Create and Manage Workspaces in Power BI

Introduction

A **workspace** is a **centralized repository in which you can collaborate with colleagues and teams** to **create collections of reports and dashboards**

Workspaces offer the following **benefits**:

- **Focused collaboration efforts**, you can use workspaces to **house reports** and **dashboards for use by multiple teams**
- **Ability to share** and **present reports** and **dashboards** in a single environment
- **Assurance** that the **highest level of security** is **maintained** by **controlling who can access datasets, reports, and dashboards**

Distribute A Report or Dashboard

By **creating a workspace** in Power BI, you can:

- **house your reports in one location**
- **make them shareable**
- **collaborate with other teams**
- **update reports**

Create a Workspace

Can be done in the **Power BI Service section**

Workspaces can be added to a **specific OneDrive** and then **choose whether this workspace** will be a **part of a decided capacity or not**

Dedicated capacities are **Power BI Premium** features that ensures that your workplace will:

have its own computational resources as opposed to sharing with other users

Assign Workspace Roles

Appropriate access should be given to those who need it.

Workspace roles allow you to **designate who can do what within a workspace**:

The **abilities of role types in a workspace include**:

Workspace roles

Capability	Admin	Member	Contributor	Viewer
Update and delete the workspace.	✓			
Add/remove people, including other admins.	✓			
Allow Contributors to update the app for the workspace	✓			
Add members or others with lower permissions.	✓	✓		
Publish, unpublish, and change permissions for an app	✓	✓		
Update an app.	✓	✓	If allowed [1]	
Share an item or share an app.[2]	✓	✓		
Allow others to reshare items.[2]	✓	✓		
Feature apps on colleagues' Home	✓	✓		
Manage dataset permissions.[3]	✓	✓		
Feature dashboards and reports on colleagues' Home	✓	✓	✓	
Create, edit, and delete content, such as reports, in the workspace.	✓	✓	✓	
Publish reports to the workspace, delete content.	✓	✓	✓	
Create a report in another workspace based on a dataset in this workspace.[3]	✓	✓	✓	
Copy a report.[3]	✓	✓	✓	
Create metrics based on a dataset in the workspace.[3]	✓	✓	✓	
Schedule data refreshes via the on-premises gateway.[4]	✓	✓	✓	
Modify gateway connection settings.[4]	✓	✓	✓	
View and interact with an item.[5]	✓	✓	✓	✓
Read data stored in workspace dataflows	✓	✓	✓	✓

To **assign these roles to users**, go to **workspace** that you've created,
=> and, in the **upper-left** corner, select **Access**

In the Access window you can add:

- **Email addresses of individual users**
- **Mail-enabled security groups**
- **Distribution lists**
- **Microsoft 365 groups**
- **Regular security groups**

and assign them specific roles

Create and Configure an App

You would want to add **content to your App workspace**

Content can be in the form of:

- **Reports**
- **Dashboards**
- **Datasets**
- **Dataflows**
- and so on.,

An app (application) is a **published, read-only window into your data for mass distribution and viewing**

When ready to share apps with your user, you can **publish the app** – **this process requires a Power BI license**

Consuming or viewing an app requires a **Pro license**

After **adding content to the app workspace**, you can then **create the app**
=> You can **choose to create a new report, dataset, streaming dataset, or dataflow** to the app through the **+ New button**

You can also **import already-existing reports** from **Power BI Desktop** and add them to the workspace app

You can also **configure your app** and **turn on the option to include the report or dashboard in the app when you publish**

if you do not want to include the report or dashboard in the app, you can **turn off this option**

The **Create App button** on the **upper-right of the ribbon** will allow you to create the app
=> A window will then pop-up,
=> Under its **Navigation tab**, you can **change the order in which the content is oriented for the user by creating a custom navigation pane**
=> You can also **add external content**, e.g., a YouTube video or PowerPoint slide deck

Under the **Permissions tab** you can:

- **Grant access to all users in your organization** or **choose which users have access**.
- **Give users build and share permissions**
  - which means they can **create and share content in the app**

Build Permission

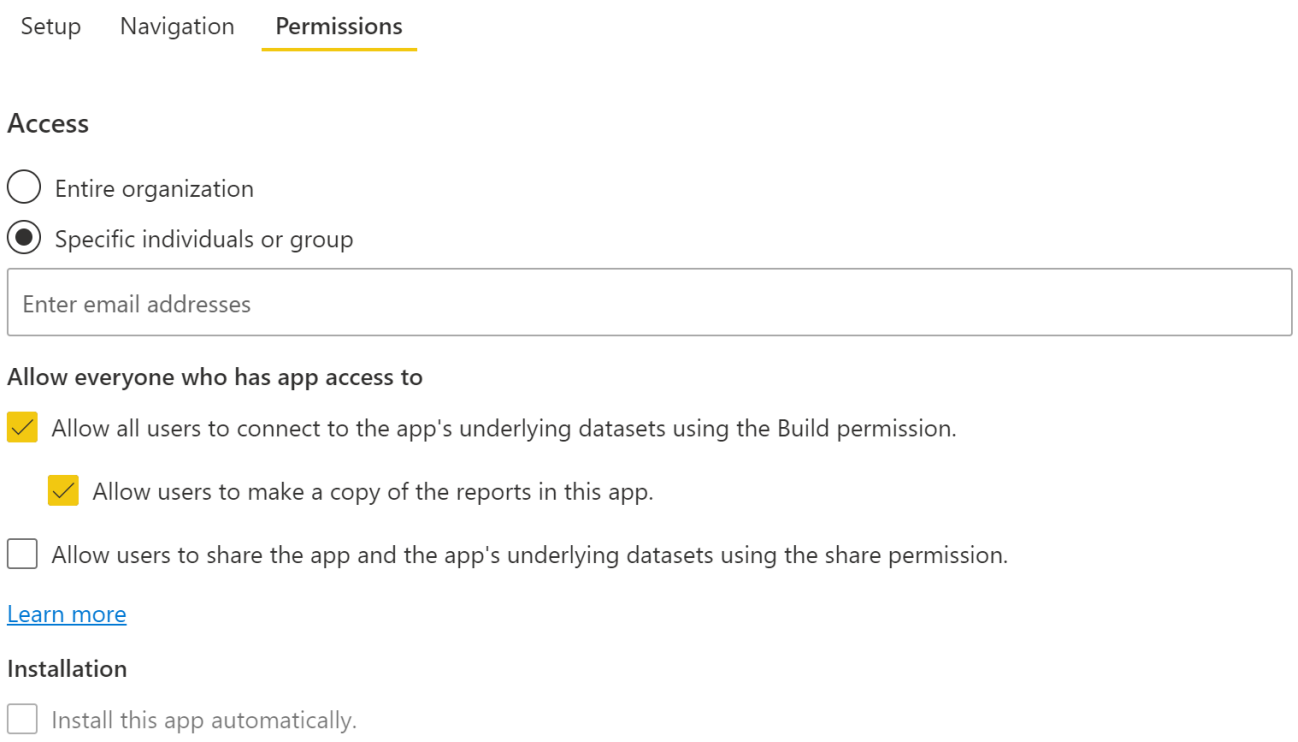
Build permissions allow your **users to connect to underlying datasets** so that they can **reuse and build their own reports** by **using the same dataset**

Build permission are **required** if your users want to **export the underlying data** or **build new content on top of the data**

You can also **allow users to only create a copy of the report to view in another workspace**, where they can **modify and delete visuals** according to their needs

You can also give your users **Share permissions** so that **they can share underlying datasets and reports**

An example of some of the **permissions can be seen below**



Update Workspaces

In order to make changes, **make the necessary updates in the reports or dashboards**

The **workspace** acts as a **staging area** where you can **make any changes you want, but they will not be added to the app**

After completion, click the **Update app** in the **upper-right corner of the ribbon**

Dataset and **dataflow updates are updated immediately**

When you make changes to an app, you can also make changes to the:

- **Setup navigation**
- **Permissions tab**

Monitor Usage and Performance

Crucial because:

- focuses your efforts for improvements
  - areas that experience worst performance – concentrate efforts for improvements in those areas
- quantifies the impact of your reports

Usage and performance metrics are available features. With these you can:

- **view who is using your reports**
- **what actions are being done on the reports**
- **what performance issues exists**

Configure and View Usage Metric Reports

Usage metric reports are available and can only be accessed by **users with the role type**:

- Admin
- Member
- Contributor

Accessible in the **pertinent workspace** through the **elipsis (…) section**

Usage metrics report will be presented in a **dashboard**.

