Azure Cloud Scale Analytics with ADX – Lab 1

Lab Summary

Land files at scale (to a maximum of 100 per second) to Azure Blob, triggering automatic file ingestion into ADLS Gen-2. File ingestion is triggered via blob create events that Azure Event Grid detects and calls a parameterised ADF pipeline. In this lab data will be queried via an external table reference from ADX to ADLS**.** In subsequent labs data will be ingested into ADX.

Azure Services

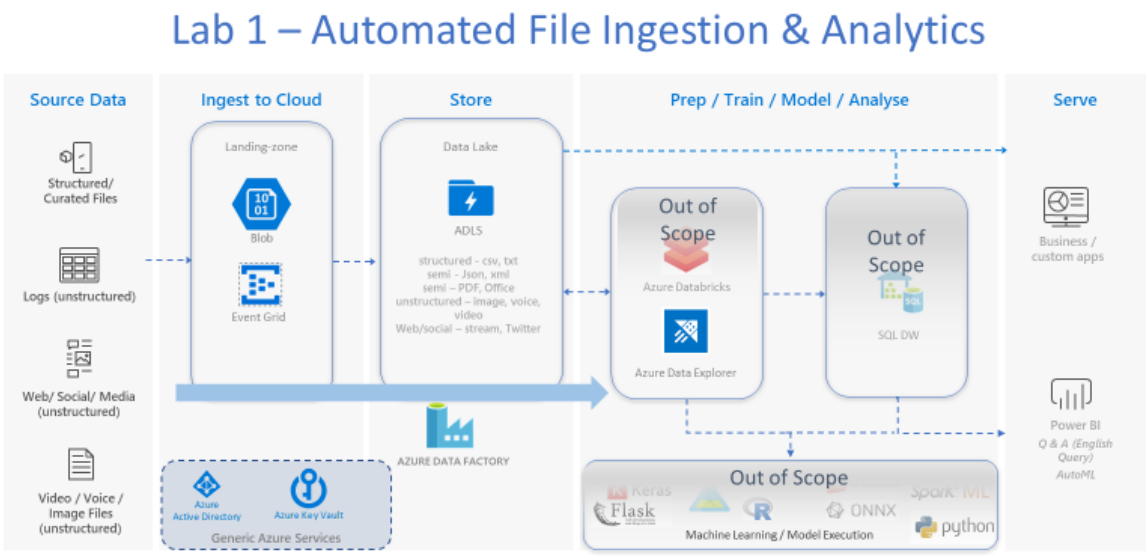
In addition to the Azure services mentioned in 'Core Azure Services' Lab1 introduces the following Azure services into the architecture:

Azure Data Factory (ADF) [link]

Azure Event Grid [link]

Lab Architecture

The diagram below shows the architecture that will be built in Lab 1.



Microsoft

Activities

The lab is broken down into several logical units, as follows:

- Activity 1 – Azure Blob, Azure Data Lake (ADLS)
  - Build these Azure services
- Activity 2 - Key Vault, Event Grid
  - Build these Azure services
- Activity 3 - Data Factory
  - Build these Azure services
- Activity 4 – Azure Data Explorer (ADX)
  - Build these Azure services
- Activity 5 – Security & Access

    - Configure storage SAS keys
- Activity 6 – ADF Pipeline
    - Build an ADF pipeline, triggered by Event Grid
- Activity 7 – Execute ADF Pipeline
    - Run the ADF pipeline with sample data
- Activity 8 – ADX integration with ADLS
    - Write KQL queries to analyse data

The first five lab activities are associated with building the environment that will be used in this and subsequent labs. Lab activities 6 & 7 build and execute an ADF pipeline associated with the file ingestion. Lab activity 8 gives a summary example of dynamic analytics with data residing in a data lake.

Activity 1 – Azure Blob, Azure Data Lake

As per the diagram "Lab1 – Activity 1 – Blob – Data Lake.pdf" we are creating an Azure Blob Storage account and an Azure Data Lake (Gen-2) account. Firstly, we will create the landing zone for all files to be dropped to.

