

Five Ways to Amplify Power BI with Azure Synapse Analytics



Five Ways to Amplify Power BI with Azure Synapse Analytics

3 /

Introduction

21 /

#3: Explore your data lake using a fully managed serverless endpoint

36 /

Summary

6 /

#1: Build and analyse Power BI dashboards directly from Azure Synapse

30 /

#4: Build code-free data pipelines to integrate more data sources and enrich insights

37 /

Next steps

15 /

#2: Manage your data lake and build data marts for BI reports

33 /

#5: Create a secure data warehouse and connect to your Power BI dashboards

Introduction

Today, all companies are looking to do more with their data. Organisations are looking at moving beyond reporting, to provide users with greater insights from their data. Using Azure Synapse Analytics, organisations can utilise the cloud as the central repository for all of their data. Information is not limited to just columns of data; with Azure Synapse, companies can move and analyse other information, including images, sound or caches of PDF documents. This information can be organised to maximise the company's ability to access the data and provide insightful information with Power BI. Data used in Power BI can come from the entire data store created by Azure Synapse. Azure Synapse is used as a central data repository, as it contains the tools needed to load data into a data lake, which can be used for machine learning or a virtual database for Power BI. For very large amounts of data, Azure Synapse can create data warehouses and data marts with optimised resources used for Power BI reporting.

Once Azure Synapse has processed the data needed in an organisation, the next step is to implement a data visualisation environment, such as Power BI. Power BI empowers users to gain more insight into their data by creating an environment where people can interact with their data to find answers to business questions. Developing the data models used in Power BI is the first step to providing access to data from all parts of the organisation. Data can come from IoT devices and social media, as well as a number of applications the organisation uses in its day-to-day environment. The data that needs to be reported upon may be streamed, perhaps with Azure Event Hubs, a data warehouse, database or data lake. The data needs to be in an accessible location so that it can be used not only for reporting, but for advanced analytics AI/ML processes. If the data is stored in one location, it will be accessible not only for Power BI reporting but also for data science modelling.

As lots of data is needed for this task, the data needs to be stored in a secure, scalable location, and to meet these requirements, the data should be stored in the cloud. Centralising your data storage in Microsoft Azure provides a secure, scalable, fault-tolerant location. Today, many companies are looking to store and analyse data that may not be text. Video, pictures and sound files are often analysed using dedicated AI libraries. For this wide variety of data, a data lake is often the solution that best fits these needs.

A data lake is the logical place to store data from daily transactions, streamed GPS location data and image files, along with any other data the organisation wishes to keep. How the data is stored determines the ability for analysis. Azure provides a hierarchical file structure that can be used for analysis as well as Power BI. Data scientists can use this data to create models to provide additional insight, which you can use within Power BI. Data scientists can also use Power BI to help them quickly analyse data to determine which elements are needed in a model. Power BI offers many different business intelligence features, many of which have been added recently, such as goals, key influencers, decomposition trees and smart narrative, which can provide additional insight into the factors contributing to organisational success.

The task of managing data used for analysis can be challenging as the data can be quite large and can include structured data stored in a data warehouse or unstructured data such as images, and anything in-between. Management of the data will need to include a method for adding more data and organising it in a format needed for business intelligence tasks. Extract, load and transfer methods can be used to gather data from different parts of the organisation or third-party applications. Based on the skills in your organisation, the data transfer may happen with low-code solutions, data flows or Python code. It does not matter where the source of the data is as you need to gather it to be able to provide operational insight. To provide answers to business questions, your organisation needs to develop solutions to provide all of the information needed for accurate decision making. Machine learning elements can further enhance the decision-making process by providing predictive analysis to determine the future state of the company or to point out anomalies with performance that may have gone undiscovered.

The task of providing this data so that Power BI can access it and the insights others may provide with the data is all available in one tool, [Azure Synapse Analytics](#) – a unified analytics platform. Power BI can be used to report on data from a data lake and Azure Synapse provides the capability to ingest data from multiple solutions, create large data warehouses, manage data lakes, build targeted data marts for BI reporting audiences, develop machine learning solutions and provide Power BI the information needed to move from providing reporting to providing business analytics solutions.

In this article, we are going to look at five different ways that you can use Azure Synapse and Power BI:

- #1: Build and analyse Power BI dashboards directly from Azure Synapse
- #2: Manage your data lake and build data marts for BI reports
- #3: Explore your data lake using a fully managed serverless endpoint
- #4: Build code-free data pipelines to integrate more data sources and enrich insights
- #5: Create a secure data warehouse and connect to your Power BI dashboards

These are five different ways in which you can use Azure Synapse with Power BI to provide greater insight into how Azure Synapse can be used to provide data to your organisation. Let's start with using Power BI to analyse data in Azure Synapse.



#1: Build and analyse Power BI dashboards directly from Azure Synapse

Power BI provides a lot of tools to help analyse data that may already be in Azure Synapse. Using a visual tool to examine the state of the current data can provide a quick way to determine the current status of the data stored in different areas of Azure Synapse, including serverless pools, dedicated pools or to examine the data created in an interim loading process. To add the ability to use Power BI within Azure Synapse, you will need an Azure Synapse workspace and Power BI instance in the same Azure tenant. Creating an Azure Synapse workspace takes a few minutes and the steps for that process can be found [here](#). Perhaps the data needs to be examined to review the quality of the data received to determine what kind of analysis it is possible to do. Perhaps you have received some data from a client and want to do a quick report on it to determine how to incorporate the data into another data model.

Learn how to create an Azure Synapse workspace with [this video tutorial](#).

Azure Synapse allows connectivity to other applications through linked services. You can include a variety of applications, including Azure Key Vault, Azure Machine Learning and of course, Power BI. Linked services provide the connection information allowing you to access the functionality of Power BI from within Azure Synapse. The linked service allows you to make a connection to a workspace in Power BI. In earlier releases, it was possible to connect to one Power BI workspace. Currently, it is possible to connect to multiple Power BI workspaces.

From within Azure Synapse, you may have a number of other linked services for elements such as an additional ADLS account or a publicly available dataset. To create a linked service connection to Power BI, select the **Manage** icon on the left side of the Azure Synapse portal and then, under **External connections**, select **Linked services**, as shown in *Figure 1*. The Power BI linked service has a purple square around it and is called **PowerbiAcctDev**:

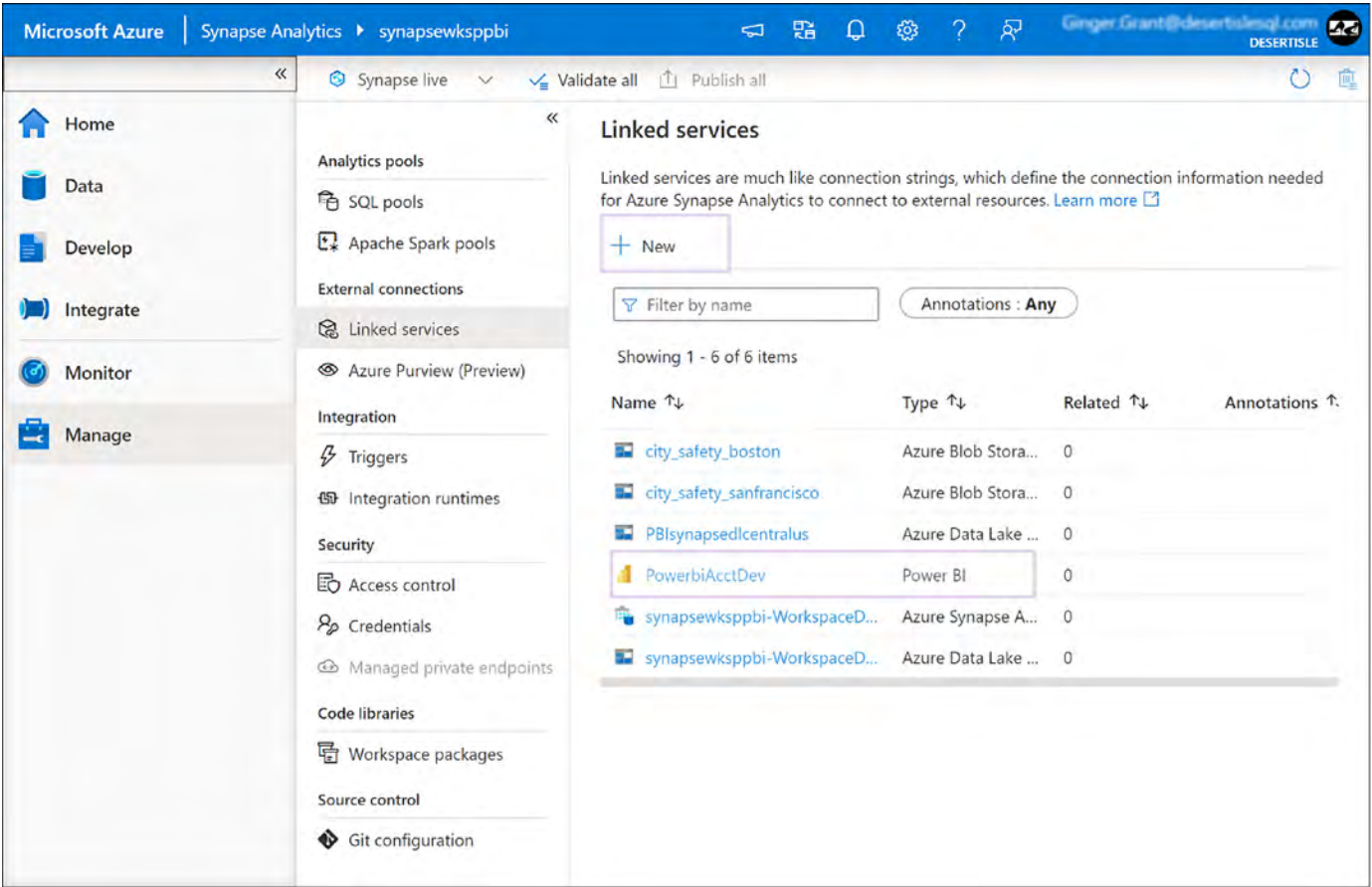


Figure 1: Linked services connections created in this Azure Synapse Analytics account

Notice that the Power BI linked service is called **PowerbiAcctDev**, which is a workspace in the same tenant as the Azure account used in this example. The Azure account used must be in the same tenant as the Power BI account. For example, if we have an Azure account tied to an Outlook.com account, we cannot link that account to our desertislesql.com tenant.