In the **report usage tab**, you can view details such as:

- **Viewers per day**
- **Unique viewers per day**
  - does not include users who returned to the same reports multiple times
- **Shares per day**
- **Total Views**
- **Total Viewers**
- **Total Shares**
- **Total views and share rankings**
  - compares how your report is doing compared to other reports in the app
- **Views by Users**
  - details about specific user that viewed the dashboard

In the **report performance tab**, you can view metrics such as:

- **Typical opening time**
  - How long it takes, at the fiftieth percentile, to open the report
- **Opening time trend**
  - How the typical opening time changes over time
  - This metric can tell you how the report is performing as the number of users start to grow
- **Daily/7-Day Performance charts**
  - Highlight the performance for 10, 50, and 90 percent of the open-report actions every day and over a seven-day period
- **Filters for date**
  - How the performance changes according to the day

Recommend a Development Life Cycle Strategy

Development process is **iterative**

It typically requires:

- **building an initial solution**
- **testing the solution in a different environment**
- **returning to make necessary revisions**

This process is known as a **development life cycle**

Deployment pipelines manages content in dashboards, reports, and datasets between different **environments** in the **development life cycle**

Deployment Pipelines (A Power BI Premium feature)

The advantages of **using a development pipeline**:

- **Increased productivity**
  - Reuse previous deployment pipelines, ensuring efforts aren't duplicated
- **Faster delivery of content**
  - Report developments become more streamlined
  - Takes less time to get to production
- **Lower human intervention required**
  - Having the **ability to reuse deployment pipelines** means a **decreased chance of error** associated with **moving content from one environment to another**

Development Environments

Development and collaboration occur in **different stages**

Reports and dashboards are **built in and iterated on a series of controlled stages, or environments** where several tasks occur:

- **Development**
  - The location in which **dashboard developers** or **data modellers** can **build new content with other developers**.
  - The stage is first in the **deployment pipeline**
- **Test**
  - Where a small group of users and user acceptance testers can **see and review new reports**, **provide feedback** and **test the reports with larger datasets for bugs** and **data inconsistencies** before it goes into production
- **Production**
  - Where an **expansive user audience** can **use tested reports** that are **reliable and accurate**
  - This stage is the final one in the deployment pipeline

Troubleshoot using Data Lineage

The **Lineage view** feature in Power BI allows you to:

- **quickly refresh datasets**
- **see the relationships between the artifacts in a workspace and their external dependencies**

Data Lineage

Data Lineage refers to the:

path that data takes from the data source to the destination

Lineage view feature is **crucial** because it:

- **Simplifies the troubleshooting process**
  - you can see the path the data takes from source to destination and determine pain points and bottlenecks
- **Allows you to manage your workspaces and observe the impact of a single change in one dataset to reports and dashboards**
- **Saves time**

- simplifies tasks of **identifying reports and dashboards that haven't been refreshed**

Use the Lineage View

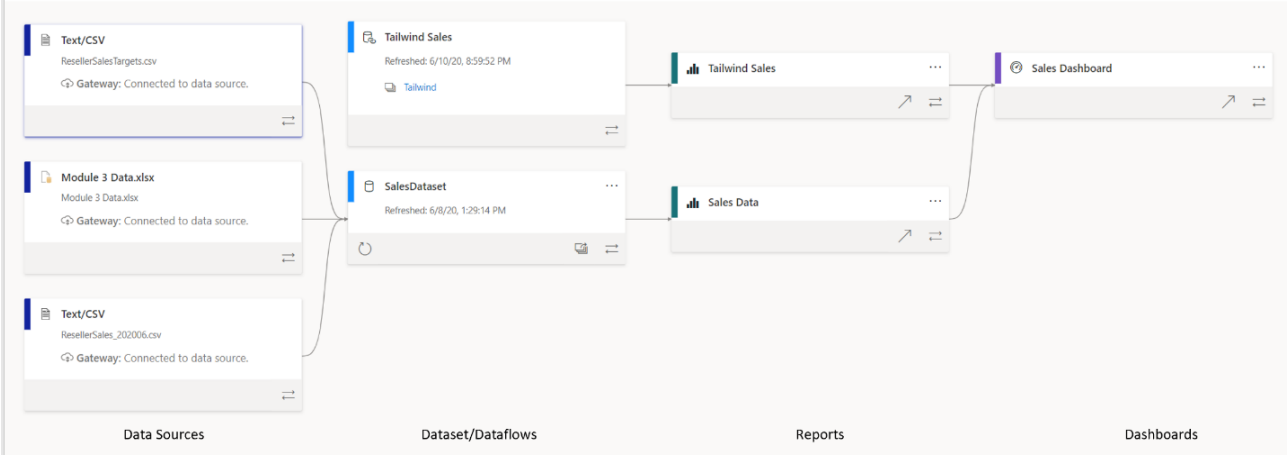
Lineage view is only accessible to:

- **Admin**
- **Contributor**
- **Member**

roles

It **requires a Power BI Pro license** and is only **available for app workspaces**

The image below is an example of a **data lineage** for an **example workspace**



The view shows **all of the artifacts in the workspace**

Artifacts include **data sources, datasets, and dataflows, reports, and dashboards**.

Each **card** represents **an artifact**

The **arrows in between the cards** represent the **flow of data or the relationship between different artifacts.**

By following the arrows from left to right, you can **observer the flow of data from the source to the destination**

Data Sources

This group of cards tells you the **type of data source** (e.g., Text/CSV) and the **Gateway**

Double clicking will get you more details about the file path and connection status

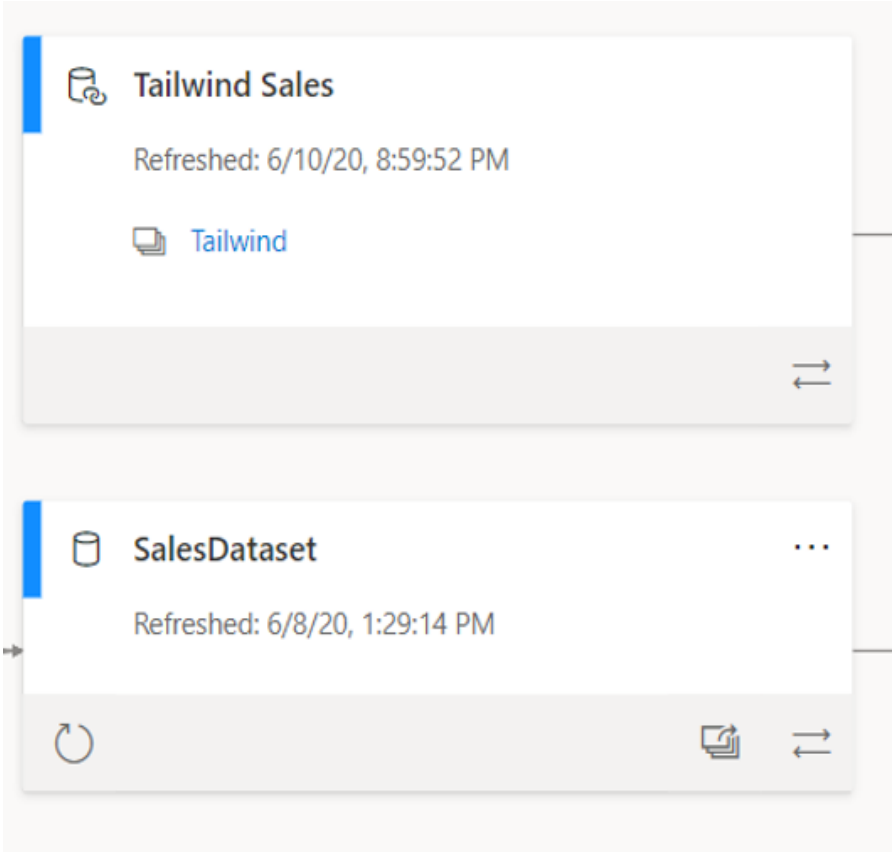
Gateway

tells you the **source of your data**

If you are connected through an **on-premise data gateway**, this card will **tell you more information about the gateway**

