

ALLGEIER

Creating a Modern Data Lakehouse with Azure Synapse Analytics

Contents

1	Workshop Preparations.....	3
2	Architecture	3
2.1	Components	4
3	Workshop Files.....	4
4	Provisioning Resources	4
4.1	Azure Account & Subscription.....	5
4.2	Create Azure Resource Group and Resources.....	6
4.3	Post Deployment	11
4.3.1	Role Assignment.....	11
4.3.2	Create Directories and Load Data into the Storage Account.....	13
5	Starting Azure Synapse Analytics.....	20
6	Create Azure Synapse Pipelines.....	22
6.1	Transform Data into Delta Format	22
6.1.1	Data Flows.....	22
6.2	Manually Trigger Pipelines (Before Data Changes).....	43
6.3	Data Changes.....	46
6.4	Manually Trigger Pipelines (After Data Changes).....	49
7	Create Azure Synapse SQL Database (OPTIONAL).....	52
7.1	Create a SQL Script	52
7.1.1	Bronze Database	52
8	Create Azure Synapse Lake Database	61
8.1	Create a Spark Notebook	61
8.1.1	Silver Database.....	61
8.1.2	Gold Database	77
9	Query Delta Formatted Data in Azure Storage with Serverless SQL Pool (OPTIONAL)	89
10	Connect to Power BI (OPTIONAL).....	93
10.1.1	Azure Synapse Analytics SQL Connector	94
10.1.2	Custom Delta Lake Connector	100
11	References	109

1 Workshop Preparations

The following prerequisites and setup must be done for successful completion of the course.

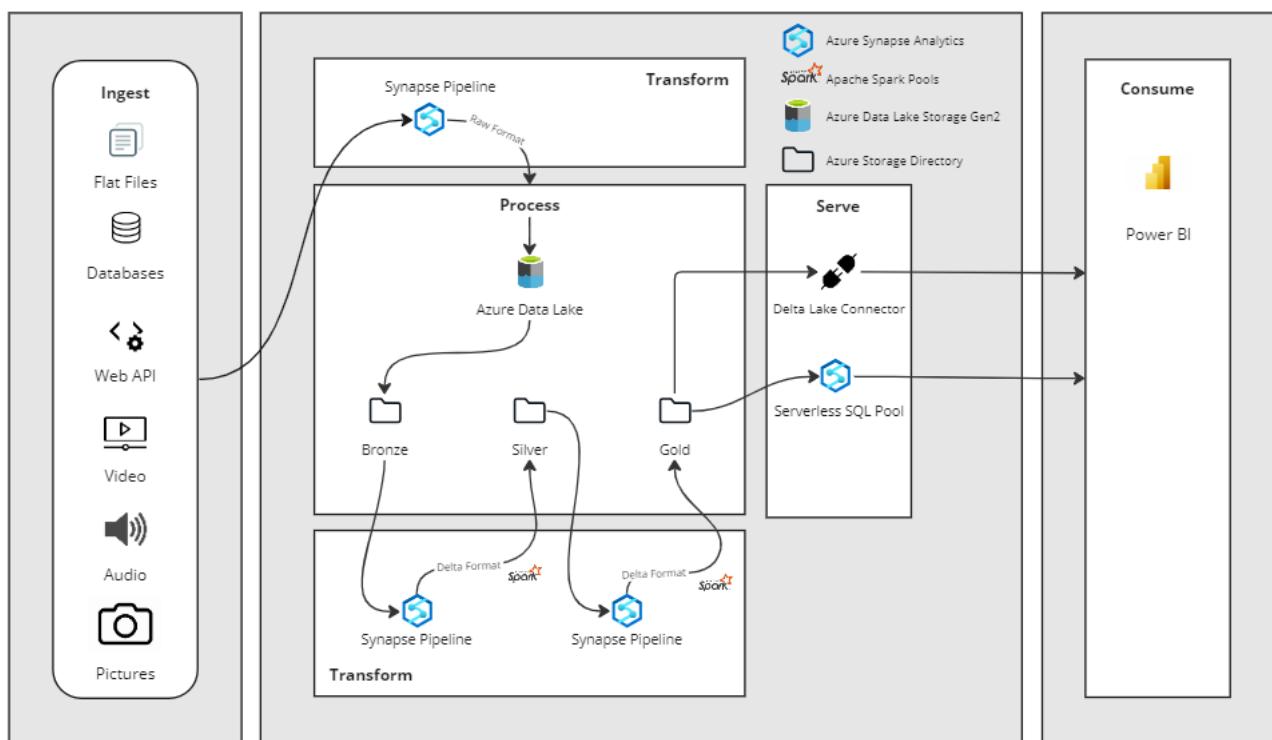
- You must be connected to the internet.
- You should use two monitors (**HIGHLY RECOMMENDED**)
- You have Microsoft Edge, or any other reliable browser installed.
- You must have created a [free Azure account](#). (see the PDF document **Creating a Free Azure Account**)
- You must **Owner** of the Azure subscription.
 - This is relevant if you are using a company Azure Subscription
- You may have [Power BI Desktop](#) installed (**OPTIONAL**).

2 Architecture

The Data Lakehouse is a data management architecture that combines the key features of data lakes and data warehouses to overcome their own drawbacks and offers low storage costs, ACID transactions, broad data access, data structures and governance features. These features enable business intelligence (BI) and machine learning (ML) on all data.

The architecture can be created using several cloud service providers such as: **Microsoft Azure**, **Databricks**, **Amazon Webservices**, **Google Cloud Platform** and **Snowflake**, to name a few.

In this workshop, we will be using **Microsoft Azure**. The architecture setup will follow the workflow of the diagram below:



2.1 Components

The architecture will be composed of the following Azure resources:

- Azure Synapse Analytics
 - Apache Spark Pool (Apache Spark Version 3.3.1, Delta Lake Version 2.2.0)
- Azure Storage Account (Azure Data Lake Gen2)

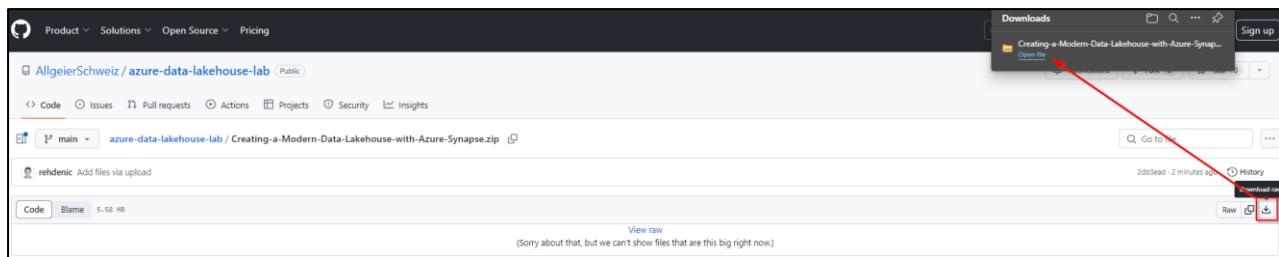
Additionally, a BI tool will be used to import and visualize the data:

- Power BI (**OPTIONAL**)

NOTE: All chapters labeled OPTIONAL may be started but are not required. During the workshop, we recommend skipping these chapters and finishing them separately.

3 Workshop Files

All files are located in a GitHub repository and can be download as a zip file from this [link](#).



4 Provisioning Resources

We will be creating the Azure components using an Azure Resource Management (ARM) template.

Additionally, the data being used comes from the [AdventureWorks](#) sample datasets made available by Microsoft.

All files, including the ARM template, can be found in the GitHub repository mentioned above, located [here](#).

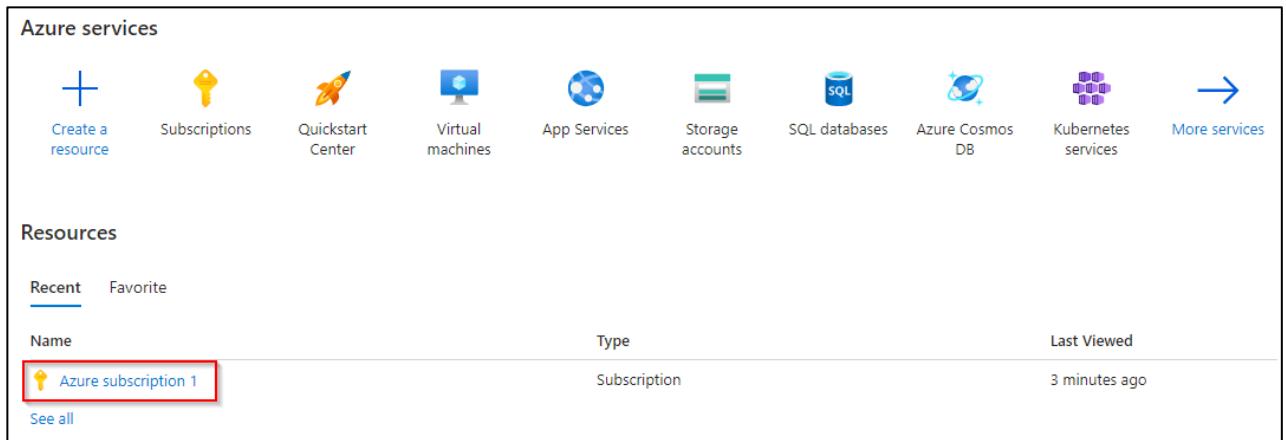
NOTE: The images in this document might differ slightly from the text in each step. This is usually the case when it comes to the naming of Azure resources. This is due to changes over time. Orient yourself primarily to the text.

4.1 Azure Account & Subscription

The Azure Account you created in the PDF document **Creating a Free Azure Account** is yours and automatically grants you the **Owner role** to the created Azure Subscription.

NOTE: If

The Azure Subscription automatically provisioned for you, when you created your free Azure account, is named **Azure Subscription 1**.



Azure services

Create a resource Subscriptions Quickstart Center Virtual machines App Services Storage accounts SQL databases Azure Cosmos DB Kubernetes services More services

Resources

Recent Favorite

Name	Type	Last Viewed
Azure subscription 1	Subscription	3 minutes ago

See all

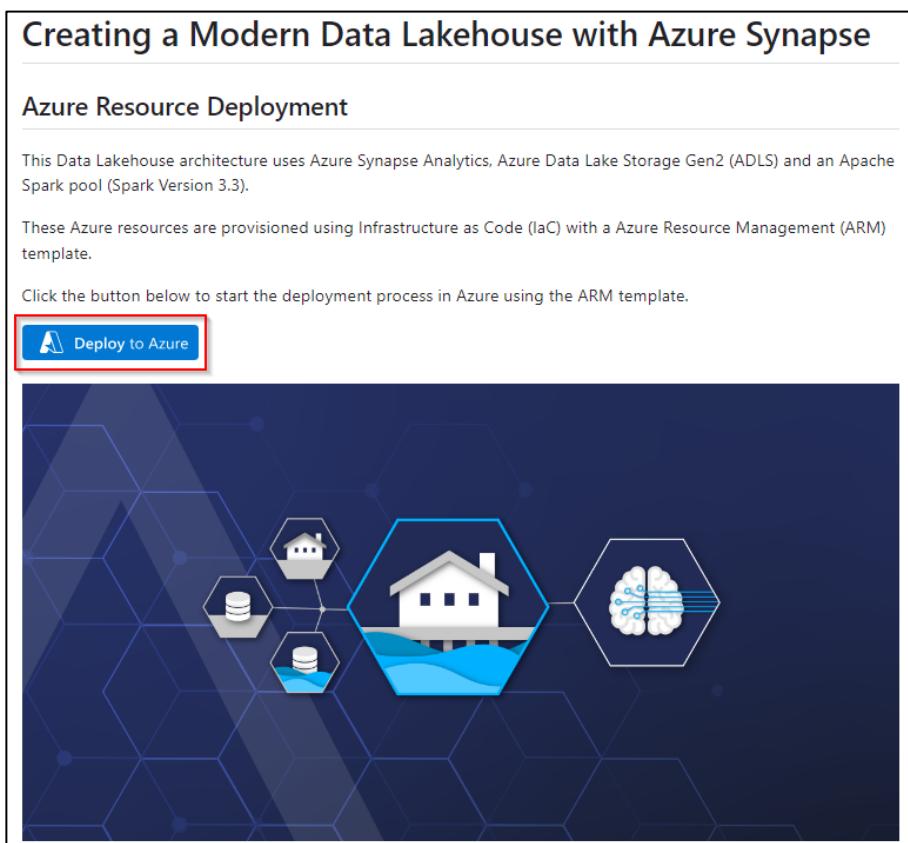
4.2 Create Azure Resource Group and Resources

You will now create the Azure Resources required to set up the Data Lakehouse. We will use an Azure Resource Management (ARM) template to help provision the required Azure resources more efficiently.

NOTE: Use the username and password you created for your free Azure account in step 1 below.

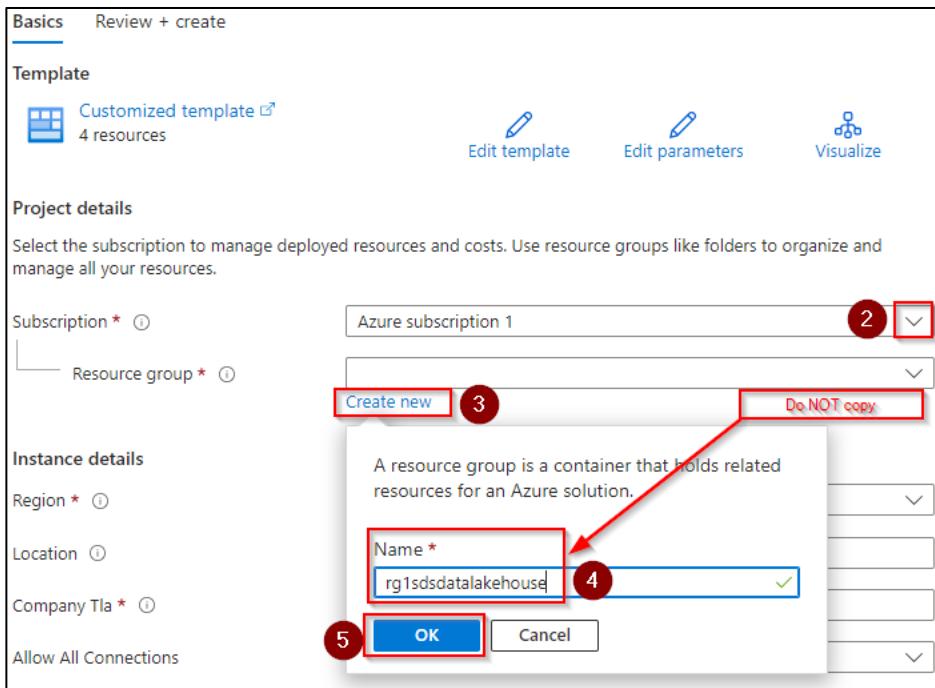
CAREFUL: Before you begin, make sure you register all the resource providers as shown in the PDF document *Creating a Free Azure Account*.

1. Open this [link](#) and select **Deploy to Azure**. This will call the *azure.portal.com* website and start the ARM template provided.



Once the template opens, there will be several entries that need to be manually input and / or changed.

2. Select the **Subscription** Azure Subscription 1.
3. Create a new **Resource Group** by selecting the **Create New** button by the Resource Group entry.
4. Add a new **Resource Group** name with the following make up:
 - a. **rg1 <dhlxyYOUR NUMBER ON THE HANDOUT> datalakehouse** (If your number is 12, the full name becomes: rg1dhlxy12datalakehouse).
5. Select **OK**



6. Set the **Region** to *the region assigned to you at the beginning of the workshop. (YOUR REGION ON THE HANDOUT)*.
7. Leave the **Location** as *[resourceGroup()].location*.
8. Set the **Company Tla** to <dhlxy*YOUR NUMBER ON THE HANDOUT*>. (If your number is 12, the full name becomes: dhlxy12).
9. Leave the **Allow All Connections** as *true*.
10. Leave the **Spark Deployment** as *true*.
11. Set the **Spark Node Size** as *Small*.
12. Leave the **Spark Version** as *3.3*.
13. Leave / Set the **Deployment type** as *poc*.
14. Define the **Sql Administrator Login** in the template as *sqladmin*
15. Define the **Sql Administrator Login Password** in the template as *5Xb%3Ns+b{bH\$7BC#\$D*

NOTE: Use the **Sql Administrator Login Password** given to you in step 15. If you select your own, the deployment might fail. This is due to Microsoft password rules not being met.

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

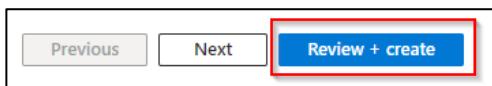
Subscription * ① Azure subscription 1

Resource group * ② (New) rg1sdsdatalakehouse
Create new Do NOT copy

Instance details

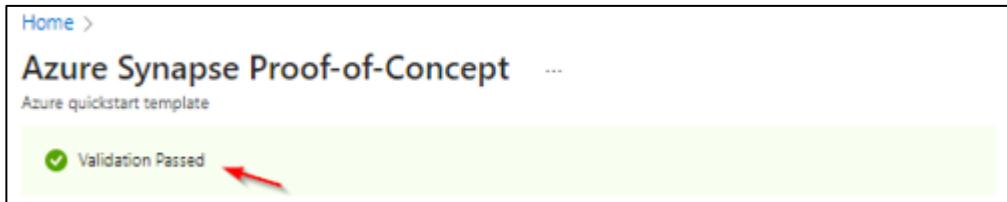
Region * ③	6 West Europe
Location ④	7 [resourceGroup().location]
Company Tla * ⑤	8 sds
Allow All Connections	9 true
Spark Deployment ⑥	10 true
Spark Node Size ⑦	11 Large
Spark Version ⑧	12 3.3
Deployment Type ⑨	13 poc
Sql Administrator Login * ⑩	14 sqladmin
Sql Administrator Login Password * ⑪	15 Do NOT copy

16. Select **Review + create** to start provisioning the Azure resources.



17. You might get a **Validation Passed** if all your template inputs are correct.

NOTE: Since you're using a free Azure account, this message might not show up. You can continue to the next step.



18. Select **Create** to start the provisioning process.



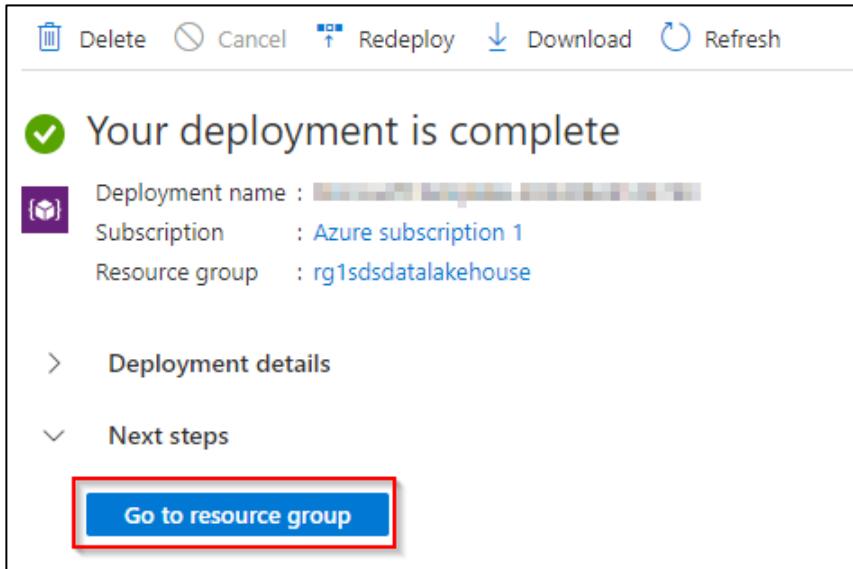
19. Once the provisioning of the resources has started, you will be brought to a page where **deployment is in progress** can be seen.

A screenshot of a deployment status page. At the top, it says "... Deployment is in progress". Below that, there are deployment details: Deployment name: [REDACTED], Subscription: [REDACTED], Resource group: rg1sdsdatalakehouse. To the right, it shows Start time: [REDACTED] and Correlation ID: [REDACTED]. A blue 'Copy' icon is also present. Below the details, there's a section titled "Deployment details" with a collapse arrow. A table lists the resources being deployed:

Resource	Type	Status
sds pocws1	Microsoft.Synapse/workspaces	Created
dlssdspoc/default/dlssdspocfs1	Microsoft.Storage/storageAccounts/blobServices/contain...	Created
dlssdspoc	Microsoft.Storage/storageAccounts	OK

NOTE: It may take up to 10 minutes for the deployment to finalize.

20. Once the deployment is completed, select Go to **resource group** to go to the provisioned resources.



Once selected, you will be brought to the resource group where you will see the resources you created using the template.

The screenshot shows the Azure Resource Group 'rg1sdsdatalakehouse' page. The title bar has the resource group name 'rg1sdsdatalakehouse' with a red box around it. The left sidebar includes links for Overview, Activity log, Access control (IAM), Tags, Resource visualizer, Events, Deployments, Security, Policies, Properties, and Locks. The main content area has tabs for 'Essentials' and 'Resources'. Under 'Essentials', it shows Subscription (move), Subscription ID, and Tags (edit). Under 'Resources', it shows a table of provisioned resources:

Name	Type	Location
dlssdpoc	Storage account	Switzerland North
sdspocws1	Synapse workspace	Switzerland North
synasp1 (sdspocws1/synasp1)	Apache Spark pool	Switzerland North

A red box highlights the first row of the table.

4.3 Post Deployment

4.3.1 Role Assignment

To run queries using the Serverless SQL pool in Azure Synapse, you first need to grant yourself access to the storage account by assigning the **Storage Blob Data Contributor** role to your user ([trouble shooting](#)).

1. Within your newly created **Resource group**, select the resource type **Storage account**.

Name ↑↓	Type ↑↓	Location ↑↓
dlssdspoc	Storage account	Switzerland North
sdspscws1	Synapse workspace	Switzerland North
synasp1 (sdspscws1/synasp1)	Apache Spark pool	Switzerland North

2. On the left panel select the **Access Control (IAM)** and add a new role assignment by selecting the **+ Add** button.

The screenshot shows the 'Access Control (IAM)' blade for a storage account named 'dlssdspoc'. The left sidebar has a tree view with 'Access Control (IAM)' selected (marked with a red circle 1). The main area shows a search bar and a list of roles: 'Add role assignment' (marked with a red circle 2), 'Add co-administrator', and 'Roles'. A tooltip for 'Add role assignment' points to the 'Add role assignment' button (marked with a red circle 3). Below this is a section for 'My access' and 'Check access'.

- Search for **Storage Blob Data Contributor**, select / highlight the **Storage Blob Data Contributor** from the list and select **Next**.

The screenshot shows the 'Add role assignment' interface. At the top, there are tabs for 'Role', 'Members' (which is selected), 'Conditions (optional)', and 'Review + assign'. Below the tabs, a note states: 'A role definition is a collection of permissions. You can use the built-in roles or you can create your own custom roles. [Learn more](#)'.

The 'Assignment type' section includes 'Job function roles' (selected) and 'Privileged administrator roles'. A note below says: 'Grant access to Azure resources based on job function, such as the ability to create virtual machines.'

In the search bar, 'storage blob data contributor' is typed, circled with a red number 1. To the right of the search bar are filters: 'Type : All' and 'Category : All'. Below the search results, the 'Selected' role is shown: 'Storage Blob Data Contributor' (circled with a red number 2). A tooltip for this role states: 'Allows for read, write and delete access to Azure Storage blob containers and data'.

At the bottom, there are buttons for 'Review + assign', 'Previous', and 'Next' (circled with a red number 3).

Toggle **User, group, or service principal** and then select the button + **Select members**. A new panel will open to the far right.

The screenshot shows the 'Add role assignment' interface with the 'Members' tab selected. The 'Selected role' is 'Storage Blob Data Contributor'. Under 'Assign access to', the radio button for 'User, group, or service principal' is selected (circled with a red number 1). Below that, 'Managed identity' is an option. The 'Members' section contains a button '+ Select members' (circled with a red number 2).

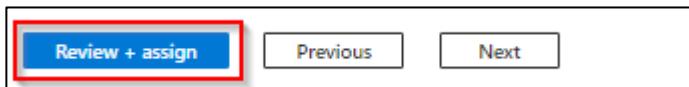
- Within this new panel, search for **your** user E-Mail address
- Select / highlight the row with your E-mail address and click **Select**.

The screenshot shows the 'Select members' dialog. It has a search bar 'Search for E-Mail' and a 'Select' button. Below the search bar, the user's email address 'nrehder@allgeier.ch' is typed into a field. In the results list, a row for 'Nicolas Rehder' is highlighted (circled with a red number 3). This row contains a small profile icon with 'NR' and the name 'Nicolas Rehder'.

Selected members:

 Nicolas Rehder
[nicholas.rehder@...](mailto:nicolas.rehder@...) [Remove](#)

6. Once selected, you will **Review + assign** the role assignment.



7. You will get the following banner once successful:



4.3.2 Create Directories and Load Data into the Storage Account

1. Go back to the [Azure portal](#) home screen and select the **Resource group** you provisioned.

- Once the resource group is open, select the **Storage account**.

Name ↑↓	Type ↑↓	Location ↑↓
<input type="checkbox"/> dlssdspoc	Storage account	Switzerland North
<input type="checkbox"/> sdspscws1	Synapse workspace	Switzerland North
<input type="checkbox"/> synasp1 (sdspscws1/synasp1)	Apache Spark pool	Switzerland North

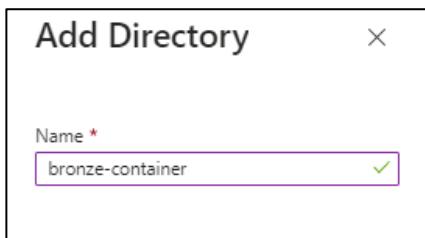
- Select the **Container** tab on the left side and select the named container.

The screenshot shows the 'Containers' blade for the storage account 'dlssdspoc'. On the left, a navigation menu includes 'Overview', 'Activity log', 'Tags', 'Diagnose and solve problems', 'Access Control (IAM)', 'Data migration', 'Events', 'Storage browser', and 'Containers'. The 'Containers' item is highlighted with a red box and a number '1'. In the main area, there's a search bar, a '+ Container' button, and a 'Change access level' button. Below these are sections for 'Search containers by prefix' and 'Name'. A container named 'dlssdspocfs1' is listed, also highlighted with a red box and a number '2'.

- Within the named container, select **+ Add Directory** to create a new folder.

The screenshot shows the 'dlssdspocfs1' container details page. The left sidebar lists 'Overview', 'Diagnose and solve problems', 'Access Control (IAM)', 'Settings' (with 'Shared access tokens', 'Manage ACL', 'Access policy', 'Properties', and 'Metadata'), and a 'Search' bar. The right side shows 'Authentication method: Access key (Switch to A)' and 'Location: dlssdspocfs1'. It has a 'Search blobs by prefix (case-sensitive)' bar and a 'Name' section where a directory named 'synapse' is listed. At the top, there are buttons for 'Upload', '+ Add Directory' (highlighted with a red box), and 'Refresh'.

5. A new panel will open on the right side, where you can define the name of the new Directory. You will create 3 new directories, and label them **bronze-container**, **silver-container**, and **gold-container**



Your storage container structure should look like this:

The screenshot shows the Azure Data Lake Storage interface. At the top, there are navigation buttons: Upload, Add Directory, Refresh, Rename, and Delete. Below that, it displays the authentication method as 'Access key (Switch to Azure AD User Account)' and the location as 'dlssdspocfs1'. There is a search bar for blobs by prefix. The main area is a table with a column labeled 'Name'. It lists five items: 'bronze-container', 'gold-container', 'silver-container', and 'synapse'. The first three items ('bronze-container', 'gold-container', and 'silver-container') are highlighted with a red box around their row.

Name
<input type="checkbox"/> bronze-container
<input type="checkbox"/> gold-container
<input type="checkbox"/> silver-container
<input type="checkbox"/> synapse

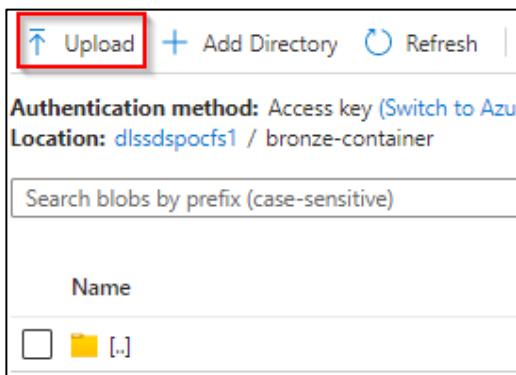
You will now add two CSV data files manually into the storage bronze-container directory.

6. Within the Data Lake, first select the **bronze-container**.

The screenshot shows the Azure Data Lake Storage interface. The 'bronze-container' directory is selected, indicated by a red box around its row in the list. The other items ('gold-container', 'silver-container', and 'synapse') are not selected.

Name
<input checked="" type="checkbox"/> bronze-container
<input type="checkbox"/> gold-container
<input type="checkbox"/> silver-container
<input type="checkbox"/> synapse

7. Within the bronze-container, select the **Upload** button.



8. A new panel will open to the right. **Drag and Drop** the CSV files **FactProductCategoryPredictions.csv** and **FactProductSales.csv** and select **Upload**.

The image consists of two parts. The top part is a screenshot of the Azure Storage Explorer showing the 'Data' folder. Inside the folder, there are two CSV files: 'FactProductCategoryPredictions' and 'FactProductSales', both highlighted with a red box. The bottom part is a screenshot of the 'Upload blob' dialog box. It shows a list with a cloud icon and the number '1'. Below the list, it says '2 file(s) selected: FactProductCategoryPredictions.csv, FactProductSales....'. There is a checkbox for 'Overwrite if files already exist' and a 'Advanced' section. At the bottom, there is a large blue 'Upload' button (highlighted with a red box) and a 'Give feedback' link.

Your bronze-container directory should look like this:

The screenshot shows the Azure Storage Explorer interface. At the top, there are buttons for Upload, Add Directory, Refresh, Rename, and Delete. Below that, it says "Authentication method: Access key (Switch to Azure AD User Account)" and "Location: dlssdspocfs1 / bronze-container". There is a search bar labeled "Search blobs by prefix (case-sensitive)". The main area is titled "Name" and lists two files: "FactProductCategoryPredictions.csv" and "FactProductSales.csv", both of which are highlighted with a red box.

Go back to the root of the container.

9. Select the **silver-container** directory.

The screenshot shows the Azure Storage Explorer interface for the "dlssdspocfs1" container. On the left, there is a sidebar with "Overview", "Diagnose and solve problems", and "Access Control (IAM)". On the right, there is a "Settings" section with "Shared access tokens", "Manage ACL", "Access policy", "Properties", and "Metadata". Below that is a "Name" column listing several directories: "bronze-container", "gold-container", "silver-container" (which is highlighted with a red box and has a red circle with the number 2), and "synapse".

10. Within the **silver-container**, select **+ Add Directory** to create a new folder.

The screenshot shows the Azure Storage Explorer interface for the "dlssdspocfs1" container, specifically within the "silver-container". The "Add Directory" button at the top is highlighted with a red box and has a red circle with the number 1. The "Location" field below it also has a red box around it. The rest of the interface is similar to the previous screenshot, showing the "Overview" tab selected and the "Properties" section on the left.

11. A new panel will open on the right side, where you can define the name of the new Directory. You will create 1 new directory called **ProductSales**



Your storage container structure should look like this:

The screenshot shows a storage container interface. At the top, there are buttons for Upload, Add Directory, Refresh, and a search bar. Below that, the authentication method is set to 'Access key' and the location is 'dlssdspocfs1 / silver-container'. A search bar labeled 'Search blobs by prefix (case-sensitive)' is present. Under the 'Name' heading, a list of directories is shown, with 'ProductSales' highlighted by a red box.

Go back to the root of the container.

12. Select the **gold-container** directory.

The screenshot shows the 'dlssdspocfs1' container. On the left, a sidebar lists options: Overview, Diagnose and solve problems, Access Control (IAM), Settings (Shared access tokens, Manage ACL, Access policy, Properties, Metadata). The 'Overview' tab is selected. On the right, the location 'dlssdspocfs1' is highlighted with a red box (1). Below it, a list of containers is shown: bronze-container, gold-container (highlighted with a red box and red circle 2), silver-container, and synapse.

13. Within **the gold-container**, select **+ Add Directory** to create a new folder.

The screenshot shows the Azure Storage Explorer interface for a container named 'dlssdspocfs1'. At the top, there's a search bar, an 'Upload' button, and a red-highlighted 'Add Directory' button. Below the search bar are three navigation links: 'Overview', 'Diagnose and solve problems', and 'Access Control (IAM)'. To the right of these links, it says 'Authentication method: Access key (Switch to Azure AD User Account)' and 'Location: dlssdspocfs1 / gold-container'. A red box highlights the 'gold-container' part of the location path. At the bottom, there's a search bar with the placeholder 'Search blobs by prefix (case-sensitive)'.

14. A new panel will open on the right side, where you can define the name of the new Directory. You will create 1 new directory called **ProductSales**

The screenshot shows a modal dialog titled 'Add Directory'. It has a text input field labeled 'Name *' containing the value 'ProductSales', which is preceded by a small green checkmark icon. There is also an 'X' button in the top right corner of the dialog.

Your storage container structure should look like this:

The screenshot shows the 'dlssdspocfs1' container in the Azure Storage Explorer. It includes the same header and sidebar as the previous screenshot. Below the header, there's a table with a single row. The first column is labeled 'Name' and contains a checkbox followed by a yellow folder icon and the text 'ProductSales'. A red box highlights the 'ProductSales' folder icon. The rest of the table row and the rest of the page are mostly blank.

5 Starting Azure Synapse Analytics

1. Go back to the [Azure portal](#) home screen and select the **Resource group** you provisioned.

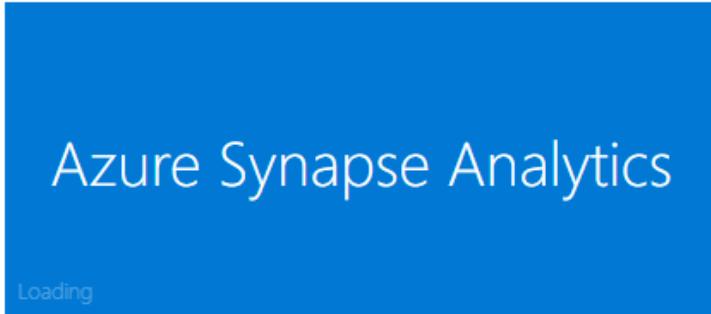
- Once the resource group is open, select the resource type **Synapse workspace**.