To add a new linked service, click on the **+ New** text in the **Linked services** pane. If you have not yet created a connection to a Power BI workspace, your menu will look like this:

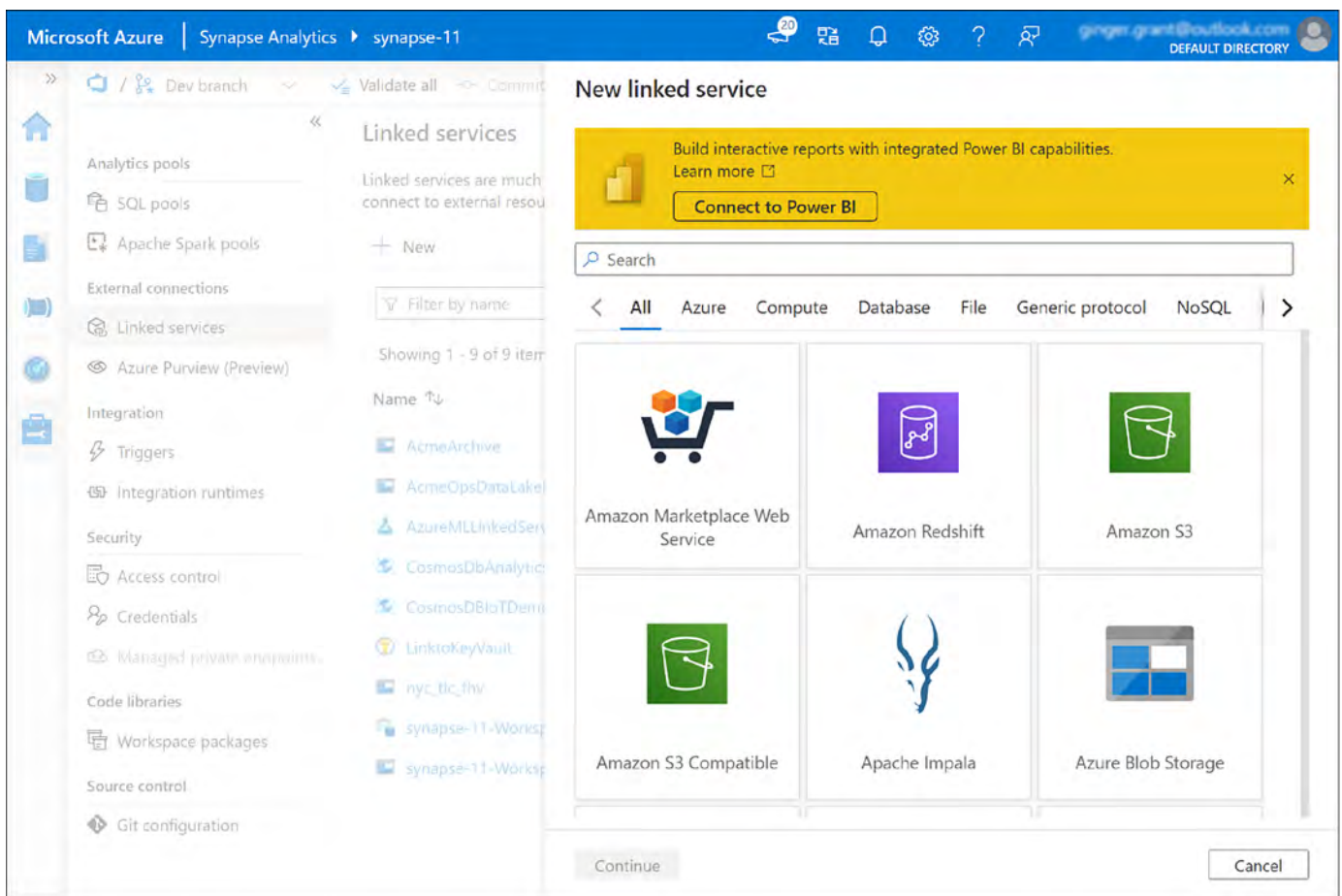


Figure 2: Azure Synapse Analytics New linked service menu

Click on the **Connect to Power BI** button. Power BI is also listed in the menus – you just need to click on the arrow on the right side of the pane to access the menu option. Both options take you to the same screen, shown in *Figure 3*, which you need to complete to connect Azure Synapse to a Power BI workspace. You can choose to name the linked service in Azure Synapse however you want, but be warned that the name cannot be changed later:

The screenshot displays the Azure Synapse Analytics interface for configuring a new linked service of type Power BI. The left sidebar shows the navigation menu with 'Linked services' selected. The main panel is titled 'New linked service (Power BI)' and includes a warning message: 'Choose a name for your linked service. This name cannot be updated later.' The form contains the following fields and options:

- Name ***: A text input field containing 'PowerBIWorkspace'.
- Description**: A text input field that is currently empty.
- Tenant**: A dropdown menu showing 'Default Directory'.
- Workspace name ***: A dropdown menu showing 'Failed to load workspace'.
- Edit**: A checkbox that is currently unchecked.
- Annotations**: A section with a '+ New' button.
- Advanced**: A section with a 'ⓘ' icon.

At the bottom of the form, there are three buttons: 'Commit', 'Back', and 'Cancel'.

Figure 3: Connecting a Power BI workspace to Azure Synapse using Linked services

You will need to select the **Tenant** from the drop-down list, and once you do this, the **Workspace name** drop-down list will be filled with all of the workspaces you have access to. The data inside a workspace can come from a number of different locations, including ADLS, and also from locations such as an Excel file stored in OneDrive or an Azure SQL workspace. After completing this form, click on the **Commit** button. You will need to publish the new linked service to complete the process of adding it.

Once Power BI has been added to Azure Synapse, you can use it from the **Develop** pane. You will notice that you have a **Power BI** section added. From within it, you can see the datasets and reports contained within your workspace. You can use Power BI from within Azure Synapse to modify an existing report, and when you do, make sure that you save it to ensure that it is available to everyone who has access to the workspace. *Figure 4* shows how to modify an existing dataset from within Power BI:

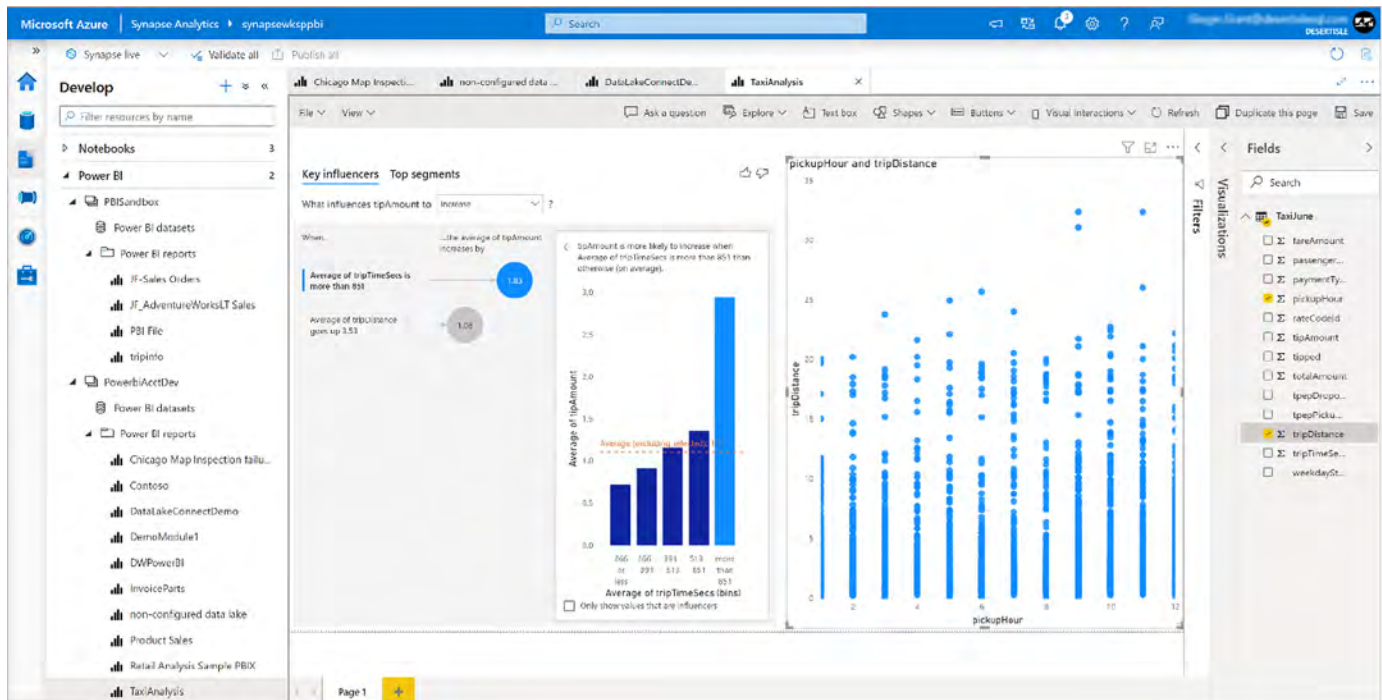


Figure 4: Power BI Linked services report development within Azure Synapse

The datasets are listed in their own window, which is also the location for creating new reports within Azure Synapse for Power BI. If you hover over any of the dataset names in the list, you will see two icons, which are shown in *Figure 5*:

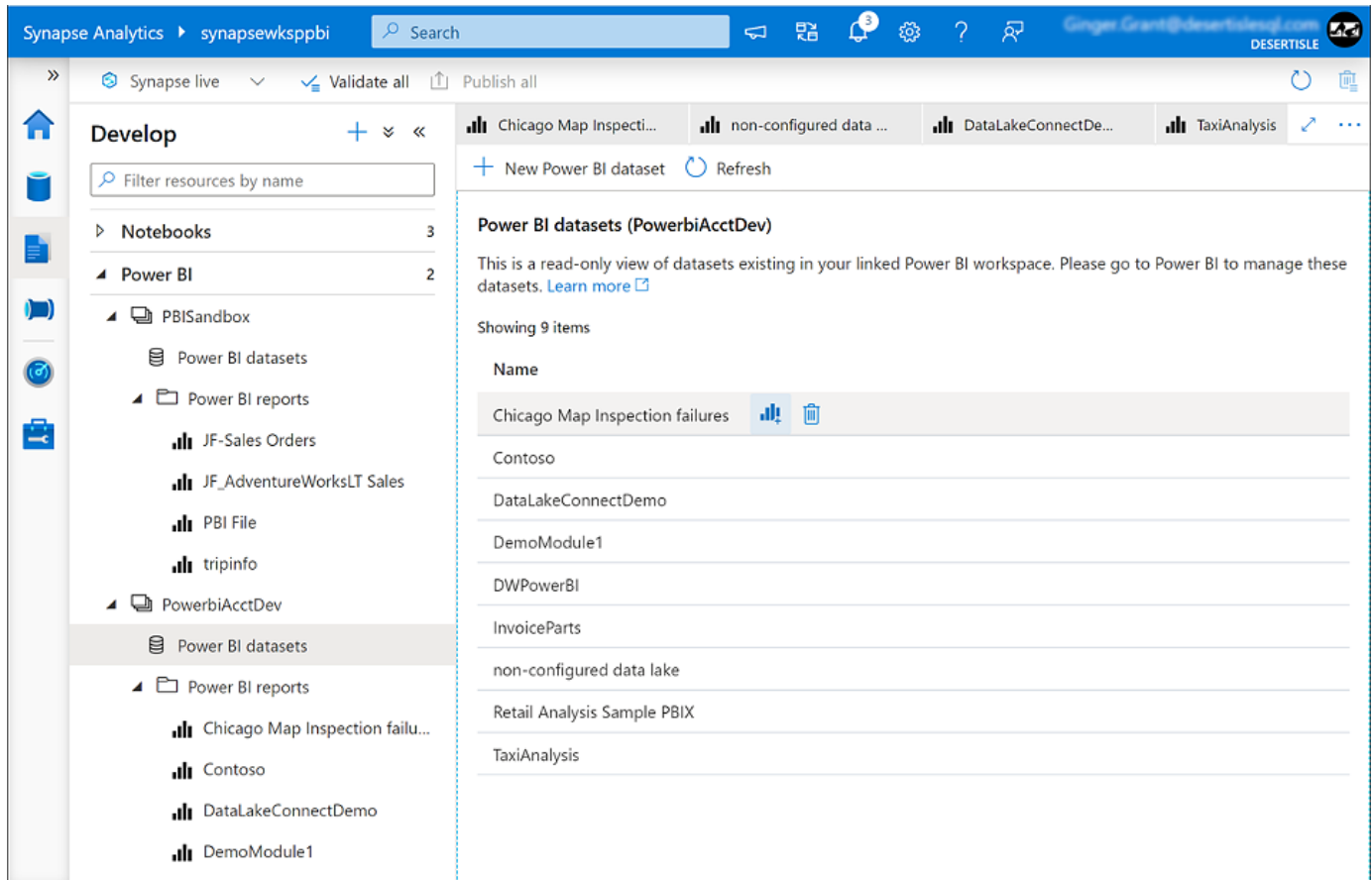


Figure 5: Dataset view of Power BI workspaces in Azure Synapse used to create new reports

The first will allow you to add a new report based upon the highlighted dataset, or you can delete the dataset. Using Power BI to analyse your data from within Azure Synapse can speed up the time it takes to examine data in your data models and use any combined data model that you might create in Power BI. From within Azure Synapse, any models from within the workspace can be used. Once new models are loaded into the workspace, they are available from within Azure Synapse. You may need to refresh the dataset using the **Refresh** button at the top of the screen shown in *Figure 5* to get the new models to appear.