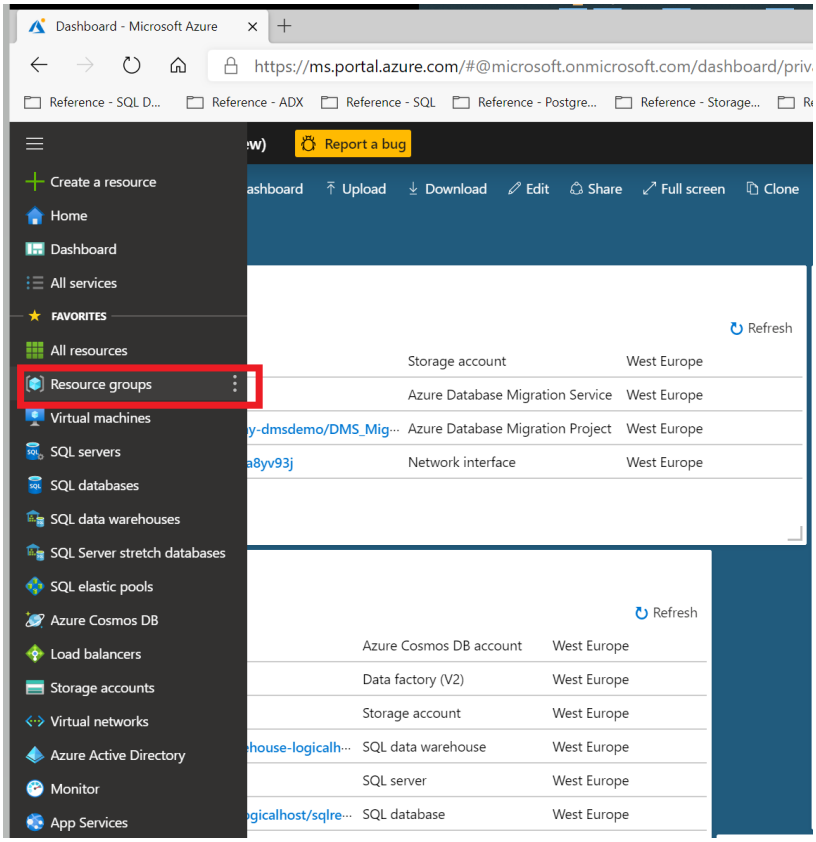
NOTE: *This lab can be further enhanced once event processing is supported on Azure Data Lake. This feature is not GA currently. Details here.*

Activity 1.1 - Create the Resource Group 'ADX'.

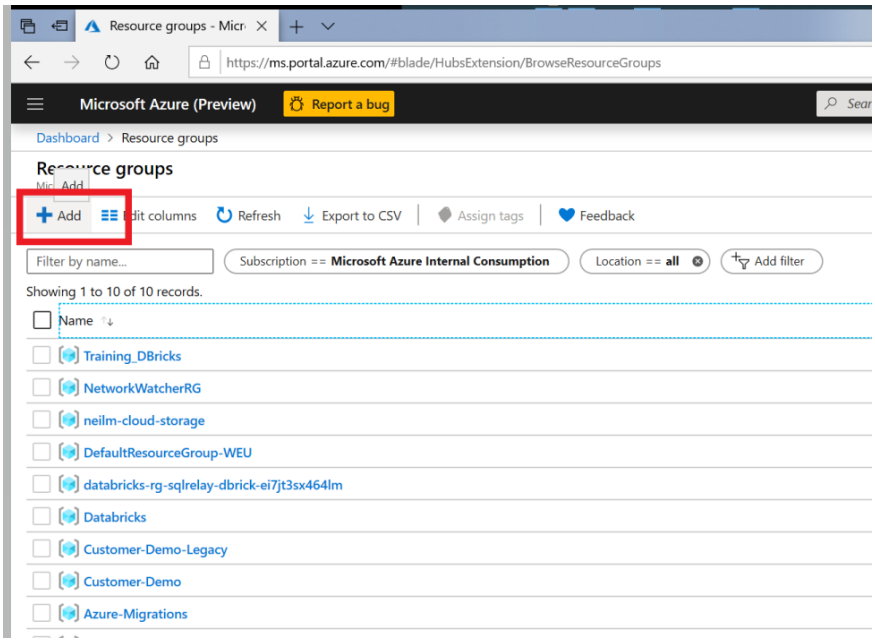
Define the Azure Resource Group that all the Lab 1 Azure Services will be created in.

