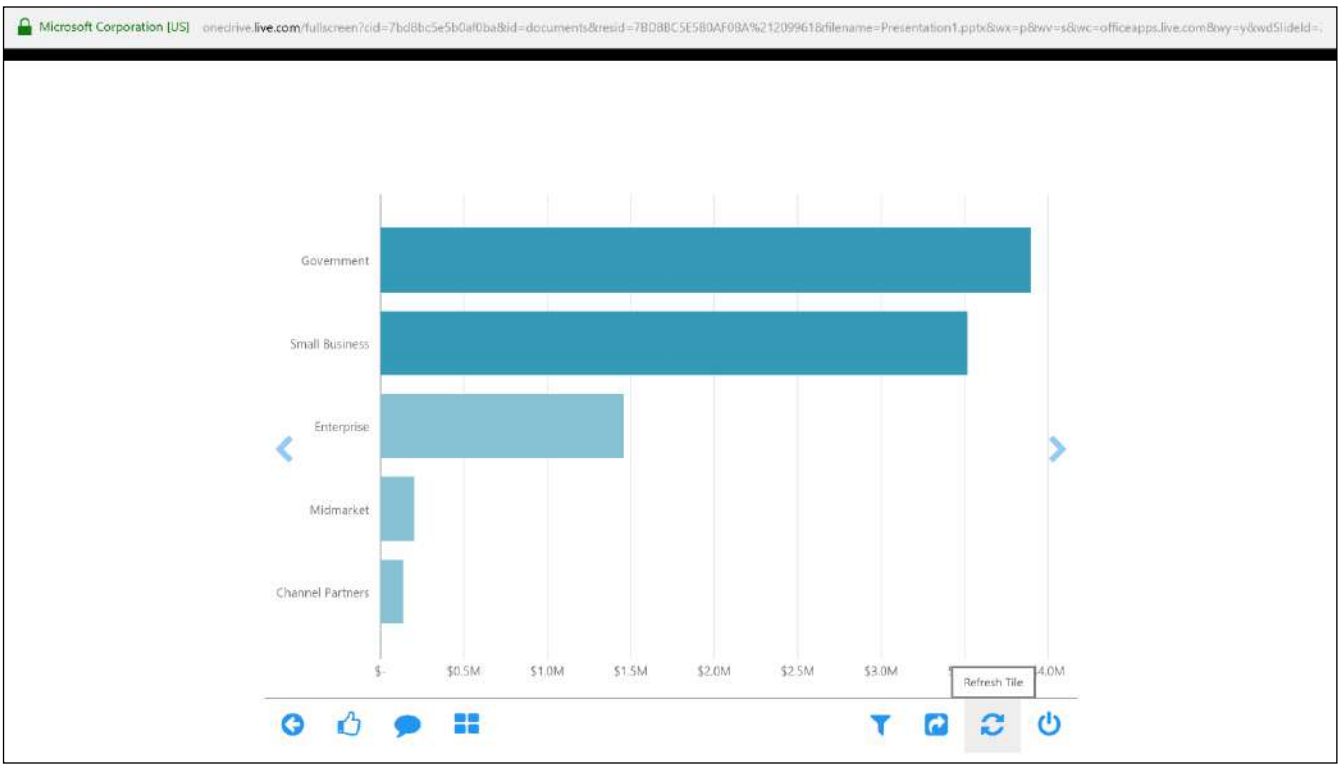


Figure 187: PowerPoint Online presentation with Power BI Dashboard

Power BI Embedded

What is Power BI Embedded

With Power BI Embedded, we can integrate the Power BI report directly into our web or mobile apps.

Power BI Embedded is an Azure service that allows app developers to bring the Power BI experience and intelligence to their applications. The applications that integrate Power BI Embedded have users with distinct licensed functionalities. Those applications can also access data elements incorporated as graphs and reports, which can now be fed through Microsoft Power BI Embedded. The users do not need a Power BI account to use the app. They can continue accessing the application as before without authentication, as well as displaying and interacting with the Power BI reports without needing possible further licenses.

Licensing for Microsoft Power BI Embedded

In the usage model of Microsoft Power BI Embedded, the licensing for Power BI is not the responsibility of the end user. Instead, the sessions and the service are bought by the app developer who has been consuming the graphics as well as paying for the subscription equipped with those resources.