Using an example dataset from the Knowledge Centre

If you do not yet have a dataset that you want to use with Azure Synapse, you can use one of the example datasets provided in the Knowledge Centre, which can be found by clicking on the **Learn** button on the workspace home page as shown in *Figure 6*, which is highlighted in purple:

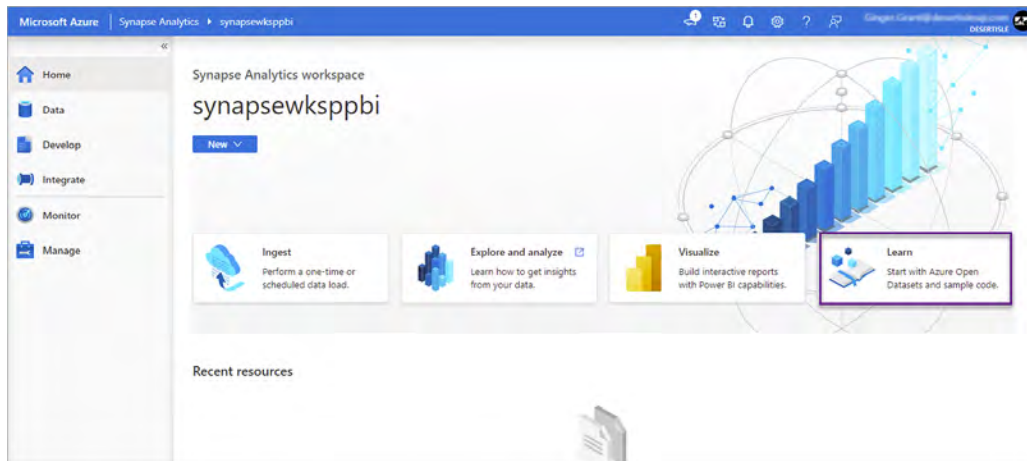


Figure 6: Azure Synapse Learn contains the Knowledge Centre where you can learn more about Azure Synapse

Azure Synapse provides a number of coding examples and datasets that you can use to get more familiar with Azure Synapse. To access them, after selecting **Learn**, the Knowledge Centre page will load. From within the Knowledge Centre, we will select **Browse gallery** and load a sample database into a new Azure Synapse logical database. You will see **Browse gallery** highlighted in purple in *Figure 7*:

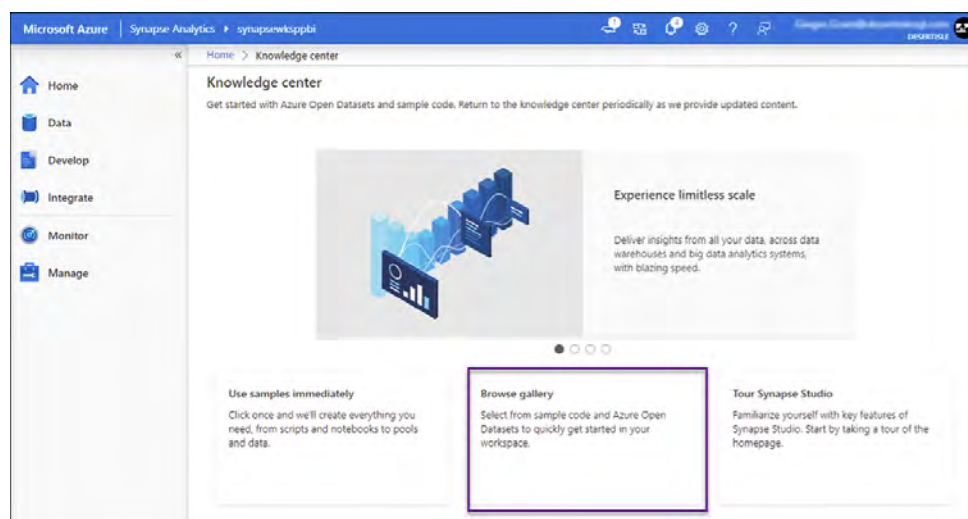


Figure 7: The Browse gallery section of the Knowledge Centre

From within the Knowledge Centre, we will select the example **US State Employment Hours and Earnings**, and then click on the blue **Continue** button. A short description will appear and a blue **Add dataset** button will appear in the upper-left corner of the window, which you should click. The dataset will then appear under **Azure Blob Storage**. Click on the ellipsis (the three dots) next to the **us-employment-hour-earnings** folder, and select **New SQL script** followed by **Create external table**, as shown in *Figure 8*:

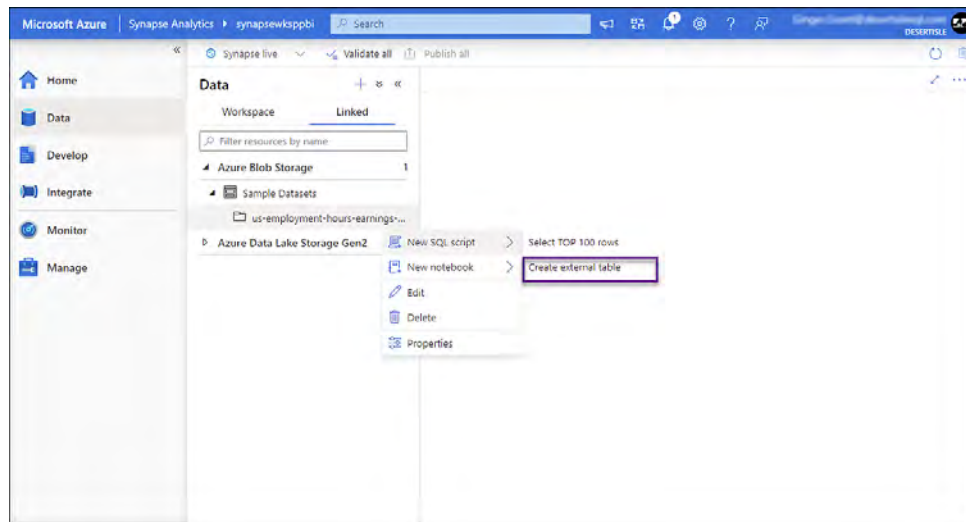


Figure 8: Create external table from the US Employment Hours Earnings sample dataset

A pop-up window will appear providing the prompts needed to create an external table. Create a new database called **USEmp** and for **External table name**, use **HoursData**, as shown in *Figure 9*:

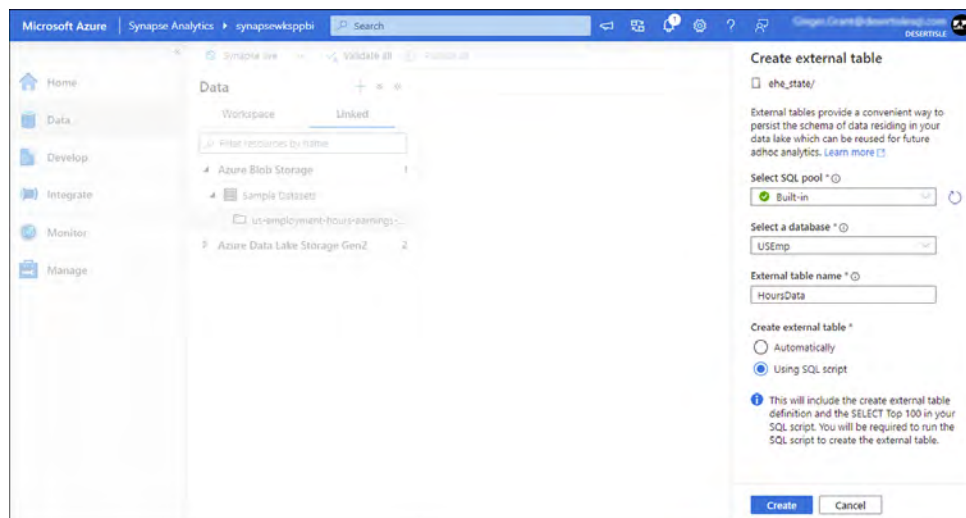


Figure 9: Create external table prompts

Make sure that the radio button, **Using SQL script**, is selected as shown here before selecting the **Create** button. Take a look at the code that is generated in the **SQL script 1** window as this can be a guide to creating external tables. Select the **Workspace** tab, which is highlighted in purple in *Figure 10*, and refresh the databases by selecting the ellipsis, and you will see the new **USEmp** database:

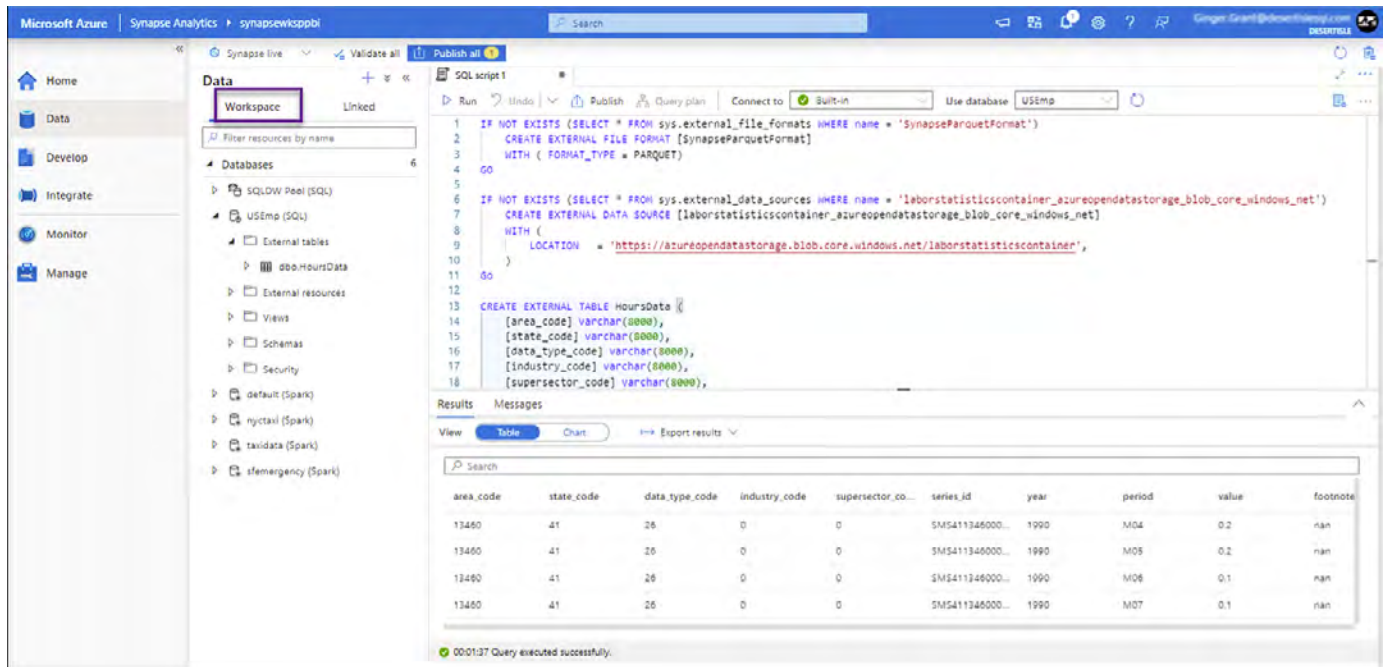


Figure 10: Newly created Azure Synapse logical database with the HoursData table shown

The code selected will generate a new database and a table that you can use in Power BI. The external table contains the dataset that you can connect to Power BI and use in a visualisation.

Next, we'll show you how Azure Synapse manages your data lake and works with Power BI for more meaningful reports.

#2: Manage your data lake and build data marts for BI reports

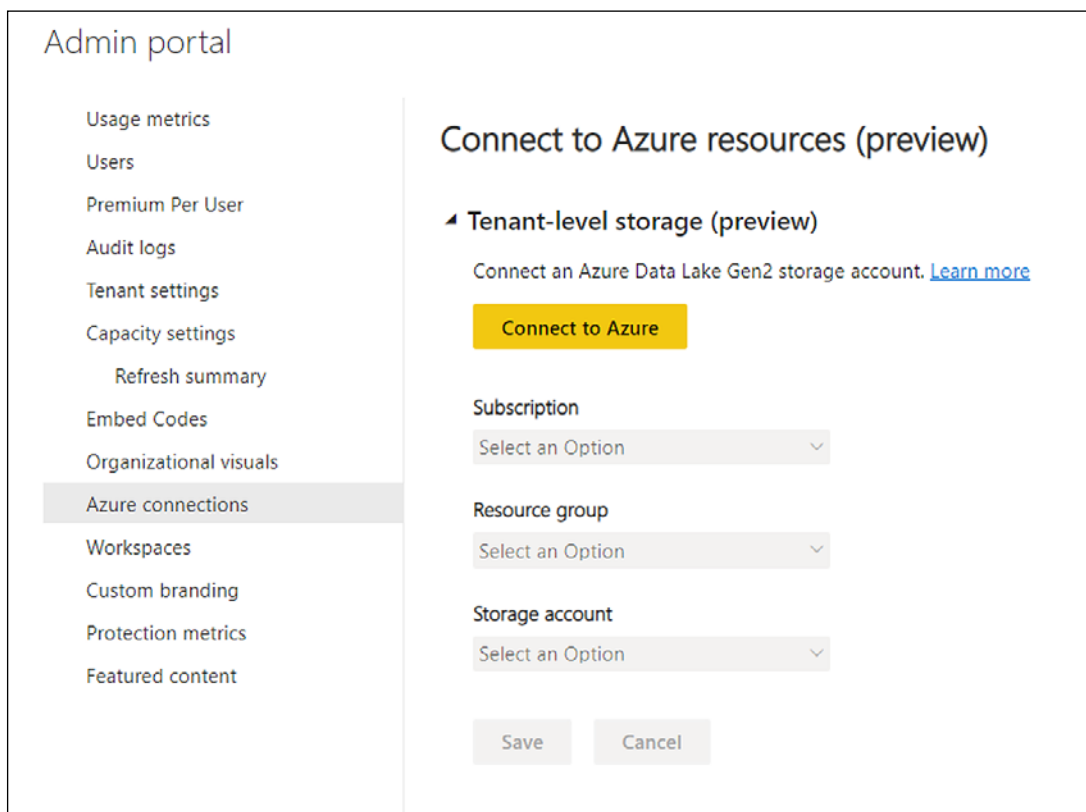
Many businesses these days are working with data lakes for long-term storage of all of the data they have in their environment. Azure Synapse can be a great solution as it provides the ability to create integration pipelines that can organise data in a data lake into different functional areas for different use cases.

The Acme company is an internet sales company that has a number of items stored in its warehouses that will need to be packed and shipped after an order is placed. While it may only want to create daily reports for the last two years, it needs to store the rest of the data in a secure, organised fashion so the data is available for ad hoc analytics or machine learning. Data may be received from IoT devices and may need to be reported upon in a summarised fashion and stored in the raw format for later machine learning analysis.

Azure Synapse contains all of the tools needed to organise data within a data lake so the data can be available for many different uses. The data could have arrived from different sources into the data lake. Using an integration pipeline, the data could be transformed from raw data stored in the bronze area of the data lake, added in some metadata in the silver area of the data lake and curated to a set of files modelled into a data warehouse star schema. Azure Synapse can help organise raw data so that it can be better utilised by organisations by providing the structure to review and organise data lakes. Once Azure Synapse has curated the data, it is ready to be used by Power BI.

Data stored in the data lake is often stored in parquet format, as this format provides columnar data compression and can be queried faster using the default serverless endpoint and read more quickly in Azure Synapse. This is not an impediment to accessing the data in Power BI. On the contrary, it provides a method for ensuring that you can load perhaps a rolling 24 months of data by loading only that data into the model. Let's examine how we can connect to the data lake that has been curated with Azure Synapse.

Accessing the data lake created by Azure Synapse is first configured in the Power BI admin portal. At the time this document was written, this was a preview feature setting in the Power BI admin portal. To access the data, a user with Power BI administrative access will need to use the admin portal and select the option **Azure connections**, as shown in *Figure 11*. When connecting to an Azure Data Lake Storage Gen 2 account, the most economic method for using the data in Power BI is to ensure that the data lake and the Power BI tenant are in the same data centre. You will also need the name of the Azure **Subscription**, **Resource group**, and the name of your Azure Data Lake **Storage account** to fill out the information in the **Azure connections** section of the admin portal, as shown in *Figure 11*:



Admin portal

- Usage metrics
- Users
- Premium Per User
- Audit logs
- Tenant settings
- Capacity settings
- Refresh summary
- Embed Codes
- Organizational visuals
- Azure connections**
- Workspaces
- Custom branding
- Protection metrics
- Featured content

Connect to Azure resources (preview)

▲ Tenant-level storage (preview)

Connect an Azure Data Lake Gen2 storage account. [Learn more](#)

Connect to Azure

Subscription
Select an Option

Resource group
Select an Option

Storage account
Select an Option

Save Cancel

Figure 11: Power BI Admin portal for Azure connections and Azure Data Lake Gen2

Once this connection is made, you can access the data in Power BI. We have used the resource group name **SynapseRGPBI** and the Azure Data Lake Storage Gen 2 account is **synapsedlpbi**. Once the Azure connection is created and saved, the data can be refreshed in the Power BI service.

Note: At the time this document was written, you could only connect to one Azure Data Lake Gen2 Storage (ADLS) account and not any other kind of storage account.

From within the Power BI desktop, Azure Data Lake Gen 2 accounts can be accessed by selecting them from within the Azure options within the **Get Data** menu. You will need to access the data with the URL with the location of your ADLS data using this pattern:

```
https://<ADLS_accountname>.dfs.core.windows.net/<filesystemname>/<subfolder>
```

This will extract all data from the folder. For example, we have an ADLS account called **PBISynapseCentralUS**. It has a folder called **powerbi**, and there is a subfolder called **Bronze**. If we wanted to access all of the files in this folder, the URL we would use in Power BI is:

```
https://synapsedlcentralus.dfs.core.windows.net/powerbi/Bronze/  
Chicagofoodinspections/2020FoodInspectionsparquet
```

The URL would be inserted as shown in *Figure 12*:



Figure 12: Getting data from Azure Data Lake Storage Gen2 using a URL

Please note, when you specify the location, make sure that the case of the letters matches exactly. If they do not match, you will get a 'Not found' error. As we did not specify a file name, all of the files in that directory were returned. Our subdirectory contains three parquet files. Parquet files are commonly used with Azure Synapse as these files will perform faster when creating tables in serverless pools and decrease the storage size. These files are binary files and are not human-readable, but they can be read by Power BI. There are three parquet files in the listed subdirectory.

These are meant to be read as one data element, the 2020 Chicago Food inspection data. Let’s take a look at these in Power BI, which you can see in *Figure 13*:

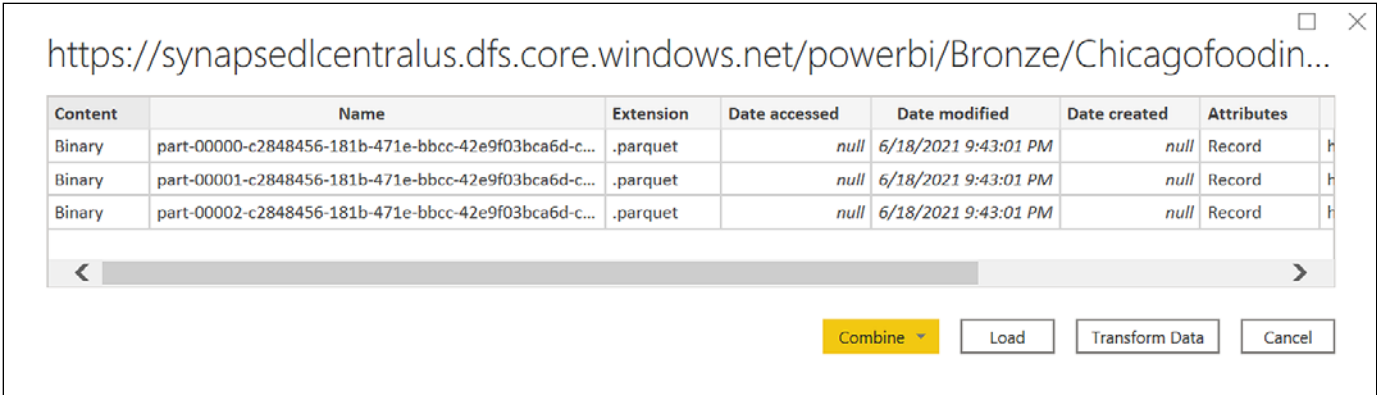


Figure 13: Parquet files to be loaded from ADLS into Power BI

The yellow **Combine** button will allow us to join the files together and incorporate the content into our data model. We are going to select **Combine** and **Transform Data** to gather the data and determine whether we need to do anything to it prior to loading it into Power BI. This step will load all of the data and transform it from parquet to a human-readable format, which you will see in *Figure 14*:

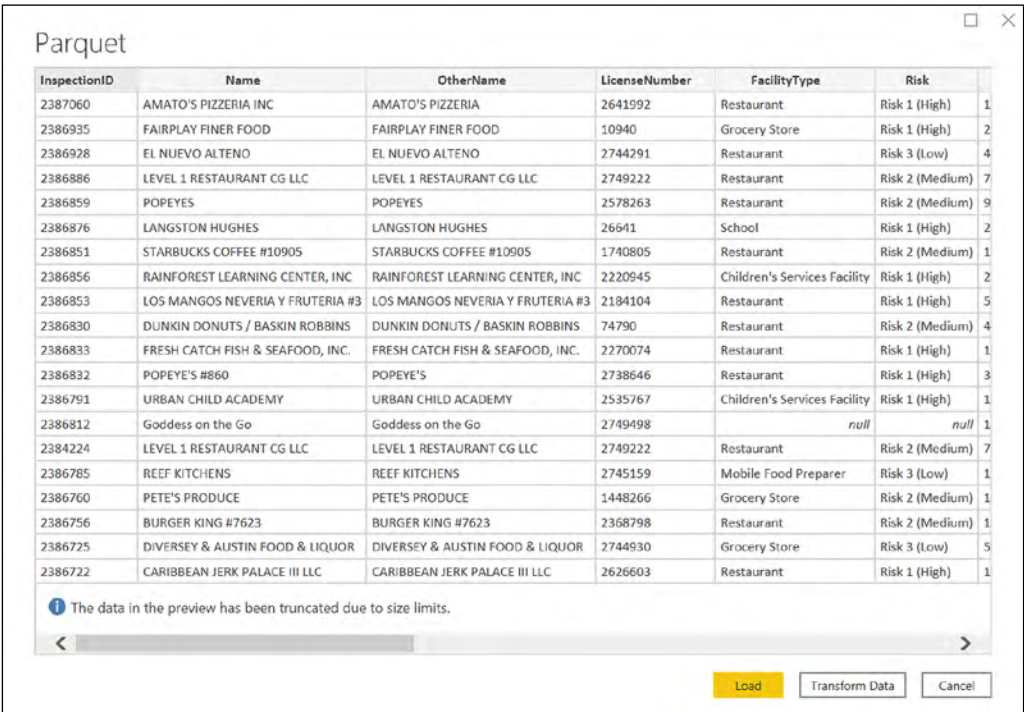


Figure 14: Sample of parquet files from ADLS transformed into Power BI

Here we can see that it is very easy-to-read data, so let’s transform it. Taking a look at the queries on the left side of the screen, we can see that Power BI has automatically performed a number of different transformation steps on the data in *Figure 15*:

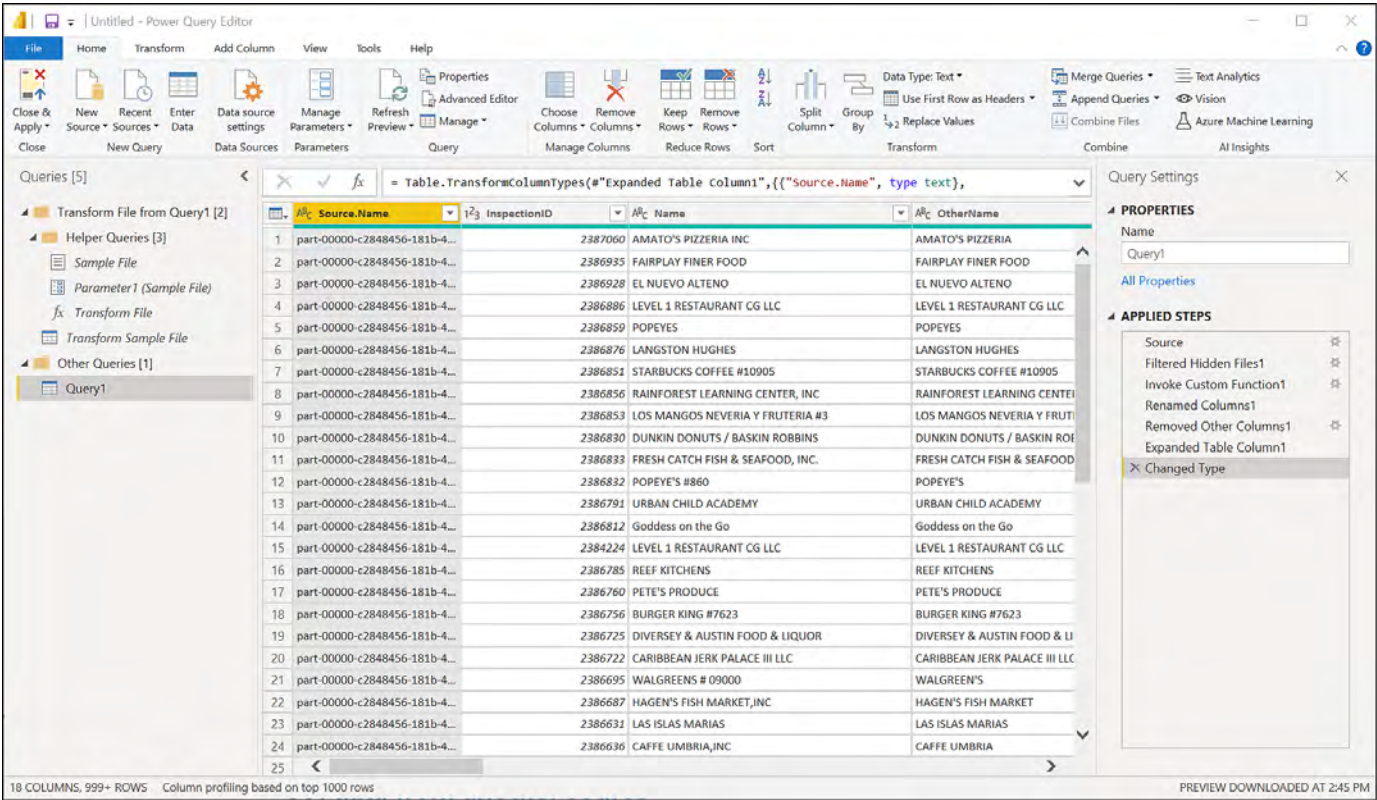


Figure 15: Transformational steps created in Power Query

The helper queries get the data from the ADLS account and transform it into human-readable form. **Query1** picks up the changes made. Here, we will remove the **Source.Name** and **InspectionID** columns as they will not be used in the data model, and rename **Query1** as **Inspection**. Using Power BI, we can use data stored in the optimal format for Azure Synapse and use Power BI to report on the data, as shown in *Figure 16*:

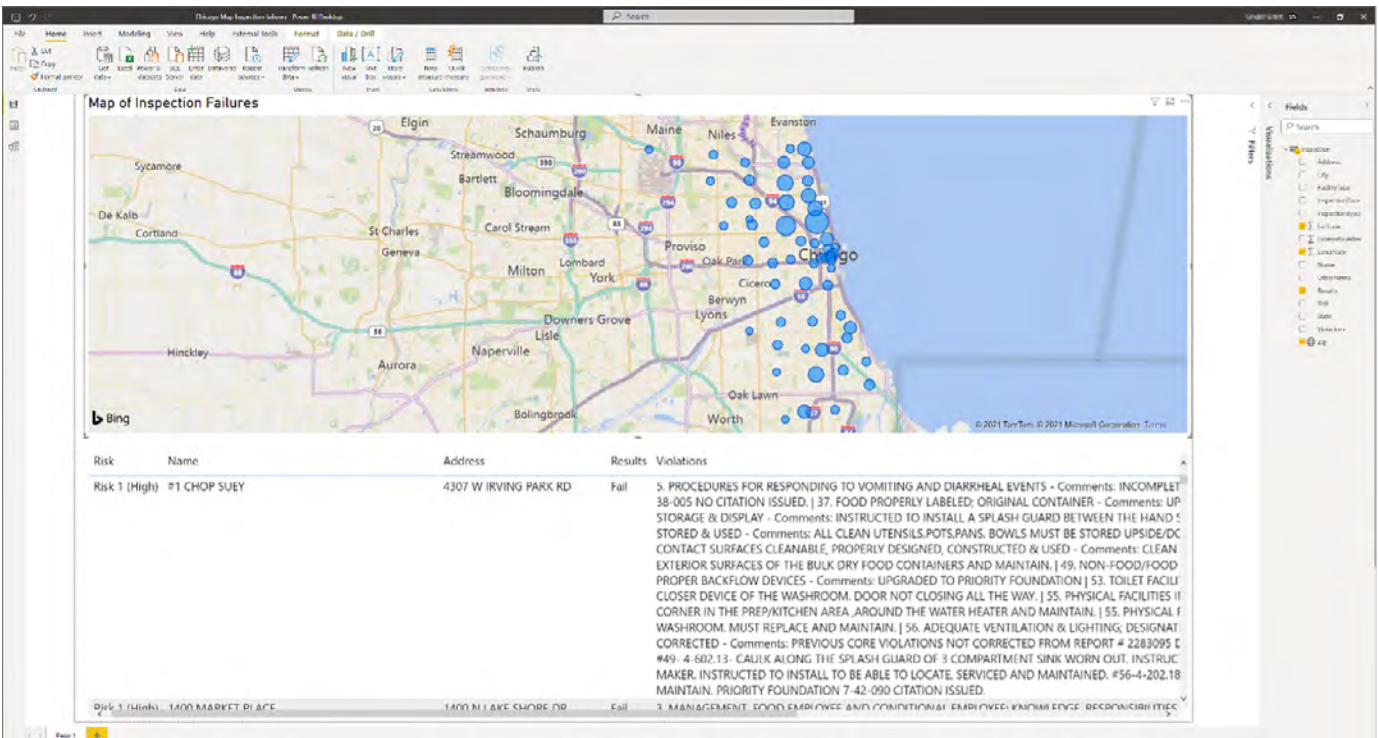


Figure 16: Power BI visualisation made with parquet data

#3: Explore your data lake using a fully managed serverless endpoint

Many companies are choosing to store a large portion of their data in a data lake. An ADLS Gen2 account can store years of data at a very low cost, especially when the data is stored in the parquet format, which offers data compression, decreasing the storage requirements of storing the data. When you create an Azure Synapse workspace, an Azure Data Lake Gen2 (ADLS) account is provisioned, providing the storage needed to create a data lake. Azure Synapse Analytics can manage the datasets and provide a logical database on top of the ADLS account where the data can be queried as if it was in a database using the included serverless endpoint. There is no need to provision anything else or stop and start anything, as it uses a pay-per-query model.

Logical databases can be created using ADLS storage so that users can explore the data contained in the data lake using T-SQL. The virtual database also includes the ability to create row-level security within the database as this feature is supported. Naturally, you can create your data model by following best practices and creating sorted views to access the data that you put in your Power BI data model. By using views, you can ensure the columns are in human-readable format and sorted by the columns with the least number of values to improve VertiPaq engine compression and Power BI report performance.