<input type="checkbox"/>	Name ↑↓	Type ↑↓	Location ↑↓
<input type="checkbox"/>	 dlssdpoc	Storage account	Switzerland North
<input type="checkbox"/>	 sdpocws1	Synapse workspace	Switzerland North
<input type="checkbox"/>	 synasp1 (sdpocws1/synasp1)	Apache Spark pool	Switzerland North

3. A new page will appear. Launch **Azure Synapse Studio** by selecting **Open Synapse Studio** located in the **overview** section of the Synapse workspace.

The screenshot shows the Azure Synapse Analytics workspace overview page. The left sidebar includes links for Activity log, Access control (IAM), Tags, Diagnose and solve problems, Settings (Azure Active Directory, Properties, Locks), Analytics pools (SQL pools, Apache Spark pools, Data Explorer pools (preview)), and Security. The main area has sections for Essentials (Resource group, Status, Location, Subscription) and Getting started. The Getting started section features a banner for 'Open Synapse Studio' with the text: 'Start building your fully-integrated analytics solution and unlock new insights.' and a 'Open' button.

Once selected, you will open a new webpage, launching Azure Synapse Studio. You will see the following banner:



6 Create Azure Synapse Pipelines

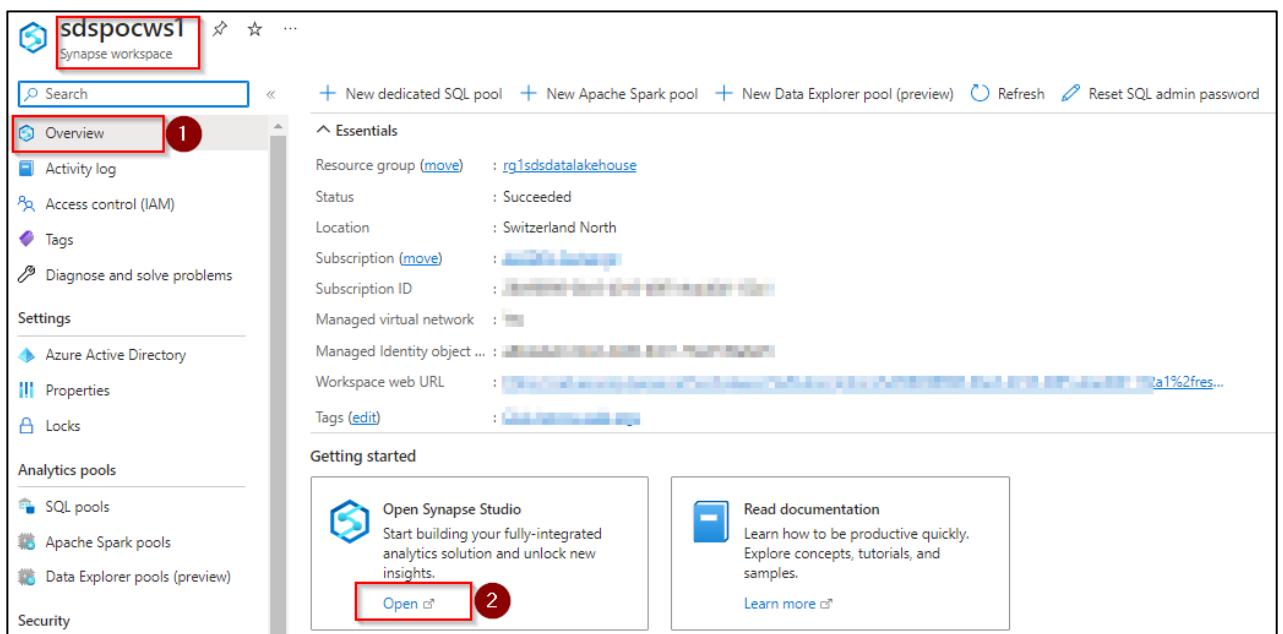
You will create two data flows using Synapse pipelines. These dataflows will move the data in the bronze container into the silver container and from the silver container into the gold containers. Each data flow will transform the data before copying it into each container in the Delta Format.

6.1 Transform Data into Delta Format

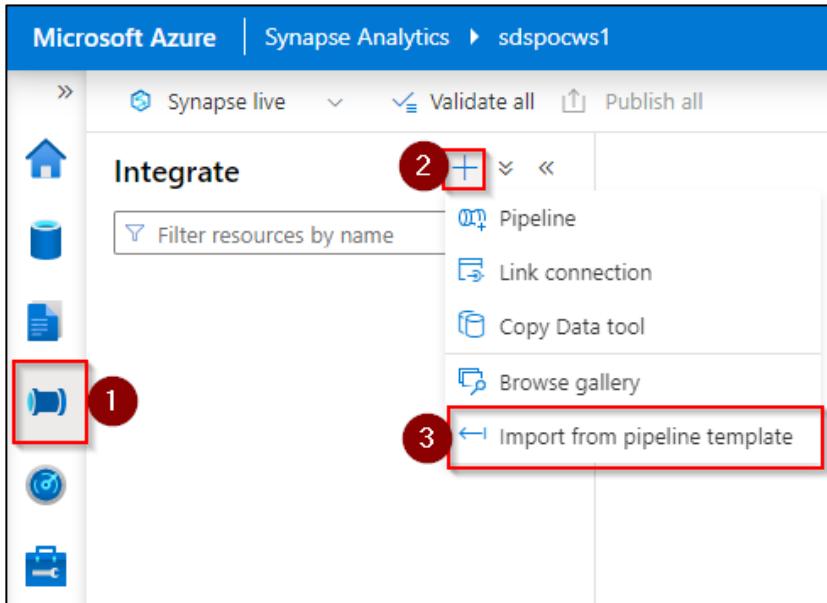
The Delta format is an open format storage layer that brings ACID (atomicity, consistency, isolation, and durability) transactions, time travelling capabilities, DML operations and many [more features](#) to Apache Spark. The Delta format stores the data in a parquet file format while also maintaining transaction logs and statistics to provide features and performance improvement over standard parquet.

6.1.1 Data Flows

1. If you haven't already, launch **Azure Synapse Studio** by selecting **Open Synapse Studio** located in the overview section of the Synapse workspace.

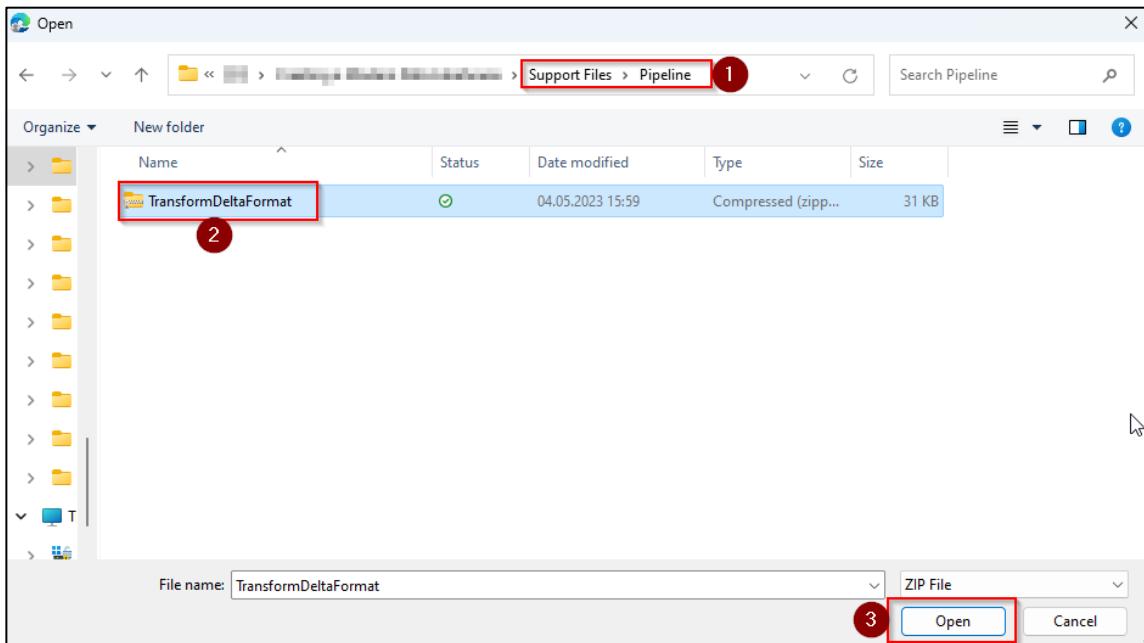


- Once **Synapse Studio** opens, on the left side, select the **Integrate** tab, the **plus sign (+)** at the top of the new panel and **Import from Pipeline Template** in the drop down, to create a new pipeline.



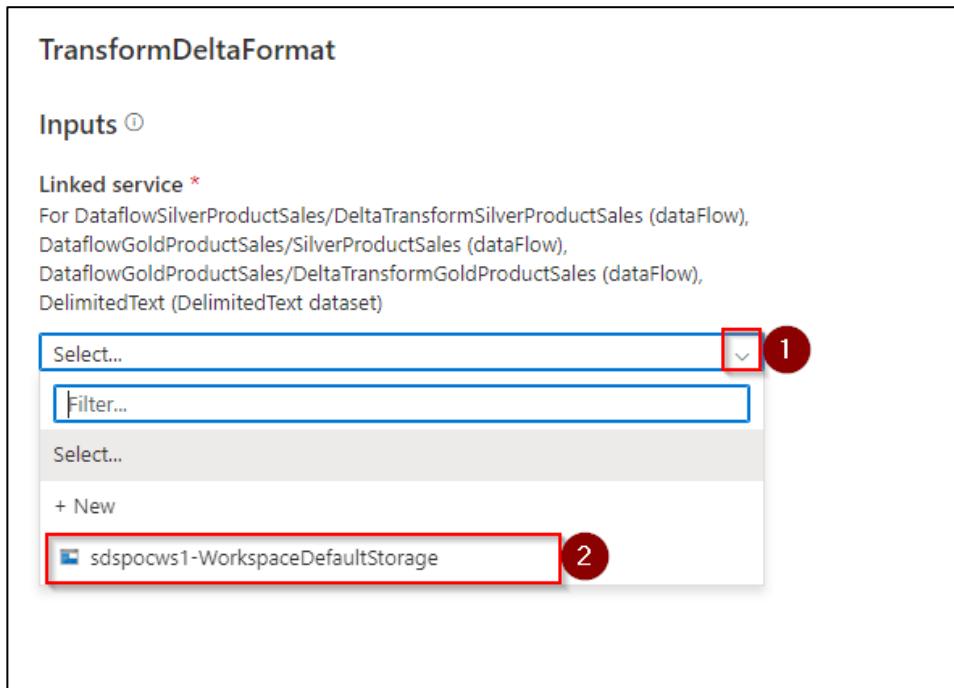
- Once **Import from Pipeline Template** is selected, a new window to browse to a ZIP file will open.
- Browse to the Folder **Support Files** and then **Pipeline**.
- Select the ZIP file **TransformDeltaFormat** and click **Open**.

NOTE: The Support Files can be found in the Git repository in this [link](#). Download these files onto your local computer if you haven't already.



A new window will open to configure the Linked Service.

6. Within the drop box under Linked Service, select the provisioned storage account.

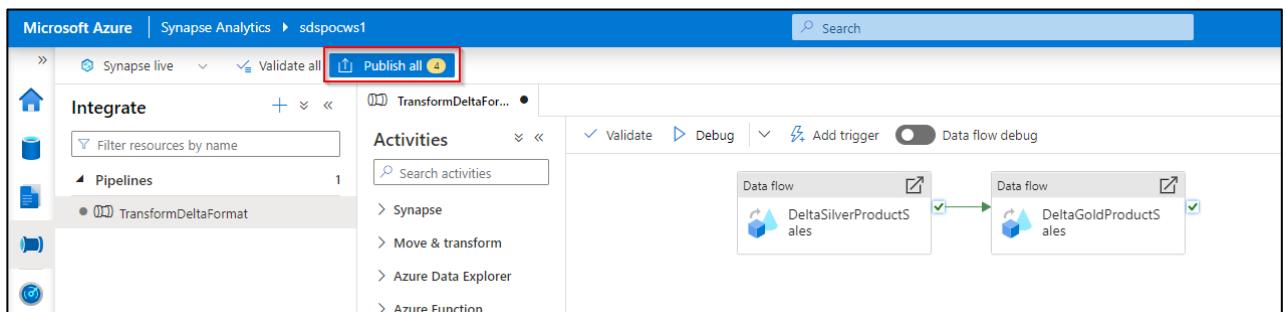


7. Once selected, click **Open pipeline** at the bottom of the page.

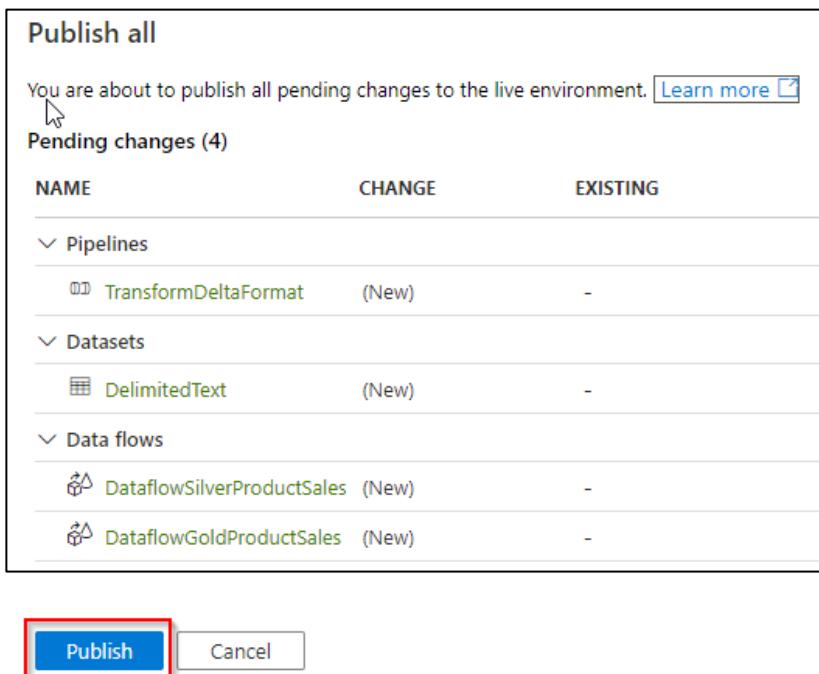


The pipeline will be imported, and you should see two data flows called **DeltaSilverProductSales** and **DeltaGoldProductSales**.

8. Select the **Publish all** button to save the work you have done so far.



9. A new panel should open to the right. Select the **Publish** button.

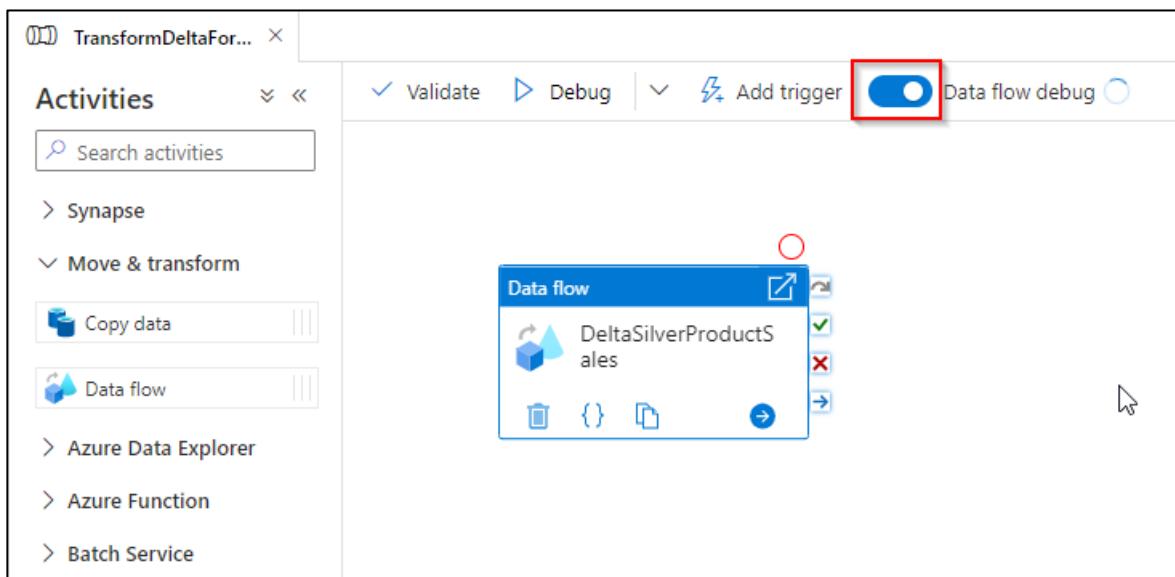


A banner should appear confirming that the content was published.

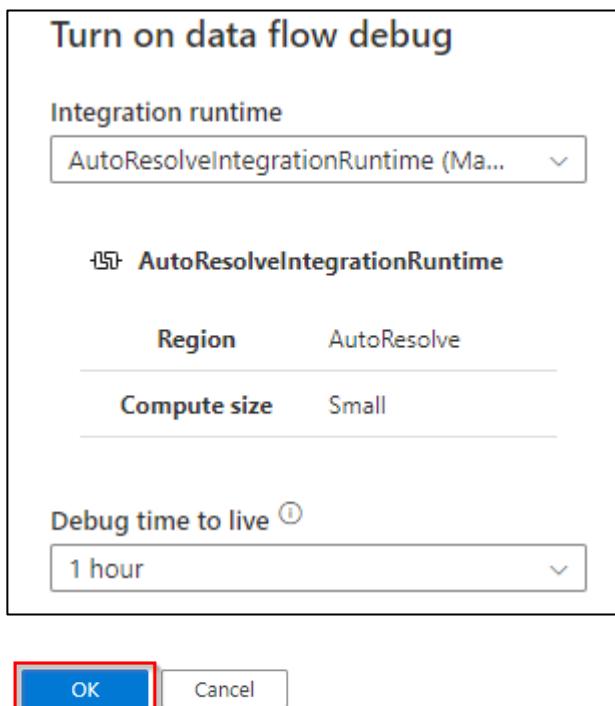


10. In the top bar of the pipeline canvas, slide the **Data flow debug** slider on.

Debug mode allows for interactive testing of transformation logic against a live Spark cluster. Data Flow clusters take 5 - 7 minutes to warm up and users are recommended to turn on debug first if they plan to do Data Flow development.



11. Once toggled, a new panel will open to the right, with the debug setting. Leave the default settings and select **OK**.



Note: The debug initialization might take 3 – 5 minutes. Please be patient!

12. You will get a green check once the debug has initialized:

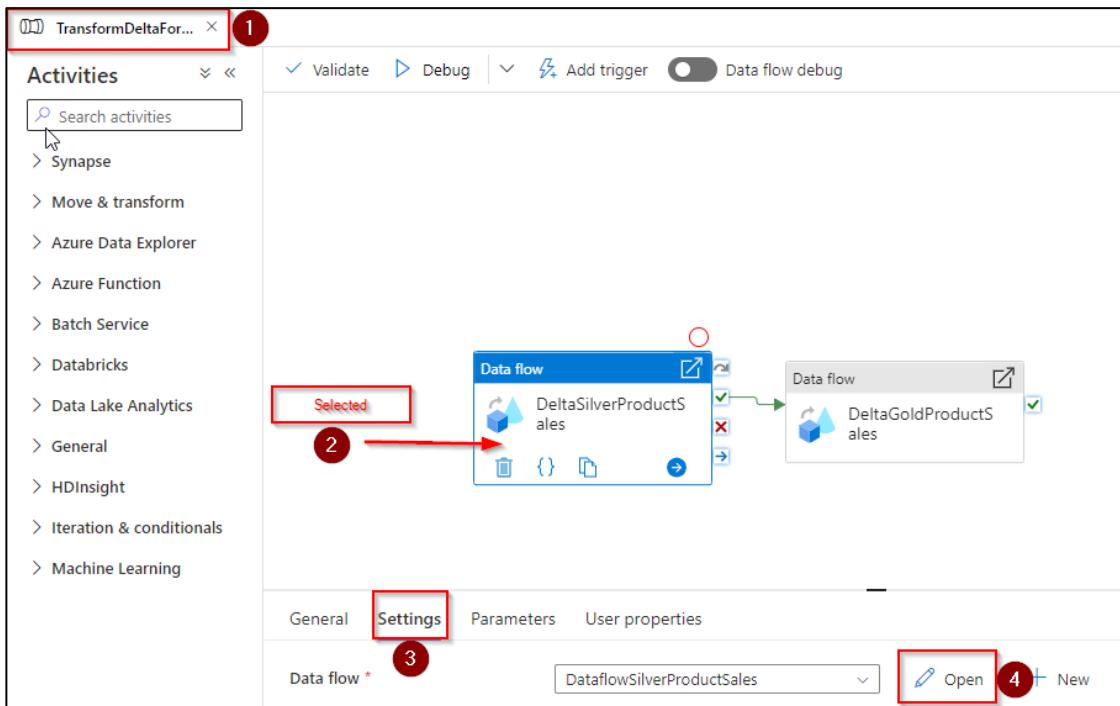


The **Debug time to live** is the amount of time that the integration runtime will wait after your last data preview before automatically shutting down your debug cluster. To avoid billing for the entire TTL, you can shut down the debug session when you are finished working.

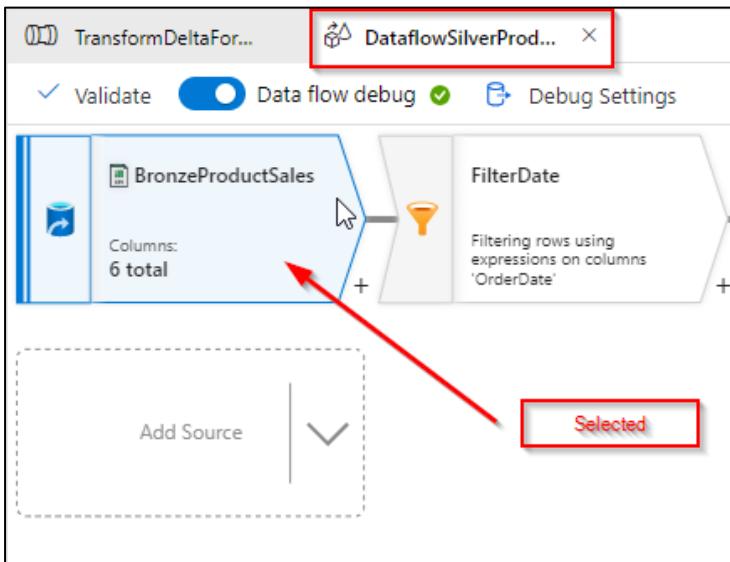
6.1.1.1 DeltaSilverProductSales

Before the data flows can be triggered, we need to make sure that the data sources i.e., the CSV files being imported into the dataflows are being found and that the sink sources are correctly mapped. This requires us to adjust parameters and change file paths.

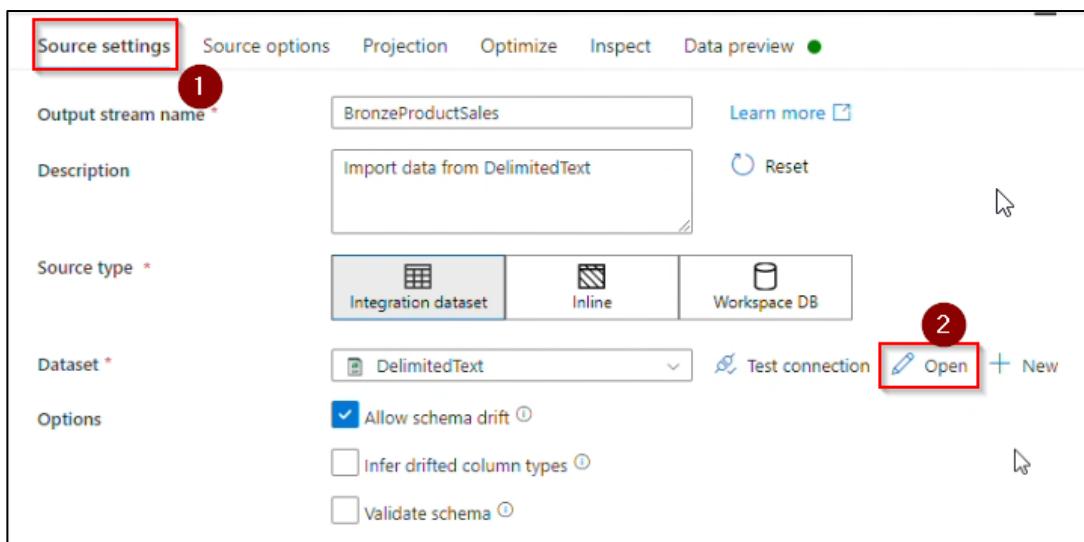
1. Within the imported **TransformDeltaFormat** pipeline, select the Data flow **DeltaSilverProductSales**.
2. In the **Settings** tab select the **Open** button next to the Data flow **DeltaSilverProductSales**.



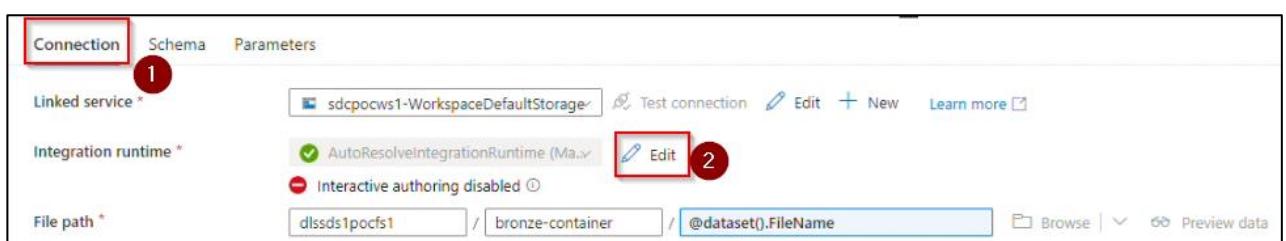
- Within the opened data flow, select the **BronzeProductSales** tile.



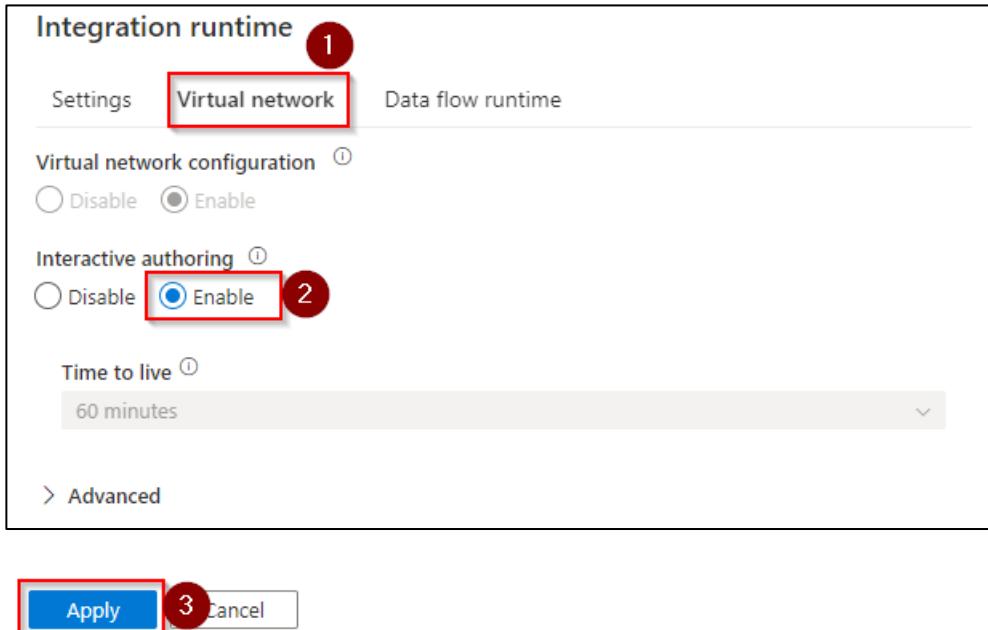
- Under the **Source settings** tab, select the **Open** button next to **Dataset (DelimitedText)**



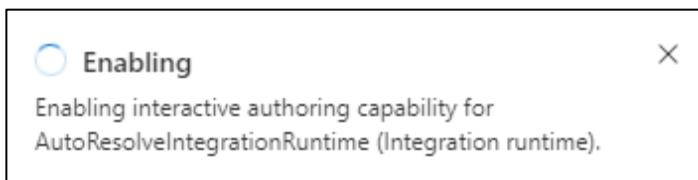
- Once selected a new tab called **DelimitedText** will open.
- Under the **Connection** tab, select the **Edit** button next to **Integration runtime**.



- A new panel called **Integration runtime** will open. Go to the **Virtual Network** tab and select **Enable** under Interactive authoring. Select **Apply**.

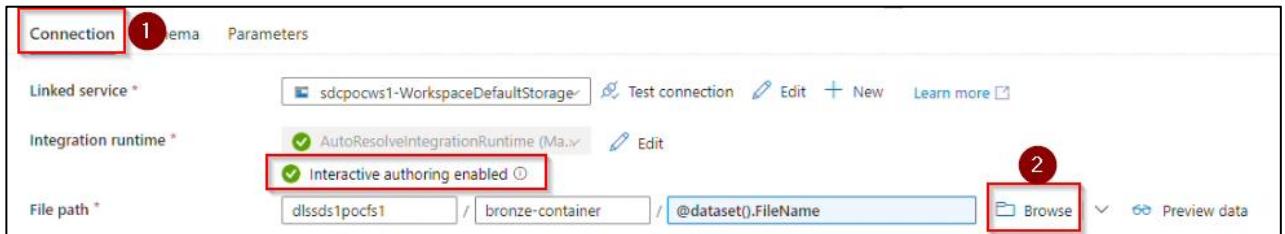


NOTE: It may take 1 – 3 minutes for interactive authoring to be enabled. You will see the banner below:



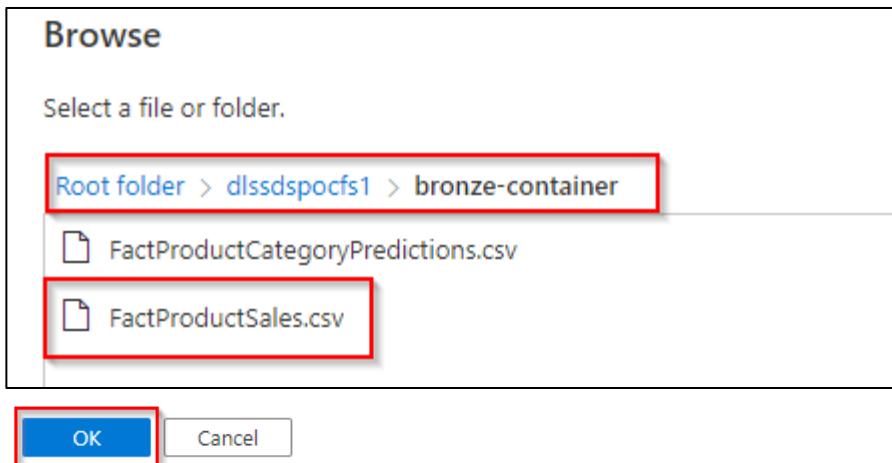
Once enabled, you will be able to Browse the directory folder.

- Under the **Connection** tab, select the **Browse** button next to **File path**.



- Browse to the **bronze-container** directory and select the **FactProductSales.csv** file.
- Once the file path has been set, select **OK**.

NOTE: The path should be similar to the one depicted below. Keep in mind that your storage account (E.g.: **dlssdsspocfs1**) will have a different name.



NOTE: Even though you selected the FactProductSales.csv file, your file path should retain the parameter @dataset().FileName. The only thing that should change is the storage name.