Conceptual model

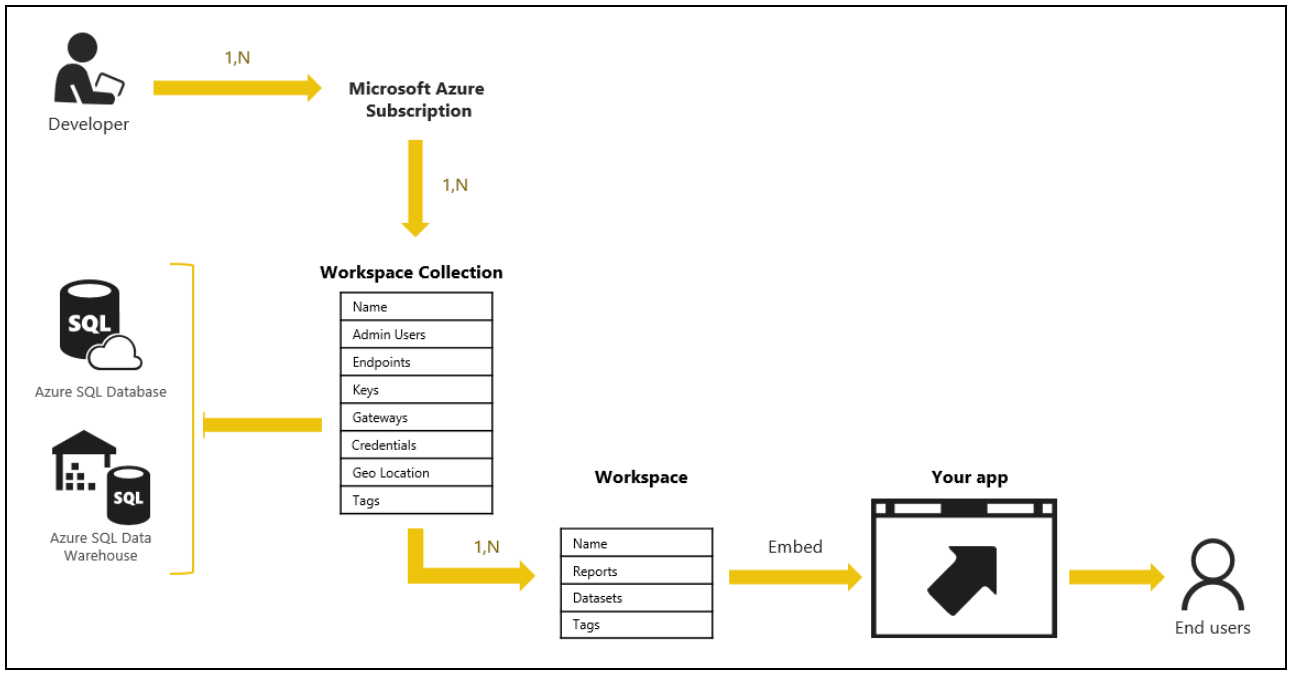


Figure 188: Conceptual Model

As with any Azure service, the resources of Power BI Embedded are provided through the Azure Resource Manager API. In this case, the resource being provided is present in the Power BI Workspace Collection.

Workspace Collection

A Workspace Collection is an upper-level container of Azure that contains resources and the possibility of zero or more Workspaces. The Workspace Collections have all the standard Azure properties:

- Access keys—keys used when we safely call the API of Power BI.
- Users—Azure Active Directory (AAD) users, who have available administrator rights to manage the Power BI Workspace Collections through the Azure ARM API.
- Region—as part of a Workspace Collection supply, we are able to select a region to carry out the supply.

Workspace

A workspace is a content container of Power BI that can include datasets, reports, and dashboards. Remember that a workspace is empty when it is created. A Power BI Desktop can be used and published in its own workspace by using the Power BI REST APIs.

Use of Workspace Collections and workspaces

Workspace Collections and workspaces are content containers, and they are used to arrange and improve the design of the application being created. We can use many ways to arrange the content within them. We can choose to put the entire content inside a workspace and then, at a later time, use the Application Authentication Tokens (app tokens) to split the content further.