Datasets/Dataflows

often **datasets/dataflows** can **connect to external data sources**, such as **SQL server**, or to external datasets in **other workspaces**



As seen above, the card tells you:

- **when the dataset was last refreshed**

and you can **manually refresh the dataset** by **selecting the circular arrow icon on the lower-left corner of the card**

This ensures that **your dataset refreshes are quick and uncomplicated**

Double-clicking on the card allows you to see the **metadata**, which tells you:

- the **sensitivity**
- **who the dataset was configured by**
- **last refresh date**
- **names and count of tables within this dataset**

You can also **view the Impact Analysis window**

Impact Analysis window

A window that pops-up after clicking the **overlapping windows icon on the lower-right corner of a dataset card**

Tells you:

- **how many workspaces, reports, and dashboards that this dataset is a part of**

- **how many views that this dataset has gathered**

Allows you to **pinpoint datasets that are not being used or looked at**.

Reports and Dashboards

Possess similar functionality as the **data source** and **dataset cards**

You can **enable/disable whether you want to include this report or dashboard within the app**

Configure Data Protection

As enterprises grow, so does their data.

Often strict requirements and regulations, must be **applied to ensure that this sensitive data is secure**.

Hence, **sensitive data must remain secure**.

Different ways this can be done:

- **Use Microsoft sensitivity labels** to **label dashboards, reports, datasets**, and **dataflows** by using the same **taxonomy** that is **used to classify and protect files in Microsoft 365**
- Add more protection measures such as **encryption** and **watermarks** when **exporting data**
- Use **Microsoft Cloud App Security** to **monitor and investigate activities in Power BI**

Sensitivity Labels

Specify which data can be exported

Configured externally to Power BI and Power BI allows you to quickly use them in your reports and dashboards

Labels allow you to **define and protect content**, even outside of Power BI

Datasets, reports, and dashboards can use **sensitivity labels** to **protect information/data**

There are **varying levels of Sensitivity labels**:

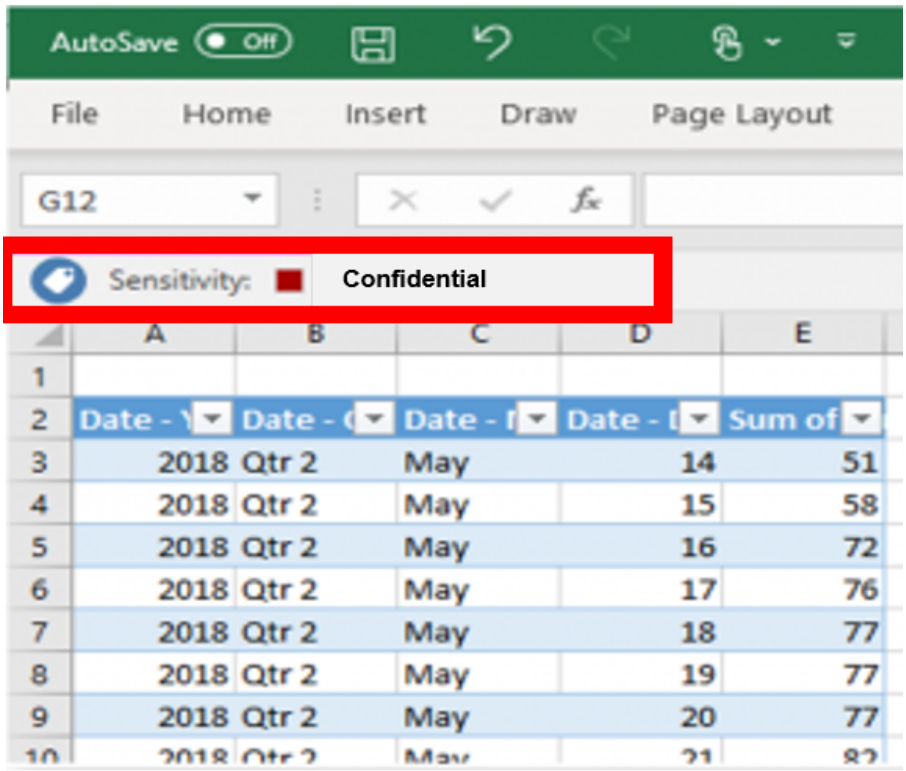
- None
- Personal
- General
- Confidential
- Highly Confidential

You can also **define your own security label**

The **varying levels of Sensitivity labels** are **applied into individual datasets, reports, or dashboards**

Hence when someone takes action, e.g., **attempts to export a Confidential database to an Excel file**

if you are an **authorized user**, you will **be able to access the data** and see the following view when **exporting into Excel**. As **sensitivity labels are enforced** when **exporting to Excel, Powerpoint and PDF**



	A	B	C	D	E
1					
2	Date - Y	Date - Q	Date - M	Date - D	Sum of
3	2018	Qtr 2	May	14	51
4	2018	Qtr 2	May	15	58
5	2018	Qtr 2	May	16	72
6	2018	Qtr 2	May	17	76
7	2018	Qtr 2	May	18	77
8	2018	Qtr 2	May	19	77
9	2018	Qtr 2	May	20	77
10	2018	Qtr 2	May	21	82

if you are an **unauthorized user**, you **will be denied access to see the data**
this **verification** ensures that **only appropriate users have access to view the data**, which helps you **make sure that your data is secured**

Manage Datasets in Power BI

Introduction

One dataset can be **used to create multiple Power BI reports**

Useful as **preparing and cleaning data can be time-consuming**, **sharing datasets can be a productivity boost for report authors**

sharing of datasets need to be **actively managed** for **optimal organizational performance**
hence you should:
automate the refresh process so that it becomes more efficient and users always have access to the latest data

promote certain datasets over others so that users can clearly identify the best dataset to use

Management also implies:

- **implementation of PARAMETERS within datasets**
  - helps with decision making and solving business problems
  - parameters can be used to **change the server or database name of your dataset** or a **file path for a data source**
  - can also be used to **configure incremental refreshes of your data**
  - also to **run "what-if" scenarios** and **conduct scenario-type analysis of data**

- **setting up and maintaining a GATEWAY**
  - so that users can **access on-premises data source from the cloud**

- you will also have to **prepare for potential issues that might arise regarding this gateway**
  - could **interrupt user access to the datasets**
    - inability to access data leads to inability to do their job
    - organizational decision-making grinds to a halt

Create Dynamic Reports with Parameters

Dynamic reports are reports in which the **data can be changed by a developer according to user specification**

Dynamic reports are **valuable** because **single reports** can **be used for multiple purposes**