Using the serverless endpoint in Azure Synapse, you are only charged for the queries you perform. The data can be accessed by creating logical databases within Azure Synapse using T-SQL statements. You can define tables on top of data from a number of different directories and data lakes and combine them into one database. This task is performed using the T-SQL command **CETAS, Create External Table AS SELECT**. This command is used to create external tables that can be accessed by Power BI or from **SSMS**.

Here is a sample of the code that I used to create a logical database within Azure Synapse and to use UTF8 collation, which we need to handle some of the text in our parquet files:

```
CREATE DATABASE EMR

GO

--Change the internal database to EMR to issue comments

USE EMR

GO

ALTER DATABASE EMR COLLATE Latin1_General_100_BIN2_UTF8;
```

The next step is to define the location of the data to be read and to set the location for retrieving the data. We have added **Not Exists** statements to allow us to run the script multiple times. If the data is stored in a number of different locations, you can create multiple external data sources to reference them:

```
IF NOT EXISTS (SELECT * FROM sys.external_file_
formats WHERE name = 'SynapseParquetFormat')

CREATE EXTERNAL FILE FORMAT [SynapseParquetFormat]

WITH ( FORMAT_TYPE = PARQUET )

GO

IF NOT EXISTS (SELECT * FROM sys.external_data_sources WHERE name = 'EMRParquetData')

CREATE EXTERNAL DATA SOURCE [EMRParquetdata]

WITH (

    LOCATION 'https://synapsedatalake.dfs.core.windows.net/root/Curated/EMR',
    CREDENTIAL = [ManagedIdentityCredential]

)

GO
```


After the format and the location have been specified, you can create the tables on top of the data. Here, we are creating two tables and formatting the query:

```
IF EXISTS (SELECT * FROM sys.external_tables WHERE name = 'MailingAddress')

DROP EXTERNAL TABLE MailingAddress

GO

CREATE EXTERNAL TABLE MailingAddress (

    [MailingAddressID] varchar(100) ,

    [StreetAddress] varchar(300),

    [City] varchar(100),

    [StateAbbr] char(2),

    [Zipcode] varchar(10),

    [ModifiedDate] varchar(13)

)

WITH (

    LOCATION = '/2021MailingAddress/Address.snappy.parquet',

    DATA_SOURCE = [RawParquetData],

    FILE_FORMAT = [SynapseParquetFormat]

)

GO
```

```
IF EXISTS (SELECT * FROM sys.external_tables WHERE name = 'Patient')

DROP EXTERNAL TABLE Patient

GO

CREATE EXTERNAL TABLE Patient (

PatientID varchar(100) ,

SSN varchar(12) ,

FirstName varchar(60) ,

MiddleName varchar(60) ,

LastName varchar(60) ,

Gender char(1) ,

MailingAddressID varchar(100) ,

Email varchar(256) ,

DateOfBirth varchar(13) ,

ModifiedDate varchar(13)

)

WITH (

LOCATION = '/2021Patient/Patient.snappy.parquet',

DATA_SOURCE = [RawParquetData],

FILE_FORMAT = [SynapseParquetFormat]

)

GO
```


Here are the views we created with that data, which we will be connecting to with Power BI:

```
CREATE VIEW vPatient
AS
SELECT TOP 999999999 [PatientID]
    ,FirstName AS [First Name]
    ,LastName AS [Last Name]
    ,[Gender]
    ,CAST([AddressID] AS INT) AS AddressID
    ,[Email]
    ,DATEDIFF(YYYY, CAST([DateOfBirth] AS DATE), GETDATE() ) as Age
FROM [EMR].[dbo].[Patient]
ORDER BY Gender, Age
GO
CREATE VIEW vAddress
AS
SELECT TOP 999999999 cast([MailingAddressID] as int) as AddressID
    ,[StreetAddress] as [Address]
    ,[City]
    ,[StateAbbr] as [State or Province]
    ,[Zipcode] as [Postal Code]
FROM [EMR].[dbo].[MailingAddress]
ORDER BY [StateAbbr] , [City]
```

Once this is complete, we can connect to the EMR database in Power BI and other database tools such as SQL Server Management Studio. When we connect to this data, it will be just like we are connected to SQL Server. The name of the database used for the serverless pool can be found on the Azure Synapse page in the Azure portal, as shown in *Figure 17*, where there is a purple box around the name of the serverless endpoint that we will be using to connect to Power BI:

The screenshot displays the Azure Synapse workspace page for 'synapse-11'. The page includes a search bar at the top, a navigation menu on the left, and a main content area with details and actions. The 'Serverless SQL endpoint' is highlighted with a purple box.

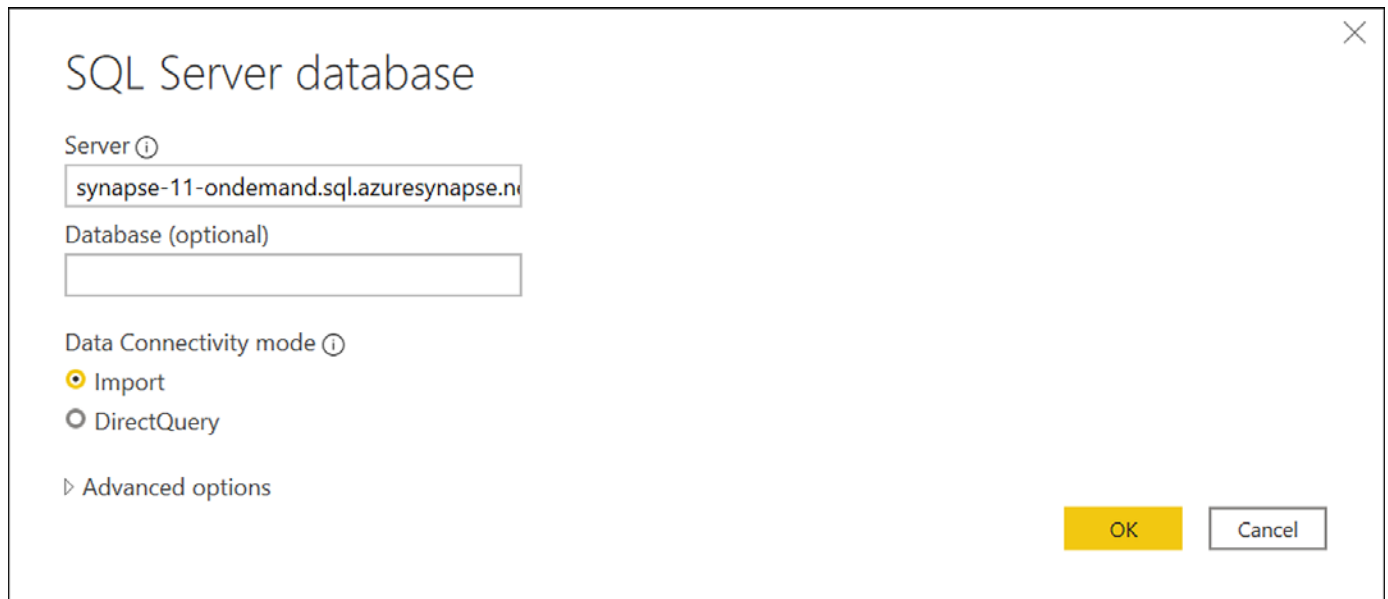
Property	Value
Resource group (change)	SynapseResources
Status	Succeeded
Location	West US 2
Subscription (change)	Microsoft Azure Sponsorship
Subscription ID	8736b1ec-0ab5-460e-8c92-9edffbaaffad5
Managed virtual network	No
Managed Identity object ...	cd72df6b-9f5a-4895-a1c8-8823299d5b6d
Workspace web URL	https://web.azuresynapse.net?workspace=%2fs...
Tags (change)	Click here to add tags
Firewalls	Show firewall settings
Primary ADLS Gen2 acco...	https://synapse11datalake.dfs.core.windows.net
Primary ADLS Gen2 file s...	root
SQL admin username	sqladminuser
SQL Active Directory ad...	live.com#ginger.grant@outlook.com
Dedicated SQL endpoint	synapse-11.sql.azuresynapse.net
Serverless SQL endpoint	synapse-11-ondemand.sql.azuresynapse.net
Development endpoint	https://synapse-11.dev.azuresynapse.net

Getting started

- Open Synapse Studio**
Start building your fully-integrated analytics solution and unlock new insights.
[Open](#)
- Read documentation**
Learn how to be productive quickly. Explore concepts, tutorials, and samples.
[Learn more](#)

Figure 17: Azure Synapse from the Azure portal showing the serverless endpoint

The next step is, of course, to connect Power BI to this data source by connecting to it from a SQL Server connection, as shown in *Figure 18*:



SQL Server database

Server ⓘ
synapse-11-ondemand.sql.azuresynapse.net

Database (optional)

Data Connectivity mode ⓘ
☒ Import
☐ DirectQuery

▸ Advanced options

OK Cancel

Figure 18: Accessing an Azure Synapse serverless pool using a SQL Server database connection

As you can see, we are choosing to import our data and we are connecting to the serverless endpoint from Azure Synapse. We are going to use the views we created from the earlier script, but you can see that we are allowed to select data from any database. When authenticating the database, you will want to use your Azure AD connection.

The screenshot shows the Power BI Navigator window. On the left, the 'Display Options' pane shows a tree view of the data source 'synapse-11-ondemand.sql.azuresynapse.net [5]'. Under the 'EMR [4]' folder, the 'vPatient' table is selected with a checkmark. Other tables like 'vAddress', 'MailingAddress', 'Patient', and 'LogicalDW' are also listed. On the right, the 'vPatient' table is displayed as a grid with columns: PatientID, First Name, Last Name, Gender, AddressID, and Email. The grid shows 25 rows of patient data. At the bottom, there are buttons for 'Select Related Tables', 'Load', 'Transform Data', and 'Cancel'.

PatientID	First Name	Last Name	Gender	AddressID	Email
99	Kelly	Phillips	F	98	valandsh
82	Teresa	Nugent	F	82	rracher@
81	Melissa	Drye	F	81	mhem@:
74	Dawn	Begay	F	75	philrieste
96	Joan	Barba	F	95	tnugen@
35	Stacy	Prickett	F	40	Irene.mir
14	Luz	Mares	F	21	debheal@
44	Kathryn	Ottosen	F	49	lbreaux@
31	Barbara	Olson	F	37	iam300i
7	Merrie	Heath	F	15	celticlady
3	Connie	Gover	F	11	benitab5
5	Susan	Schulte	F	13	bromgn@
95	Susan	Maestas	F	94	tindle358
20	Sheri	Williams	F	27	dprince0
34	Cynthia	Knutson	F	4	acdpc50(
11	Deborah	Jackson	F	19	Comp@h
53	Julee	Gilson	F	56	mboro2@
41	Bonnie	Russell	F	46	kle@ne
77	Debra	Sawyer	F	78	rayban92
40	Sue	Budman	F	45	KNUTSOI
80	Irene	Russell	F	80	rmerricks
100	Rosalind	Racher Kirk	F	99	WDHOU5
25	Lynne	Myers	F	31	GregFlyH

Figure 19: Selecting data from a logical database in Azure Synapse in Power BI

Using our data from Azure Synapse Analytics, we can now create Power BI reports using serverless databases, as shown in *Figure 19*.