Connection Schema Parameters

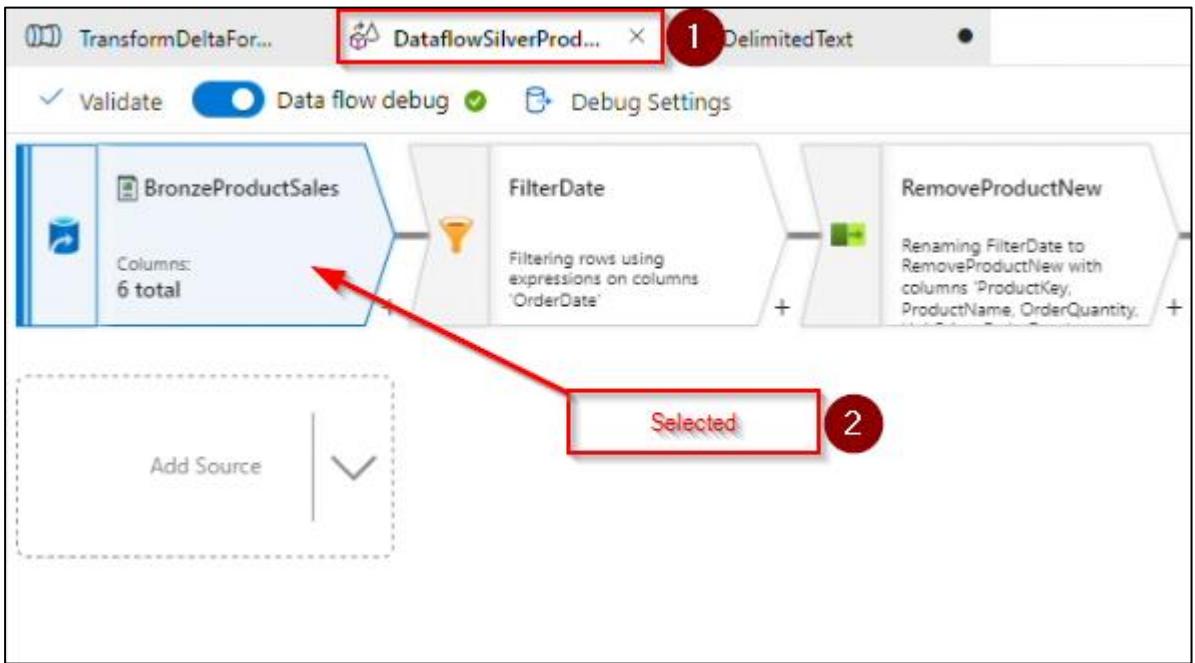
Linked service * sdcpcws1-WorkspaceDefaultStorage Test connection Edit + New Learn more

Integration runtime * AutoResolveIntegrationRuntime (Ma.) Edit

Interactive authoring enabled

File path * dlssdpcfs1 / bronze-container / @dataset().FileName Browse Preview data

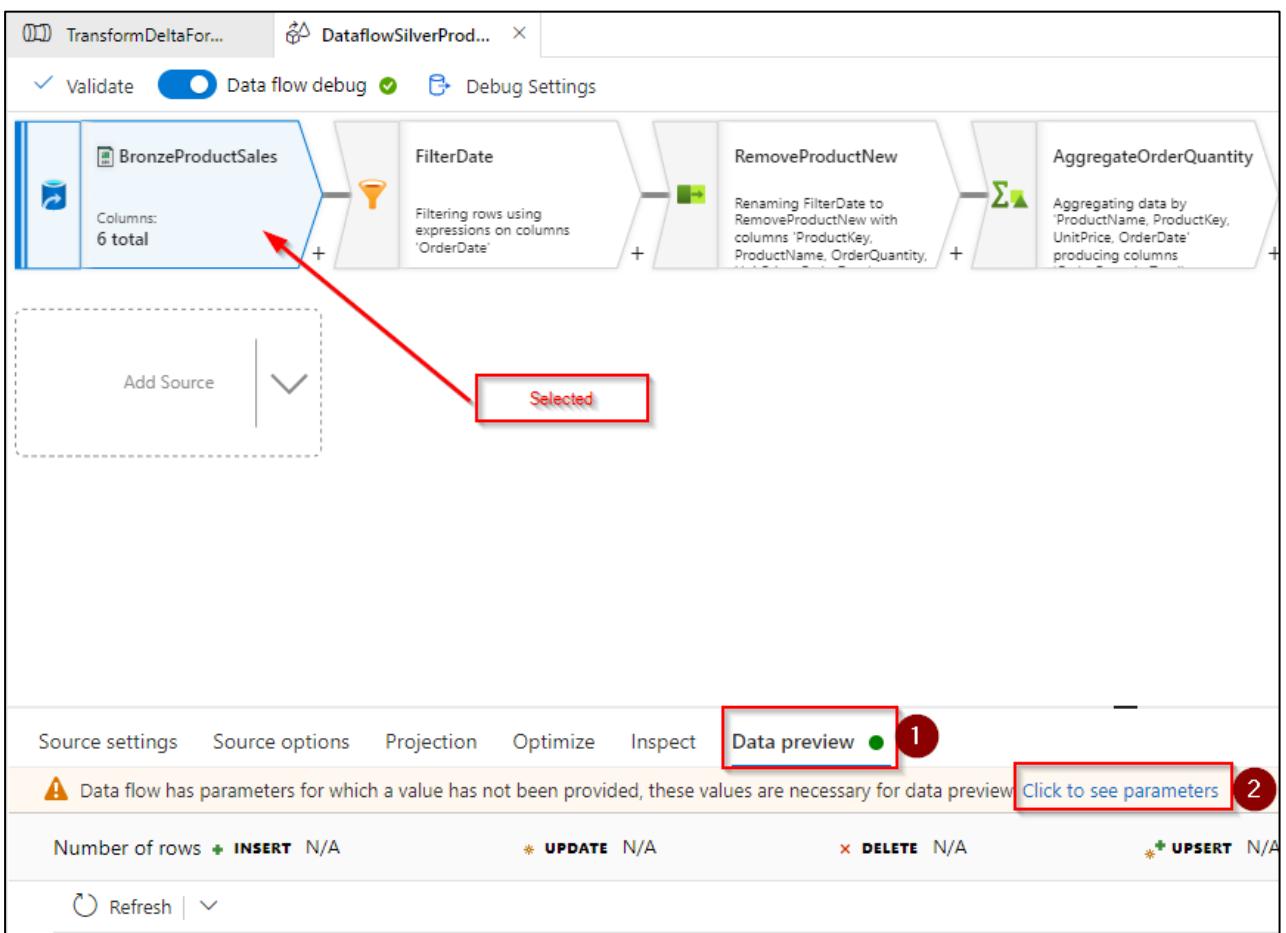
16. Go back to the **DataflowSilverProductSales** tab and select the **BronzeProductSales** tile.



17. Select the **Data Preview** tab.

18. Once selected, there will be a yellow ribbon notifying you of parameters that require values.

Select the field **Click to see parameters**.



A new panel will open to the right. You will now populate these parameters.

19. Select the arrow next to **BronzeProductSales**, add the name of the Sales CSV file **FactProductSales.csv** and select **Save**.

The screenshot shows the 'Debug Settings' panel with the 'Parameters' tab selected. Under 'Dataset parameters', there is a section for 'BronzeProductSales'. A red circle labeled '1' is over the expand arrow for 'BronzeProductSales'. A red box labeled '2' surrounds the 'Value' field, which contains 'FactProductSales.csv'. Below this, a red box surrounds the 'Save' button.

The panel will close, and you will be brought back to the Data preview tab.

20. Under the Data preview tab, select the **Refresh** button.

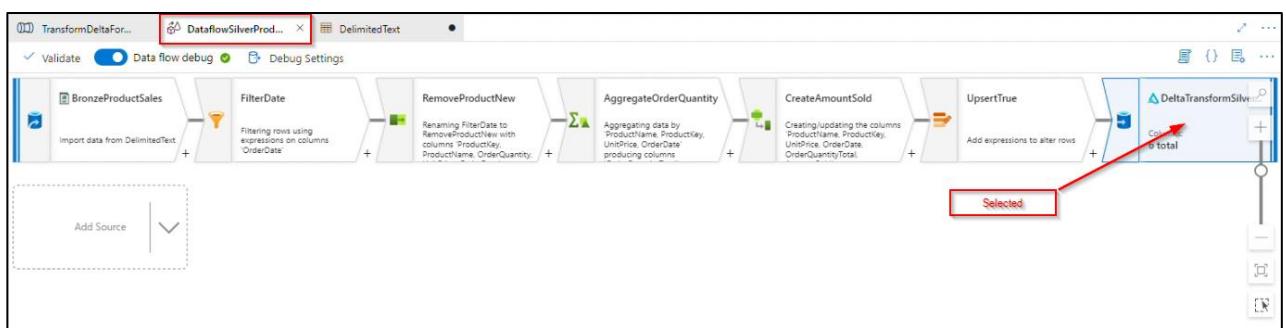
The screenshot shows the 'Data preview' tab. At the top, there are tabs for 'Source settings', 'Source options', 'Projection', 'Optimize', 'Inspect', and 'Data preview'. A red box labeled '1' surrounds the 'Data preview' tab. Below it, there are sections for 'Number of rows' (with 'INSERT' and 'N/A' buttons), 'UPDATE' (with 'N/A' button), and 'DELETE' (with 'N/A' button). A red box labeled '2' surrounds the 'Refresh' button, which has a circular arrow icon.

The data will be fetched, and you should see a preview of your table.

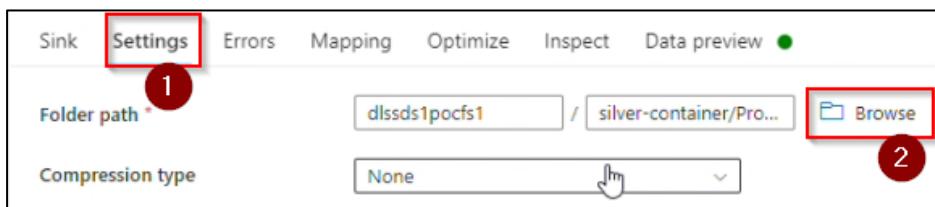
A screenshot of the Data preview tab in a data management interface. The top navigation bar includes tabs for Source settings, Source options, Projection, Optimize, Inspect, and Data preview. Below the tabs, it shows 'Number of rows' (100), 'INSERT 100', 'UPDATE 0', and 'DELETE 0'. The main area displays a table with columns: ProductKey, ProductName, ProductNew, and OrderQuantity. The first seven rows are highlighted with a red border. The data shows multiple entries for ProductKey 360, all with ProductName 'Mountain-200 Black...' and OrderQuantity 1, while ProductNew is NULL.

ProductKey	ProductName	ProductNew	OrderQuantity
360	Mountain-200 Black...	NULL	1
360	Mountain-200 Black...	NULL	1
360	Mountain-200 Black...	NULL	1
360	Mountain-200 Black...	NULL	1
360	Mountain-200 Black...	NULL	1
360	Mountain-200 Black...	NULL	1
360	Mountain-200 Black...	NULL	1
360	Mountain-200 Black...	NULL	1
360	Mountain-200 Black...	NULL	1

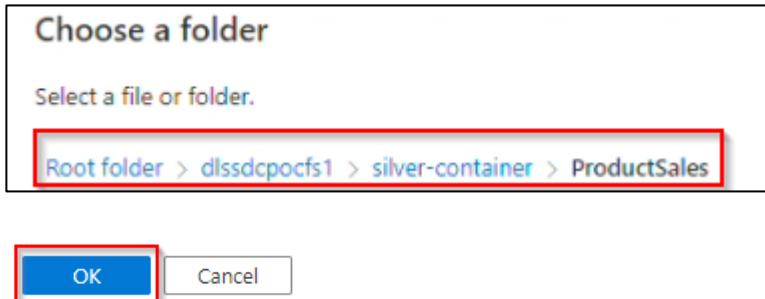
21. Within the **DataflowSilverProductSales** tab, select the **DeltaTransformSilver** tile.



22. Select the **Settings** tab and click the **Browse** button next to **Folder path**.

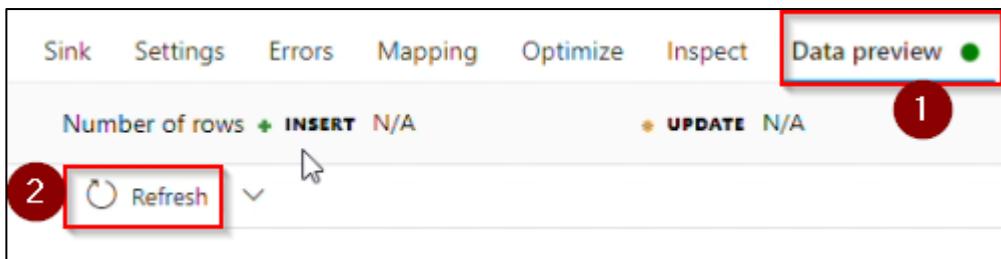


23. Browse to the **silver-container** directory and select the **ProductSales** folder.
24. Once the file path has been set, select **OK**.



The panel will close, and you will be brought back to the Settings tab.

25. Navigate to the **Data preview** tab and select the **Refresh** button.



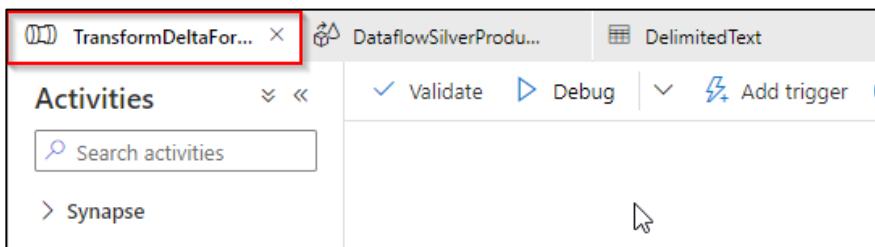
The data will be fetched, and you should see a preview of your table.

ProductName	ProductKey	UnitPrice	OrderDate	OrderQuantityTotal	AmountSold
Mountain-200 Black...	360	2049.0981	2018-01-02	1	2049.0981
Mountain-200 Black...	360	2049.0981	2018-01-08	1	2049.0981
Mountain-200 Black...	360	2049.0981	2018-01-20	2	4098.1963
Mountain-200 Black...	360	2049.0981	2018-01-21	1	2049.0981
Mountain-200 Black...	360	2049.0981	2018-01-24	2	4098.1963
Mountain-200 Black...	360	2049.0981	2018-01-30	1	2049.0981
Mountain-200 Black...	360	2049.0981	2018-01-31	1	2049.0981

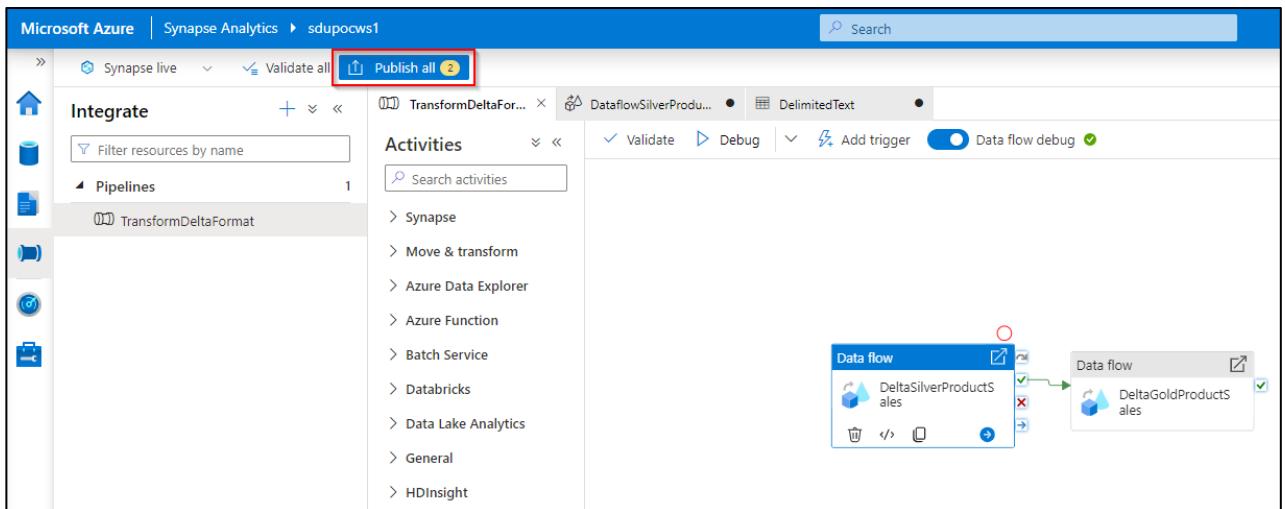
The **DeltaSilverProductSales** dataflow is ready.

We now need to configure the second dataflow called **DeltaGoldProductSales** in a similar manner.

26. Go back to the **TransformDeltaFormat** tab.



27. Select the **Publish all** button to save the work you have done so far.



28. A new panel should open to the right. Select the **Publish** button.

Publish all

You are about to publish all pending changes to the live environment. [Learn more](#)

Pending changes (2)

NAME	CHANGE	EXISTING
DelimitedText	(Edited)	DelimitedText
Dataflows		
DataflowSilverProductSales	(Edited)	DataflowSilverProductSales

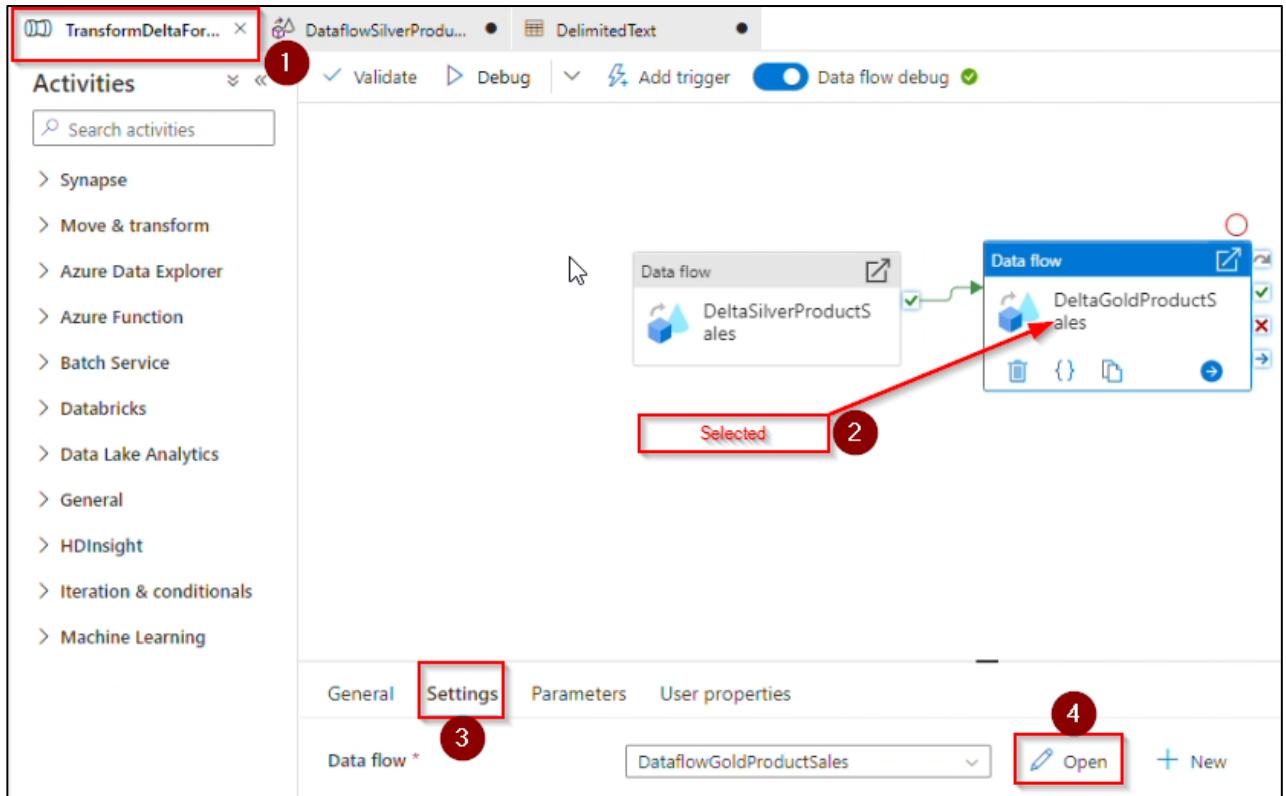
Publish **Cancel**

A banner should appear confirming that the content was published.

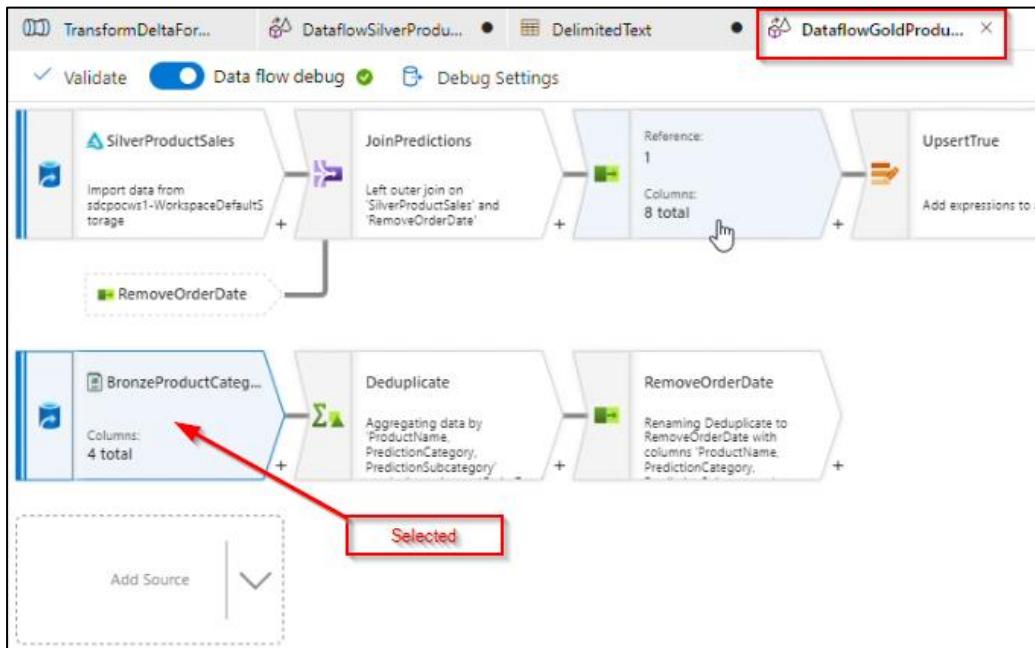


6.1.1.2 DeltaGoldProductSales

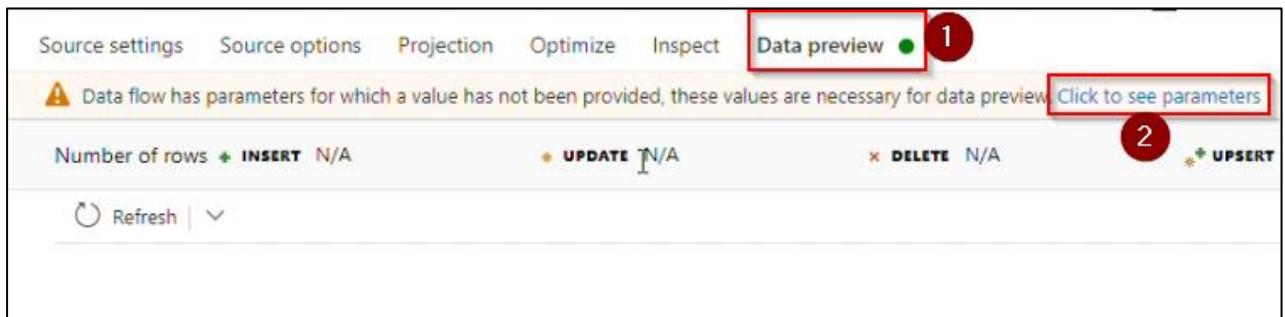
1. Go back to the **TransformDeltaFormat** tab and select the Data flow **DeltaGoldProductSales**.
2. In the **Settings** tab select the **Open** button next to the Data flow **DeltaGoldProductSales**.



- Within the opened data flow, select the **BronzeProductCategoryPredictions** tile.

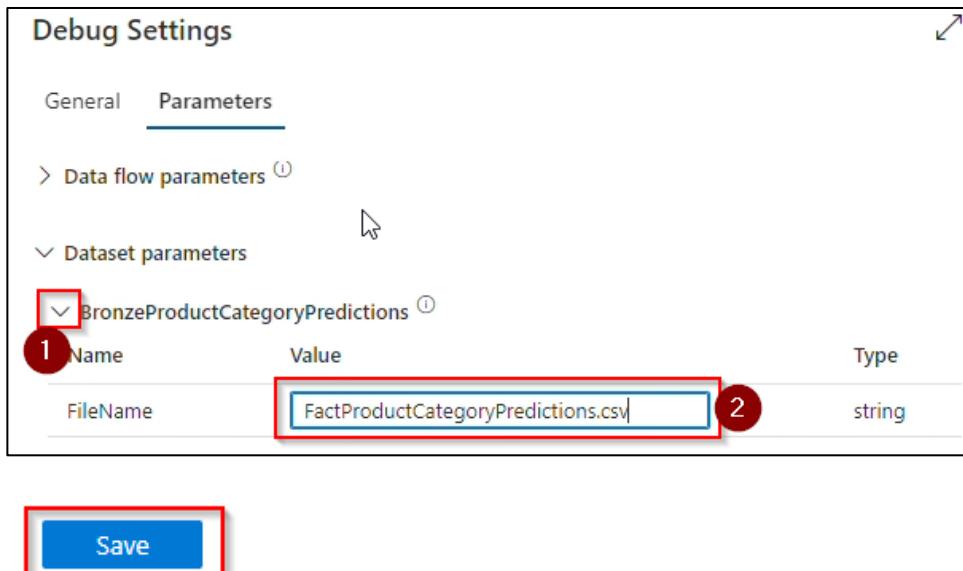


- Select the **Data Preview** tab.
- Once selected, there will be a yellow ribbon notifying you of parameters that require values. Select the field **Click to see parameters**.



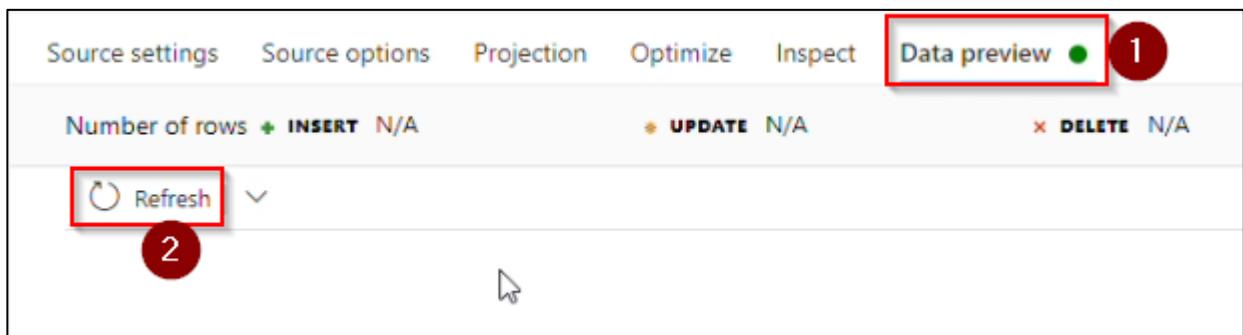
A new panel will open to the right. You will now populate these parameters.

- Select the arrow next to **BronzeProductCategoryPredictions**, add the name of the Prediction CSV file **FactProductCategoryPredictions.csv** and select Save.



The panel will close, and you will be brought back to the Data preview tab.

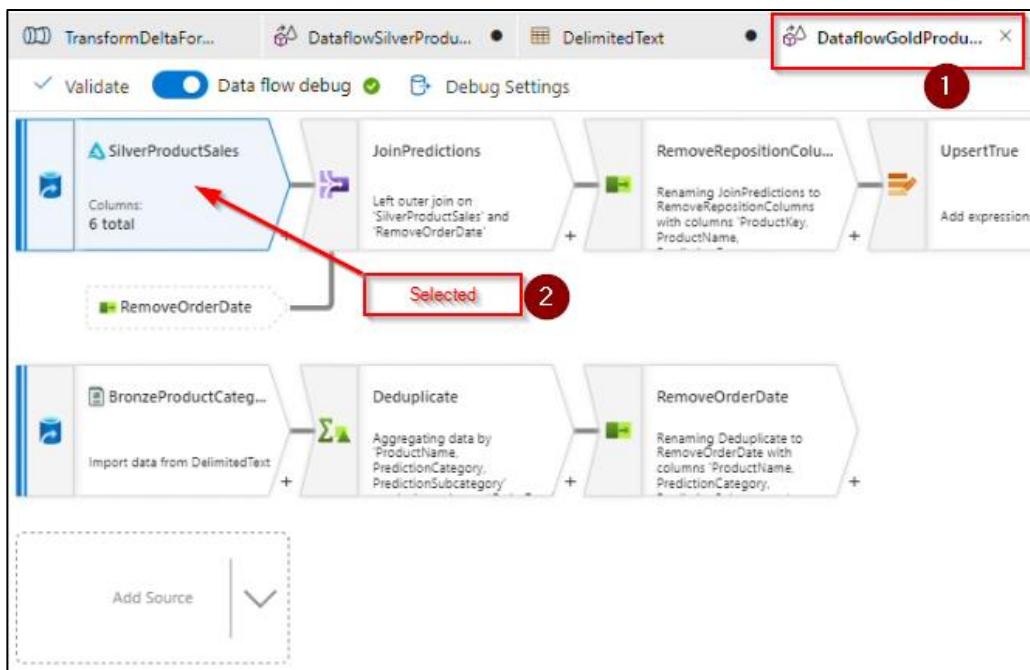
7. Under the Data preview tab, select the Refresh button.



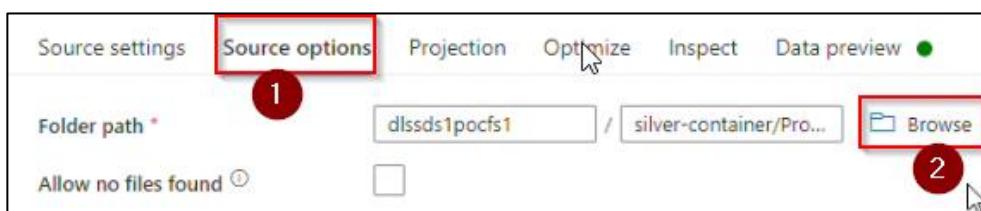
The data will be fetched, and you should see a preview of your table.

Product Name	Prediction Category	Prediction Subcategory	Order Date
Mountain-200 Black...	Bikes	Mountain Bikes	2018-01-02
Mountain-200 Black...	Bikes	Mountain Bikes	2018-01-08
Mountain-200 Black...	Bikes	Mountain Bikes	2018-01-20
Mountain-200 Black...	Bikes	Mountain Bikes	2018-01-20
Mountain-200 Black...	Bikes	Mountain Bikes	2018-01-21
Mountain-200 Black...	Bikes	Mountain Bikes	2018-01-24
Mountain-200 Black...	Bikes	Mountain Bikes	2018-01-24
Mountain-200 Black...	Bikes	Mountain Bikes	2018-01-24
Mountain-200 Black...	Bikes	Mountain Bikes	2018-01-24

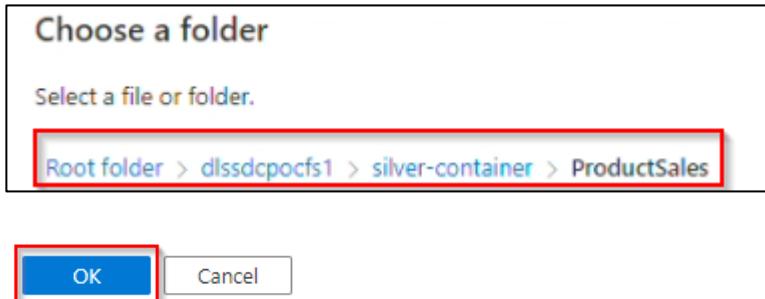
- Within the **DeltaGoldProductSales** tab, select the **SilverProductSales** tile.



- Select the **Source options** tab and click the **Browse** button next to **Folder path**.

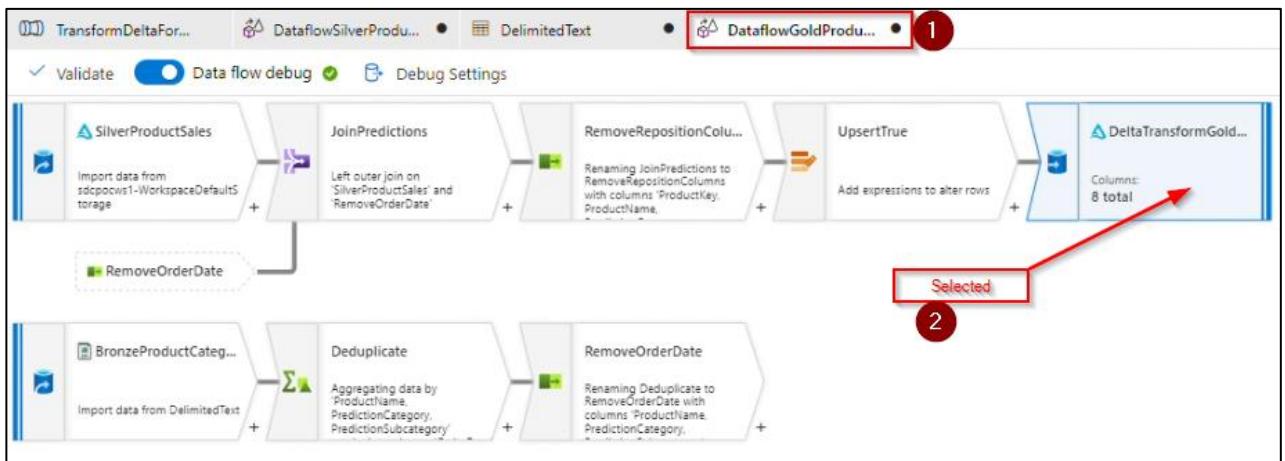


10. Browse to the **silver-container** directory and select the **ProductSales** folder.
11. Once the file path has been set, select **OK**.

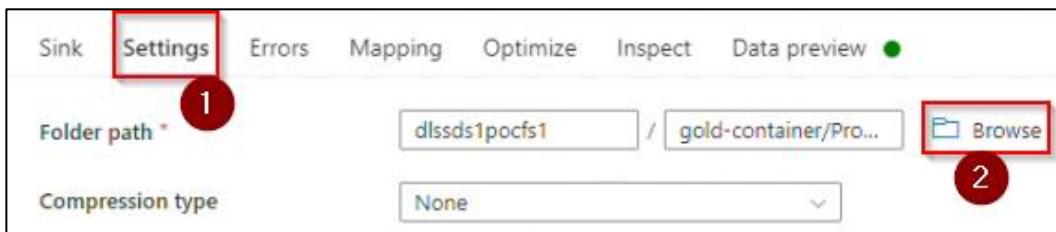


The panel will close, and you will be brought back to the Settings tab.

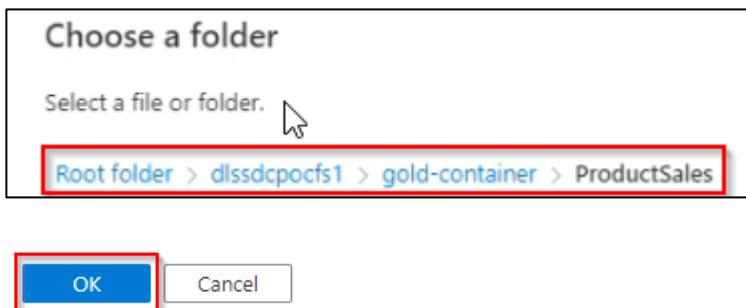
12. Within the **DeltaGoldProductSales** tab, select the **DeltaTransformGoldProductSales** tile.



13. Select the **Settings** tab and click the **Browse** button next to **Folder path**.



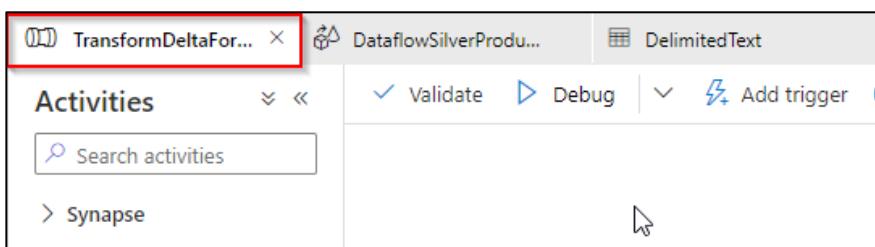
14. Browse to the **gold-container** directory and select the **ProductSales** folder.
15. Once the file path has been set, select **OK**.