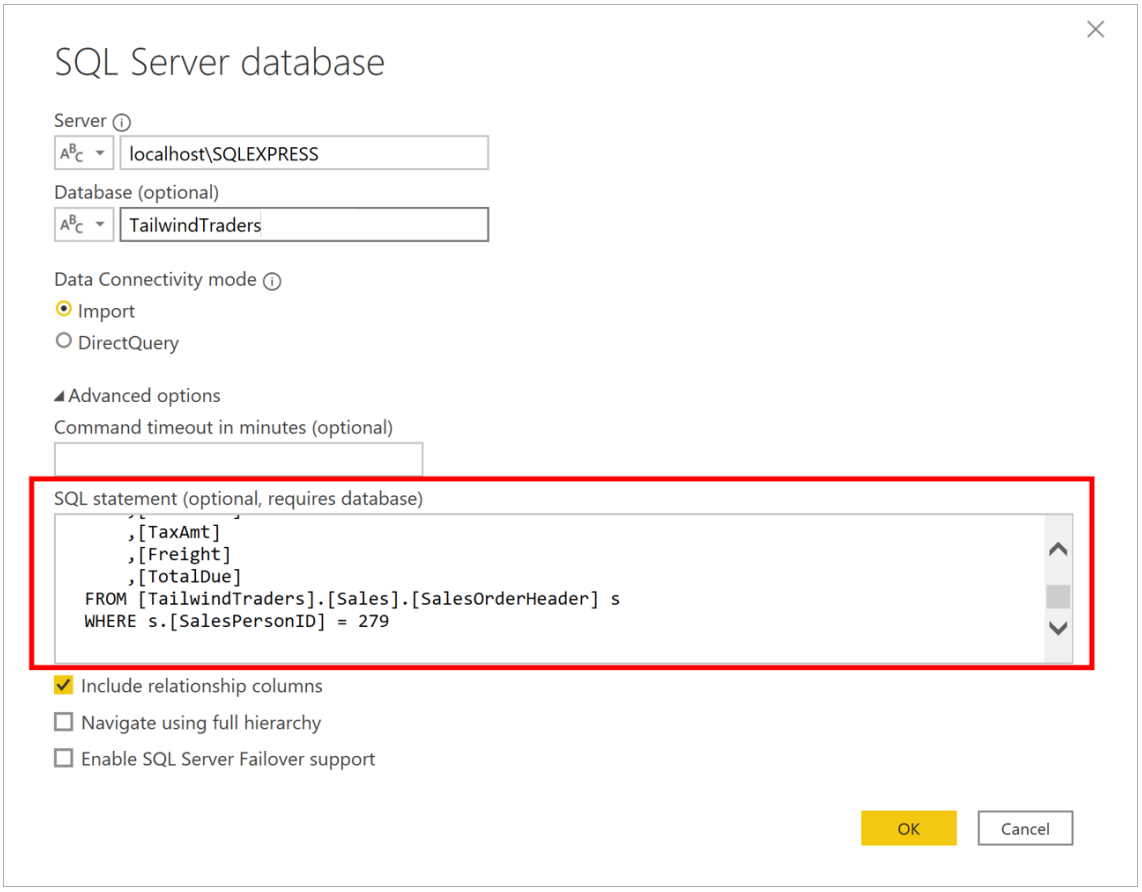
You can use **parameters** by **determining the values that you want to see data for in the report**
report updates accordingly by filtering the data for you

Dynamic reports **give users more power over the data** that is displayed in the reports, they can **change the data source** and **filter the data by themselves**

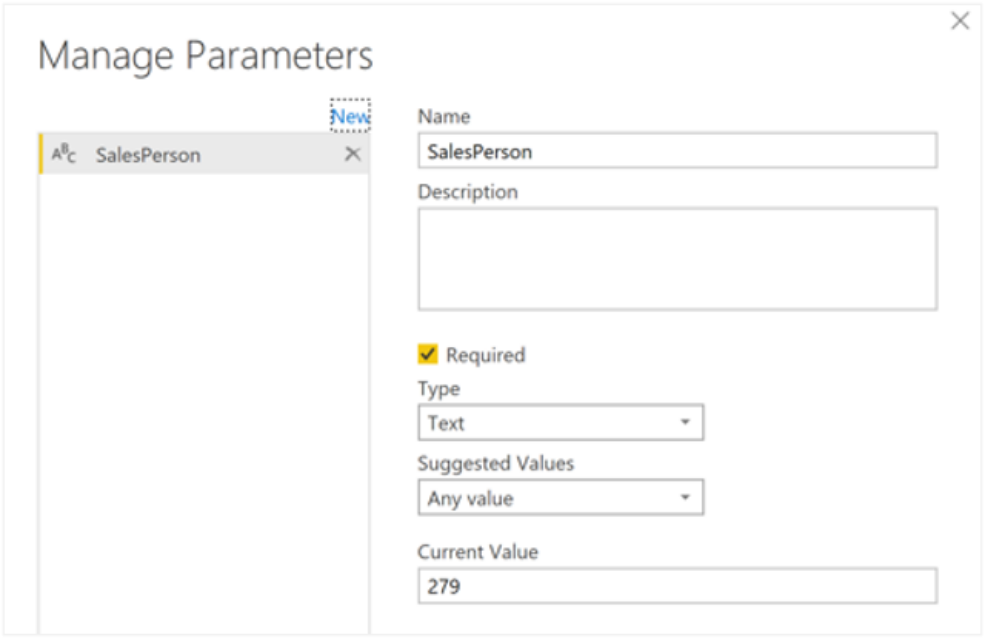
Create Dynamic Reports for Individual Values

To **create a dynamic report**, you first need to write an SQL query. Then you will use the **Get data feature** in **Power BI desktop** to **connect to the database**

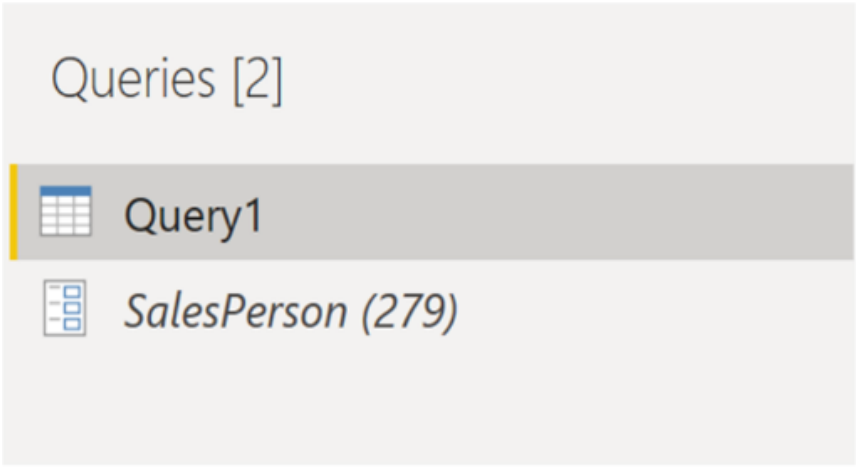
An example of a **dynamic report** being **queried using the Get Data for SQL server** selection is seen below:



You then **select Manage parameter => New parameter** to create a new parameter as seen below

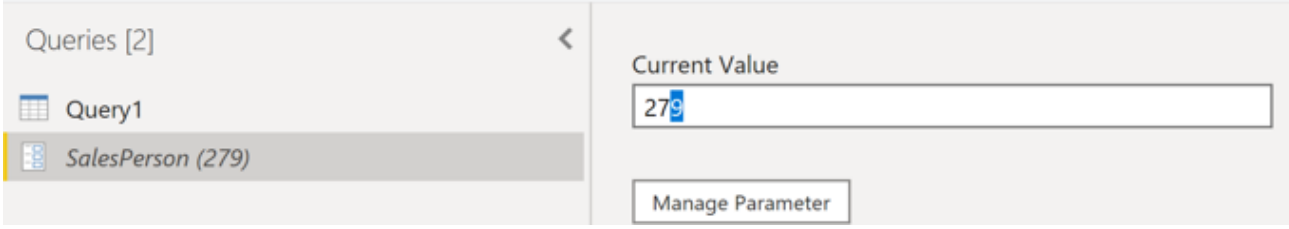


The result is a **new query** for the parameter **being shown to have created**, as seen below:



Click on the **Advanced editor** which allows you **replace the existing value and enter a parameter name**

All of this then allows you to **search by a stated query as the SQL query has been configured to pull information by a parameter**



After executing the above value, the **data for the value set to the parameter (in the example 279) is then pulled from the SQL database**, the result of which can be seen below:

	SalesOrderNumber	SalesOrderID	SalesPersonID	OrderDate
1	SO43659	43659	279	31/05/2011 00:00:00
2	SO43660	43660	279	31/05/2011 00:00:00
3	SO43681	43681	279	31/05/2011 00:00:00
4	SO43684	43684	279	31/05/2011 00:00:00
5	SO43685	43685	279	31/05/2011 00:00:00
6	SO43694	43694	279	31/05/2011 00:00:00
7	SO43695	43695	279	31/05/2011 00:00:00
8	SO43696	43696	279	31/05/2011 00:00:00

What-If Parameters

can be used to **run scenarios and scenario-type analysis on the data.**

Powerful additions to your **Power BI data models** because they **enable you to look at historical data to analyse potential outcomes**