Using the logical database provides the ability to access data stored in different formats, such as parquet, and explore the data stored in a data lake to provide meaningful results without the need to create additional services, as all of this functionality is provided from within Azure Synapse.

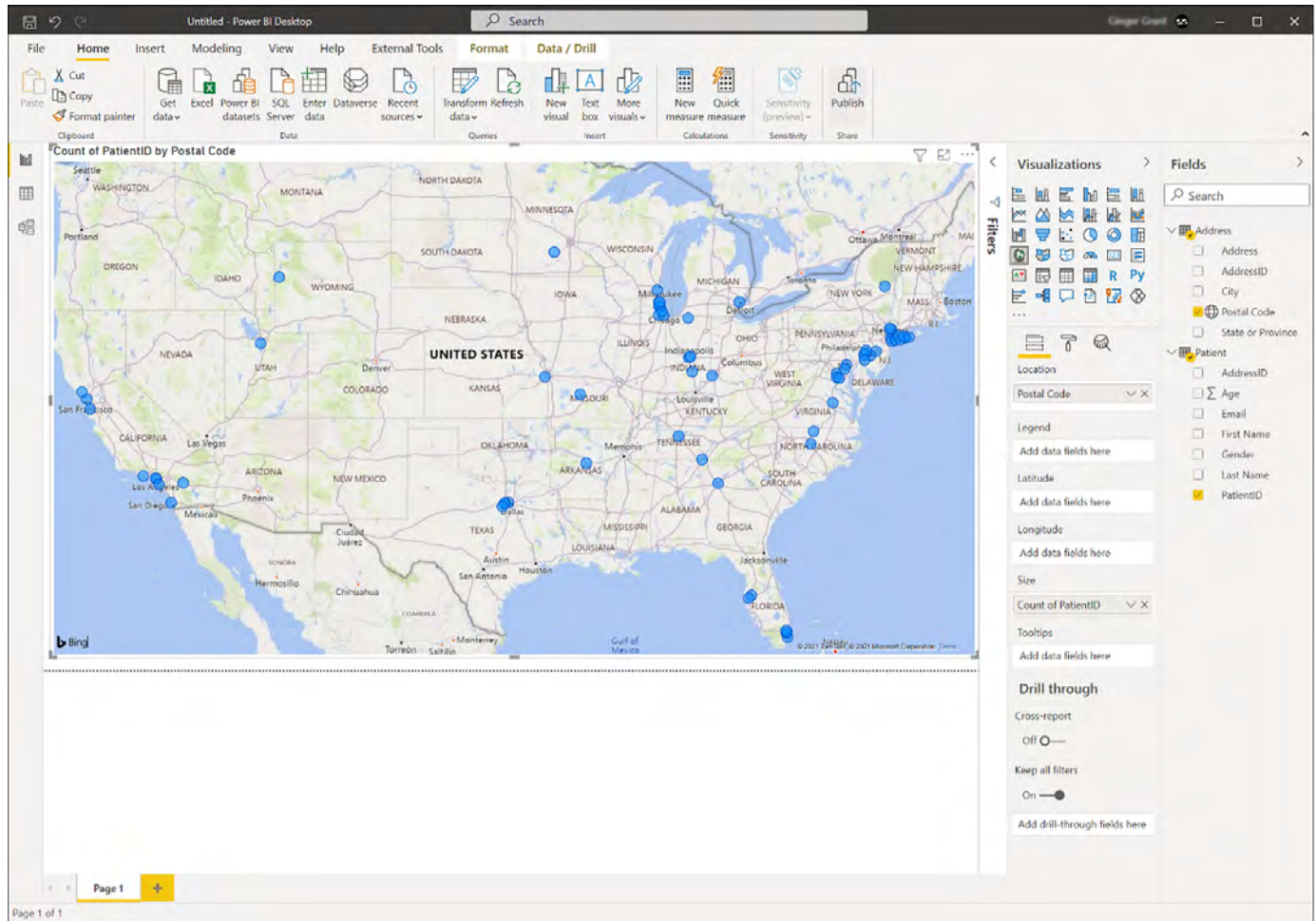


Figure 20: Power BI report with Azure Synapse serverless pool data

#4: Build code-free data pipelines to integrate more data sources and enrich insights

Azure Synapse can gather data from a variety of different data sources and store it centrally in an Azure Data Lake Gen2 storage account, which then acts as a single repository for all other applications that need access to that data. Data scientists may want to combine data from web applications such as Workday together with company data using data flows to gather data from different sources. This data may also be transformed into a data warehouse prior to being accessed. As part of the data-gathering process, data may also be analysed using machine learning models, with Power BI used to visualise the results. For example, as part of a nightly process, customers could be analysed to determine the likelihood of customer churn using machine learning. The data analysed could be from unstructured feedback with customer service personnel combined with their transaction history stored in the data warehouse, which is stored in a dedicated pool in Azure Synapse. This analysis could be done as part of a batch process to feed data to Power BI, and then the customer churn analysis could be included in Power BI using integration pipelines in Azure Synapse.

Azure Synapse integrations provide the ability to create data pipelines in the same fashion as Azure Data Factory, while also providing the capability to include PySpark notebooks to write data to serverless pools that can be included in a Power BI model. Let's take a look at how to do that with Azure Synapse.

First, we are going to create a new pipeline to process the raw customer service data. Here is an example pipeline in *Figure 21* we have created to analyse and score the extracted customer service data, which we are going to output as part of an Azure Synapse pipeline:

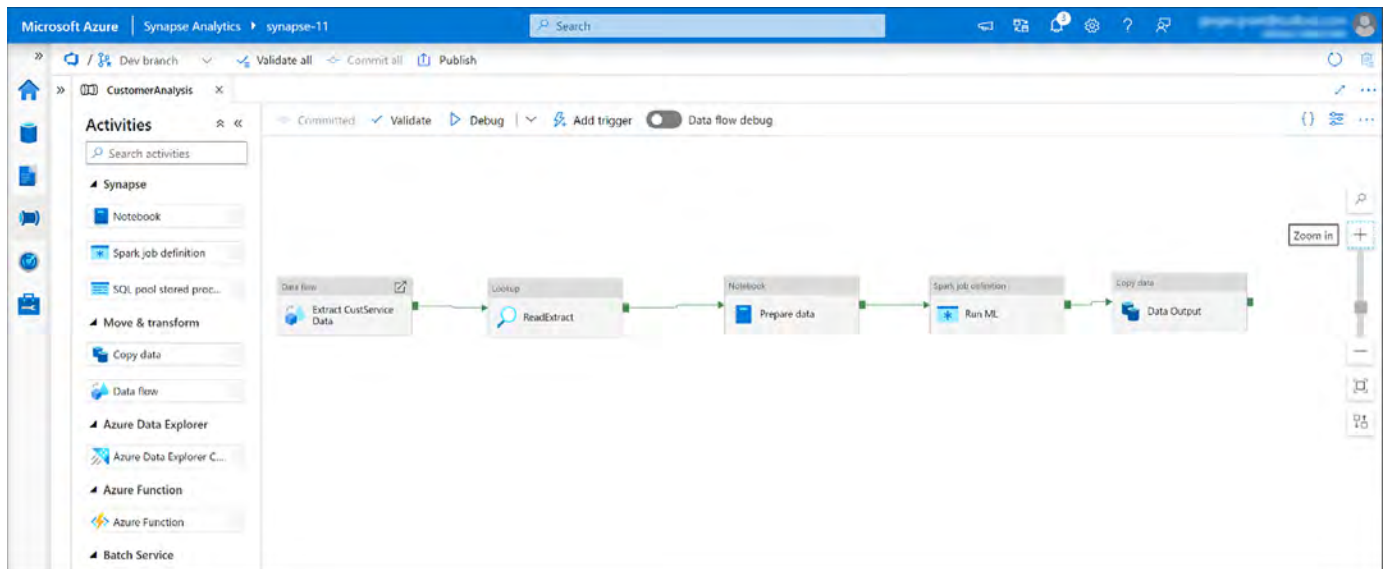


Figure 21: Azure pipeline to extract customer service data and analyse it with Spark

The output of the machine learning analysis is to create a transaction from the raw customer service data, process the data using PySpark and then use the processed data to feed a machine learning object to score the customer transaction and write the data to a serverless pool in Azure Synapse. Let's take a look at our new data elements in Power BI as the data was written to the LogicalDW.

We will be connecting using a SQL Server connection to our serverless pool the same way we did in our previous example, as shown in *Figure 21*. We will then be able to use the information gathered from our analysis of the customer service sentiment and other customer data to show average churn probability along with a score of customer satisfaction based upon an analysis of the data in Azure Synapse and provide that information as part of a Power BI report, as shown in *Figure 22*:

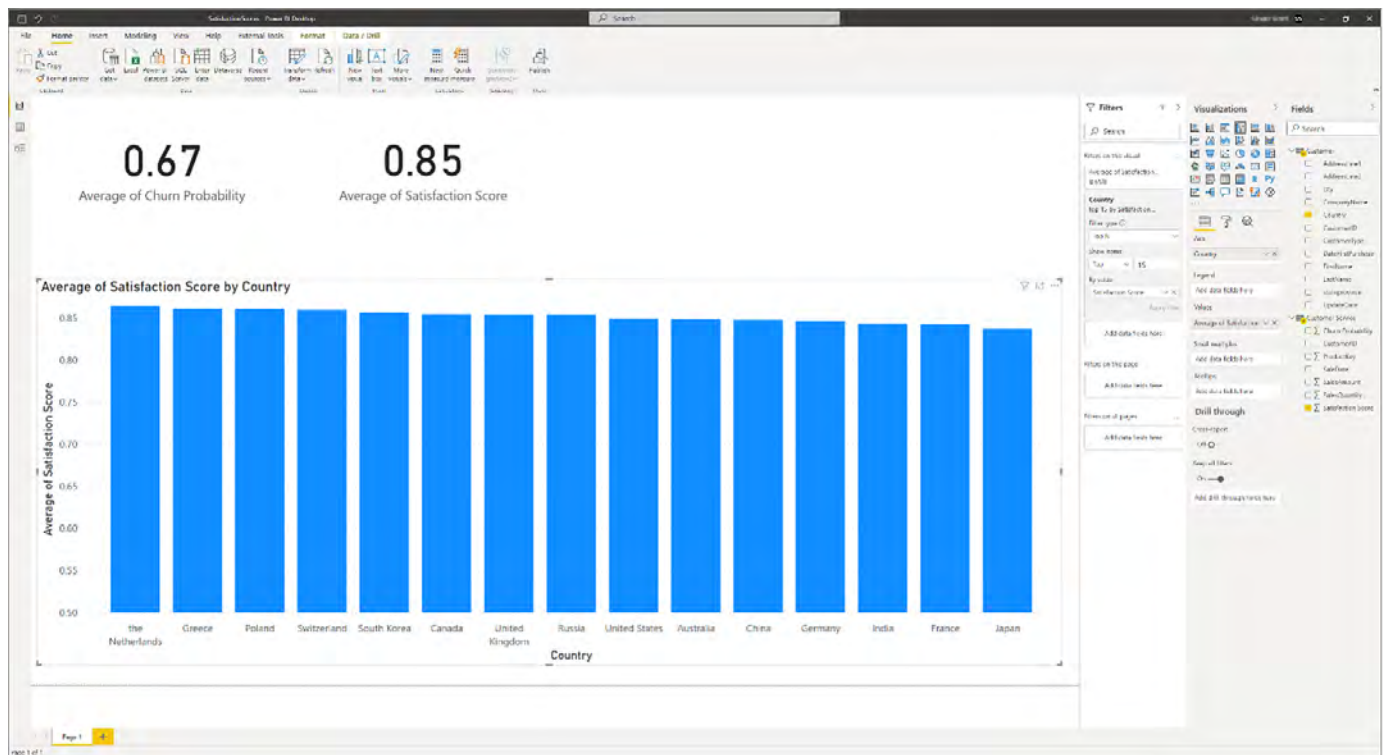


Figure 22: The output of processing data in Azure Synapse to do churn analysis on the customer service dataset and the customer satisfaction score