Cached datasets

The datasets saved in the caches can be previewed. However, it is not possible to update the saved data in the cache once it has been uploaded on Power BI Embedded.

Authentication and authorization through the app tokens

Power BI Embedded refers to the execution of the necessary user authentication as well as the authorization back to the application. Azure AD works differently—the application conveys the authorization to carry out the rendering of a Power BI report to Power BI Embedded by using app tokens. These app tokens are created when the application requires the rendering of a report.

The app tokens are used for authentication in Power BI Embedded. There are three types of app tokens:

- Provisioning Tokens—used during the supply of a new workspace inside a Workspace Collection.
- Development Tokens—used in case of calls carried out directly to the REST APIs of Power BI.
- Embedding Tokens—used in the case of calls carried out for the rendering of a report incorporated in the iframe.

These tokens are used for the various phases of the interaction with Power BI Embedded. The tokens are designed in order to delegate authorizations from our app to Power BI.

With Microsoft Power BI Embedded, we can integrate Power BI reports directly into our own web application or mobile applications.

Here are several official online examples:

- Sample dashboard web app
- Power BI Embedded API reference
- Power BI Embedded .NET SDK (available via NuGet)

Configure the example application

We need Visual Studio to display the examples. Even the 2015 or 2017 Communty Edition versions will work okay.

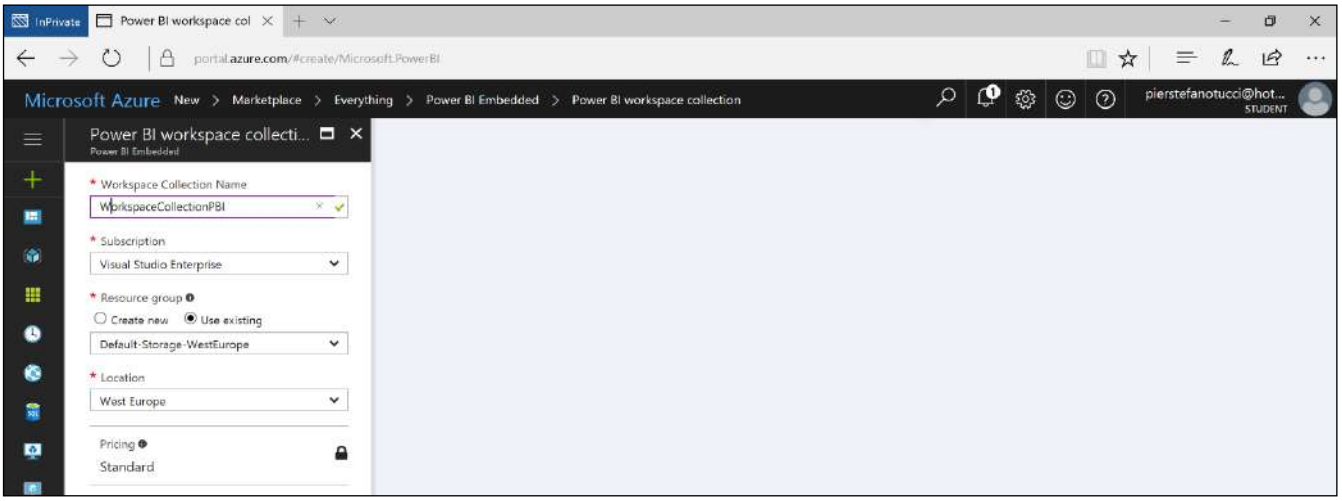
1. Download and decompress: Power BI Embedded. Integrate a report into a web app sample on GitHub.

2. Open PowerBI-embedded.sln in Visual Studio. You might need to carry out the commands to update the NuGET packets used in the solution.
3. Build the solution.
4. Run the console ProvisionSample application. In this example, the console application provisions a workspace, and it is possible to import a PBIX file.
5. Provision a new workspace in a collection of existing workspaces.

```
What do you want to do?
=======================================================================
1. Provision a new workspace collection
2. Get workspace collection metadata
3. Retrieve a workspace collection's API keys
4. Get list of workspaces within a collection
5. Provision a new workspace in an existing workspace collection
6. Import PBIX Desktop file into an existing workspace
7. Update connection string info for an existing dataset
```

Figure 189: Console Application

Next, we insert the name of the Workspace Collection and the Access Key. Figure 190 shows this procedure in the Azure portal.