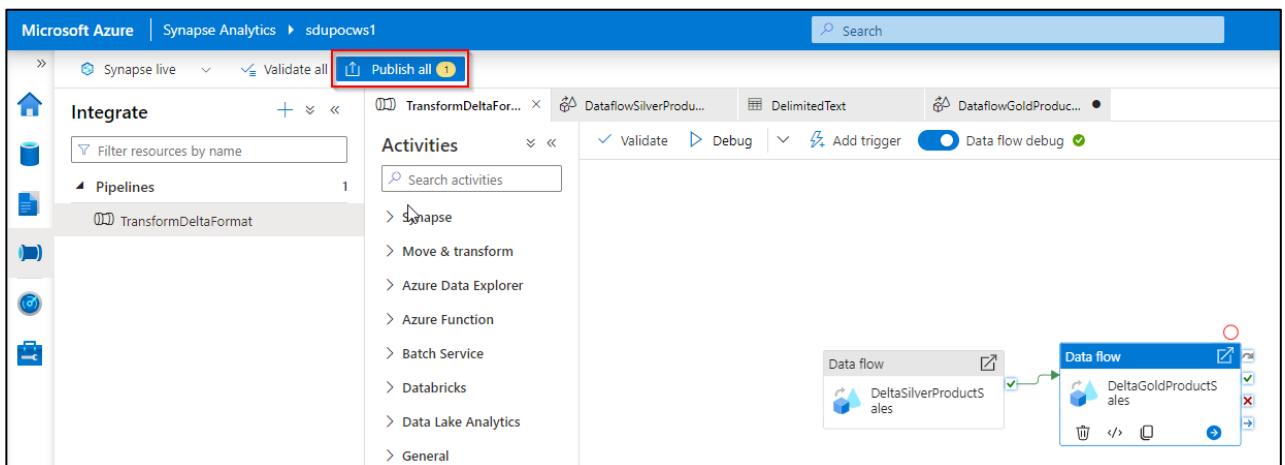
The panel will close, and you will be brought back to the Settings tab.

The **DeltaGoldProductSales** dataflow is now configured and the **TransformDeltaFormat** pipeline is ready to be triggered.

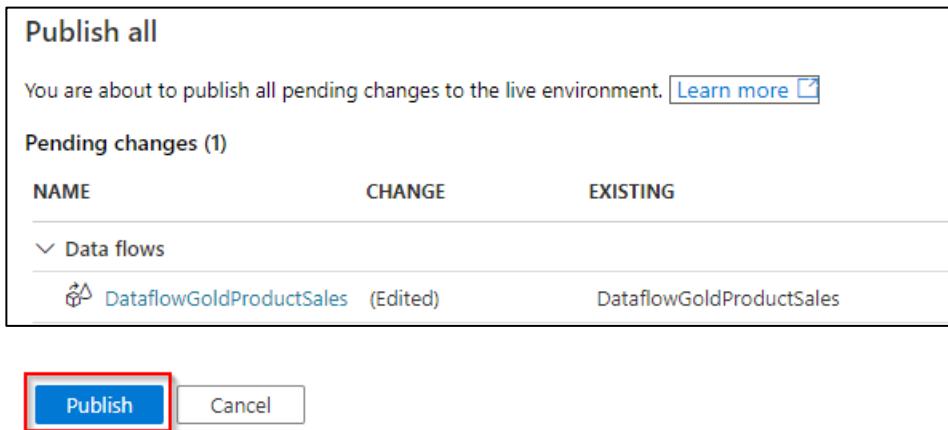
16. Go back to the **TransformDeltaFormat** tab.



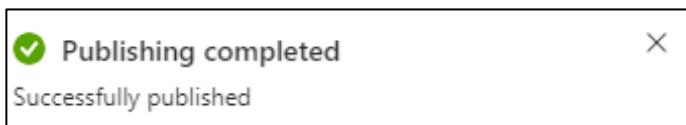
17. Select the **Publish all** button to save the work you have done so far.



18. A new panel should open to the right. Select the **Publish** button.



A banner should appear confirming that the content was published.



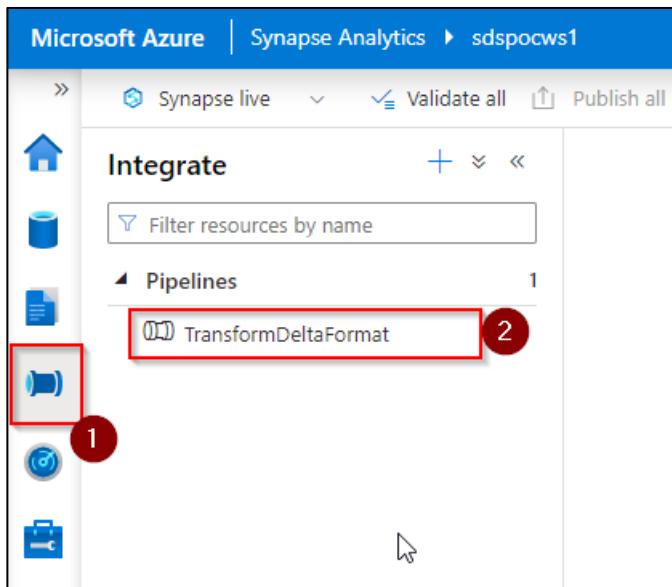
We will now manually trigger the dataflows we configured.

6.2 Manually Trigger Pipelines (Before Data Changes)

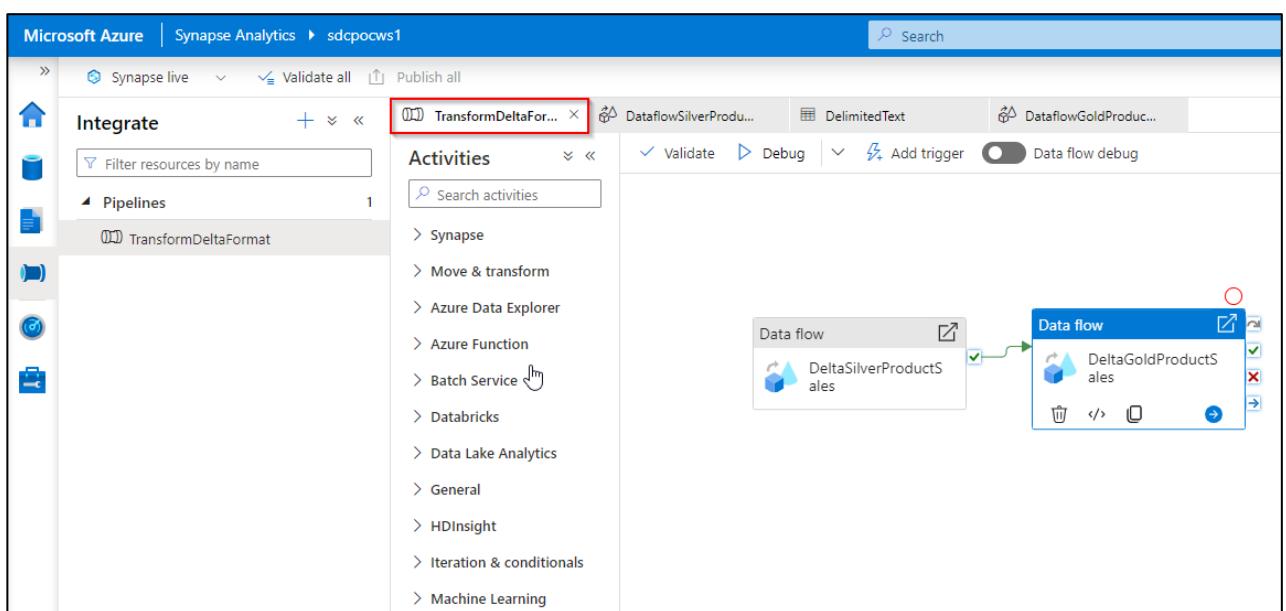
Let's trigger our synapse pipelines to extract, transform and load the raw csv data in our bronze-container as delta formatted data into our silver and gold-containers.

Azure Synapse Pipelines allow for event-based triggers, that start the pipeline when new data is added to the Azure storage. But to keep this example simple, we will be manually triggering the pipelines we have created.

1. Within the Azure Synapse Workspace, select the **Integrate** tab and select the pipeline **TransformDeltaFormat**.

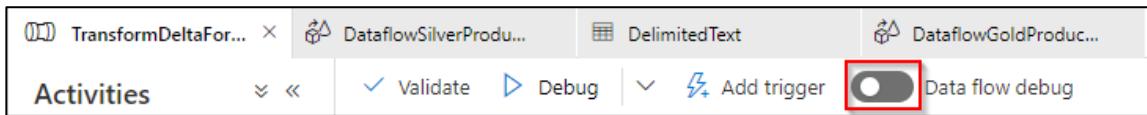


2. Once the pipeline is selected, a new tab with the dataflow activities we configured earlier will appear.

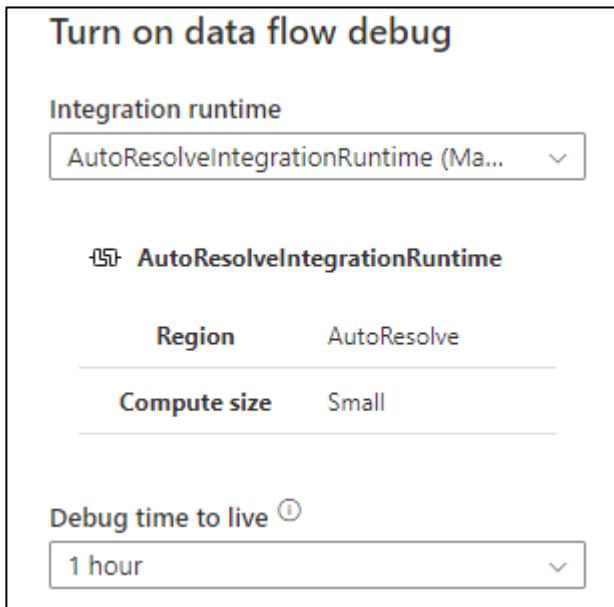


- Select the **Data flow debug** toggle button, to initiate the integration runtime.

NOTE: If the debug is still on [green], you may continue directly on step 5

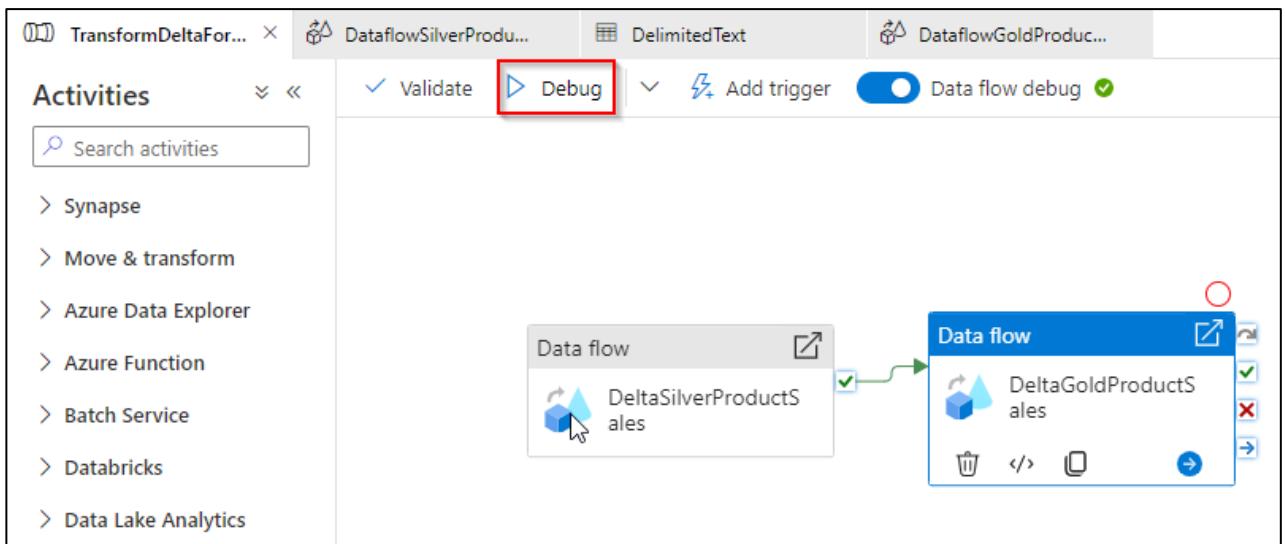


- Once toggled, a new panel will open to the right, with the debug setting. Leave the default settings and select **OK**.

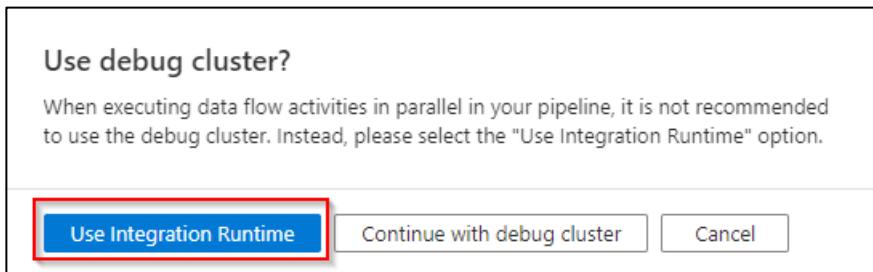


NOTE: The debug initialization might take 3 – 5 minutes. Please be patient!

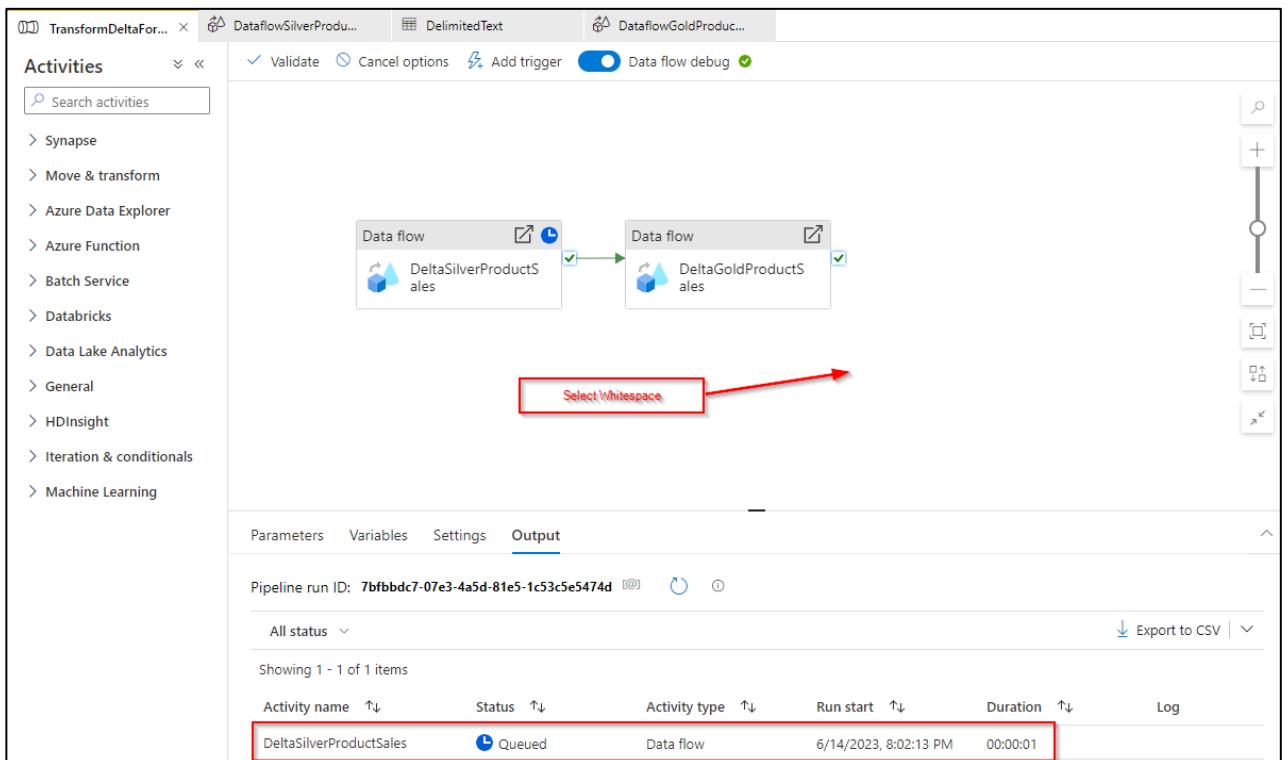
- Select the **Debug** button to run the **TransformDeltaFormat** pipeline we configured and verify that the data flow is working.



6. If prompted by the message **Use debug cluster?** Select **Use Integration Runtime**.



7. Once the Debug is selected, click on the whitespace of the **TransformDeltaFormat** canvas. You should now see the pipelines run.



8. The status of both data flows should change to Succeeded after approximately **3 - 5 minutes**.

Parameters	Variables	Settings	Output			
Pipeline run ID: 564e30ba-4763-48ad-a8c3-6f807dd706ef						
Name	Type	Run start	Duration			
DeltaGoldProductSales	Data flow	5/1/2023, 8:32:23 PM	00:02:21			
DeltaSilverProductSales	Data flow	5/1/2023, 8:29:20 PM	00:03:03			
			<table border="1"> <thead> <tr> <th>Status</th> </tr> </thead> <tbody> <tr> <td>✓ Succeeded</td> </tr> <tr> <td>✓ Succeeded</td> </tr> </tbody> </table>	Status	✓ Succeeded	✓ Succeeded
Status						
✓ Succeeded						
✓ Succeeded						

6.3 Data Changes

In real business scenarios, the data you upload will go through changes on a periodic or even aperiodic basis. To reflect those changes, we will make manual changes to our **productsales** data.

The delta format keeps track of these changes and allows us later to view these changes through audit logs and to time travel i.e., view different version of that same data.

For the sake of simplicity, we will create this change manually by overwriting the **FactProductSales.csv** file we uploaded in [Create Directories and Load Data into the Storage Account](#) with a new version containing additional rows.

1. Go back to the [Azure portal](#) home screen and select the **Resource group** you provisioned.

Azure services

Create a resource Private endpoints Subscriptions All resources Storage accounts Power BI Embedded Azure Active Directory Microsoft Purview... Resource groups More services

Resources

Recent Favorite

Name	Type	Last Viewed
sdspocws1	Synapse workspace	2023-09-01
rg1sdssdatalakehouse	Resource group	2023-09-01
sdspocapp	Resource group	2023-09-01
sdspocstorage	Resource group	2023-09-01
sdspoc	Resource group	2023-09-01
sdspocblobstorage	Storage account	2023-09-01
sdspocappservice	Resource group	2023-09-01
sdspocfunction	Function app	2023-09-01
sdspocwebapp	Web app	2023-09-01
sdspocvms	Virtual machine	2023-09-01
sdspoclogAnalytics	Log Analytics workspace	2023-09-01

See all

2. Once the resource group is open, select the **Storage account**.

	Name ↑↓	Type ↑↓	Location ↑↓
<input type="checkbox"/>	 dlssdpoc	Storage account	Switzerland North
<input type="checkbox"/>	 sdspocws1	Synapse workspace	Switzerland North
<input type="checkbox"/>	 synasp1 (sdspocws1/synasp1)	Apache Spark pool	Switzerland North

- Select the **Container** tab on the left side and select the named container.

The screenshot shows the Azure Storage account interface for 'dlssdspoc'. On the left sidebar, the 'Containers' tab is highlighted with a red box and the number '1'. In the main pane, a container named 'dlssdspocfs1' is selected, indicated by a red box and the number '2'. Other tabs like 'Overview', 'Activity log', and 'Data storage' are visible but not selected.

- Within the named container, select the **bronze-container**.

The screenshot shows the blob storage interface for the 'bronze-container'. It displays a list of blobs with their names: 'bronze-container', 'gold-container', 'silver-container', and 'synapse'. The file 'bronze-container' is highlighted with a red box.

- Upload the **FactProductSales.csv** file which can be found in the **Changes** subfolder within the **Data** Folder.

NOTE: The Support Files can be found in the Git repository in this [link](#). Download these files onto your local computer if you haven't already.

The screenshot shows a file explorer window with the path 'Data > Changes'. The 'Changes' folder is highlighted with a red box. Inside, there is a single file named 'FactProductSales.csv', which is also highlighted with a red box. The file is a Microsoft Excel file from May 2, 2023, at 20:52, with a size of 2'549 KB.

The changed **FactProductSales.csv** file contains additional rows.

	A	B	C	D	E	F
1	ProductKey	ProductName	ProductNew	OrderQuantity	UnitPrice	OrderDate
57574	486	All-Purpose Bike Stand		1	159	02.05.2023
57575	486	All-Purpose Bike Stand		1	159	02.05.2023
57576	537	HL Mountain Tire		1	35	02.05.2023
57577	225	AWC Logo Cap		1	8.99	03.05.2023
57578	485	Fender Set - Mountain		1	21.98	03.05.2023
57579	485	Fender Set - Mountain		1	21.98	04.05.2023
57580	360	Mountain-200 Black, 42		1	2049.0982	04.05.2023
57581	360	Mountain-200 Black, 42		1	2049.0982	04.05.2023

6. Within the bronze-container, select the **Upload** button.

Upload Add Directory Refresh Rename

Authentication method: Access key (Switch to Azure AD User Acc)

Location: dlssdspocfs1 / bronze-container

Search blobs by prefix (case-sensitive)

Name

FactProductCategoryPredictions.csv

FactProductSales.csv

7. A new panel will open to the right. Drag and Drop the **FactProductSales.csv** file from the Changes folder, toggle the **Overwrite if files already exist** and select **Upload**.

Upload blob

1 file(s) selected: FactProductSales.csv 1

Drag and drop files here or Browse for files

2

Overwrite if files already exist

3

Upload Give feedback

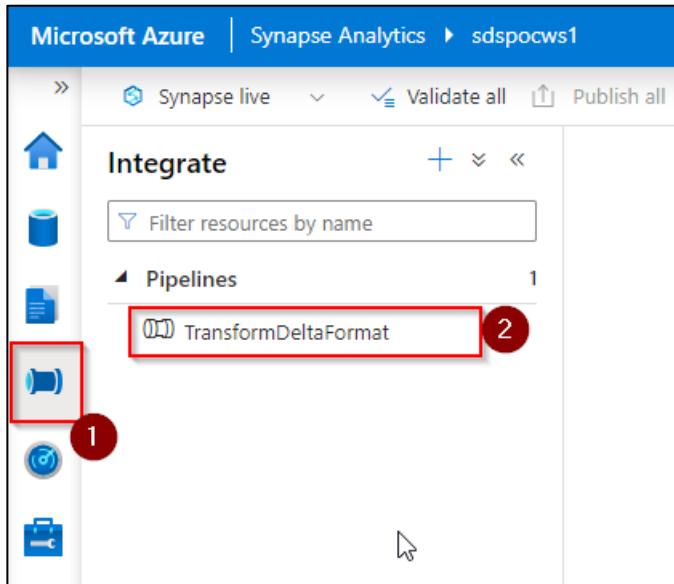
You should get a successfully uploaded banner:



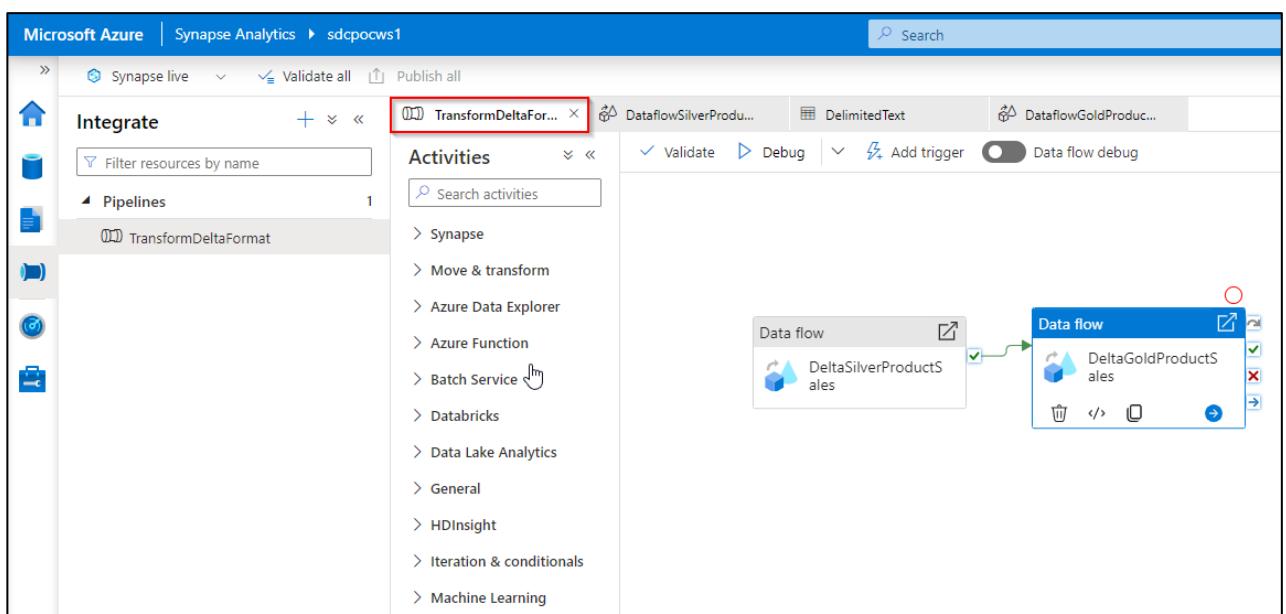
6.4 Manually Trigger Pipelines (After Data Changes)

We uploaded our changed data into the storage in chapter [Data Changes](#). Let's trigger our synapse pipelines to extract, transform and load these changes into our silver and gold-containers.

1. Go back to your Azure Synapse Workspace and select the **Integrate** tab and select the pipeline **TransformDeltaFormat**.

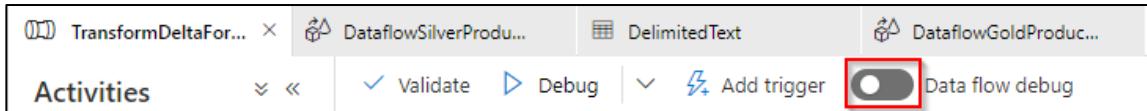


2. Once the pipeline is selected, a new tab with the dataflow activities we created earlier will appear.

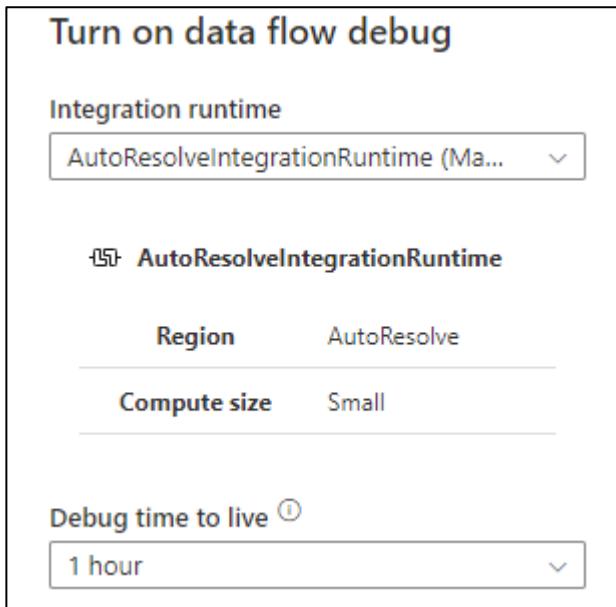


- Select the **Data flow debug** toggle button, to initiate the integration runtime.

NOTE: If the debug is still on [green], you may continue directly on step 5.

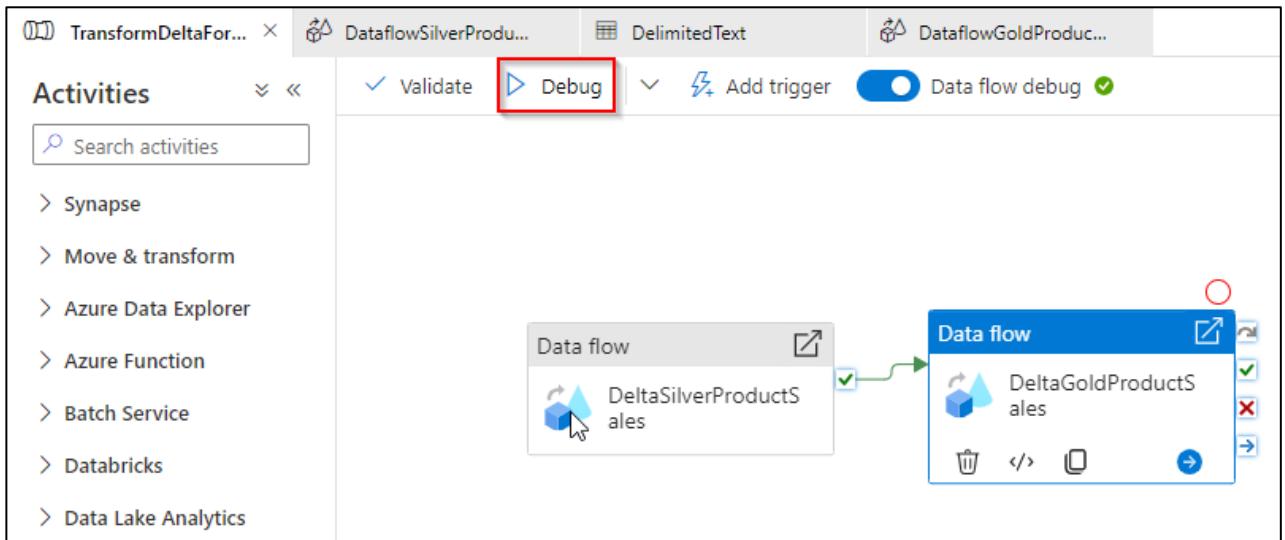


- Once toggled, a new panel will open to the right, with the debug setting. Leave the default settings and select **OK**.

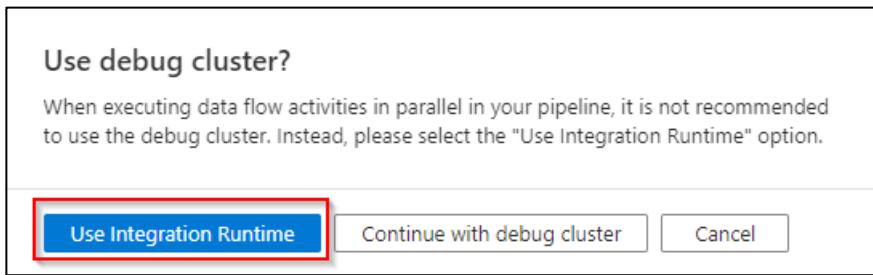


NOTE: The debug initialization might take 3 – 5 minutes. Please be patient!

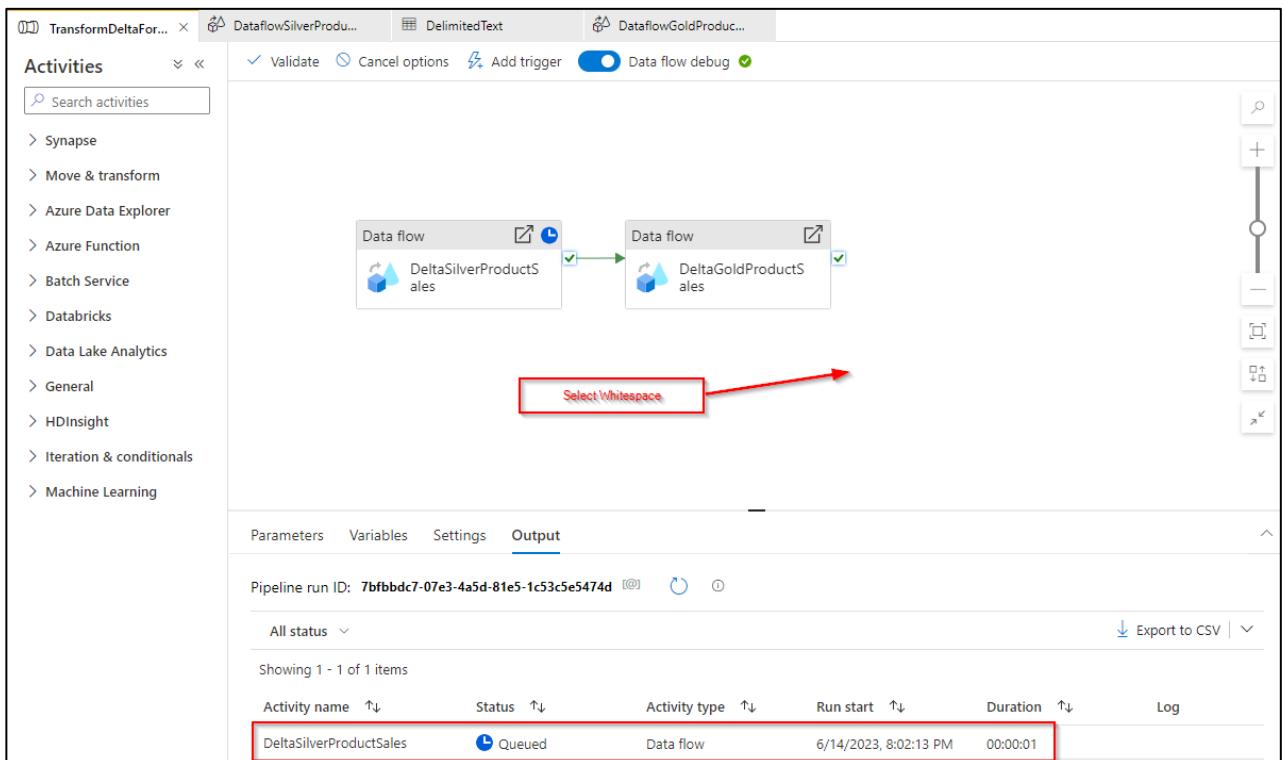
- Select the **Debug** button to run the pipeline we created and verify that the data flow is working.



6. If prompted by the message **Use debug cluster?** Select **Use Integration Runtime**.



7. Once the Debug is selected, click on the whitespace of the **TransformDeltaFormat** canvas. You should now see the pipelines run.