NOTE: *Creating the Resource Group can also be achieved via PowerShell here.*

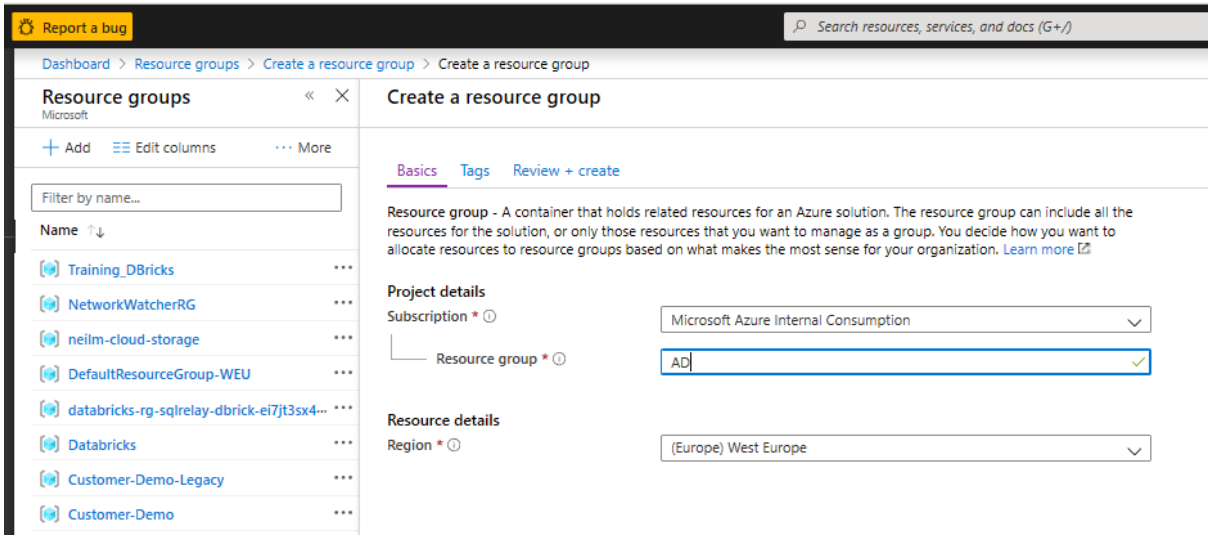
1. Sign in to the Azure portal.
2. In the Azure portal, select **Resource Groups** and the **ADX** resource group you have created above. Select **Add** (see below):



3. When the blade loads, select **Add:**



4. Enter the Resource Group Details:



Property	Description	Required
Subscription	The value must be set to the subscription the ADX resource group is to be created in.	Yes
Resource Group	The name of the workshop resource group. Enter **ADX**.	Yes
Region	Enter the Azure region your resource group is to be located in. This should be a region that supports the Azure services required in this workshop and located as close as possible to your geographic location.	Yes

5. Select **Create.**

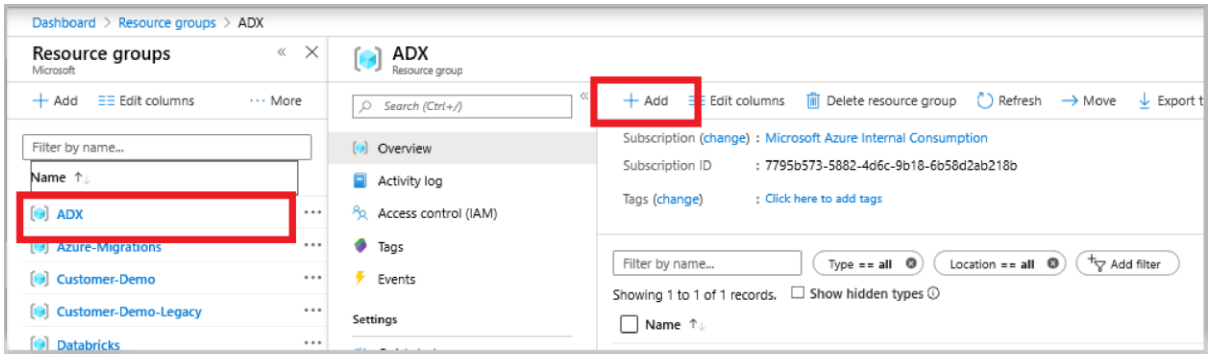
After a few minutes you will a resource group ADX into which the 'Lab1 Automated File Ingestion & Analytics' architecture can be deployed into.