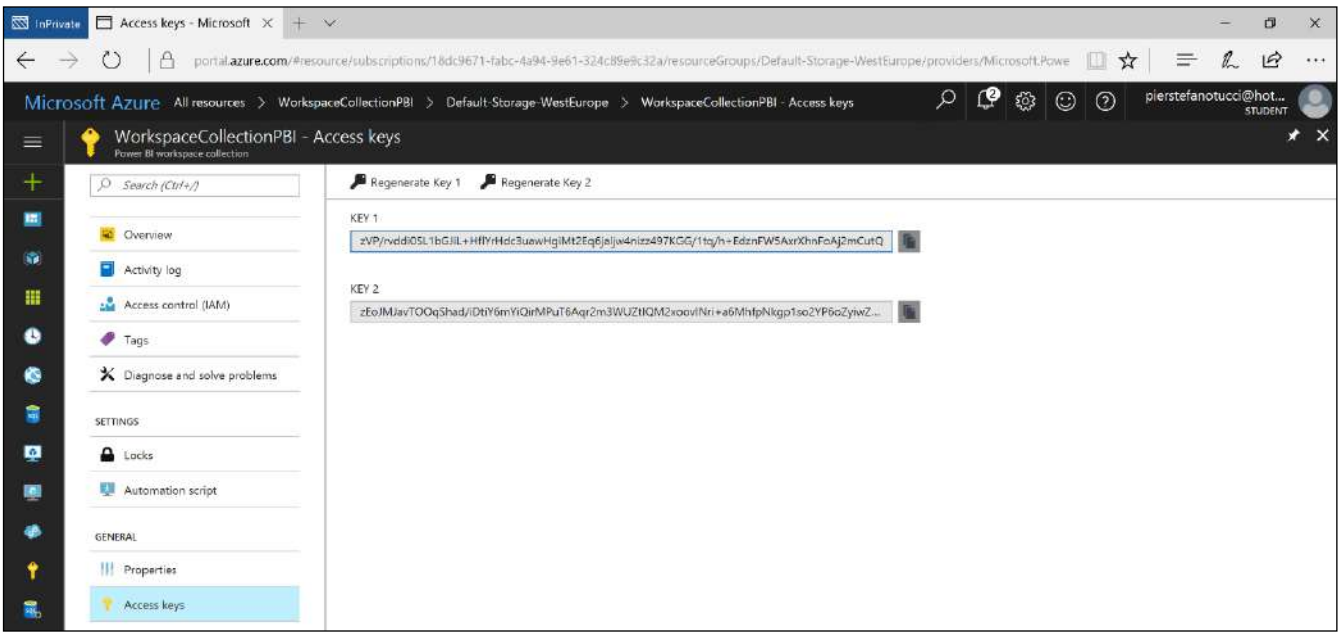


Figure 190: Azure Portal—Power BI Embedded, Create a workspace collection—sequence

Copy and save the newly created Workspace ID. You will find this in the Azure portal.