8. The status of both data flows should change to Succeeded after approximately **3 - 7 minutes**.

Parameters	Variables	Settings	Output			
Pipeline run ID: 564e30ba-4763-48ad-a8c3-6f807dd706ef						
Name	Type	Run start	Duration			
DeltaGoldProductSales	Data flow	5/1/2023, 8:32:23 PM	00:02:21			
DeltaSilverProductSales	Data flow	5/1/2023, 8:29:20 PM	00:03:03			
			<table border="1"> <thead> <tr> <th>Status</th> </tr> </thead> <tbody> <tr> <td>✓ Succeeded</td> </tr> <tr> <td>✓ Succeeded</td> </tr> </tbody> </table>	Status	✓ Succeeded	✓ Succeeded
Status						
✓ Succeeded						
✓ Succeeded						

7 Create Azure Synapse SQL Database (OPTIONAL)

This chapter is **OPTIONAL**. We advise you to do this chapter after the workshop and continue the workshop on chapter [Create Azure Synapse Lake Database](#).

The serverless SQL databases are like that of a traditional Azure SQL Database/Dedicated Pool (formerly SQL DW). The primary difference is that the serverless SQL database queries over data stored in a data lake, rather than stored on disk on the SQL Server machine. This is also the primary feature that makes the data lakehouse architecture possible.

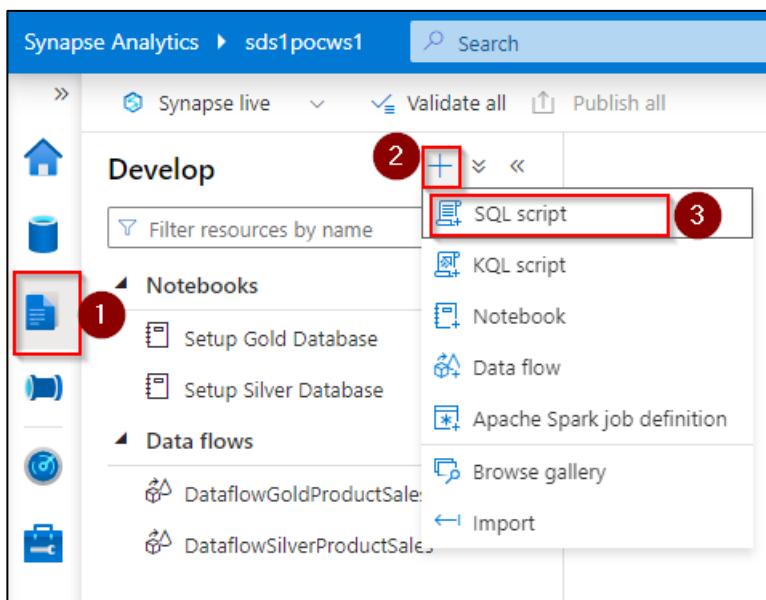
7.1 Create a SQL Script

Before we create the delta tables based off the delta formatted data in the silver-container and the gold-container, we can also create a SQL Database to query the raw CSV data in the bronze-container using a view with the serverless SQL pool.

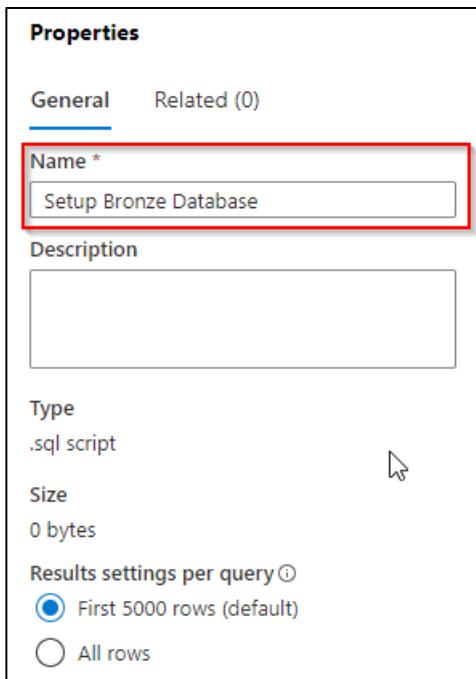
NOTE: Creating views or external tables from the raw data is usually not necessary, since they can be queried directly from the Azure storage using Serverless SQL pools as described in chapter [Query Delta Formatted Data with Serverless SQL Pool \(OPTIONAL\)](#).

7.1.1 Bronze Database

1. Select the **Develop** tab to the left, select the + button and create a new SQL script.



2. Rename the newly created SQL script to **Setup Bronze Database**.



You will start by creating a bronze database using SQL.

3. Within the SQL Script **Setup Bronze Database** field, add the following SQL code to create the database:

```
USE MASTER  
GO  
  
CREATE DATABASE bronze COLLATE Latin1_General_100_BIN2_UTF8
```

4. Select the Run button to run the script.



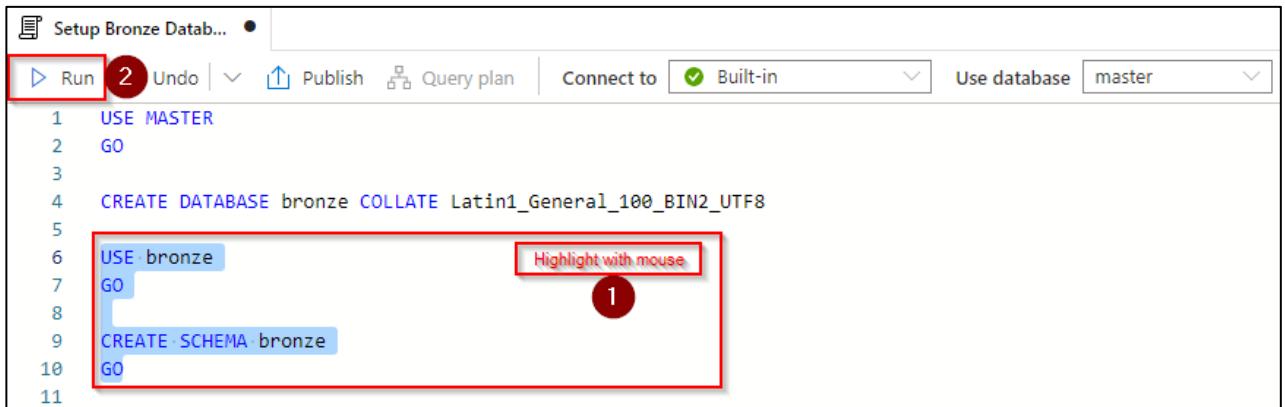
You will get a notification on the bottom that the **Query executed successfully**.

00:00:02 Query executed successfully.

5. Below the SQL code you added in step 3, add the following SQL code to create a schema for your bronze database:

```
USE bronze  
GO  
  
CREATE SCHEMA bronze  
GO
```

6. Highlight this new portion of SQL code and select the run statement.



You will get a notification on the bottom that the **Query executed successfully**.

✓ 00:00:02 Query executed successfully.

7. Below the SQL code you added in step 5 add the following SQL code to create a view of your raw data:

NOTE: The BULK command below needs to be adjusted. See Step 8 to find the correct file path.

```
CREATE OR ALTER VIEW bronze.productsales AS  
SELECT *  
FROM  
OPENROWSET(  
    BULK 'see step 8 to find correct path,  
    FORMAT = 'CSV',  
    PARSER_VERSION = '2.0',  
    FIELDTERMINATOR = ';',  
    FIRSTROW = 1,  
    HEADER_ROW = TRUE  
) AS [result]
```

Before running the SQL code, we need to make sure the **BULK** command looks at the correct file path.

8. Select the **Data** tab to the left, select **Linked** and open your **primary data storage** i.e., the storage provisioned in chapter [Create Resource Group and Resources](#).
9. Select the **named container** of this storage.
10. Select the **bronze-container**.

The screenshot shows the Microsoft Azure Synapse Analytics Data blade. On the left, there's a sidebar with icons for Home, Data, Datasets, and Integration datasets. The 'Data' icon is highlighted with a red box and has a red circle with the number 1 above it. The main area shows a 'Workspace' section with a 'Linked' tab selected (highlighted with a red box and red circle 2). Below it is a 'Filter resources by name' input field. Under 'Azure Data Lake Storage Gen2', there are two entries: 'sds1pocws1 (Primary - dlssds1pocfs1)' (highlighted with red box 3 and red circle 4) and 'dlssds1pocfs1 (Primary)' (highlighted with red box 5 and red circle 5). To the right, a 'bronze-container' is selected in a list (highlighted with red box 6 and red circle 6). At the top right, there are buttons for 'Publish all' (with a yellow notification badge), 'New SQL script', 'New data flow', and 'New integration dataset'.

11. Within the **bronze-container**, right click the **FactproductSales.csv** file, select **New SQL Script** and select **Select TOP 100 rows**.

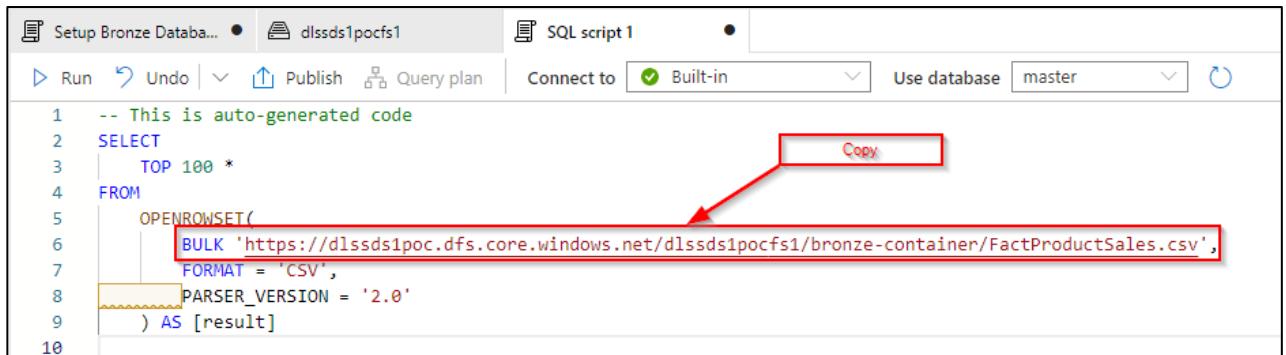
The screenshot shows the context menu for a file named 'FactProductSales.csv' within the 'bronze-container'. The menu items are:

- 1. The container name 'bronze-container' is highlighted with a red box and red circle 1.
- 2. The file 'FactProductSales.csv' is highlighted with a red box and red circle 2. A red box labeled 'Right Click' points to this item.
- 3. 'New SQL script' is highlighted with a red box and red circle 3.
- 4. 'Select TOP 100 rows' is highlighted with a red box and red circle 4, and a mouse cursor is hovering over it.

Other menu options include 'New notebook', 'New data flow', 'New integration dataset', 'Manage access...', 'Rename...', 'Download', 'Delete', and 'Properties...'.

This will create a SQL statement for you, with the required OPENROWSET command to call the file.

12. Copy the **BULK** statement.

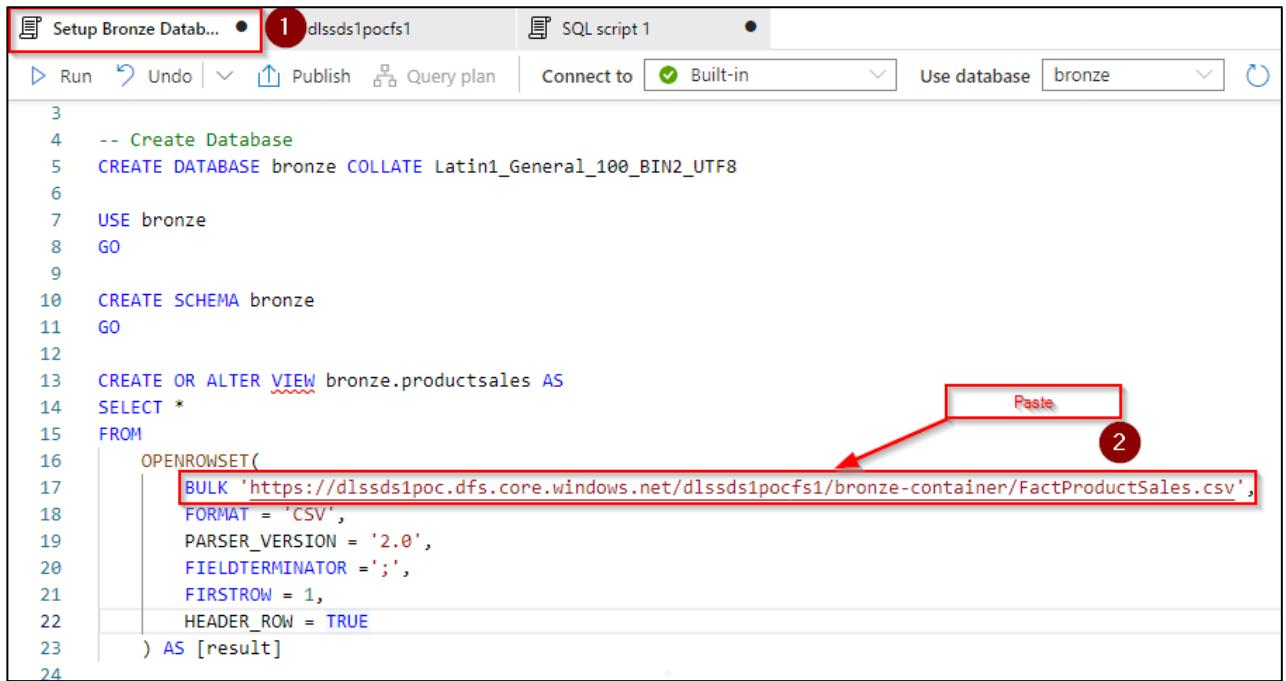


The screenshot shows a SQL Server Management Studio window with two tabs: "Setup Bronze Database..." and "SQL script 1". The "SQL script 1" tab is active, displaying a SQL script. A red box highlights the BULK statement, and a red arrow points from it to a red-bordered box labeled "Copy". The script content is as follows:

```
1 -- This is auto-generated code
2 SELECT
3     TOP 100 *
4 FROM
5     OPENROWSET(
6         BULK 'https://dlssds1poc.dfs.core.windows.net/dlssds1pocfs1/bronze-container/FactProductSales.csv',
7         FORMAT = 'CSV',
8         PARSE_VERSION = '2.0'
9     ) AS [result]
10
```

13. Go back to the **Setup Bronze Database** SQL Script by selecting the **Setup Bronze Database** tab.

14. Paste the copied BULK statement into the SQL code from step 7.



The screenshot shows a SQL Server Management Studio window with two tabs: "Setup Bronze Database..." (circled in red) and "SQL script 1". The "Setup Bronze Database..." tab is active, displaying a SQL script. A red box highlights the BULK statement, and a red arrow points from it to a red-bordered box labeled "Paste". A red circle with the number "2" is positioned to the right of the arrow. The script content is as follows:

```
3
4 -- Create Database
5 CREATE DATABASE bronze COLLATE Latin1_General_100_BIN2_UTF8
6
7 USE bronze
8 GO
9
10 CREATE SCHEMA bronze
11 GO
12
13 CREATE OR ALTER VIEW bronze.productsales AS
14 SELECT *
15 FROM
16     OPENROWSET(
17         BULK 'https://dlssds1poc.dfs.core.windows.net/dlssds1pocfs1/bronze-container/FactProductSales.csv',
18         FORMAT = 'CSV',
19         PARSE_VERSION = '2.0',
20         FIELDTERMINATOR = ';',
21         FIRSTROW = 1,
22         HEADER_ROW = TRUE
23     ) AS [result]
24
```

15. Make sure that the **Use database** is set to **bronze** (this should happen automatically when you run the script in step 6)
16. Highlight this new portion of SQL code you added in step 7 and select **Run**.

```

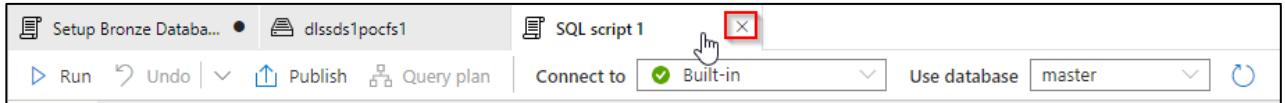
1 USE MASTER
2 GO
3
4 CREATE DATABASE bronze COLLATE Latin1_General_100_BIN2_UTF8
5
6 USE bronze
7 GO
8
9 CREATE SCHEMA bronze
10 GO
11
12 CREATE OR ALTER VIEW bronze.productsales AS
13 SELECT *
14 FROM OPENROWSET(
15     BULK 'https://dlssdspoc.dfs.core.windows.net/dlssdpsocfs1/bronze-container/FactProductSales.csv',
16     FORMAT = 'CSV',
17     PARSER_VERSION = '2.0',
18     FIELDTERMINATOR = ';',
19     FIRSTROW = 1,
20     HEADER_ROW = TRUE
21 ) AS [result]
22
23

```

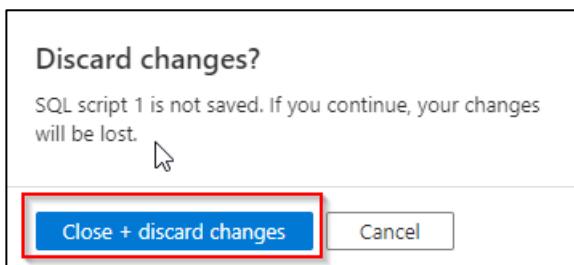
You will get a notification on the bottom that the **Query executed successfully**.

00:00:02 Query executed successfully.

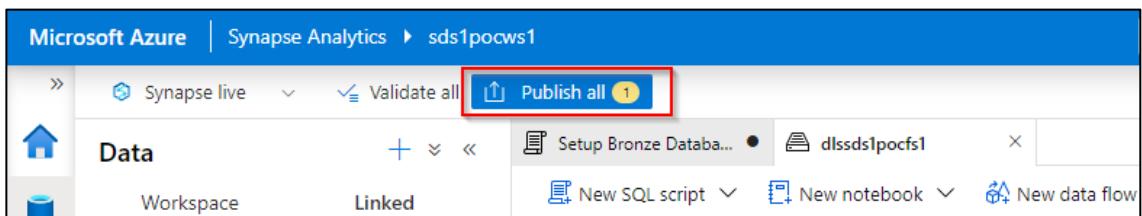
17. You may now close the tab **SQL script 1** which you created in step 11 when querying the TOP 100 rows.



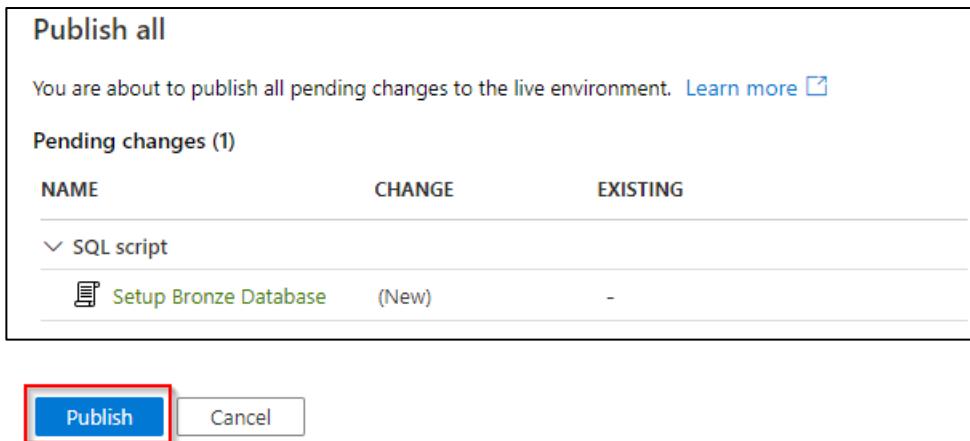
18. Select **Close + discard changes** to confirm the deletion.



19. Select the **Publish all** button to save the work you have done so far.



20. A new panel should open to the right. Select the **Publish** button.



A banner should appear confirming that the content was published.

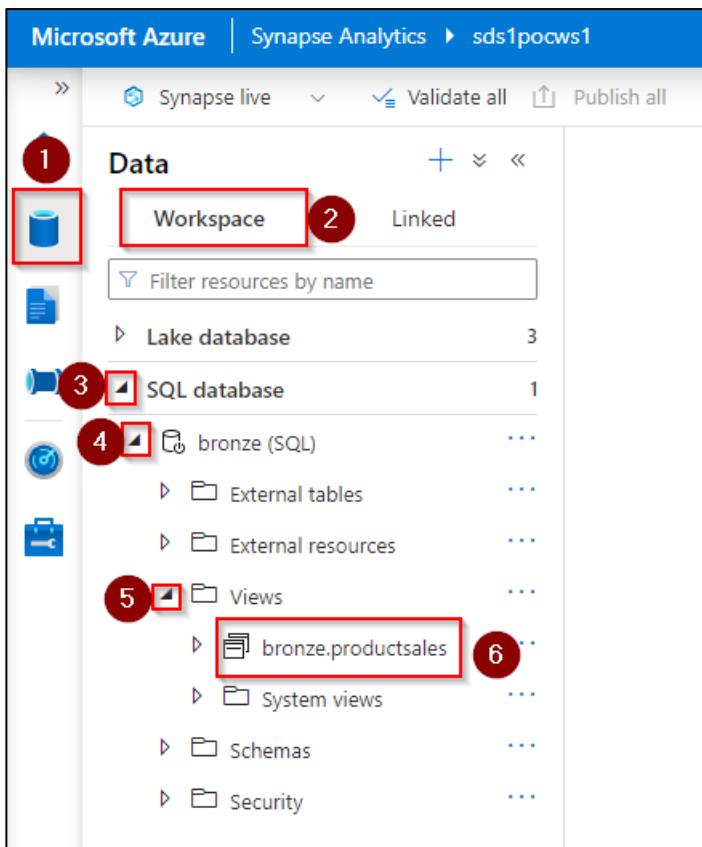


7.1.1.1 Query View with SQL Serverless Pool

We will now query the view we created from the raw file **FactproductSales.csv**.

NOTE: You may need to refresh your browser to view the newly created SQL database in the Data tab. Make sure you publish everything and save your work before refreshing the browser. See Step 19.

21. Select the **Data** tab to the left, select **Workspace** and browse to the **Views** folder. Within the Views folder you will see the **bronze.productsales** view we created earlier.



22. Right click the **bronze.productsales** and select New SQL script, followed by Select TOP 100 rows. This will create a new tab with a SQL statement to query the view.

23. Select the **Run** button to run the SQL statement.

The screenshot shows a SQL editor window titled 'SQL script 1'. At the top, there are buttons for 'Run' (highlighted with a red box), 'Undo', 'Publish', 'Query plan', 'Connect to' (set to 'Built-in'), 'Use database' (set to 'bronze'), and a refresh icon. The main area contains a SQL script:

```
1 SELECT TOP (100) [ProductKey]
2 ,[ProductName]
3 ,[ProductNew]
4 ,[OrderQuantity]
5 ,[UnitPrice]
6 ,[OrderDate]
7 | FROM [bronze].[productsales]
```

You will now see the data related to the **FactproductSales.csv** file.

The screenshot shows the Azure Data Studio interface. At the top, there's a toolbar with various icons like Run, Undo, Publish, and Query plan. Below the toolbar, a dropdown menu says 'Connect to' with 'Built-in' selected, and another dropdown says 'Use database' with 'bronze' selected. The main area contains a SQL script window with the following code:

```
1  SELECT TOP (100) [ProductKey]
2  ,[ProductName]
3  ,[ProductNew]
4  ,[OrderQuantity]
5  ,[UnitPrice]
6  ,[OrderDate]
7  | FROM [bronze].[productsales]
```

Below the script, there are tabs for 'Results' and 'Messages'. The 'Results' tab is active, showing a table view of the query results. The table has columns: ProductKey, ProductName, ProductNew, OrderQuantity, UnitPrice, and OrderDate. There are six rows of data, all with ProductKey 360, ProductName 'Mountain-200 Black, 42', and ProductNew '(NULL)'. The OrderQuantity is 1, UnitPrice is 2049.0982, and OrderDate is 02.01.2018. The entire table is highlighted with a red border.

ProductKey	ProductName	ProductNew	OrderQuantity	UnitPrice	OrderDate
360	Mountain-200 Black, 42	(NULL)	1	2049.0982	02.01.2018
360	Mountain-200 Black, 42	(NULL)	1	2049.0982	08.01.2018
360	Mountain-200 Black, 42	(NULL)	1	2049.0982	20.01.2018
360	Mountain-200 Black, 42	(NULL)	1	2049.0982	20.01.2018
360	Mountain-200 Black, 42	(NULL)	1	2049.0982	21.01.2018
360	Mountain-200 Black, 42	(NULL)	1	2049.0982	24.01.2018

You have successfully queried a view with data directly from a data lake using a serverless SQL pool.

8 Create Azure Synapse Lake Database

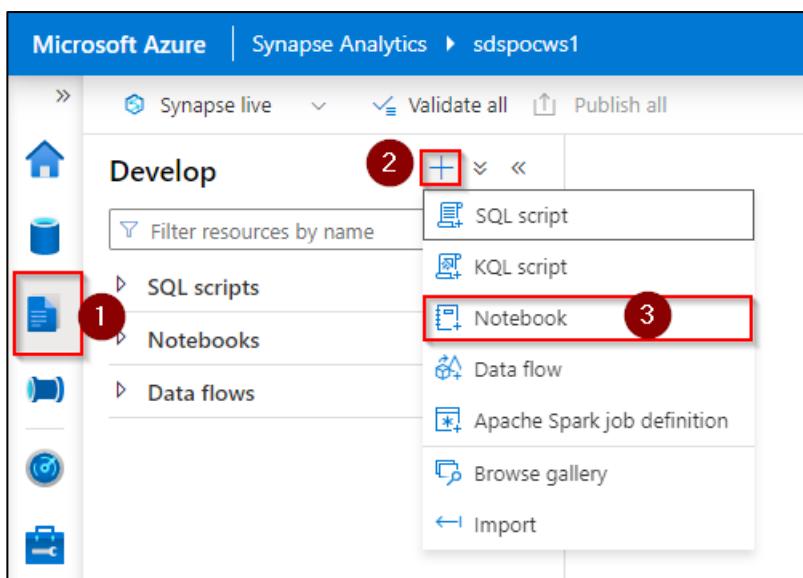
The Lake Database is a new way of defining data structures on data that is hosted on Azure Data Lake Storage and processed with Azure Synapse. This Database type is synchronized between the Spark and the SQL Serverless engines in Azure Synapse and allows interoperability between the different compute engines (Spark and SQL). It is within this database type that you can create and query delta formatted data using delta tables.

8.1 Create a Spark Notebook

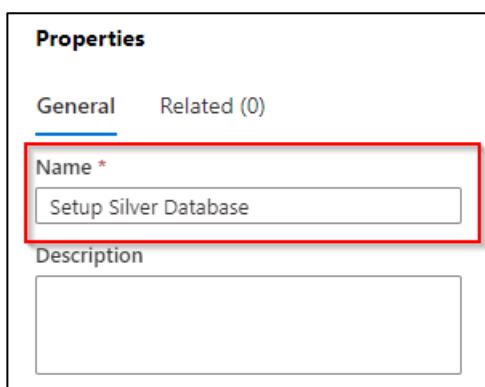
We can now create delta tables based off the delta formatted data in the silver-container and the gold-container using the Apache Spark Pool.

8.1.1 Silver Database

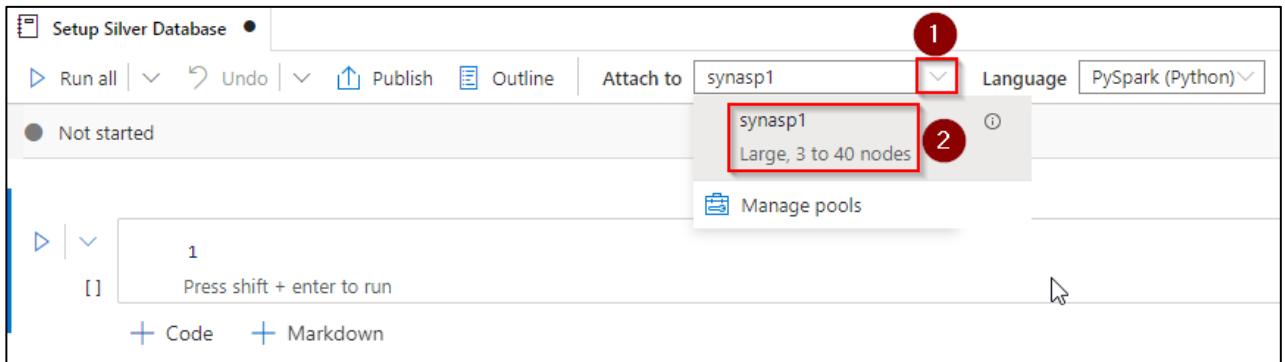
24. Select the **Develop** tab to the left, select the + button and create a new Notebook.



25. Rename the newly created Notebook to **Setup Silver Database**.



26. To begin using this new Notebook, attach an Apache Spark Pool by selecting the **Attach to** dropdown. Select the provisioned spark pool (Large).



The Spark Notebooks allow you to run code using Spark flavors of SQL, Python, R and Scala programming languages. In this example, we will focus on SQL and Python.

8.1.1.1 Lake Databases

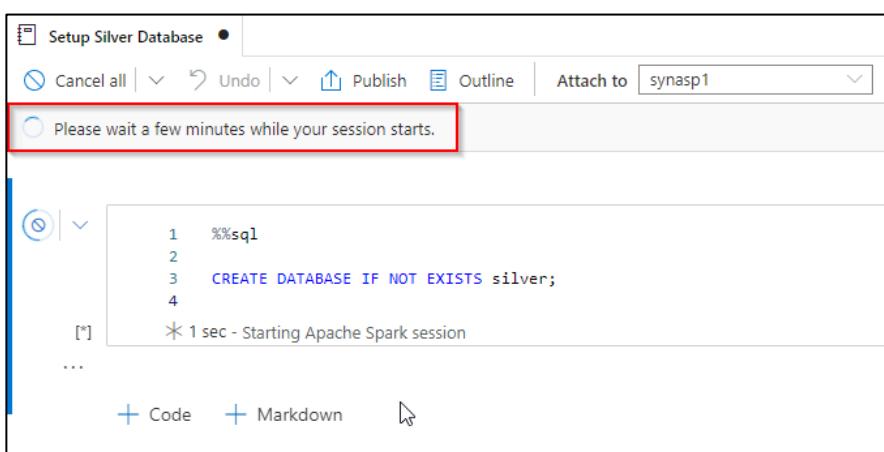
You will start by creating a silver database using Spark SQL.

27. Within the Notebook **Setup Silver Database** code field, add the following SQL code to create the databases:

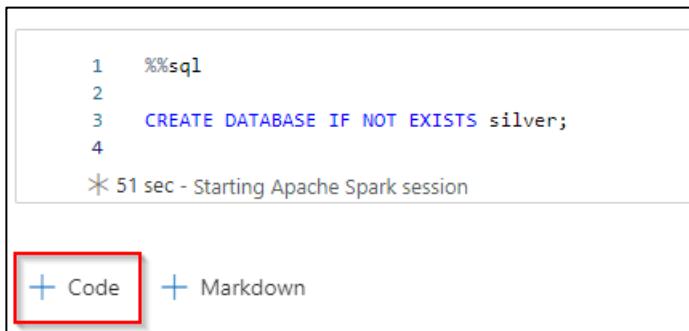
```
%%sql
CREATE DATABASE IF NOT EXISTS silver;
```

28. Select the Run button to start the Spark Pool and run the script.

NOTE: Starting the Apache Spark Pool may take 3 – 5 minutes. Please be patient!



29. Once the code has run, select the **+ Code** button, to add a new code field to the Notebook.



8.1.1.2 Azure Blob File System

To create a delta table, you will need the Azure Blob File System (ABFS) driver. This has the URI structure:

abfss://**container_name**@**storage_account_name**.dfs.core.windows.net/

NOTE: Pay attention to the different colors in the URI above. First the Container name (blue), then the storage name (orange). This is a source of errors.

30. Go back to the [Azure portal](#) home screen and select the **Resource group** you provisioned.

31. Once the resource group is open, find the **Storage account** and **copy** the storage name (in the example below, this would be **dlssdspoc**).

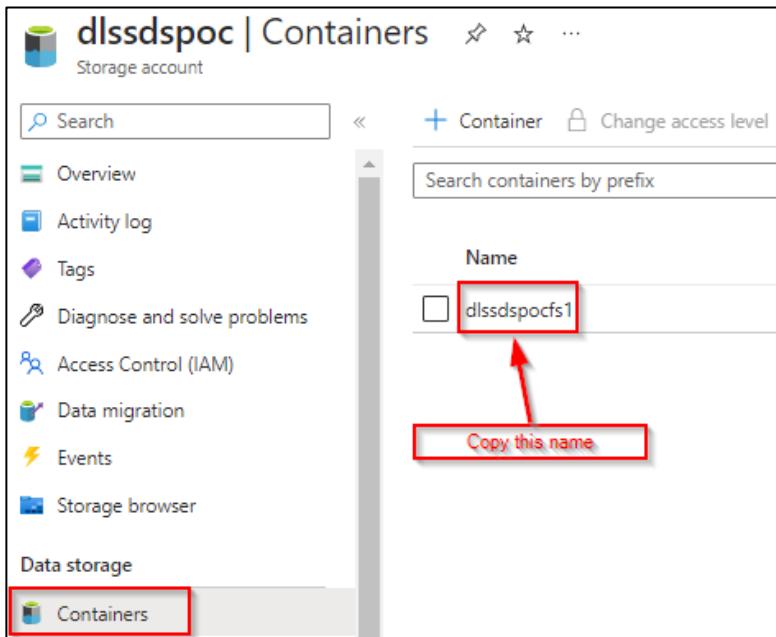
This is the **storage_account_name** from the URI.

Name ↑	Type ↑	Location ↑
<input type="checkbox"/>  dlssdspoc	<input type="checkbox"/> Storage account	Switzerland North
<input type="checkbox"/>  sdspocws1	Synapse workspace	Switzerland North
<input type="checkbox"/>  synasp1 (sdspocws1/synasp1)	Apache Spark pool	Switzerland North

32. Once the storage account name is copied, **select** the **Storage account** to open it.

33. Once the Storage account is open, select the **Container** tab and **copy** the **container name** (in the example below, this would be **dlssdspocfs1**).

This is the **container_name** from the URI.



The screenshot shows the Azure Storage account interface for 'dlssdspoc'. On the left, there's a sidebar with links like Overview, Activity log, Tags, etc., and a 'Containers' link which is highlighted with a red box. The main area has a 'Containers' tab selected, showing a list of containers. One container, 'dlssdspocfs1', is highlighted with a red box, and a red arrow points to a 'Copy this name' button below it. There's also a search bar and a 'Container' creation button.