#5: Create a secure data warehouse and connect to your Power BI dashboards

Power BI can connect to data sources, including very large datasets – including those found in a dedicated pool in Azure Synapse. There can be challenges to reporting on a large amount of data, which Power BI practitioners are already familiar with doing. Within Power BI, there are different methods for performance tuning your data, including aggregation tables and query folding to name but two. Very large datasets often use Direct Query or Dual connections to the data, as the data models are typically too large to import. Azure Synapse also contains methods to assist in improving report performance through workload management classification, which offers the ability to prioritise resources for Power BI reporting tasks. Dedicated pools are often paused to save Azure resources. Make sure the dedicated pool is running by going to the **Manage** section in Azure Synapse and looking at the SQL pools. You will notice in *Figure 23* that we have two dedicated pools, **SmallPool** and **Warehouse**. The status shows that **SmallPool** is running:

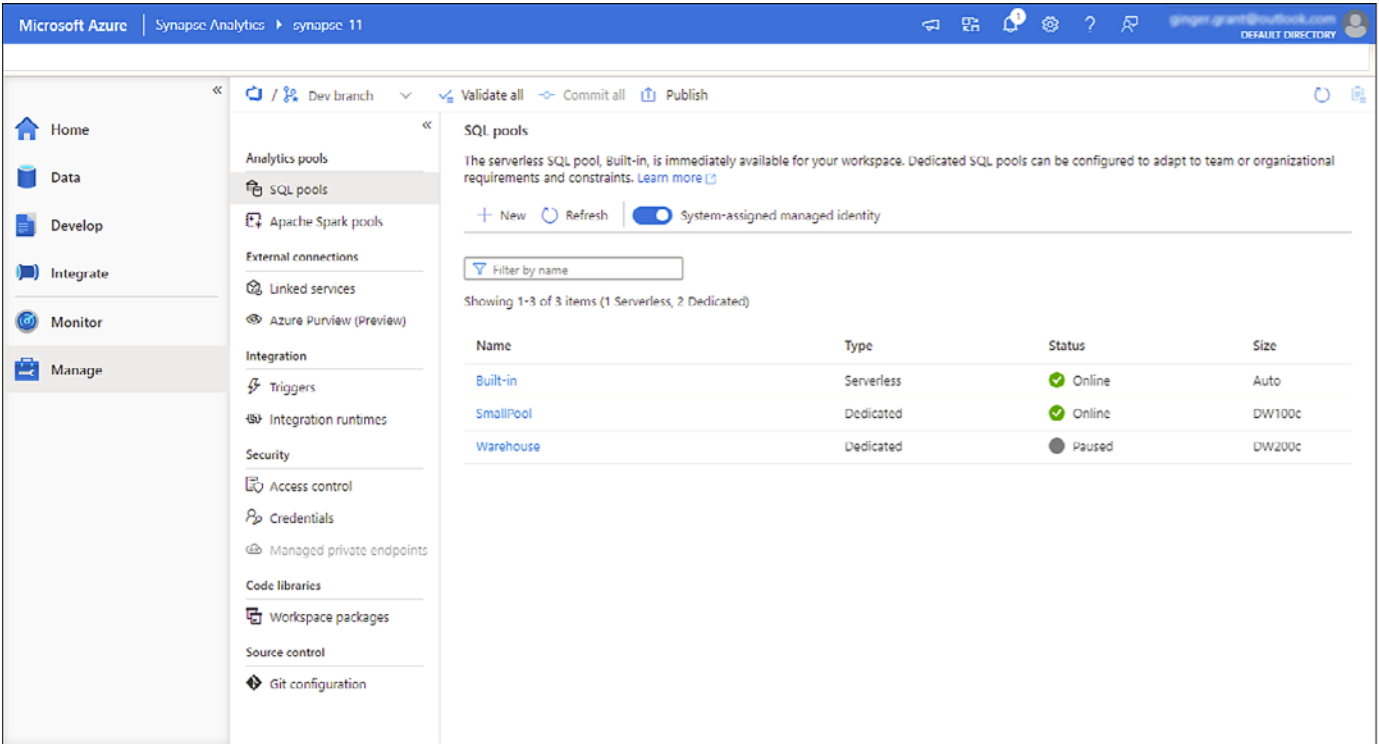


Figure 23: Managing SQL pools

If it was not running, I would click on the **Resume** button.

The first step in providing more resources for Power BI reporting is to create a workload group in Azure Synapse. The workload group will specify the amount of workload isolation associated with a given group. We will need to create a new SQL script for this code and make sure that we are connecting to **SmallPool** when we run it. *Figure 24* shows the **Develop** section, where a new SQL script has been created, which is connected to the running dedicated pool, **SmallPool**:

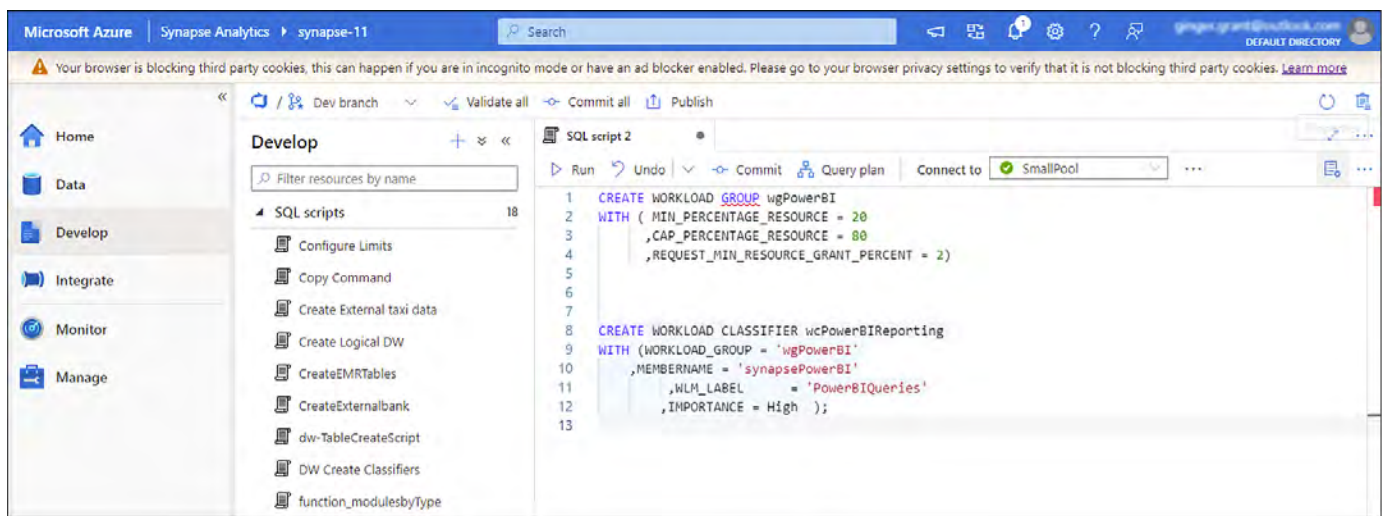


Figure 24: Developing a SQL script for creating workloads

To run the SQL query in a SQL notebook within Azure Synapse, as shown in the following statement, make sure that your dedicated pool is started as you cannot run any queries against it until it is running:

```
CREATE WORKLOAD GROUP wgPowerBI

WITH ( MIN_PERCENTAGE_RESOURCE = 20

      ,CAP_PERCENTAGE_RESOURCE = 80

      ,REQUEST_MIN_RESOURCE_GRANT_PERCENT = 2)
```

The workload group **wgPowerBI** will provide a maximum of 80% of the resources of Azure Synapse Analytics to elements within this workload. Each member will be granted a minimum of 2% of the resources when initially connecting. Make sure that when you create resource groups, the total of all of the minimum percentage resources adds up to 100%, as over-allocating resources can create performance issues. **REQUEST_MIN_RESOURCE_GRANT_PERCENT** is the number of chunks of resources that can be allocated at one time. Workload isolation allows you to dedicate resources to your workload groups. Here we see this workload group ELT is assigned at least 20% of the available resources using **MIN_PERCENTAGE_RESOURCE**. In the absence of workload isolation, requests operate in the shared pool of resources. Access to resources in the shared pool is not guaranteed and is assigned on the basis of importance.

Workload classification provides the ability to rate Power BI queries to be given more overall priority and resources over less important requests. There are five different weights you can assign to a workload classifier: *user*, *role*, *label*, *context* and *time*. These different elements are listed here in the order they are weighted. The element with the highest priority is *user*. If you include *user* in a classifier, anything executed by that user has the highest priority. The next highest value is *role*. Users in a given role are given less priority than a user, but are easier to manage. As users may have multiple rows, the highest priority is used in workload classifiers:

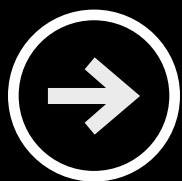
```
CREATE WORKLOAD CLASSIFIER wcPowerBIReporting  
  
WITH (WORKLOAD_GROUP = 'wgPowerBI'  
  
      ,MEMBERNAME = 'synapsePowerBI'  
  
      ,WLM_LABEL    = 'PowerBIQueries'  
  
      ,IMPORTANCE = High );
```

This classifier will use the dedicated resources defined in the workload group with members included in the Azure directory group called **synapsePowerBI** to improve the performance of reports connecting to data within a dedicated pool in Azure Synapse.



Summary

In this paper, we looked at five different ways to harness different features of Azure Synapse and use them with Power BI. Azure Synapse provides the ability to create data marts on top of data lakes for use within Power BI. You can use the data exploration features of Power BI to analyse data from within Azure Synapse to improve analytic capabilities. From within Azure Synapse, you can create virtual databases for use within Power BI. You can also add the ability to include machine learning functions created in Azure Synapse and report on the results. We looked at methods within Azure Synapse for performance tuning Power BI by adding workload management to ensure Power BI has all of the resources needed to gather data. These are a few ways of harnessing Azure Synapse's capabilities and once you start using it with Power BI, you can take advantage of all of these elements. Incorporating the elements discussed in this book will provide you with the abilities you need to take advantage of Power BI with Azure Synapse.



Next steps

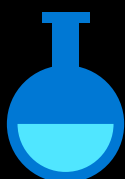
Your three steps to taking your BI and analytics to the next level:



Create a free [Azure Synapse workspace](#)



Watch the free [hands-on training series](#)



Use the [free quantities in Azure Synapse](#) to continue experimenting