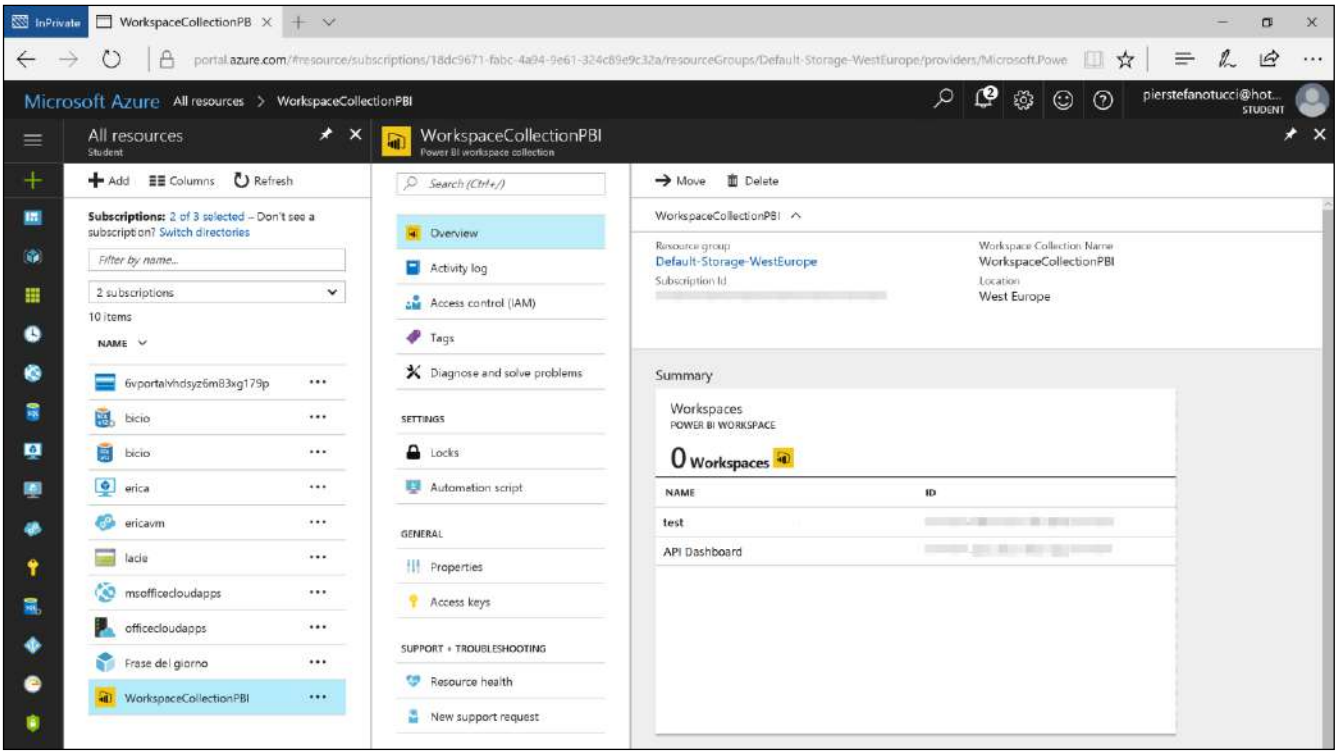


Figure 191: Azure Portal—Power BI Embedded, Workspace Collection

In order to import a PBIX (Power BI desktop file) into an Azure Workspace, you can use an existing workspace or create a new one. If you do not have a PBIX file, you can experiment with an example Retail Sales Analysis PBIX file from Microsoft.

Insert a descriptive name for the dataset content if one is requested.

At this point, the Power BI PBIX file has been imported into its own workspace.

Run the sample web app

The web application example is an example dashboard that carries out the rendering of the reports imported in the workspace.

Here are the steps for setting up the sample web app:

1. In the Power BI Embedded Visual Studio solution, select the web application **EmbedSample**, followed by **Set as StartUp project**.
2. Open the web.config file associated with the EmbedSample web application, then edit the appSettings section to supply values for the AccessKey, WorkspaceCollection, and WorkspaceId keys.

Code Listing 7 web.config

```
<appSettings>
    <add key="powerbi:AccessKey" value="" />
    <add key="powerbi:ApiUrl" value="https://api.powerbi.com" />
    <add key="powerbi:WorkspaceCollection" value="" />
    <add key="powerbi:WorkspaceId" value="" />
</appSettings>
```

3. Run the web application Embed Sample.
4. When the Embed Sample web application has been run, the navigation panel on the left displays the reports. Extend the reports to display the imported reports and select **Reports**. If you imported the sample “Retail Analysis Sample PBIX,” you will have something like the following image from Figure 192.

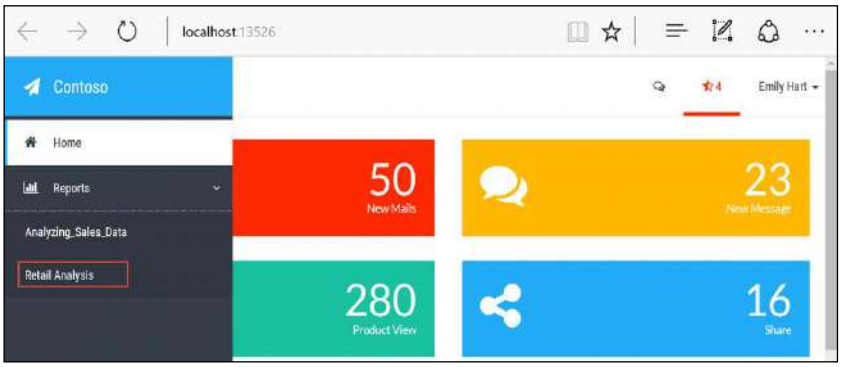


Figure 192: Power BI Embedded Sample Web Application

After selecting a report, the web application Embed Sample will look like Figure 193.