Can be used to help you **look forward, to predict or forecast what could happen in the future**

e.g.,
determine the **effect of increased sales to deeper discounts,** or to **let sales consultants see their compensation if they meet certain sales goals or percentages**

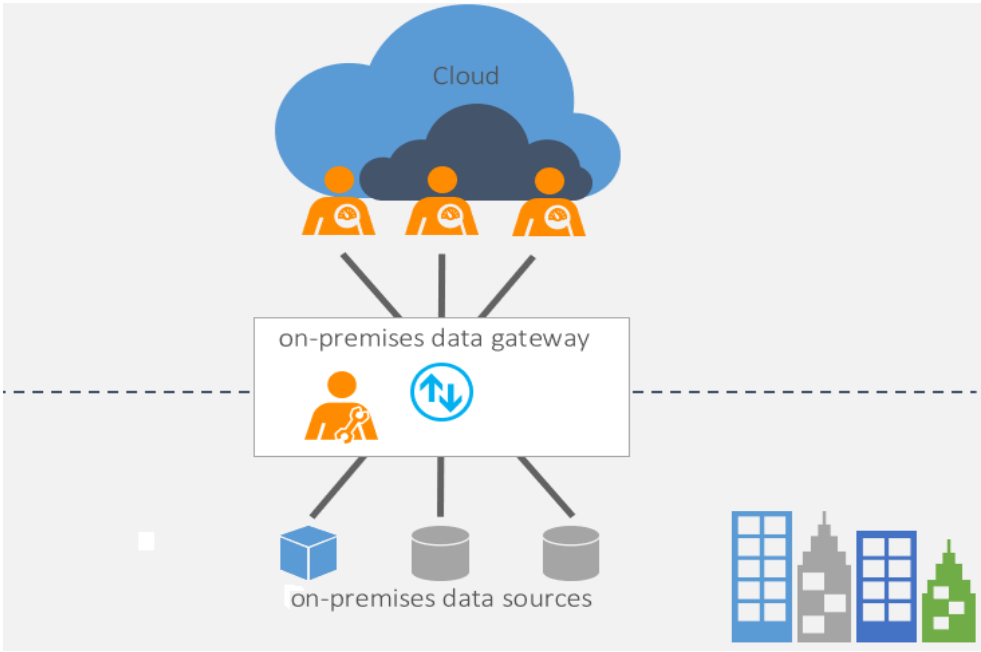
Use a Power BI Gateway to Connect to On-Premise Data Sources

Gateway software acts like a bridge

It allows organizations to **retain databases** and **other data sources on their on-premise networks** and **access that on-premises data in cloud services**, such as Power BI and Microsoft Azure Analysis Services

A **gateway** **facilitates quick, behind-the-scenes communication** that **flows from a user in the cloud** to your **on-premises data source** and then **back again to the cloud**

A visual depiction of a gateway can be seen below:



Two types of **on-premises gateways** are:

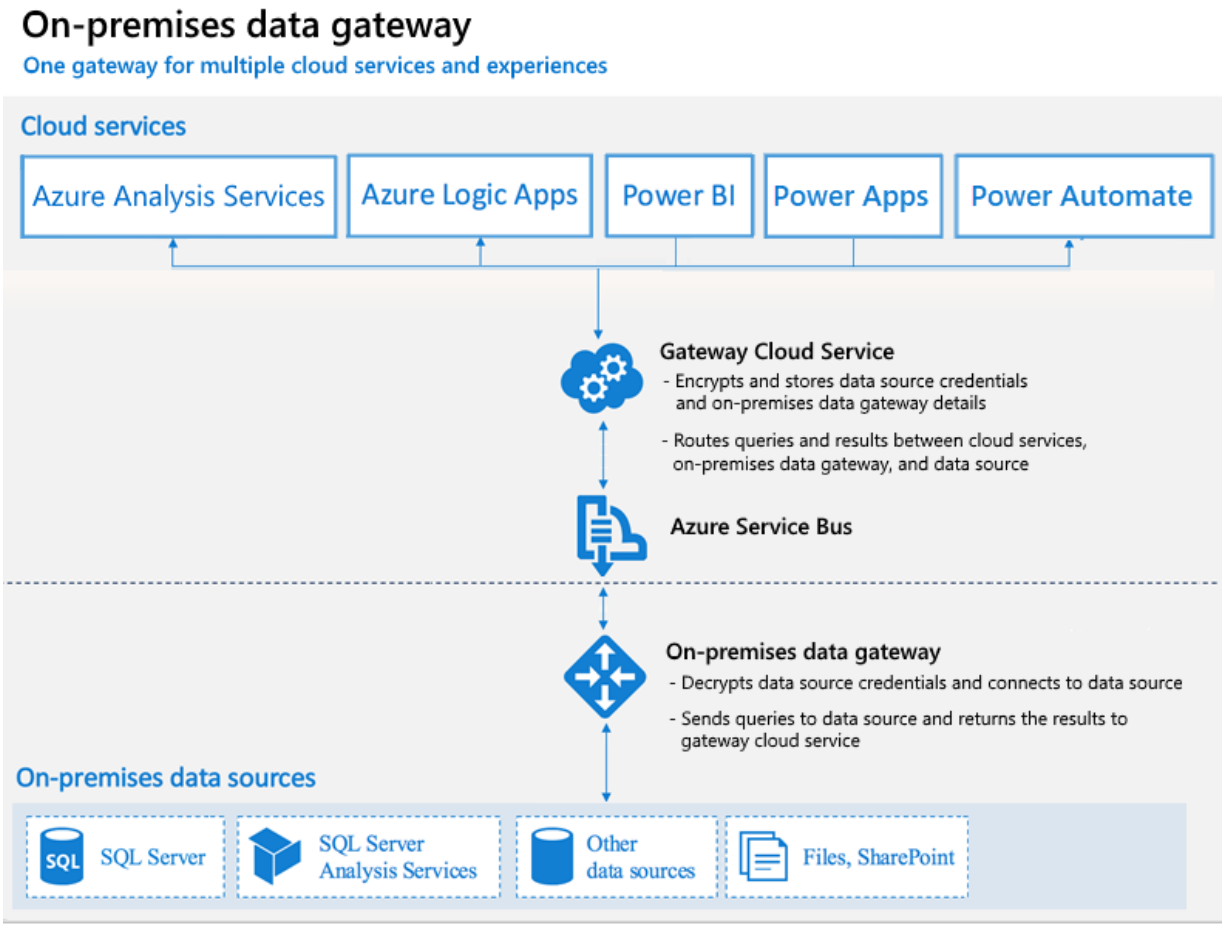
- **Organization mode**
  - Allows **multiple users** to **connect to multiple on-premises data sources** and is **suitable for complex scenarios**

- **Personal mode**
  - Allows **one user** to **connect to data sources**.
    - This type of gateway can be used only with Power BI
    - Can't be shared with other users
  - **Suitable in situations** where **you're the only one in your organization that creates reports**

Use an On-Premise Gateway

Connecting to an on-premise data source **requires installation of an on-premises data gateway** and then **configure it to suit your organizational needs**

The following occurs when **connecting/connected to an on-premises data source**



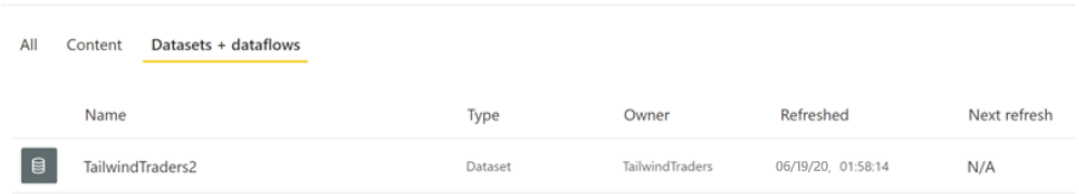
Configure a Dataset Scheduled Refresh

The **scheduled refresh feature** in Power BI allows you to:

define the frequency and time slots to refresh a particular dataset

Scheduling the refresh of your data will:

- **save you time** because **you don't have to manually refresh the data**
- **ensure that users** can **access the most up-to-date data**



An example of a query whose **last refresh date was 10 days ago**, and **does not have an upcoming data for a scheduled data refresh.**