You can now populate the URI; it should look like the one below:

- abfss://**container_name@storage_account_name**.dfs.core.windows.net/

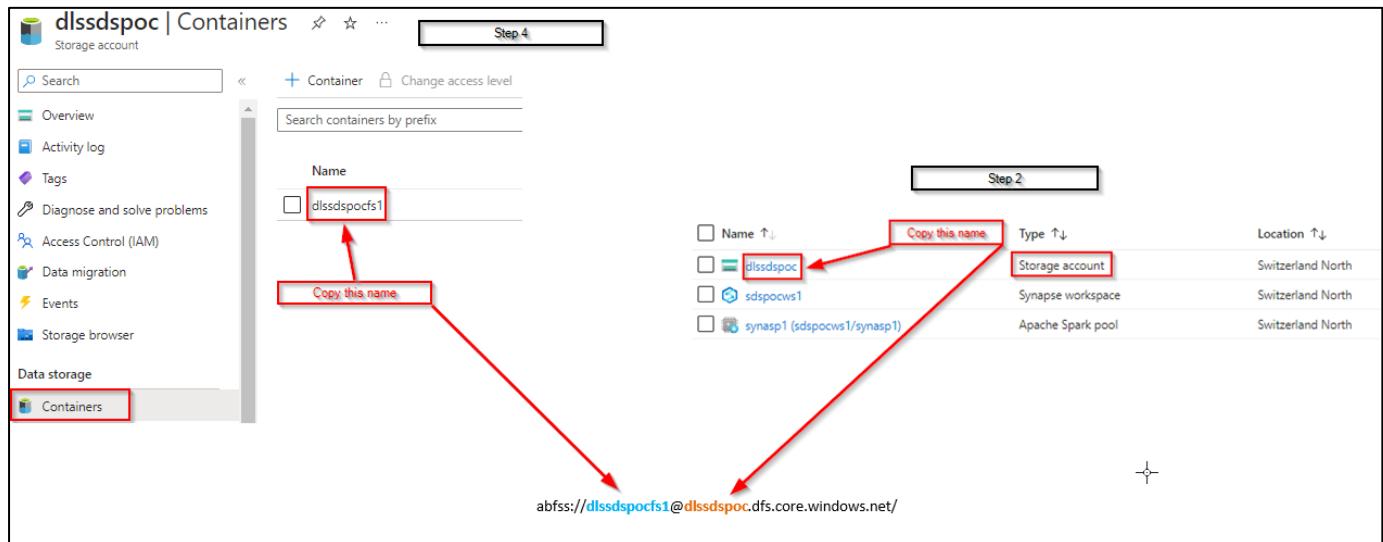
becomes in this example:

- abfss://**dlssdspocfs1@dlssdspoc**.dfs.core.windows.net/

This URI will be used in the **LOCATION** variable in the next code script.

NOTE: Do NOT copy the URI abfss://dlssdspocfs1@dlssdspoc.dfs.core.windows.net/ you see above. This will NOT work for you. You must input the names of the container and storage account you provisioned as described in chapter [Azure Blob File System](#) above.

Overview:



8.1.1.3 Lake Database Delta Tables

1. Go back to your Azure Synapse Workspace.
2. Within the Notebook **Setup Silver Database** code field, add the SQL code below to create the delta table.

```
%%sql

CREATE TABLE IF NOT EXISTS silver.productsales
USING DELTA
LOCATION 'abfss://dlssdspocfs1@dlssdspoc.dfs.core.windows.net/silver-container/ProductSales/';
```

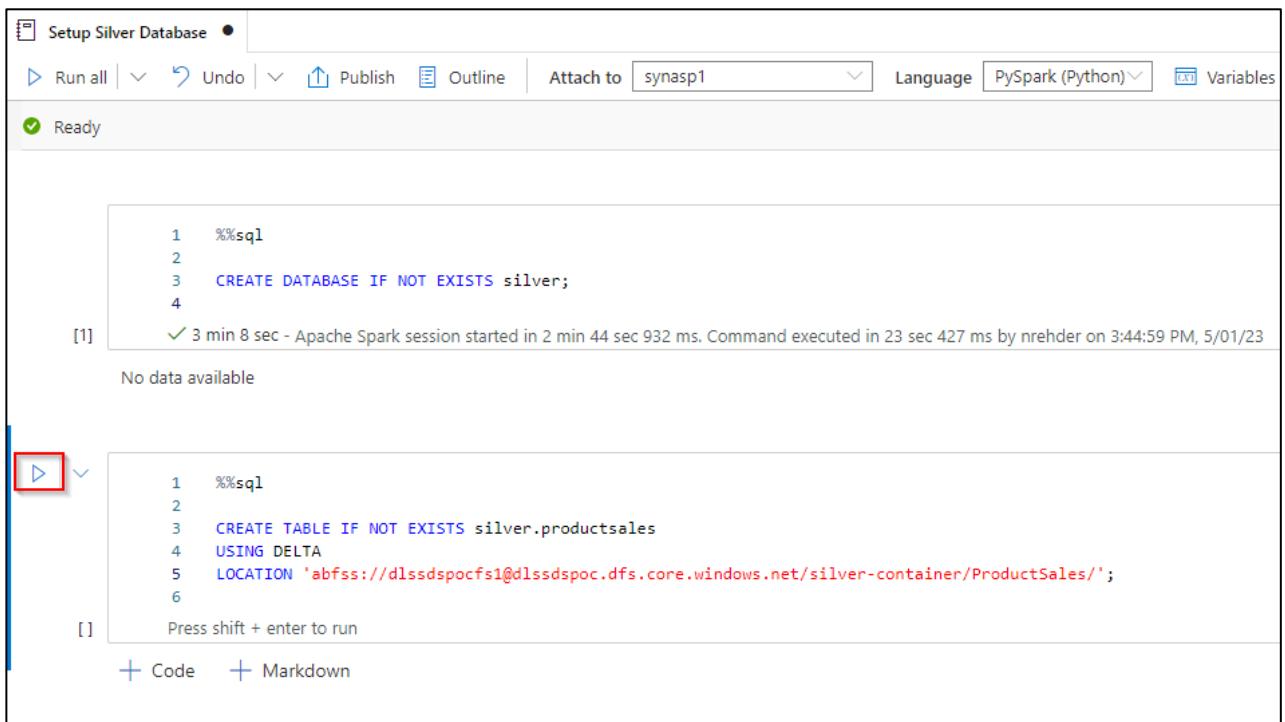
NOTE: The %%sql command explicitly tells the notebook to use the SQL language to run the code.

3. Change the **LOCATION** parameter within the SQL code to your URI as described in chapter [Azure Blob File System](#).

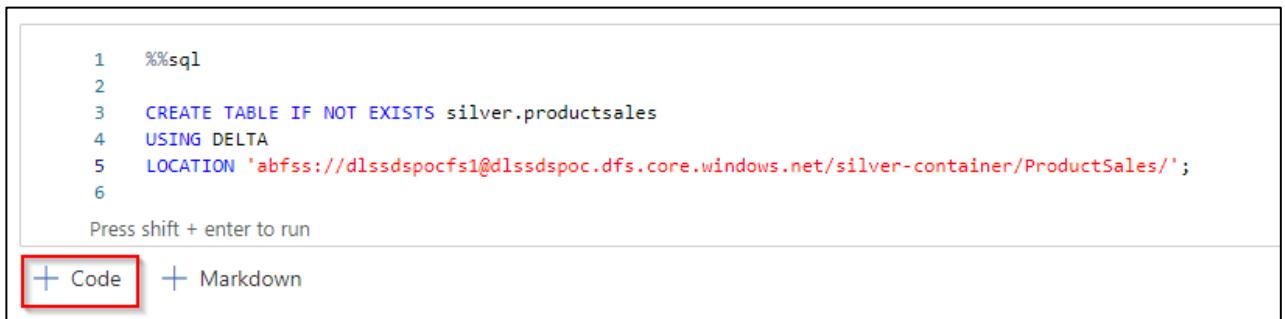
```
%%sql

CREATE TABLE IF NOT EXISTS silver.productsales
USING DELTA
LOCATION 'abfss://dlssdspocfs1@dlssdspoc.dfs.core.windows.net/silver-
container/ProductSales/';
```

4. Select the Run button to run the script.



5. Once the code has run, select the **+ Code** button, to add a new code field to the Notebook.

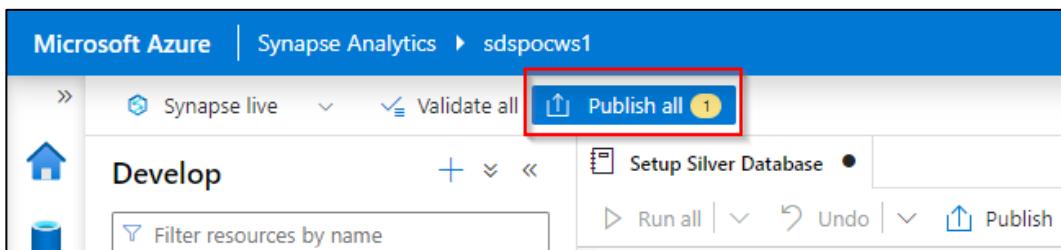


```
1 %%sql
2
3 CREATE TABLE IF NOT EXISTS silver.productsales
4 USING DELTA
5 LOCATION 'abfss://dlssdspocfs1@dlssdspoc.dfs.core.windows.net/silver-container/ProductSales/' ;
6
```

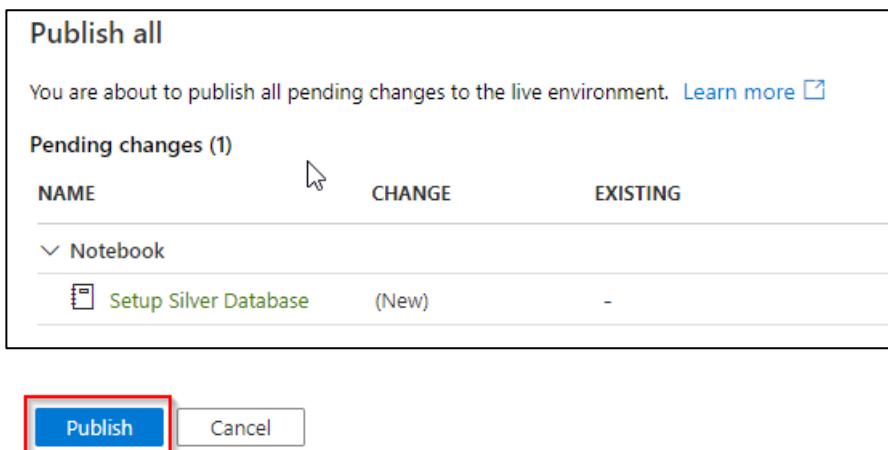
Press shift + enter to run

+ Code + Markdown

6. Select **Publish** to save the new notebook.



7. A new panel should open to the right. Select the **Publish** button.



A banner will appear confirming that the content was published.



8.1.1.4 Audit Logs

Delta Lake format stores data in Parquet files and information regarding DML operations in the `_delta_log` metadata folder. With this metadata available you can view audit logs and time travel between different versions of a Delta table. Time travelling can be achieved by table version or by timestamp.

Now that the data has been changed, we want to be able to view different versions of this data i.e., time travel.

8. Go back to the **Setup Silver Database** tab. Once selected, you will see your Notebook.

```
%%sql
CREATE DATABASE IF NOT EXISTS silver;
[1] ✓ - Apache Spark session started in 2 min 44 sec 932 ms. Command executed in 23 sec 427 ms on 3:44:59 PM, 5/01/23
...
No data available

+ Code + Markdown

%%sql
CREATE TABLE IF NOT EXISTS silver.productsales
USING DELTA
LOCATION 'abfss://dlssdspocfs1@dlssdspoc.dfs.core.windows.net/silver-container/ProductSales/';
[8] ✓ - Command executed in 2 sec 944 ms on 3:54:13 PM, 5/01/23
No data available
```

9. Within the Notebook **Setup Silver Database** code field, add the following SQL code to view the audit logs of your delta table. This gives you the change history of your data.

In the audit log you will see the operations we executed after manually changing the data in chapter [Data Changes](#).

```
%%sql
DESCRIBE HISTORY 'abfss://dlssdspocfs1@dlssdspoc.dfs.core.windows.net/silver-container/ProductSales/';
```

10. Within the SQL code above change the **abfss URI** to your URI as described in chapter [Azure Blob File System](#).

```
%%sql
DESCRIBE HISTORY 'abfss://dlssdspocfs1@dlssdspoc.dfs.core.windows.net/silver-
container/ProductSales/'
```

Change to your URI

11. Select the Run button to run the script.

```
1 %%sql
2
3 CREATE DATABASE IF NOT EXISTS silver;
4
```

[1] ✓ - Apache Spark session started in 2 min 44 sec 932 ms. Command executed in 23 sec 427 ms on 3:44:59 PM, 5/01/23
No data available

```
1 %%sql
2
3 CREATE TABLE IF NOT EXISTS silver.productsales
4 USING DELTA
5 LOCATION 'abfss://dlssdspocfs1@dlssdspoc.dfs.core.windows.net/silver-container/ProductSales/';
6
```

[8] ✓ - Command executed in 2 sec 944 ms on 3:54:13 PM, 5/01/23
No data available

```
1 %%sql
2
3 DESCRIBE HISTORY 'abfss://dlssdspocfs1@dlssdspoc.dfs.core.windows.net/silver-container/ProductSales/';
4
```

[] Press shift + enter to run
+ Code + Markdown

You should see the Changes, specifically the operations that took place and the versions available as an output:

version	timestamp	userId	userName	operation	operationParameters	job
1	2023-05-01T18:39:29Z	null	null	MERGE	▶ {"predicate": "(source.'ProductKey' null"}	
0	2023-05-01T18:32:12Z	null	null	WRITE	▶ {"mode": "Append", "partitionBy": "l" null"}	

12. Once the code has run, select the **+ Code** button, to add a new code field to the Notebook.

version	timestamp	userId	userName	operation	operationParameters	job
1	2023-04-28T06:19:46Z	null	null	MERGE	▶ {"predicate": "(source.'ProductKey' null"}	
0	2023-04-27T18:57:36Z	null	null	WRITE	▶ {"mode": "Append", "partitionBy": "l" null"}	

8.1.1.5 Time Traveling

13. Add the following SQL code to view a previous version of your product sales data i.e. **version 0** or the initial state before the manual changes. For the sake of simplicity, we will filter the data.

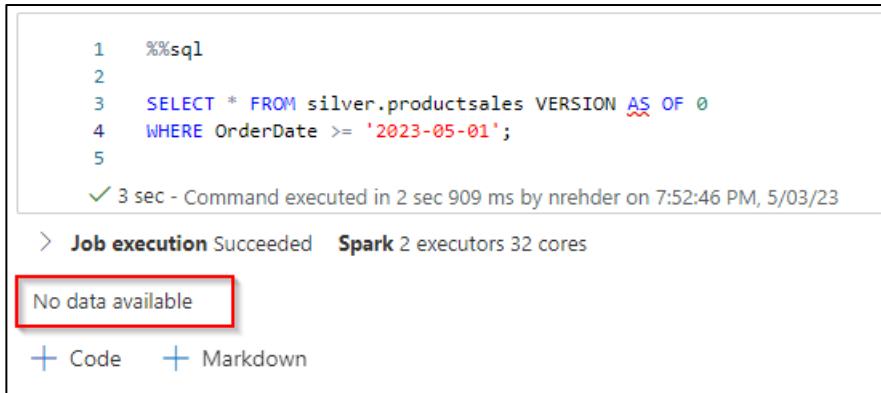
```
%%sql
SELECT * FROM silver.productsales VERSION AS OF 0
WHERE OrderDate >= '2023-05-01';
```

14. Select the Run button to run the script.



```
1 %%sql
2
3 SELECT * FROM silver.productsales VERSION AS OF 1
4 WHERE OrderDate >= '2023-05-01';
5
[] Press shift + enter to run
```

In Version 0, you will not see any values, since the previous version did not have any new OrderDate rows after May 01 2023.



```
1 %%sql
2
3 SELECT * FROM silver.productsales VERSION AS OF 0
4 WHERE OrderDate >= '2023-05-01';
5
✓ 3 sec - Command executed in 2 sec 909 ms by nrehder on 7:52:46 PM, 5/03/23
> Job execution Succeeded Spark 2 executors 32 cores
No data available
+ Code + Markdown
```

15. Change the previous SQL code from **VERSION AS OF 0** to **VERSION AS OF 1** and run the script again.

```
%%sql
SELECT * FROM silver.productsales VERSION AS OF 1
WHERE OrderDate >= '2023-05-01';
```

You will now see the changed data from the uploaded file in chapter [Data Changes](#) together with the transformations we implemented in data flow **DeltaSilverProductSales**.

ProductName	ProductKey	UnitPrice	OrderDate	OrderQuantityTotal
All-Purpose Bike Stand	486	159	2023-05-02	2
AWC Logo Cap	225	8.989999771118164	2023-05-03	1
Fender Set - Mountain	485	21.979999542236328	2023-05-04	1
Mountain-200 Black, 42	360	2049.09814453125	2023-05-04	2
HL Mountain Tire	537	35	2023-05-02	1
Fender Set - Mountain	485	21.979999542236328	2023-05-03	1

Steps 15 and 16 are **optional**. We advise you to do these steps after the workshop and continue the workshop on step 18.

(OPTIONAL)

Instead of using the command **VERSION AS OF**, we can also view previous versions using a timestamp with the command **TIMESTAMP AS OF**.

16. Add the following SQL code to view the initial version of your product sales data using the **TIMESTAMP AS OF** command.

NOTE: The timestamp "**2023-05-03T17:41:21Z**" from the code below needs to be replaced. See the next steps.

```
%%sql
SELECT * FROM silver.productsales TIMESTAMP AS OF "2023-05-03T17:41:21Z"
WHERE OrderDate >= '2023-05-01';
```

The Timestamps we can query can be found in the **DESCRIBE HISTORY** command we used previously.

```

1 %%sql
2
3 DESCRIBE HISTORY `abfss://dlssds1pocfs1@dlssds1poc.dfs.core.windows.net/silver-container/ProductSales/`;
4

```

✓ 11 sec - Command executed in 10 sec 900 ms by nrehder on 7:51:12 PM, 5/03/23

> Job execution Succeeded Spark 2 executors 32 cores [View in monitoring](#)

View [Table](#) [Chart](#) [Export results](#)

version	timestamp	userId	userName	operation
1	2023-05-03T17:41:21Z	null	null	MERGE
0	2023-05-03T16:03:37Z	null	null	WRITE

17. Copy a timestamp from the **DESCRIBE HISTORY** command and paste it in the SQL code from above.

```

1 %%sql
2
3 DESCRIBE HISTORY `abfss://dlssds1pocfs1@dlssds1poc.dfs.core.windows.net/silver-container/ProductSales/`;
4

```

✓ 11 sec - Command executed in 10 sec 900 ms by nrehder on 7:51:12 PM, 5/03/23

> Job execution Succeeded Spark 2 executors 32 cores [View in monitoring](#)

View [Table](#) [Chart](#) [Export results](#)

version	timestamp	userId	userName	operation
1	2023-05-03T17:41:21Z	null	null	MERGE
0	2023-05-03T16:03:37Z	null	null	WRITE

```
%> %%sql
SELECT * FROM silver.productsales, TIMESTAMP AS OF "2023-05-03T17:41:21Z"
WHERE OrderDate >= '2023-05-01';
```

We can now see the new rows from the uploaded file in chapter [Data Changes](#).

```

1 %%sql
2
3 SELECT * FROM silver.productsales VERSION AS OF 1
4 WHERE OrderDate >= '2023-05-01';
5

```

✓ 5 sec - Command executed in 5 sec 511 ms by nrehder on 7:53:26 PM, 5/03/23

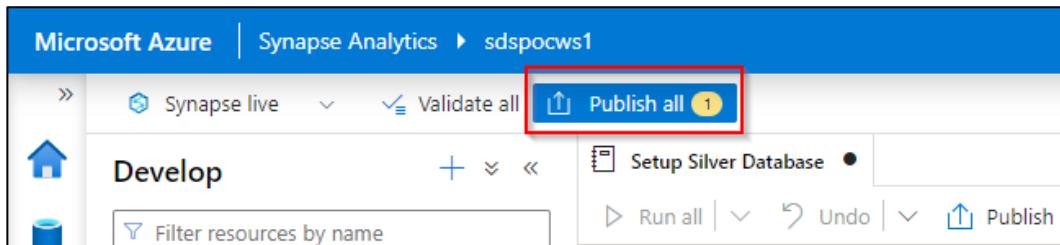
> Job execution Succeeded Spark 2 executors 32 cores [View in monitoring](#) [Open Spark UI](#)

View [Table](#) [Chart](#) [Export results](#)

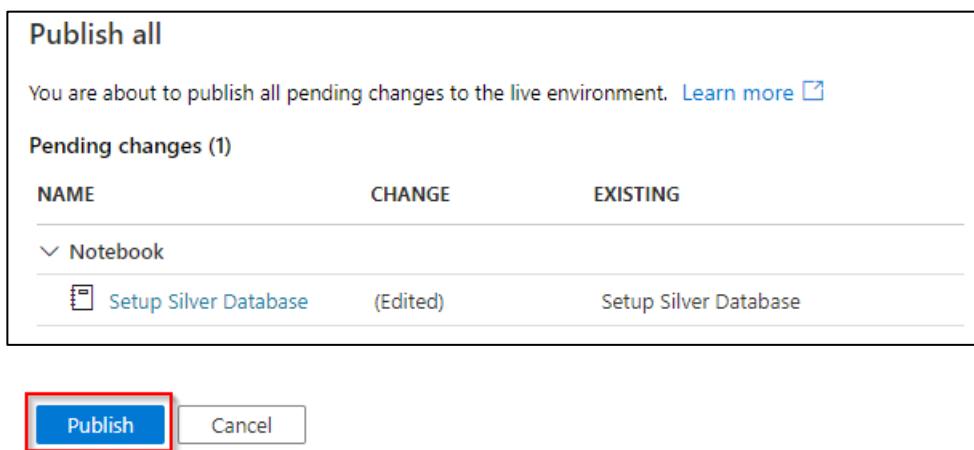
ProductName	ProductKey	UnitPrice	OrderDate	OrderQuantityTotal
All-Purpose Bike Stand	486	159	2023-05-02	2
AWC Logo Cap	225	8.989999771118164	2023-05-03	1
Fender Set - Mountain	485	21.979999542236328	2023-05-04	1
Mountain-200 Black, 42	360	2049.09814453125	2023-05-04	2
HL Mountain Tire	537	35	2023-05-02	1
Fender Set - Mountain	485	21.979999542236328	2023-05-03	1

Let's save our work.

18. Select the **Publish All** button.



19. A new panel should open to the right. Select the **Publish** button.



A banner should appear confirming that the content was published.



You have now learned how to view different versions of your delta formatted data being used as a delta table in your lake database using Spark SQL.

You have successfully queried a delta table using a serverless SQL pool.

8.1.1.6 Removing Historic Delta Logs (OPTIONAL)

This chapter is **OPTIONAL**. We advise you to do these steps after the workshop and continue the workshop on chapter [Gold Database](#).

Delta Lake maintains a history of all the changes by default. That means over a period, the historical data will grow. Based on your business requirements you would like to keep historical data for a past certain duration like last one month, last one year etc. to optimize your storage costs.

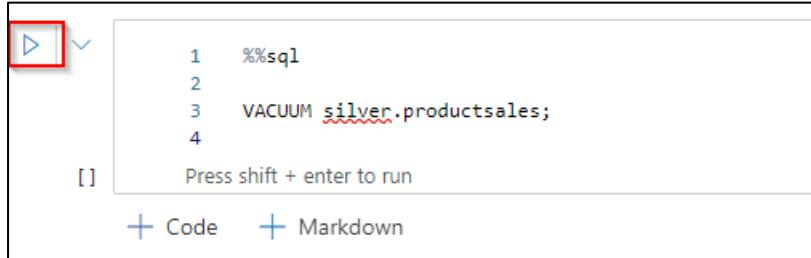
Name	Modified
[..]	
_delta_log	
part-00000-a367e0e5-cf99-4a8a-bfeb-e886f592e2c9-c000.snappy.parquet	5/13/2023, 11:01:59 AM
part-00001-8488466f-c7de-4d0c-9798-ec88b8c06b7c-c000.snappy.parquet	5/13/2023, 11:01:59 AM
part-00002-ddfa9f46-3c51-490e-9d3f-e8aa9d819a44-c000.snappy.parquet	5/13/2023, 11:01:59 AM
part-00003-e8b95a83-7668-4908-a1d2-2b4c4ecc8c08-c000.snappy.parquet	5/13/2023, 11:01:59 AM
part-00004-913ea121-5088-42d1-9bc5-71c33872813f-c000.snappy.parquet	5/13/2023, 11:02:02 AM
part-00005-4817b5a3-4461-4c5c-b347-443cfefbefc9-c000.snappy.parquet	5/13/2023, 11:02:02 AM
part-00006-9d25989f-18e7-4630-8612-dae49d8ca74e-c000.snappy.parquet	5/13/2023, 11:02:00 AM

1. Within the Notebook **Setup Silver Database** code field, add the following SQL code to clear historic delta logs.

NOTE: This same command can be used later in the Setup Gold Database to clear historic delta logs.

```
%%sql  
VACUUM silver.productsales;
```

2. Select the Run button to run the script.

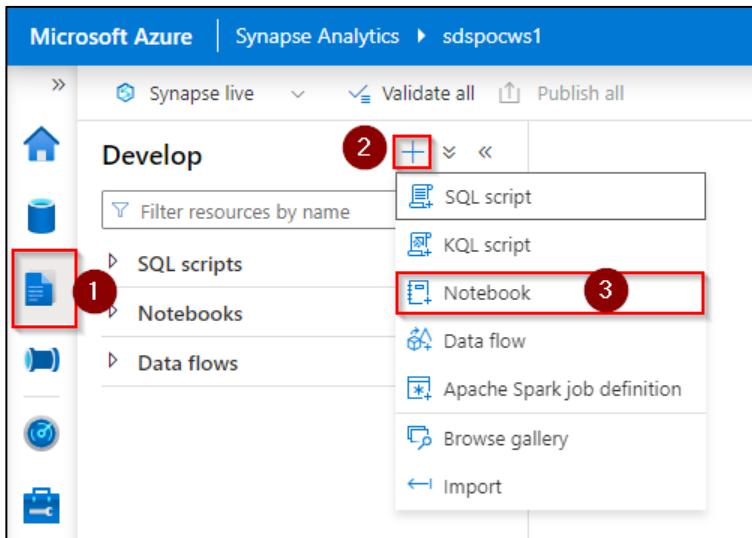


NOTE: You cannot delete historical data within the last 7 days by default and that is to maintain consistency in data.

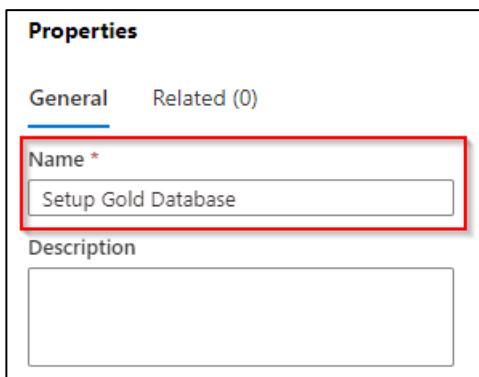
8.1.2 Gold Database

We will now create our gold database, where the data is made ready for consumption. The steps here are similar to what we configured in chapter [Silver Database](#).

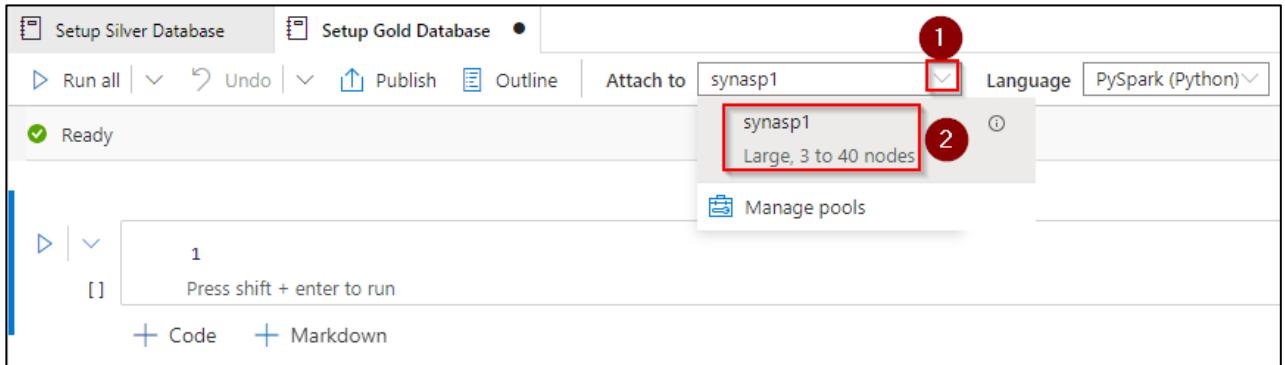
1. Select the **Develop** tab to the left, select the + button and create a new Notebook.



2. Rename the newly created Notebook to **Setup Gold Database**.



- To begin using this new Notebook, attach an Apache Spark Pool by selecting the **Attach to** dropdown. Select the provisioned spark pool.



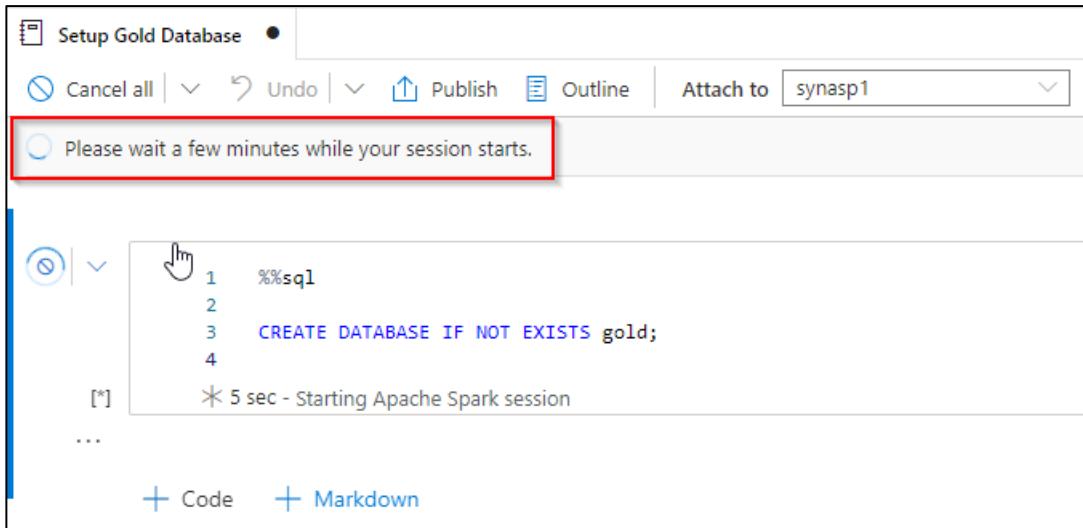
8.1.2.1 Lake Database

- Within the Notebook **Setup Gold Database** code field, add the following SQL code to create the database:

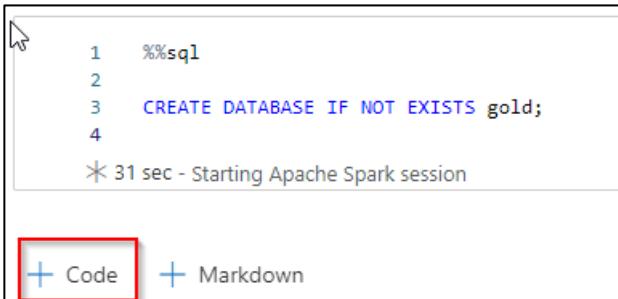
```
%%sql  
CREATE DATABASE IF NOT EXISTS gold;
```

- Select the Run button to start the Spark Pool and run the script.

Note: Starting the Spark Pool may take 3 – 5 minutes. Please be patient!



6. Once the code has run, beneath the initial code field we created in step 4. Select the **+ Code** button, to add a new code field to the Notebook.



```
1 %%sql
2
3 CREATE DATABASE IF NOT EXISTS gold;
4
5 * 31 sec - Starting Apache Spark session
```

+ Code + Markdown

8.1.2.2 Lake Database Delta Tables

7. Within the Notebook **Setup Gold Database** code field, add the following SQL code to create the delta tables in the gold lake database.

```
%%sql

CREATE TABLE IF NOT EXISTS gold.productsales
USING DELTA
LOCATION 'abfss://dlssdpocfs1@dlssdpoc.dfs.core.windows.net/gold-container/ProductSales/';
```

8. Change the **LOCATION** parameter within the SQL code to your URI as described in chapter [Azure Blob File System](#).



```
%%sql
CREATE TABLE IF NOT EXISTS gold.productsales
USING DELTA
LOCATION 'abfss://dlssdpocfs1@dlssdpoc.dfs.core.windows.net/gold-
container/ProductSales/';
```

9. Select the Run button to run the script.

The screenshot shows the Synapse Notebook interface with the title "Setup Gold Database". The toolbar includes "Run all", "Undo", "Publish", "Outline", "Attach to" (set to "synasp1"), "Language" (set to "PySpark (Python)"), and a "View" dropdown. A green checkmark icon indicates the notebook is "Ready".

Cell [1] contains the following PySpark SQL code:

```
1 %%sql
2
3 CREATE DATABASE IF NOT EXISTS gold;
4
```

Output for cell [1]:

```
[1] ✓ 1 min 7 sec - Apache Spark session started in 52 sec 171 ms. Command executed in 15 sec 47 ms by nrehder on 8:55:20 PM, 5/01/23
No data available
```

Cell [2] contains the following PySpark SQL code:

```
1 %%sql
2
3 CREATE TABLE IF NOT EXISTS gold.productsales
4 USING DELTA
5 LOCATION 'abfss://dlssdspocfs1@dlssdspoc.dfs.core.windows.net/gold-container/ProductSales/';
6
7
```

Output for cell [2]:

```
[2] ✓ 33 sec - Command executed in 33 sec 764 ms by nrehder on 9:01:50 PM, 5/01/23
> Job execution Succeeded  Spark 2 executors 8 cores
...
No data available
```

At the bottom of the interface, there are two buttons: "+ Code" (highlighted with a red box) and "+ Markdown".