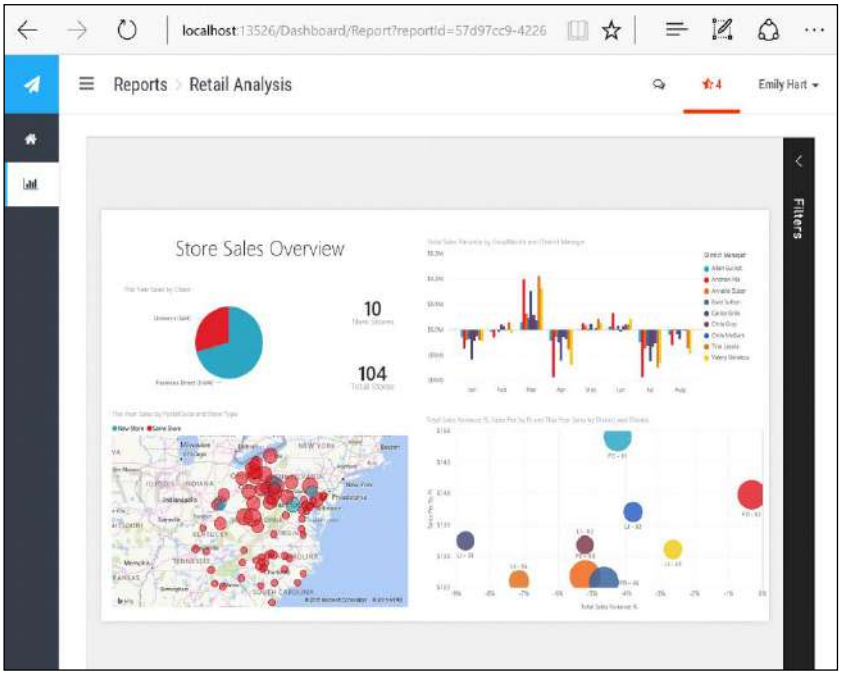


Figure 193: Power BI Embedded Sample Web Application

Explore the sample code

The sample Microsoft Power BI Embedded is an example of a dashboard web app that displays how to integrate the Power BI reports into an application. It uses a Model-View-Control (MVC) design pattern.

The sample code is separated into sections that include the filename of the PowerBI-embedded.sln solution so that the code can be easily found in the sample.

Model

The sample is composed of ReportsViewModel and ReportViewModel.

ReportsViewModel.cs in Code Listing 8 represents the Power BI reports.

Code Listing 8 ReportsViewModel.cs

```
public class ReportsViewModel

{

    public List<Report> Reports { get; set; }
```

```
}
```

ReportViewModel.cs in Code Listing 9 represents a single Power BI report.

Code Listing 9 classReportViewModel.cs

```
public class classReportViewModel

{

    public IReport Report { get; set; }

    public string AccessToken { get; set; }

}
```

Connection string

The connection string must be composed as shown in Code Listing 10.

Code Listing 10 Connection String

```
Data Source=tcp:MyServer.database.windows.net,1433;Initial
Catalog=MyDatabase
```

Using the common server and database attributes will bring a negative result. For instance, see Code Listing 11.

Code Listing 11 Connection String

```
Server=tcp:MyServer.database.windows.net,1433;Database=MyDatabase
```

View

The view displays and manages both reports and report.

Reports.cshtml interacts with Model.Reports to create an ActionLink. The ActionLink is composed of:

Title (the name of the report) and the **QueryString** (a link to the Report ID).