Set-up a Refresh Schedule

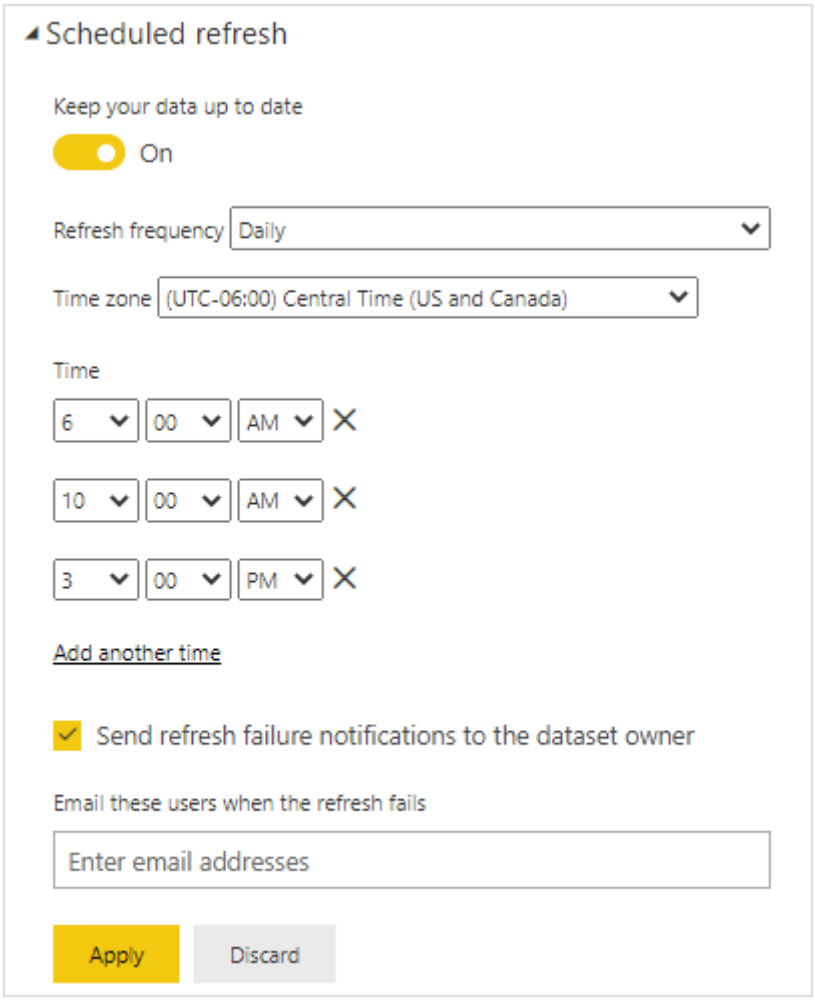
To set-up a **refresh schedule**, you need to have **create a gateway connection**

Go to the **Datasets + Dataflows page**, hover over the dataset and **select the Schedule refresh icon**
=> on the **Settings page,** turn **on the Scheduled refresh feature**
=> select the **Refresh frequency,** and **ensure that the correct time zone is selected**
=> add the time(s) that **you want the refresh to occur**
=> when you **finished configuring the scheduled refresh**, select **Apply**

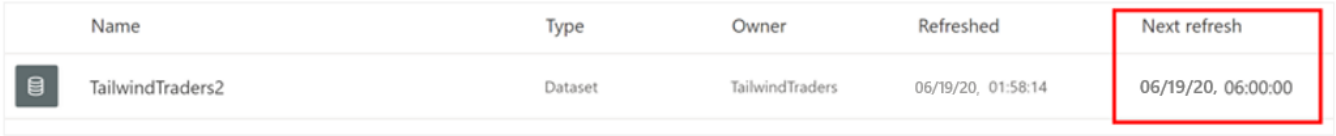
You can **configure up to:**

- **8 daily time slots** if your **dataset is on shared capacity**
- **48 time slots** on **Power BI Premium**

The example below illustrates the **sales data being refreshed on a DAILY basis at 6:00 AM, 10:00 AM and 3:00 PM**



When you have **configured a refresh schedule**, the **dataset settings page** informs you of the **next refresh times**, as shown in the following image:

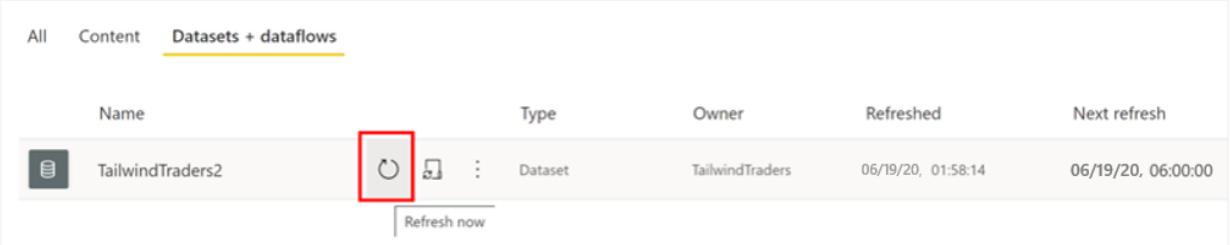


Perform an On-Demand Refresh

You can **refresh a dataset at any time** by **performing an on-demand refresh**.

This type of refresh **doesn't affect the next scheduled refresh time**

This might be used when **testing your gateway** and **data source configuration**



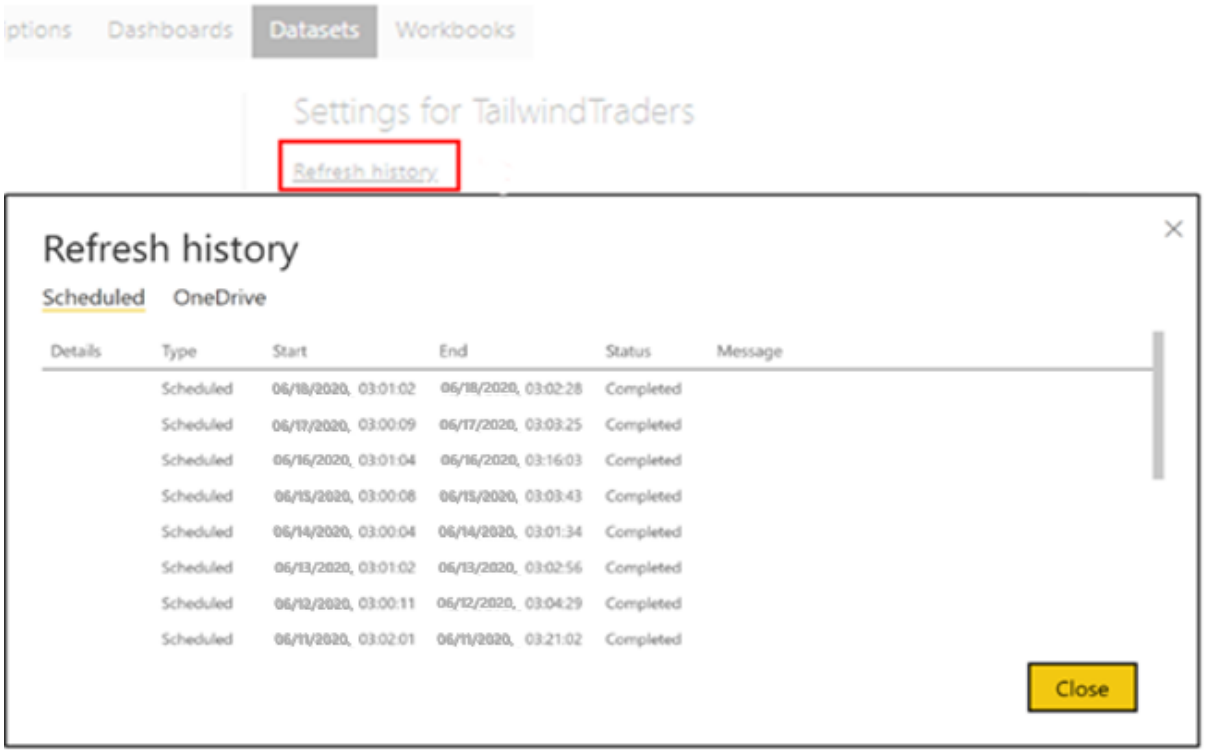
Check the Refresh Status and History

This feature is helpful when you want to find out:

- **when the last refresh occurred** and **when the next one is scheduled**
- It is also **good practice to check the status of your dataset occasionally** to **check if refresh errors have occurred**

Good to check **refresh history occasionally** to **review the success or failure status** of **past synchronization cycles**

An example can be seen below:



Configure Incremental Refresh Settings

The **incremental refresh** feature in Power BI is a **popular feature** because it **allows you to refresh large datasets quickly** and as **often as you need**, **without having to reload historical data each time**

Incremental refresh should only be **used on data sources and queries that support query folding**

Incremental refresh allows the **following benefits**:

- **Quicker refreshes** – Only **data that needs to be changed gets refreshed**
  - e.g., if you only need to refresh data **from the last 10 days** you can do so instead of refreshing the entire dataset
- **More reliable refreshes**
  - Do not need to **keep your long-running data connections open to schedule a refresh**
- **Reduced resource consumption**
  - Since you only need to **refresh the smaller amount of data**, the **overall consumption of memory and other resources is reduced**

Hence, you can **define an incremental refresh policy to solve this business problem**. This process involves the following steps:

- **Define the filter parameters**

Large datasets are commonly filtered when imported to Microsoft Power BI

For **incremental refresh**, the datasets are **filtered by two date/time parameters**:
RangeStart and **RangeEnd**

These two **parameters** are used as a **filtering window** because they **restrict the used data to the range that is listed in the start and end dates**

- **Use the parameters to apply a filter**
- **Define the incremental refresh policy**
- **Public changes to Power BI service**

Manage and Promote Datasets

Useful for when you have many different datasets that can be accessed by users

This allows you to **direct users to the most up-to-date and highest-quality datasets in your workspace**

Hence, it is important to **create and share optimized datasets** and **endorse those datasets** as the **one source of truth**

Report creators can then **reuse those endorsed datasets** to **build accurate, standardized reports**

Power BI provides **two ways to endorse your datasets**:

- **Promotion**
  - **Promote your dataset** when they're **ready for broad usage**.
  - Power BI **Admins** have **permissions to promote datasets**

Promote a Dataset

You can only **promote a dataset** if you're a **Power BI Admin User** or the **Owner of that dataset**

A badge in the **endorsement column** for that dataset will appear, indicating that **its ready for viewing by all of your users**.

- **Certification**
  - **Request certification** for a **promoted dataset** from an **admin user** that is defined in the **Dataset Certification tenant admin setting**

    - Certification **adds another layer of security for the dataset**
    - Certification can be a **highly selective process**
        - So that the **only truly reliable and authoritative datasets are used across the organization**

Certification requires **permission** from the **dataset owner** to **access to the dataset**

Certify a Dataset

Can be done by **selecting the Certified option** in the **Endorsement settings** selection

Technically,
A **certified dataset** is **above** a **promoted dataset**

Query Cache

Using the **local caching services** of **Power BI** to **process query results**

Instead of **relying on the dataset to calculate queries**, which when **overloaded** can **reduce performance**

you can use **CLOUD RESOURCES** on your **PREMIUM capacities** on the Power BI service to **load your report, and thereby, ensure constant performance**

Query Caching

a **local caching feature** that **maintains results on a user and report basis**.

This service is only available to users with **Power BI Premium or Power BI Embedded**

When using **query caching**, the **query results are only specific to a user**, and you can only use **query caching on a specific page of a report**

Several benefits of using **query caching include**:

- **Improvement of the performance** of **reports, dashboards, and dashboard tiles** by **reducing loading time** and **increasing query speed**
- **Respects bookmarks** and **default filters**, so enabling query caching keeps any presently created bookmarks
- **Cached query results** are specific to the user
- **All security labels** are followed

***** Row-Level Security

Power BI can help you secure reports and workspaces by **allowing you to share them to active directory users and groups**

You can also **share a single report** but **have users see different data according to their job role**

An example scenario

employees can only see sales from their department, Maria from Clothing should **ONLY BE ABLE to see sales from the CLOTHING department**

Row-level security uses a **DAX** filter as the **core logic mechanism** to ensure that **only the appropriate person can view the appropriate records**

DAX filters are (usually) **applied on the dimension table**, hence **Row-level security performs better** when the **data is organized in a star schema**

Two ways of **implementing row-level security** in Power BI:

Row-level security involves several configuration steps, which should be completed in the following order:

1. Create a report in Microsoft Power BI Desktop
    a. Import the data
    b. Confirm the data model between both table
    c. Create the report visuals

2. Create RLS roles in Power BI Desktop by using DAX

Select the Modelling tab, then select **Manage Roles**
=> On the **Manage roles** page, select **Create**

Power BI **row-level security (RLS) uses DAX** to **control who can see which data**.

Consider it as **always adding another filter to the appropriate users**, regardless of the filters, slicers, or interactions that the **user chooses on the Power BI report**

3. Test the roles in Power BI Desktop

You can **validate that the filter is working** by selecting the **Modelling tab** and then selecting **View as Roles**
=> e.g., select a **role to view as** (e.g., “Game”). **The report now renders as if you were in that role**, and you will only see the records that are **included in the Game Department**

4. Deploy the report to Microsoft Power BI service

5. Add members to the role in Power BI service

In the **Row-Level Security screen** accessible after clicking on the **elipsis (…) icon** and then the **Security tab**, you can add:

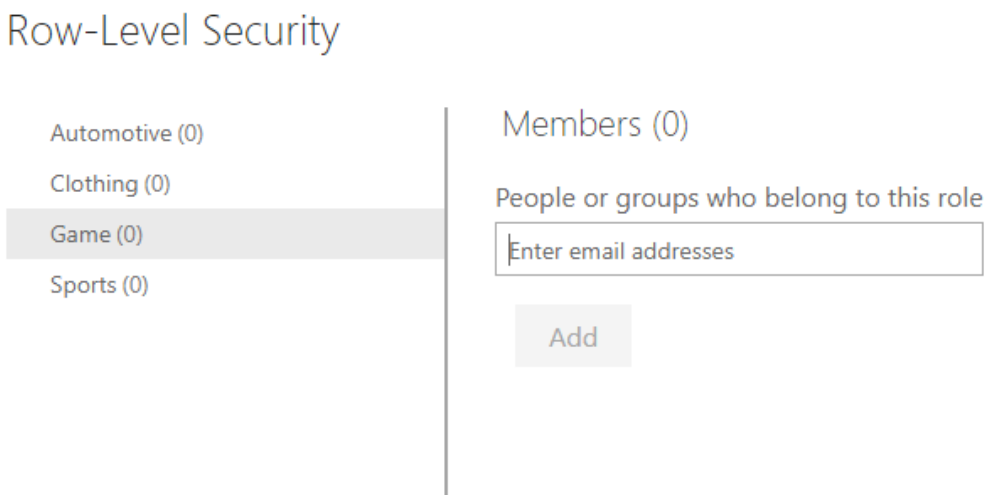
- **Microsoft Azure Active Direction (Azure AD) users**
- **Security groups**

to the **created security roles**

row-level security (RLS) will **only apply to members who are ADDED TO THE ROLE ON POWER BI SERVICE.**

Hence, if **members were not added to the role**, but **have access to the report**, **RLS will NOT apply to them**

An example of configuring **Row-Level Security (RLS) on the Power BI Service page** is seen below:

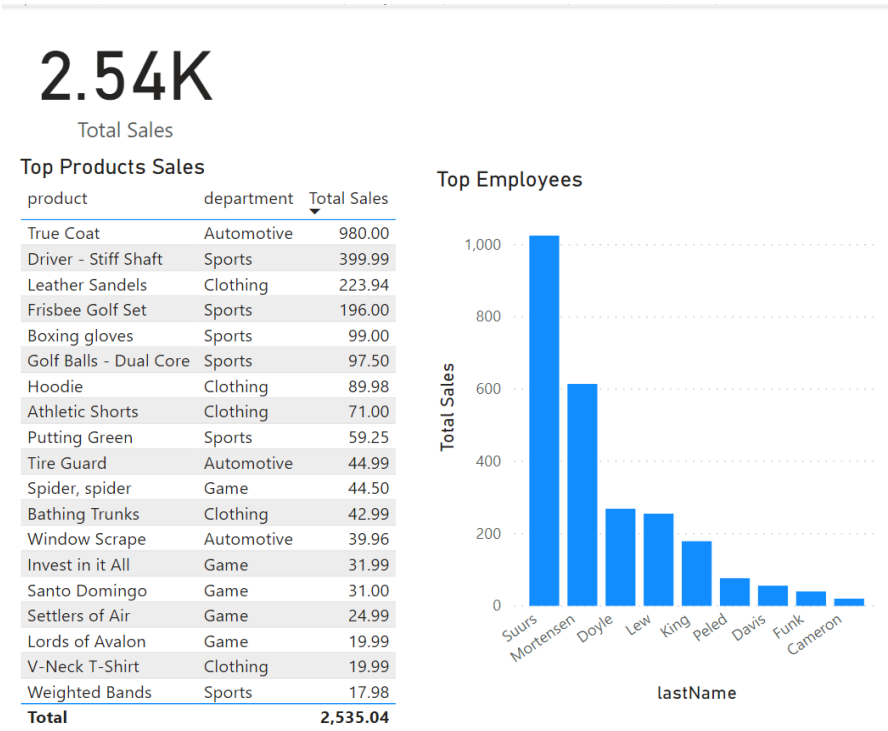


6. Test the roles in Power BI service

- **Static method**

Uses a FIXED/STATIC value in the DAX filter

Once you've reached **step 2**, the **view should be similar to this**



Hence, we will be **limiting visibility of the columns** so that **only employees of a specific department can see their own sales**

On the **Manage Roles** page:

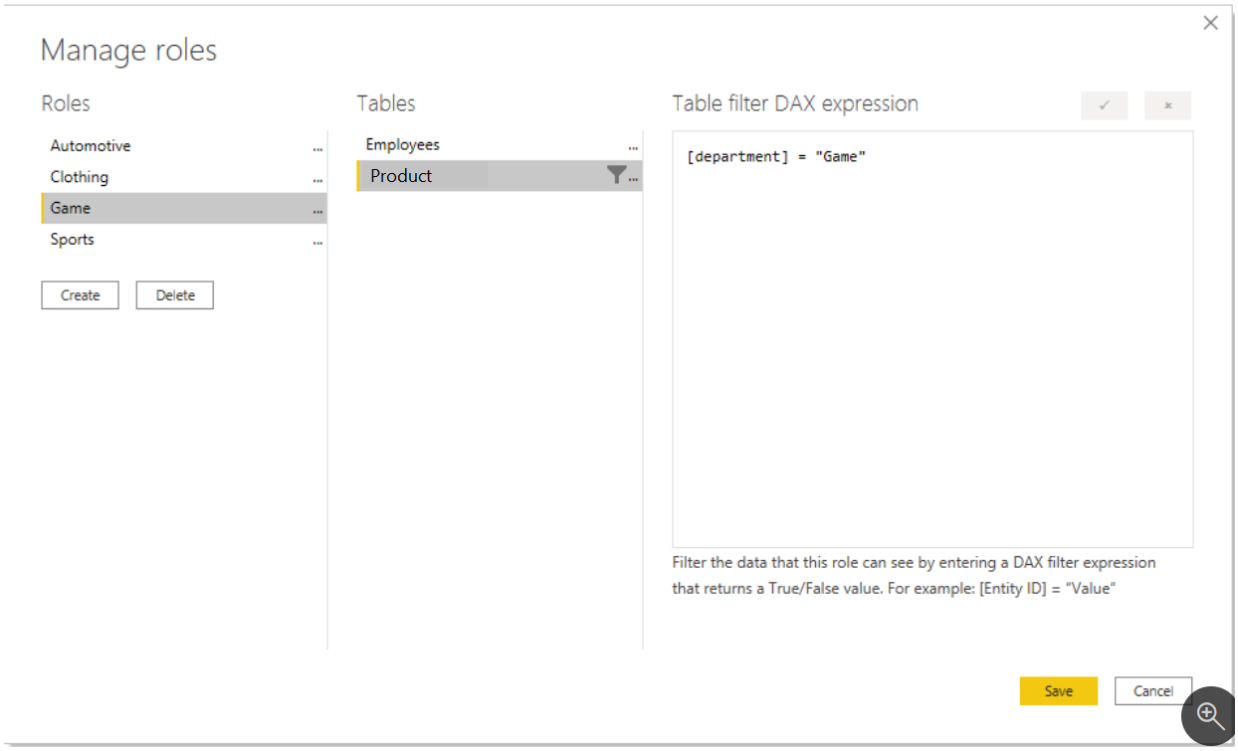
- **Create a role for each department**, and then **add a DAX expression to it**
- For instance, you can **create a role called Game**

- And then add a **DAX expression** [department] = “Game”. Then **whenever a member of that role interacts with the report**, Power BI **will add a filter to their interaction**, thus **limiting what they see**

NOTE: **a fixed value is used in the filter on the right side of the equal sign**, e.g., (“Game”).
Hence, the intention is that
if you ever need to add a category, you will need to create a new role with a new value in the DAX expression

Notice that the **DAX filter** is **applied on the dimension table**.

An example is seen below,



- **Dynamic method**

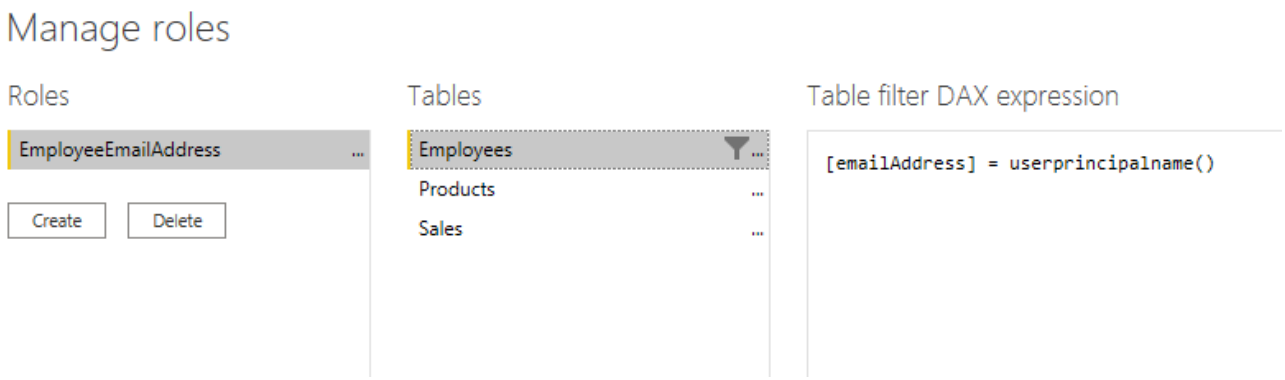
Uses a DAX function

You can set up **row-level security (RLS)** only once, **WITHOUT the need to continue maintaining it dynamically**

an example scenario:
Configuring row-level security (RLS) to **only show sales to the person who made them**.
e.g., Russell King has made four sales. When viewing the report, Russell should only see the sales that he is responsible for and no other sales

Row-level security (RLS) can then be configured in the same manner as before, with one change

Instead of **creating four roles**, we only need to **create one role**. The **DAX filter** for that **one role** would look similar to the following image:



Notice that **instead of the fixed string,** such as Game or Clothing, this **uses a DAX function in the row-level security filter**

The **userprinciplename() function** will
compare the email address from the Employees table with the **email address that the user entered when signing into the Power BI service**

If he used a company email address **similar to the one stored in the Employees table to sign-in** to **Power BI service**, the **system will compare that value** to the **email address in the Employees table**

However, this **ASSUMES** that **a relationship has been created between Employees and Sales**
if it has then Russell will only see his four sales

Hence the **USERPRINCIPALNAME()** DAX function will:
tell you the username of the person who is signed in to Power BI service