10. Once the code has run, select the **+ Code** button, to add a new code field to the Notebook.

The screenshot shows the Synapse Notebook interface with the same setup as the previous screenshot. Cell [2] has already been run, showing its output.

At the bottom of the interface, the "+ Code" button is highlighted with a red box, while the "+ Markdown" button is visible next to it.

8.1.2.3 Audit Logs

11. Add the following SQL code to view the change history of your product data i.e., the data that we manually changed in chapter [Data Changes](#).

```
%%sql  
DESCRIBE HISTORY 'abfss://dlssdspocfs1@dlssdspoc.dfs.core.windows.net/gold-container/ProductSales/' ;
```

12. Within the SQL code above change the **abfss URI** to your URI as described in chapter [Azure Blob File System](#).

```
%%sql  
DESCRIBE HISTORY 'abfss://dlssdspocfs1@dlssdspoc.dfs.core.windows.net/gold-  
container/ProductSales/' ;
```



Change to your URI

13. Select the Run button to run the script.

```
1 %%sql
2
3 CREATE DATABASE IF NOT EXISTS gold;
4
[1] ✓ 1 min 7 sec - Apache Spark session started in 52 sec 171 ms. Command executed in 15 sec 47 ms by nrehder on 8:55:20 PM, 5/01/23
No data available

1 %%sql
2
3 CREATE TABLE IF NOT EXISTS gold.productsales
4 USING DELTA
5 LOCATION 'abfss://dlssdspocfs1@dlssdspoc.dfs.core.windows.net/gold-container/ProductSales/';
6
7
[2] ✓ 33 sec - Command executed in 33 sec 764 ms by nrehder on 9:01:50 PM, 5/01/23
> Job execution Succeeded Spark 2 executors 8 cores
No data available

[3] ✓ 2 min 57 sec - Apache Spark session started in 2 min 14 sec 917 ms. Command executed in 42 sec 110 ms by nrehder on 8:48:18 AM, 4/28/23
Press shift + enter to run
+ Code + Markdown
```

Once the code has run you will see the change operations WRITE (the initial trigger) and MERGE (the second trigger after the data changes)

14. Select the + Code button, to add a new code field to the Notebook.

version	timestamp	userId	userName	operation	operationParameters	job
1	2023-04-28T06:19:46Z	null	null	MERGE	["predicate": "(source.ProductKey` null"]	
0	2023-04-27T18:57:36Z	null	null	WRITE	["mode": "Append", "partitionBy": ""]	

+ Code + Markdown

8.1.2.4 Optimization (Z-Ordering)

Z-ordering will allow for greater read performance by taking advantage of data skipping. One or multiple columns can be specified for a Z-order. Ideal column choices are those that are commonly used as filters when reading data.

15. Add the following SQL code to implement Z-ordering optimization using the **OrderDate** column.

```
%%sql
OPTIMIZE gold.productsales ZORDER BY (OrderDate);
```

8.1.2.5 Time Traveling

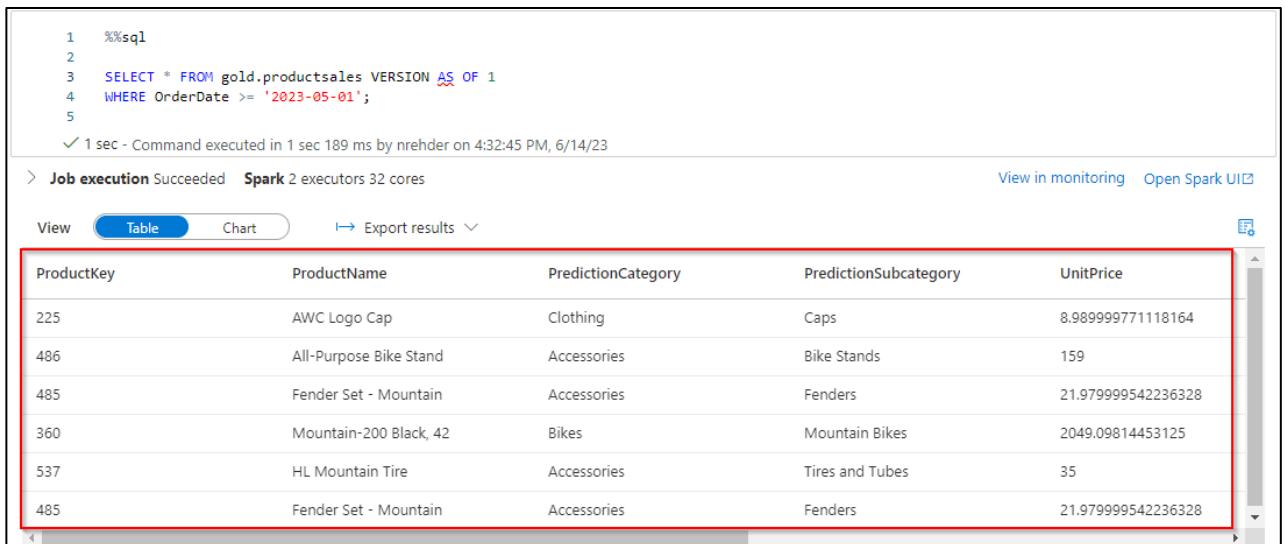
16. Add the following SQL code to view **version 1**.

```
%%sql
SELECT * FROM gold.productsales VERSION AS OF 1
WHERE OrderDate >= '2023-05-01';
```

17. Select the **Run** button to run the script.

```
1 %%sql
2
3 SELECT * FROM gold.productsales VERSION AS OF 1
4 WHERE OrderDate >= '2023-05-01';
5
[17] ✓ 1 sec - Command executed in 1 sec 189 ms by nrehder on 4:32:45 PM, 6/14/23
```

You will now see the same changed data we queried previously in chapter [Silver Database](#), but transformed based on the transformations activities we implemented in data flow **DeltaGoldProductSales**.



1 %%sql
 2
 3 SELECT * FROM gold.productsales VERSION AS OF 1
 4 WHERE OrderDate >= '2023-05-01';
 5

✓ 1 sec - Command executed in 1 sec 189 ms by nrehder on 4:32:45 PM, 6/14/23

> Job execution Succeeded Spark 2 executors 32 cores

View Table Chart Export results

ProductKey	ProductName	PredictionCategory	PredictionSubcategory	UnitPrice
225	AWC Logo Cap	Clothing	Caps	8.98999971118164
486	All-Purpose Bike Stand	Accessories	Bike Stands	159
485	Fender Set - Mountain	Accessories	Fenders	21.979999542236328
360	Mountain-200 Black, 42	Bikes	Mountain Bikes	2049.09814453125
537	HL Mountain Tire	Accessories	Tires and Tubes	35
485	Fender Set - Mountain	Accessories	Fenders	21.979999542236328

8.1.2.6 Restoring

Let's assume, we are not happy with the changes made and would like to revert back to version 0 i.e. without the data changes we implemented in chapter [Data Changes](#).

18. Select the **+ Code** button, to add a new code field to the Notebook.
19. Add the following SQL code to restore **version 0** as the primary data structure.

```
%%sql
RESTORE gold.productsales TO VERSION AS OF 0;
```

20. Select the **Run** button to run the script.



We can now view this restoration process in our audit log. Re-run the DESCRIBE HISTORY script from step 13. You should now see the RESTORE command in the logs.

NOTE: Restoring is database specific. If you restore a version within the gold database, you will also have to do so manually in the silver database.

```

1 %%sql
2
3 DESCRIBE HISTORY abfss://dlssdcpcocfs@dlssdcpcoc.dfs.core.windows.net/gold-container/ProductSales/';
4

[6] ✓ 1 sec - Command executed in 1 sec 145 ms by nrehder on 9:07:01 PM, 6/14/23
> Job execution Succeeded Spark 2 executors 32 cores
View in monitoring
...
```

version	timestamp	userId	userName	operation
3	2023-06-14T19:06:38Z	null	null	RESTORE
2	2023-06-14T19:05:53Z	null	null	OPTIMIZE
1	2023-06-14T18:44:41Z	null	null	MERGE
0	2023-06-14T18:06:29Z	null	null	WRITE

Let's verify that we are viewing the initial data structure (version 0).

21. Select the **+ Code** button, to add a new code field to the Notebook.
22. Add the following SQL code to view the data.

```

%%sql

SELECT * FROM gold.productsales VERSION AS OF 3
WHERE OrderDate >= '2023-05-01';

```

23. Select the **Run** button to run the script.

```

1 %%sql
2
3 SELECT * FROM gold.productsales VERSION AS OF 3
4 WHERE OrderDate >= '2023-05-01';
5

[7] ✓ 1 sec - Command executed in 1 sec 154 ms by nrehder on 9:08:05 PM, 6/14/23
> Job execution Succeeded Spark 2 executors 32 cores
...
```

No data available

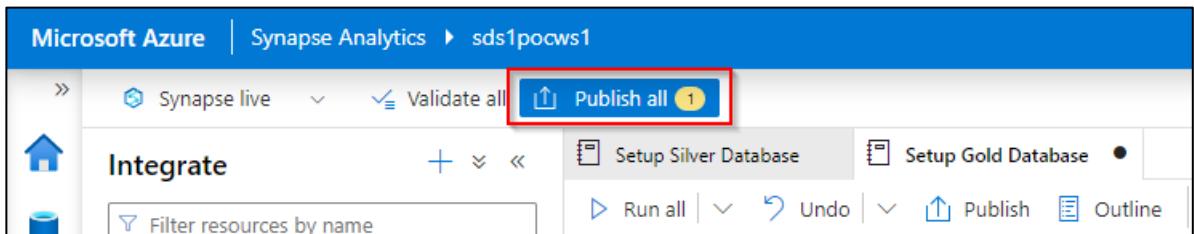
NOTE: We select **VERSION AS OF 3** because version 2 is the Z-Ordering optimization we implemented in chapter [Optimization \(Z-Ordering\)](#)

Running this script should return no data since the initial data i.e., version 0 did not have any new OrderDate rows after May 01, 2023.

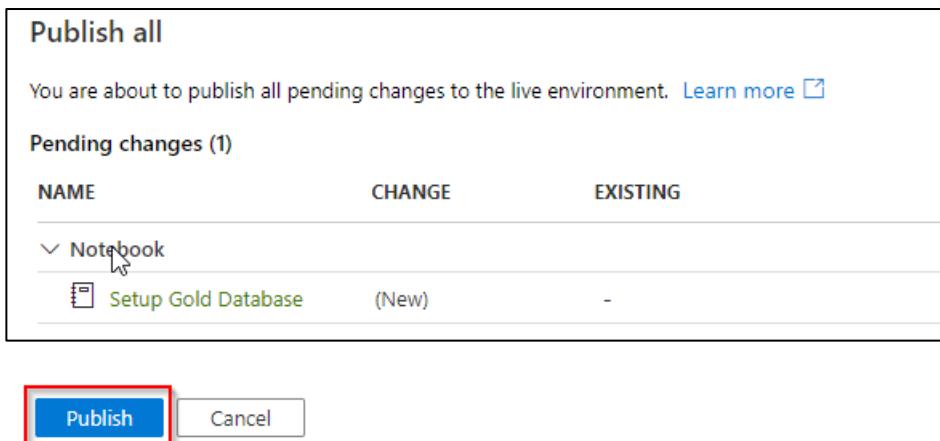
We have therefore successfully reversed our data changes.

Let's save our work.

24. Select the Publish All button.



25. A new panel should open to the right. Select the **Publish** button.



A banner should appear confirming that the content was published.

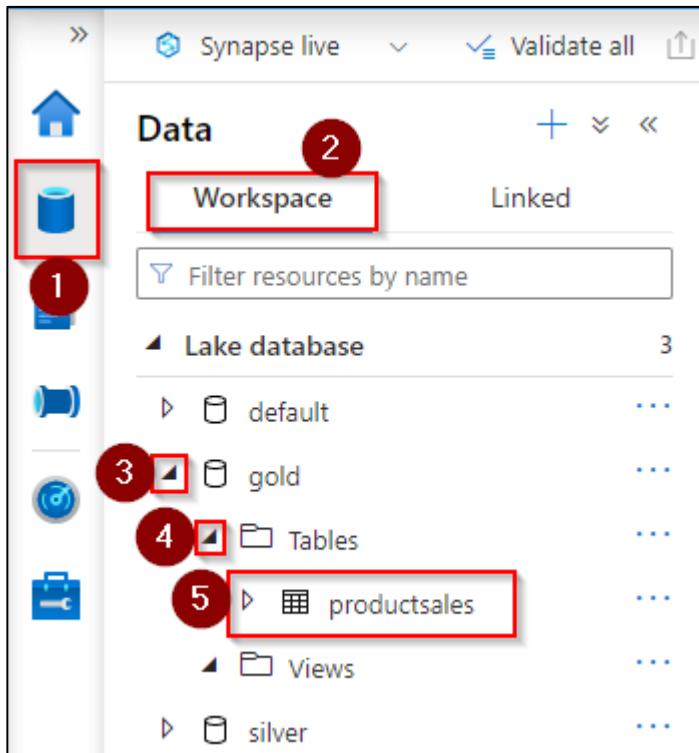


8.1.2.7 Query Delta Table with SQL Serverless Pool

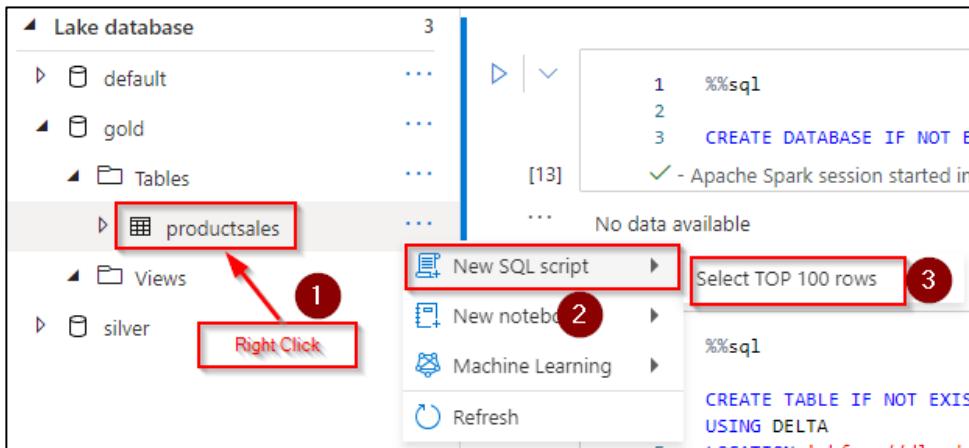
We will now query the delta table we created from the delta formatted data located in the gold-container in chapter [Gold Database](#).

NOTE: You may need to refresh your browser to view the newly created Lake Database in the Data tab. Make sure you publish everything and save your work before refreshing the browser. See Step 24.

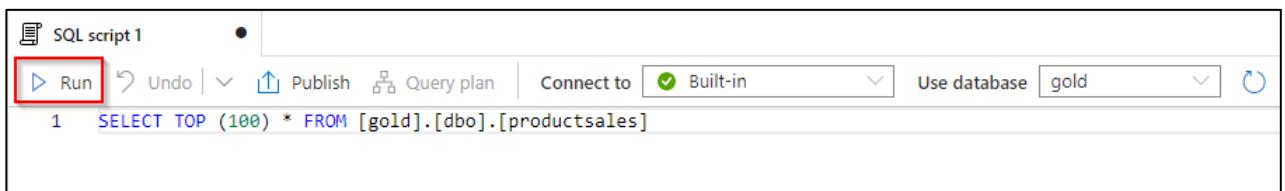
34. Select the **Data** tab to the left, select **Workspace** and browse to the **Tables** folder. Within the **Tables** folder you will see the **productsales** table we created earlier.



35. Right click the **productsales** table and select **New SQL script**, followed by **Select TOP 100 rows**.
 This will create a new tab with a SQL statement to query the table.



36. Select the **Run** button to run the SQL statement.



After query execution, you will now see the data related to the **productsales** delta table in the gold database.

ProductKey	ProductName	PredictionCategory	PredictionSubCategory	UnitPrice	OrderDate	OrderQuantity	AmountSold
214	Sport-100 Helmet, Red	Accessories	Helmets	34.99	2018-10-24T00:00:00	1	34.99
217	Sport-100 Helmet, Black	Accessories	Helmets	34.99	2019-01-07T00:00:00	12	419.88
217	Sport-100 Helmet, Black	Accessories	Helmets	34.99	2019-02-22T00:00:00	5	174.95
217	Sport-100 Helmet, Black	Accessories	Helmets	34.99	2019-03-04T00:00:00	7	244.93
222	Sport-100 Helmet, Blue	Accessories	Helmets	34.99	2018-12-22T00:00:00	5	174.95
222	Sport-100 Helmet, Blue	Accessories	Helmets	34.99	2019-01-21T00:00:00	8	279.92
222	Sport-100 Helmet, Blue	Accessories	Helmets	34.99	2019-10-01T00:00:00	5	174.95
222	Sport-100 Helmet, Blue	Accessories	Helmets	34.99	2019-10-13T00:00:00	6	209.94
231	Long-Sleeve Logo Jersey, M	Clothing	Jerseys	49.99	2019-06-09T00:00:00	2	99.98

9 Query Delta Formatted Data in Azure Storage with Serverless SQL Pool (OPTIONAL)

This chapter is **OPTIONAL**. We advise you to do this chapter after the workshop.

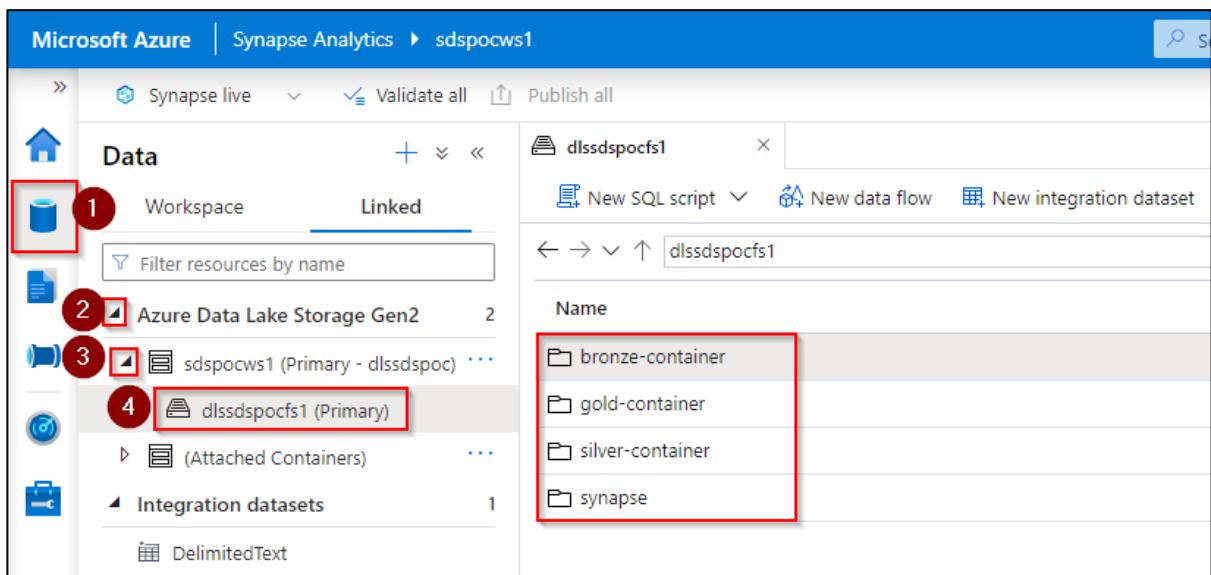
We have been querying the delta tables using the Apache Spark Pool and the Serverless SQL Pool. Additionally, we can also query the delta formatted files directly from the Azure storage using the Serverless SQL Pool. This method is usually reserved for ad-hoc queries.

NOTE: Querying the delta files using the Serverless SQL Pool comes with limitations. One of those is the inability to time travel. The data queried is always in the most resent state. Additional limitations can be found [here](#).

NOTE: We are querying the gold-container in this chapter, but any container with delta formatted data could be used.

1. Select the **Data** tab to the left, select **Linked** and open your **primary data storage** i.e., the storage you provisioned in chapter [Create Resource Group and Resources](#). Select the **named container** of this storage.

Selecting the container will open a **new tab** on the Synapse canvas. You will see the medallion structure you created earlier (*bronze-container, silver-container, and gold-container*).



The first data flow we created (**DeltaSilverProductSales**) imported data from the *bronze-container*, transformed it and saved it to the Delta format in the *silver-container*. Later we created a second data flow (**DeltaGoldProductSales**) that imported data from the *silver-container* and joined it with predicted data, which in turn was saved in the Delta format in the *gold-container*. To review these steps, see chapter [Data Flows](#).

2. Within the **gold-container**, select the **ProductSales** directory.

You will see that a **delta log folder** (_delta_log) was created together with a delta formatted file.

We can now query this delta formatted file using the SQL Serverless pool directory from the data lake.

3. Go back to the **gold-container**, right-click on the **ProductSales** directory, select **New SQL script** and **Select Top 100 rows**.

4. Once selected, a new panel will open to the right. Select the File Type **Delta format** and select **Apply**.

Select TOP 100 rows

ProductSales

Source folder format settings
Specify the format and layout of your data.

Folder path
<https://dlssdspocdfs.core.windows.net/dlssdspocfs1/gold-container/ProductSales/>

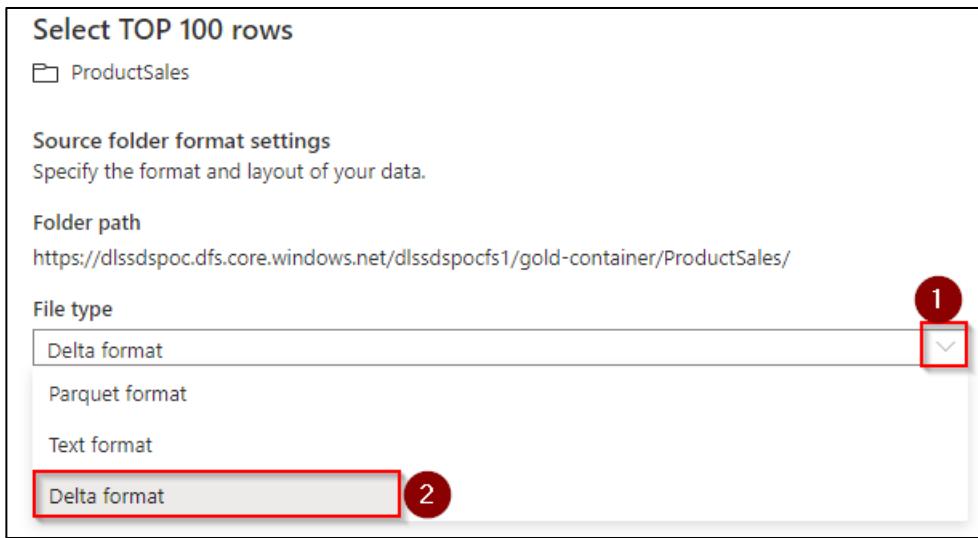
File type

1 Delta format

Parquet format

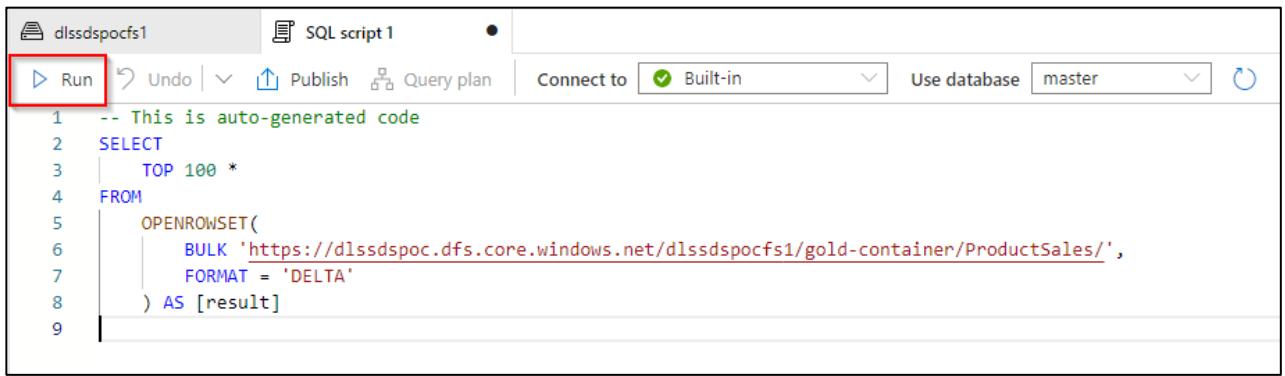
Text format

2 Delta format



This will create a SQL statement querying the first 100 rows of ProductSales directory located in the gold-container using the delta format.

5. Run the query.



```
1 -- This is auto-generated code
2 SELECT
3     TOP 100 *
4 FROM
5     OPENROWSET(
6         BULK 'https://dlssdspocdfs.core.windows.net/dlssdspocfs1/gold-container/ProductSales/',
7         FORMAT = 'DELTA'
8     ) AS [result]
```

You will see a query output below:

The screenshot shows a SQL Server Management Studio (SSMS) window. The top pane displays a T-SQL script named "SQL script 1" with auto-generated code for reading data from an Azure Data Lake Store table. The code uses OPENROWSET with BULK and FORMAT = 'DELTA' options. The bottom pane shows the results of the query, which is a table of product sales data. A red arrow points from the top of the code pane down to the results table.

```
1 -- This is auto-generated code
2 SELECT
3 | TOP 100 *
4 FROM
5 OPENROWSET(
6 BULK 'https://dlssdspoc.dfs.core.windows.net/dlssdspocfs1/gold-container /ProductSales/',
7 FORMAT = 'DELTA'
8 ) AS [result]
```

ProductKey	ProductName	PredictionCategory	PredictionSubCategory	UnitPrice	OrderDate	OrderQuantity	AmountSold
217	Sport-100 Helmet, Black	Accessories	Helmets	34.99	2019-06-09T00:00:00.000	6	209.94
217	Sport-100 Helmet, Black	Accessories	Helmets	34.99	2019-10-05T00:00:00.000	5	174.95
222	Sport-100 Helmet, Blue	Accessories	Helmets	34.99	2019-05-03T00:00:00.000	6	209.94
225	AWC Logo Cap	Clothing	Caps	8.99	2019-03-23T00:00:00.000	5	44.95
225	AWC Logo Cap	Clothing	Caps	8.99	2019-07-03T00:00:00.000	5	44.95
225	AWC Logo Cap	Clothing	Caps	8.99	2019-09-21T00:00:00.000	5	44.95

00:00:07 Query executed successfully.

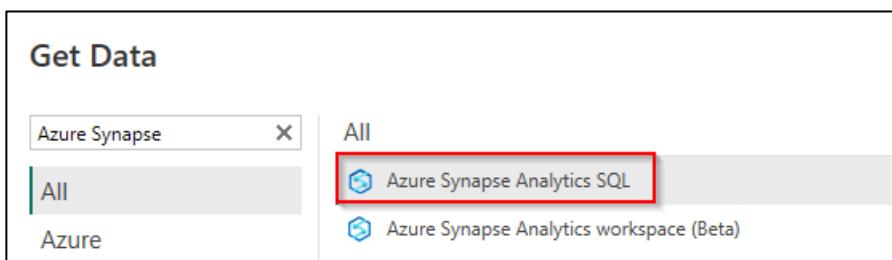
10 Connect to Power BI (OPTIONAL)

This chapter is **OPTIONAL**. We advise you to do this chapter after the workshop.

We have created a data lakehouse, the next step is to import and visualize the data using a BI Tool such as Power BI.

There are currently two methods of importing delta formatted data into Power BI if you've created the data lakehouse using Azure Synapse Analytics:

1. Azure Synapse Analytics SQL Connector



This native connector uses the **Serverless SQL endpoint** for Azure Synapse to query the delta tables we created in chapter [Create Azure Synapse Lake Database](#). This is a great option if the report consumers expect live data, as you can connect to the Serverless SQL endpoint using the **DirectQuery mode**.

Since we are using the Serverless SQL endpoint, we cannot time travel. The delta formatted data we query using this connector will always be the most current version.

In the future, Microsoft will most likely make a Serverless Spark endpoint for Azure Synapse available, allowing us to query the delta formatted data using spark, which enables time travel. But for now, this is not yet available.

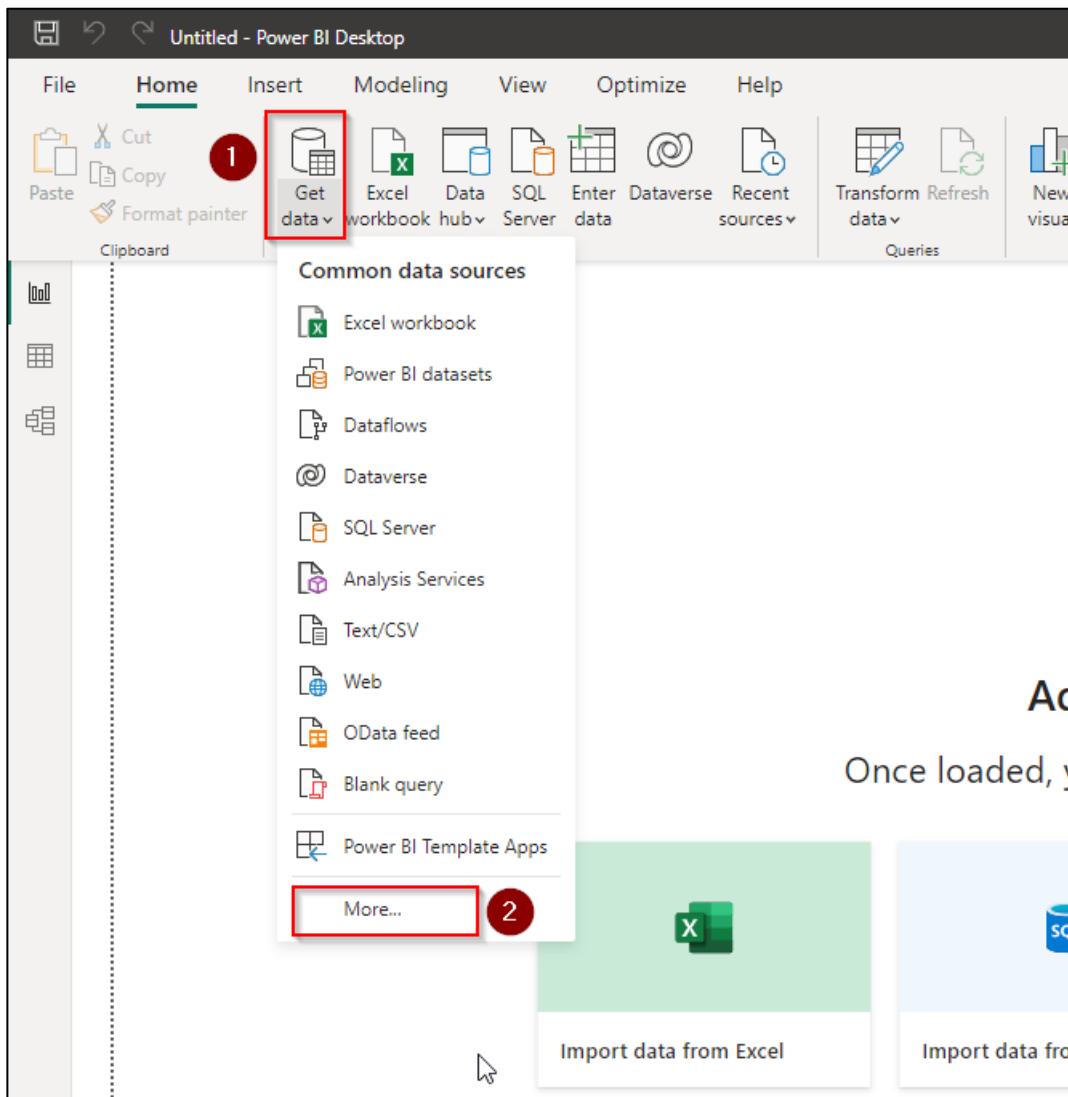
2. Custom Delta Lake Connector

This [custom connector](#) allows you to import your delta lake files directly from your storage account. This option allows you to define the version number directly in the Power Query Editor in Power BI. Keep in mind that you can only use the **import mode** with this option.

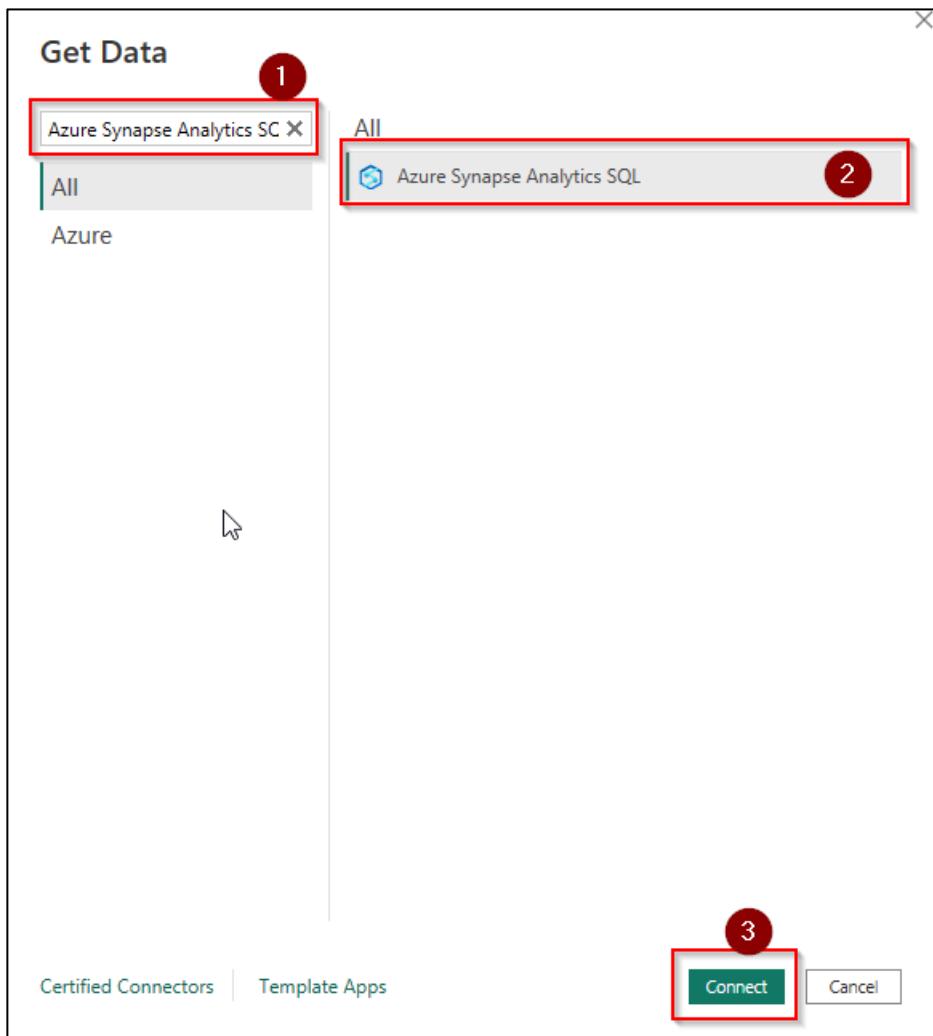
10.1.1 Azure Synapse Analytics SQL Connector

Let's connect our delta table in the gold database located in the Azure Synapse using the Power BI Azure Synapse Analytics SQL Connector.