Code Listing 12 Reports.cshtml

```
<div id="reports-nav" class="panel-collapse collapse">

    <div class="panel-body">

        <ul class="nav navbar-nav">

            @foreach (var report in Model.Reports)

            {

                var reportClass = Request.QueryString["reportId"]
== report.Id ? "active" : "";

                <li class="@reportClass">

                    @Html.ActionLink(report.Name, "Report", new {
reportId = report.Id })

                </li>

            }

        </ul>

    </div>

</div>
```

The Report.cshtml sets up the Model.AccessToken and the Lambda expression for the PowerBIReportFor.

Code Listing 13 Reports.cshtml

```
@model ReportViewModel

…

<div class="side-body padding-top">

    @Html.PowerBIAccessToken(Model.AccessToken)

    @Html.PowerBIReportFor(m => m.Report, new { style =
"height:85vh" })

</div>
```

Controller

DashboardController.cs creates a PowerBIClient and passes to it the app token. A JSON Web Token (JWT) is generated from the Signing Key of the credentials. The credentials are used to create an application request for PowerBIClient. Once the PowerBIClient has been instantiated, GetReports() and GetReportsAsync() can be called.

Code Listing 14 DashboardController.cs

```
CreatePowerBIClient()

private IPowerBIClient CreatePowerBIClient()

{

    var credentials = new TokenCredentials(accessKey, "AppKey");

    var client = new PowerBIClient(credentials)

    {

        BaseUri = new Uri(apiUrl)

    };

    return client;

}

ActionResult Reports()

public ActionResult Reports()

{

    using (var client = this.CreatePowerBIClient())

    {

        var reportsResponse =
client.Reports.GetReports(this.workspaceCollection,
this.workspaceId);

        var viewModel = new ReportsViewModel

        {

            Reports = reportsResponse.Value.ToList()
```

```
        };

        return PartialView(viewModel);

    }

}

Task Report(string reportId)

public async Task<ActionResult> Report(string reportId)

{

    using (var client = this.CreatePowerBIClient())

    {

        var reportsResponse = await
client.Reports.GetReportsAsync(this.workspaceCollection,
this.workspaceId);

        var report = reportsResponse.Value.FirstOrDefault(r => r.Id
== reportId);

        var embedToken =
PowerBIToken.CreateReportEmbedToken(this.workspaceCollection,
this.workspaceId, report.Id);

        var viewModel = new ReportViewModel

        {

            Report = report,

            AccessToken = embedToken.Generate(this.accessKey)

        };

        return View(viewModel);

    }

}
```

Integrate a report into your app

When you have a report available, you can use an iframe to encapsulate the Power BI Report.

Figure 194 shows a snippet of powerbi.js present in the example of Power BI Embedded.

```
init: function () {
    var embedUrl = this.getEmbedUrl();
    var iframeHtml = '<iframe style="width:100%;height:100%;" src="' + embedUrl +
        '" scrolling="no" allowfullscreen="true"></iframe>';
    this.element.innerHTML = iframeHtml;
    this.iframe = this.element.childNodes[0];
    this.iframe.addEventListener('load', this.load.bind(this), false);
},
```

Figure 194: Power BI Embedded—init function

Filter reports embedded in your application

We can filter encapsulated reports by using the URL syntax.

In order to do so, we have to add a $filter to the interrogation parameters with a URL-escaped **eq** operator for the iframe src url with the specific filter. Code Listing 15 shows a query syntax.

Code Listing 15 Query Syntax URL

```
https://app.powerbi.com/reportEmbed?reportId=d2a0ea38-...-9673-
ee9655d54a4a&$filter=tableName/fieldName}%20eq%20'{fieldValue}'
```

Chapter 8 Power BI Gateway: Data Security

If the data source we have is on-premises, we can't update from Power BI automatically, which instead runs in the cloud. Therefore, we need to implement a communication channel between what is in the cloud and what runs on-premises. This is possible through the gateways that are made available by Power BI. In particular, we have the personal gateway and the on-premises data gateway, both free and directly downloadable from the Power BI portal.

The personal gateway is a Windows service that can act as a data source for Windows applications. If the user is an administrator, the personal gateway will run as a Windows service by using the same credentials of the user who installed it. But if the user is not an administrator of the machine, the personal gateway will run as a Windows application. The personal gateway service runs as a 64-bit Windows application that communicates with Power BI using the Azure Service Bus, which allows a secure channel for data transfer. It is very important to specify here that we do not need to open firewall ports in the infrastructure in order to accept incoming connections because the Azure Service Bus works in "relay" mode and therefore all communicating entities are clients of the Services bus. This means we must be open to the possibility of open outgoing connections, although incoming connections are not required. It is a little bit like the chat functionality in which all the users use chat clients, but in fact the communication among them is bidirectional.

The personal gateway cannot run in combination with other gateways, therefore it must be the only gateway present on the machine on which it is installed. It is called a personal gateway because it is meant for personal use. For instance: the user who works with Excel workbooks daily and decides to load the workbooks on Power BI would need a data refresh because they are working on-premises. We can reach the same result by installing the personal gateway in a personal laptop. The personal gateway will be linked to the personal Power BI workspace of the user who installed it. At that point, we will be able to update the data exactly as it occurred when the workbook was on-premises. But if we think about an organization with more users, even 10 users, each of them should install their own personal gateway, so that they will have the ability to update the data. To solve this, the on-premises data gateway is available, which is indeed meant for enterprise environments.

The on-premises data gateway has been designed to run on-premises 24/7. The results of the queries are sent safely (SSL) through the Azure Service Bus.

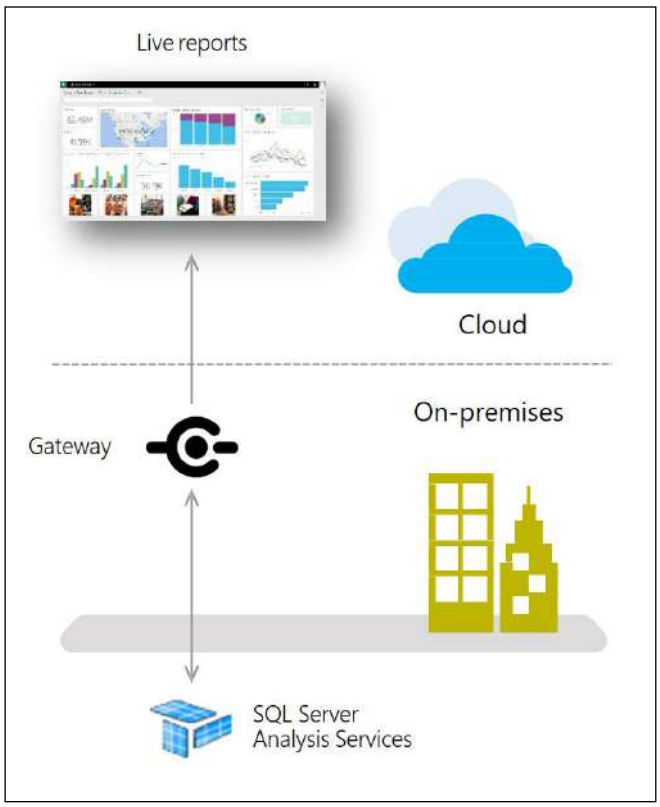


Figure 195: Power BI Gateway rule

With Power BI, the Enterprise Gateway is installed on a dedicated server. There is also a recovery procedure in case of failure. In such a case, if we know the recovery key, we can carry out the refresh even on a different server. Also remember that the personal gateway interrupts the refresh functionality when the user's is shut down. With the Power BI Enterprise Gateway, we can have a constant refresh, seven days a week and 24 hours a day. The on-premises data gateway works by using a channel of Azure Service Bus similar to how the personal gateway works. Azure Service Bus gives us the ability to interact with data sources such as Analysis Services. The gateway allows us to accept the account from which the query comes. In the specific case of a connection to a data source of Analysis Services, we can accept the so-called Effective User Name. The gateway will receive the users as well as carry out the query on behalf of the specific user, and the result of the query will depend only on the users who updated the report on Power BI. Another essential aspect of the on-premises data gateway is that it allows us to manage data sources in a centralized way. The user who installs an on-premises data gateway is the administrator of the data sources. Furthermore, the administrator can add other administrators and set up different types of data sources, such as SQL Server, Database Oracle, and many others. Access to the data sources can be set up on the basis of the destination user. As administrators, we can decide that only one category of users will be able to access that specific data source, while the others cannot.