1. Open the **Power BI Desktop** application.
2. Once opened, select the **Get data** button and select **More** to view all the available connectors.



3. A new window will open, search for the **Azure Synapse Analytics SQL** and select it from the list.
4. Select the **Connect** button.



A new window will appear requesting the SQL Server database information.



We require the SQL Serverless Pool endpoint for Azure Synapse.

3. Go back to the [Azure portal](#) home screen and select the **Resource group** you provisioned.

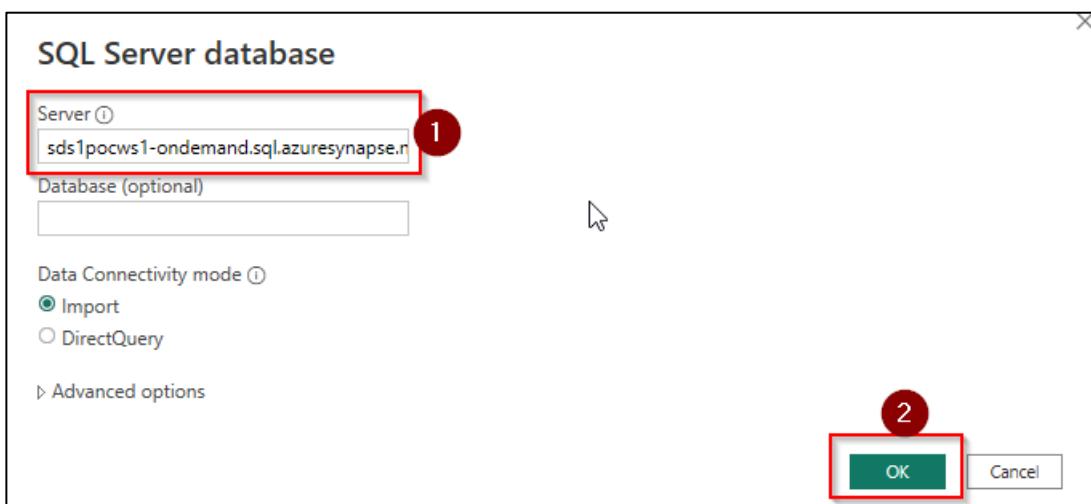
- 4. Within the Resource group, select the resource type Synapse workspace.**

<input type="checkbox"/>	Name ↑↓	Type ↑↓	Location ↑↓
<input type="checkbox"/>	dlssdpoc	Storage account	Switzerland North
<input type="checkbox"/>	 sdspocws1	Synapse workspace	Switzerland North
<input type="checkbox"/>	 synasp1 (sdspocws1/synasp1)	Apache Spark pool	Switzerland North

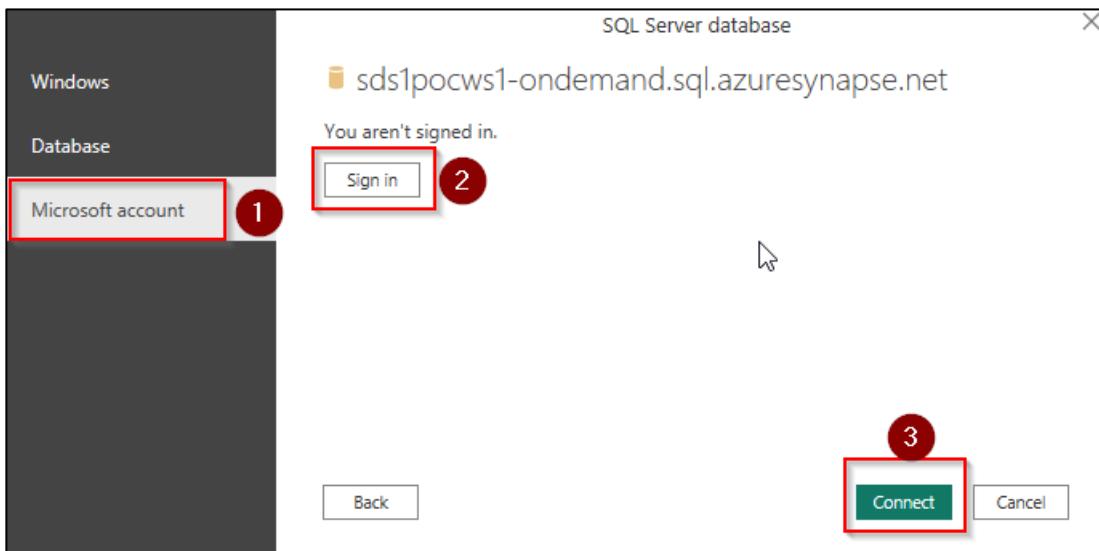
5. A new page will appear. The Serverless SQL endpoint is in the **overview** section of the Synapse workspace.

The screenshot shows the Azure portal's 'Overview' page for a Synapse workspace. The 'Serverless SQL endpoint' field is highlighted with a red box and a red arrow points to it from the Power BI dialog.

6. **Copy** the Serverless SQL endpoint.
7. Go back to your Power BI Desktop application and paste the Serverless SQL endpoint in the **Server** field and select **OK**.



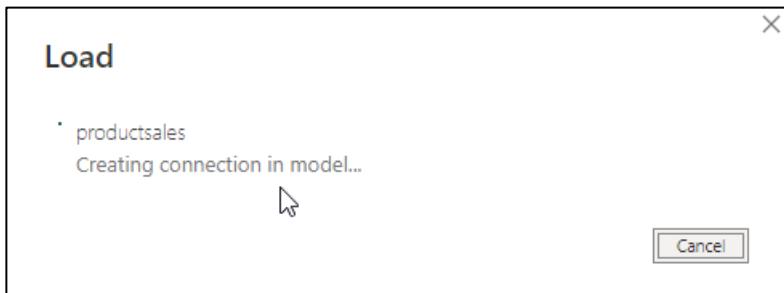
8. A new window will open asking for the credentials. Select **Microsoft account**, select **Sign in** and select the **Connect** button.



9. A new window will open showing the databases we created in chapters [Create Azure Synapse SQL Database \(OPTIONAL\)](#) and [Create Azure Synapse Lake Database](#).
 10. Open the **gold** database and select the **productsales** delta table.
 11. Select **Load** to import the data directly into Power BI without calling the Power Query Editor.

ProductKey	ProductName	PredictionCategory
214	Sport-100 Helmet, Red	Accessories
214	Sport-100 Helmet, Red	Accessories
217	Sport-100 Helmet, Black	Accessories
225	AWC Logo Cap	Clothing
228	Long-Sleeve Logo Jersey, S	Clothing
231	Long-Sleeve Logo Jersey, M	Clothing
231	Long-Sleeve Logo Jersey, M	Clothing
237	Long-Sleeve Logo Jersey, XL	Clothing
237	Long-Sleeve Logo Jersey, XL	Clothing
327	Road-650 Red, 44	Bikes
353	Mountain-200 Silver, 38	Bikes
354	Mountain-200 Silver, 42	Bikes
357	Mountain-200 Silver, 46	Bikes
357	Mountain-200 Silver, 46	Bikes
357	Mountain-200 Silver, 46	Bikes
357	Mountain-200 Silver, 46	Bikes
358	Mountain-200 Black, 38	Bikes
359	Mountain-200 Black, 38	Bikes
361	Mountain-200 Black, 42	Bikes
368	Road-250 Red, 44	Bikes
369	Road-250 Red, 48	Bikes
374	Road-250 Black, 44	Bikes
377	Road-250 Black, 52	Bikes

12. A loading screen will appear.



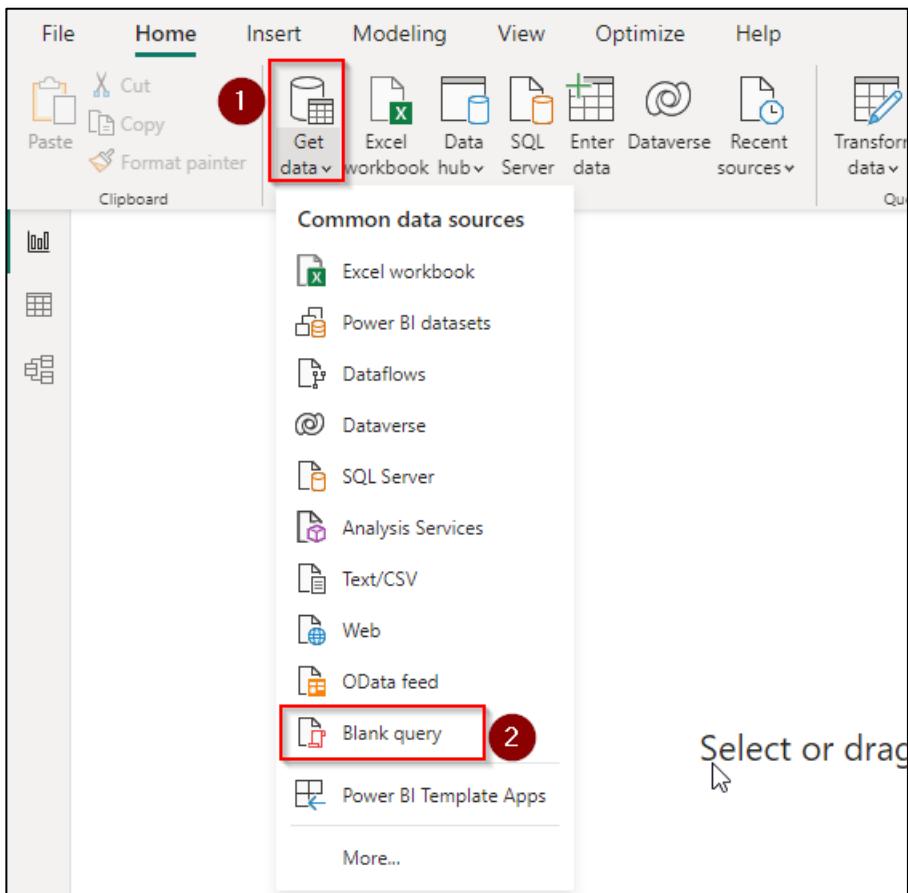
13. The data is now imported and available to be used for visualizations in Power BI.

A screenshot of the Power BI 'Build visual' interface. On the left, there's a large workspace labeled 'Build visual' with the sub-instruction 'Build visual with your data'. Below this, a message says 'Select or drag fields from the Data pane onto the report canvas.' In the center, there's a small preview area showing a green bar chart with a checkmark icon. To the right is the 'Data' pane, which contains a tree view of the imported 'productsales' data. The entire 'productsales' node and its children are highlighted with a red border. The data fields listed under 'productsales' include: AmountSold, OrderDate, OrderQuantityTotal, PredictionCategory, PredictionSubCategory, ProductKey, ProductName, and UnitPrice. Below the tree view, there are sections for 'Values', 'Drill through', 'Cross-report', 'Keep all filters', and 'Add drill-through fields here'.

10.1.2 Custom Delta Lake Connector

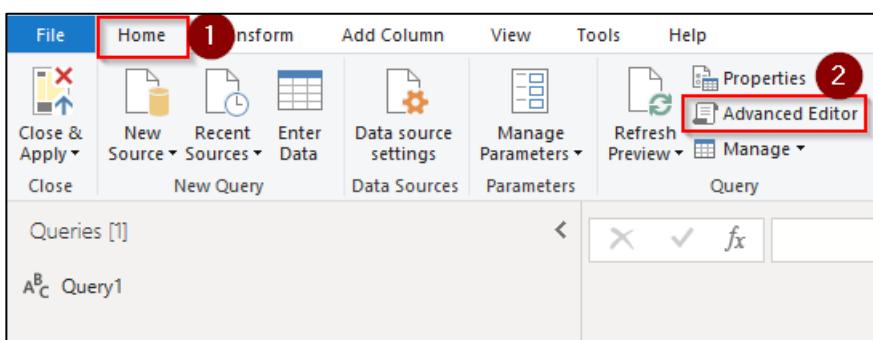
The provided connector allows you to read a Delta Lake table directly from any storage supported by Power BI such as the Azure Data Lake Gen2 we provisioned. Additional information can be found [here](#).

1. Open the **Power BI Desktop** application.
2. Once opened, select the **Get data** button and select **Blank query**.

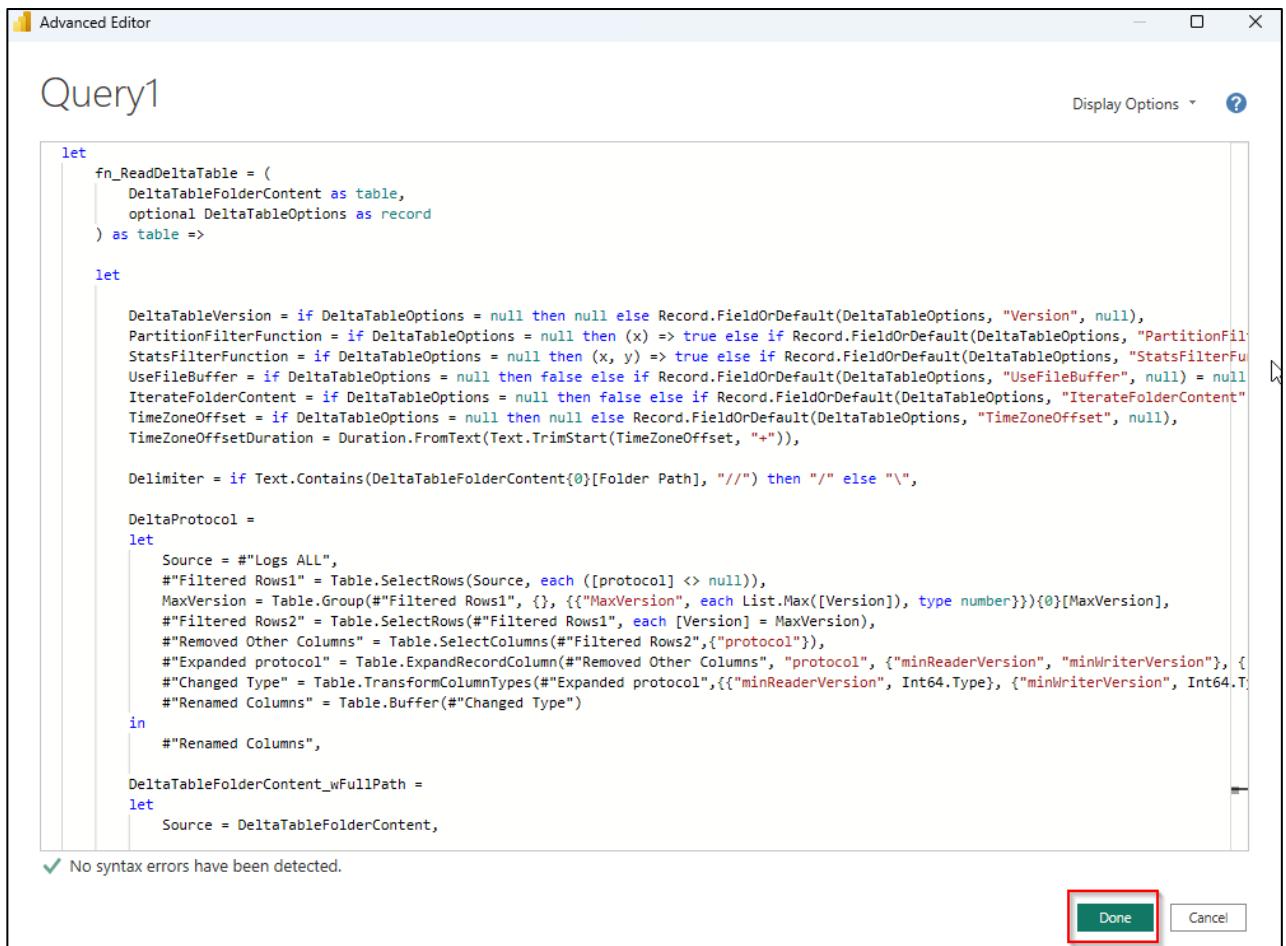


This will open the Power Query Editor.

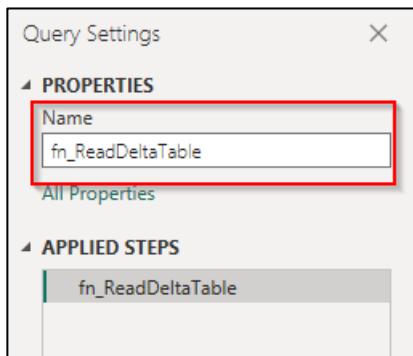
3. Within the Power Query Editor, in the **Home** tab, select **Advanced Editor**.



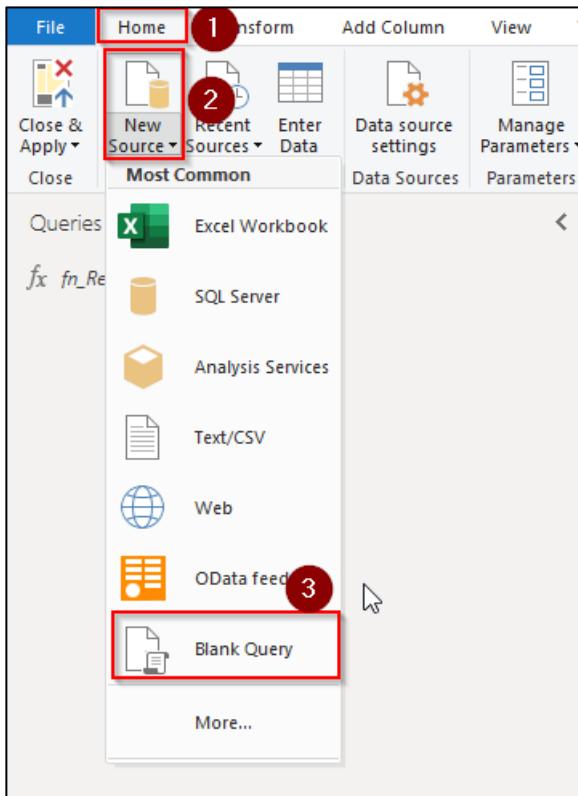
4. A new window will appear.
5. **Copy and paste** the custom M-Code function made available [here](#) into the **Advanced Editor** window.
6. Select **Done**.



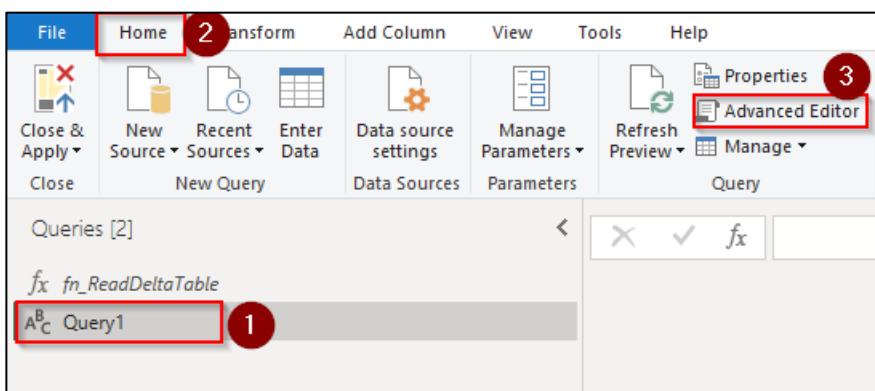
7. Under **Query Settings** located on the right panel, rename this Blank query to **fn_ReadDeltaTable**.



8. Create a new Blank Query within the Power Query Editor by selecting **New Source** followed by **Blank Query**.



9. Within the Power Query Editor, with the new Blank Query selected, go to the **Home** tab, select **Advanced Editor**.



10. A new window will appear.

11. Add the following M-Code into the **Advanced Editor** window.

NOTE: You will need to change the AzureStorage.DataLake function below with your own data lake storage endpoint and container name. Follow the next steps in chapter [Data Lake Storage endpoint](#) and [Container Name](#) and add these missing values.

```
let
  #"Product Sales" = fn_ReadDeltaTable(
    AzureStorage.DataLake(
      "Your Data Lake Storage Endpoint/Your Container Name/gold-container/ProductSales",
      [HierarchicalNavigation = false]),
      [Version = 0])
in
  #"Product Sales"
```

10.1.2.1 Data Lake Storage endpoint

The **Data Lake Storage endpoint** and **container name** can be found in our provisioned storage.

1. Go back to the [Azure portal](#) home screen and select the **Resource group** you provisioned.

Azure services

Create a resource Private endpoints Subscriptions All resources Storage accounts Power BI Embedded Azure Active Directory Microsoft Purview... Resource groups More services

Resources

Recent Favorite

Name	Type	Last Viewed
sdspocws1	Synapse workspace	2023-09-16
rg1sdssdatalakehouse	Resource group	2023-09-16
sdssdss	Resource group	2023-09-16
sdss	Storage account	2023-09-16
sdssdss (sdspocws1)	File	2023-09-16
sdssdss	Resource service	2023-09-16
sdssdss	Resource pool	2023-09-16
sdssdss	Storage account	2023-09-16
sdssdss	Resource (V2) pool	2023-09-16
sdssdss	Log Analytics	2023-09-16
sdssdss	Resource (link) pool	2023-09-16
sdssdss	Management group	2023-09-16

See all

2. Once the resource group is open, select the **Storage account**.

Name ↑↓	Type ↑↓	Location ↑↓
dlssdsspoc	Storage account	Switzerland North
sdspocws1	Synapse workspace	Switzerland North
synasp1 (sdspocws1/synasp1)	Apache Spark pool	Switzerland North

3. Within the storage account, select **Endpoints** on the left panel (You will need to scroll down).
4. **Copy** the Data Lake Storage Endpoint.

dlssds1poc | Endpoints

Storage account

Search Refresh

Provisioning state

Created

Storage account resource ID

Blob service

Resource ID

Blob service

File service

Resource ID

File service

Queue service

Resource ID

Queue service

Table service

Resource ID

Table service

Data Lake Storage

Resource ID

Data Lake Storage **Copy** https://dlssds1poc.dfs.core.windows.net/

Static website

Resource ID

Static website

Networking

Access keys

Shared access signature

Encryption

Microsoft Defender for Cloud

Redundancy

Data protection

Blob inventory

Static website

Lifecycle management

Configuration

Resource sharing (CORS)

SFTP

Advisor recommendations

Endpoints **2**

Locks

Monitoring

Insights

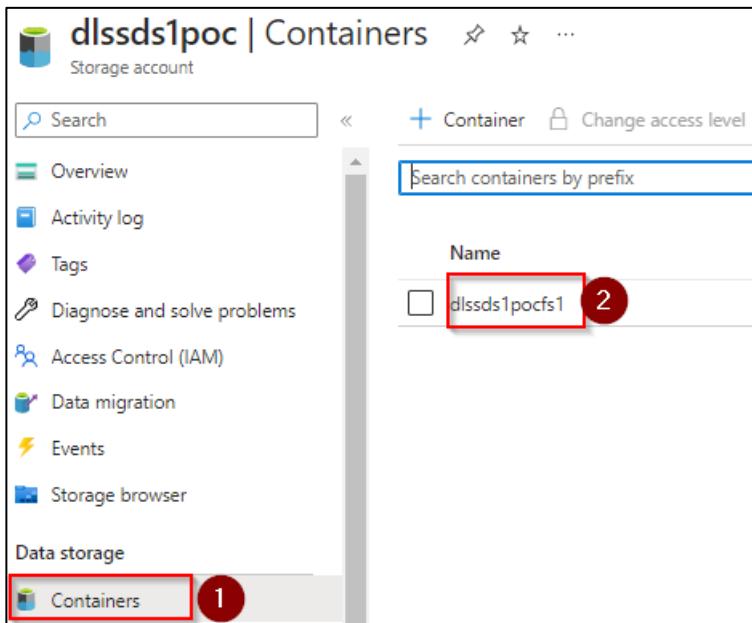
Alerts

Metrics

Workbooks

10.1.2.2 Container Name

5. Within the storage account, select **Container** on the left panel.
6. **Copy** the **Container** name (step 2 in the image).



7. Go back to the **Power Query Editor**.
8. Adjust the M-Code in the **Advanced Editor** with the **Data Lake Storage Endpoint** and **container name** you copied before.

NOTE: Do NOT copy the same data lake storage endpoint and container name as the M-Code below (marked in red). This is only an example.

```
let
    "#Product Sales" = fn_ReadDeltaTable(
        AzureStorage.DataLake(
            "https://dlssdspoc.dfs.core.windows.net/dlssdpscfs1/gold-container/ProductSales",
            [HierarchicalNavigation = false]),
            [Version = 0])
in
    "#Product Sales"
```

9. Select **Done**.

The screenshot shows the 'Advanced Editor' window with a query titled 'Query1'. The M-Code is as follows:

```
let
    #"Product Sales" = fn_ReadDeltaTable(
        AzureStorage.DataLake(
            "https://dlssdpscoc.dfs.core.windows.net/dlssdpscocfs1/gold-container/ProductSales",
            [HierarchicalNavigation = false]),
        [Version = 0])
in
    #"Product Sales"
```

A red box highlights the URL 'https://dlssdpscoc.dfs.core.windows.net/dlssdpscocfs1/gold-container/ProductSales'. A callout bubble points to it with the text 'Data Lake Storage Endpoint and container name needs to be changed'. Below the code, a green checkmark indicates 'No syntax errors have been detected.' At the bottom right are 'Done' and 'Cancel' buttons, with 'Done' being highlighted by a red box.

Notice that you can adjust the Version of the Delta table you are querying, by adjusting the Version parameter within the M-Code. This is identical to the time traveling we did in chapter [Time Traveling](#). Unfortunately, time traveling using a Timestamp is not currently supported by this connector.

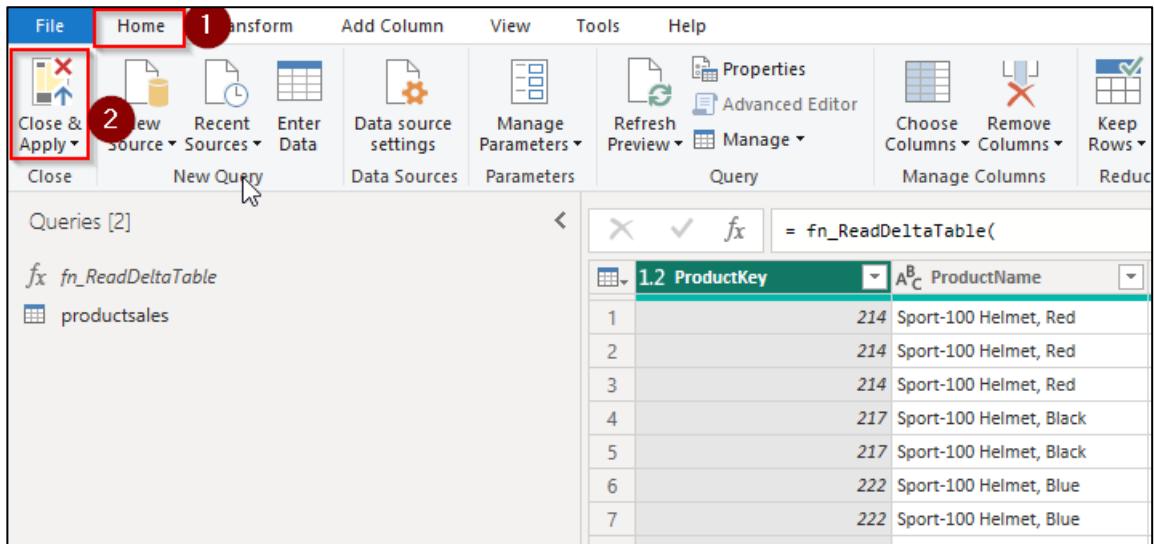
The screenshot shows the 'Advanced Editor' window with a query titled 'Query1'. The M-Code is as follows:

```
let
    #"Product Sales" = fn_ReadDeltaTable(
        AzureStorage.DataLake(
            "https://dlssdpscoc.dfs.core.windows.net/dlssdpscocfs1/gold-container/ProductSales",
            [HierarchicalNavigation = false]),
        [Version = 0])
in
    #"Product Sales"
```

A red box highlights the '[Version = 0]' parameter. A callout bubble points to it with the text 'The Version can be adjusted'. The 'Done' button at the bottom right is highlighted by a red box.

You can now import your data into Power BI.

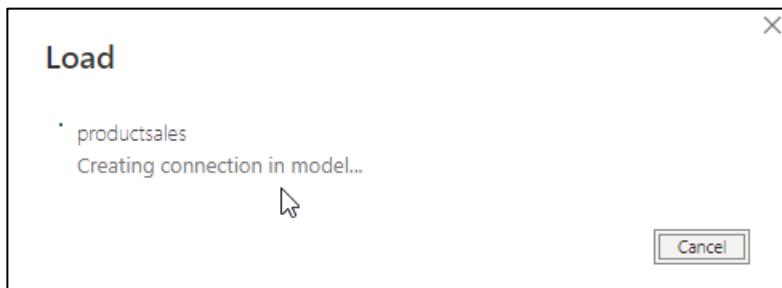
10. Select **Close & Apply** button within the Power Query Editor to load the data into Power BI



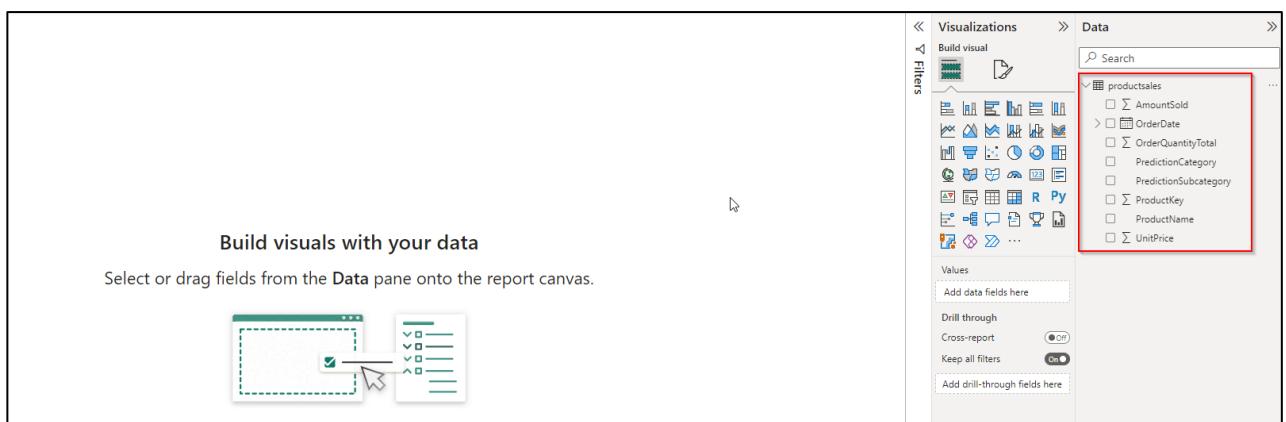
The screenshot shows the Power Query Editor interface. The 'Home' tab is selected (indicated by a red circle with '1'). The 'Close & Apply' button is highlighted with a red circle and labeled '2'. The 'Queries [2]' pane on the left lists 'fn_ReadDeltaTable' and 'productsales'. The main area displays a table with columns 'ProductKey' and 'ProductName'. The table data is as follows:

	A	B
	C	ProductName
1	214	Sport-100 Helmet, Red
2	214	Sport-100 Helmet, Red
3	214	Sport-100 Helmet, Red
4	217	Sport-100 Helmet, Black
5	217	Sport-100 Helmet, Black
6	222	Sport-100 Helmet, Blue
7	222	Sport-100 Helmet, Blue

14. A loading screen will appear.



15. The data is now imported and available to be used for visualizations in Power BI.



The screenshot shows the Power BI desktop interface. The 'Data' pane on the right is open, showing the 'productsales' table with fields: ProductKey, ProductName, UnitPrice, OrderDate, OrderQuantityTotal, PredictionCategory, and PredictionSubcategory. A red box highlights the 'productsales' table in the Data pane. The main canvas area has a placeholder message 'Build visuals with your data'.

11 References

- [Browse Azure Architectures - Azure Architecture Center | Microsoft Learn](#)
- [What is the medallion lakehouse architecture? - Azure Databricks | Microsoft Learn](#)
- [Medallion architecture: best practices for managing Bronze, Silver and Gold | by Piethein Strengtholt | Medium](#)
- [The Fundamentals of Delta Lake | Medium](#)
- [Delta lake ETL with data flows - Azure Data Factory | Microsoft Learn](#)
- [Building the Lakehouse - Implementing a Data Lake Strategy with Azure Synapse - Microsoft Community Hub](#)
- [What is a Lake Database in Azure Synapse Analytics? \(endjin.com\)](#)
- [Microsoft BI Tools: Synapse - Add existing Delta Table to Lake Database \(microsoft-bitools.blogspot.com\)](#)
- [What is the difference between "SQL Database" and "Lake Database" in Synapse - Microsoft Q&A](#)
- [Azure Synapse Analytics | Spark pool | Delta Lake - Part 1 - YouTube](#)
- [Microsoft BI Tools: Synapse - Using Spark SQL to time travel Delta Tables \(microsoft-bitools.blogspot.com\)](#)
- [Synapse Espresso: Timetravel with Delta tables in Azure Synapse Spark! - YouTube](#)
- [Azure Synapse Analytics | Spark pool | Delta Lake - Part 1 - YouTube](#)
- [Catching Up With Delta Lake in Azure Synapse - Iteration Insights](#)
- [SQL language reference | Databricks on AWS](#)