Activity 1.2: Create the Blob storage account

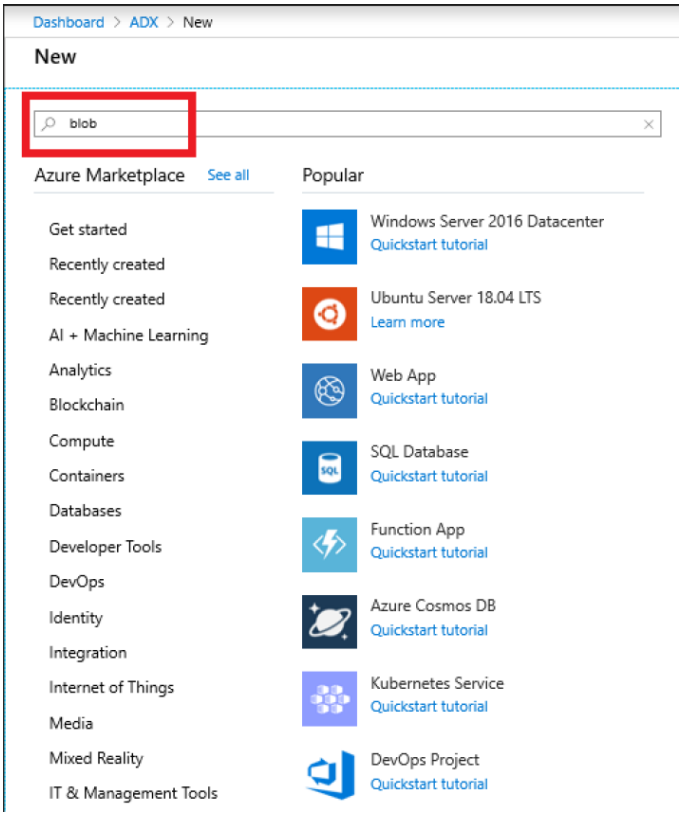
The blob storage account will act as the *landing zone* for files that will then be automatically ingested.

An explanation of Azure Blob storage is here: [link]

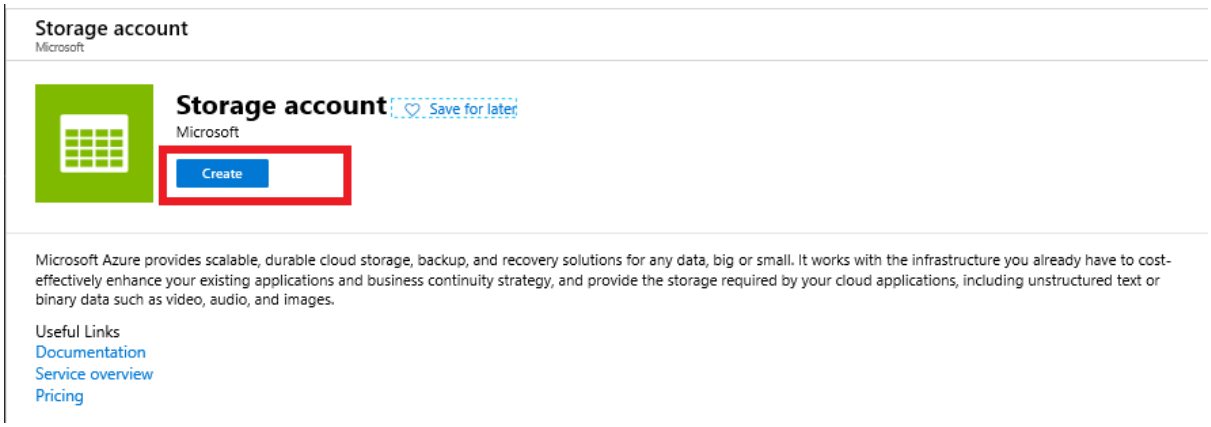
1. Within the ADX resource Group, select **+ Add** or **Create Resource**



2. Type **blob** or **storage** in the search window (see below) and then select **Storage Account** from the results displayed:

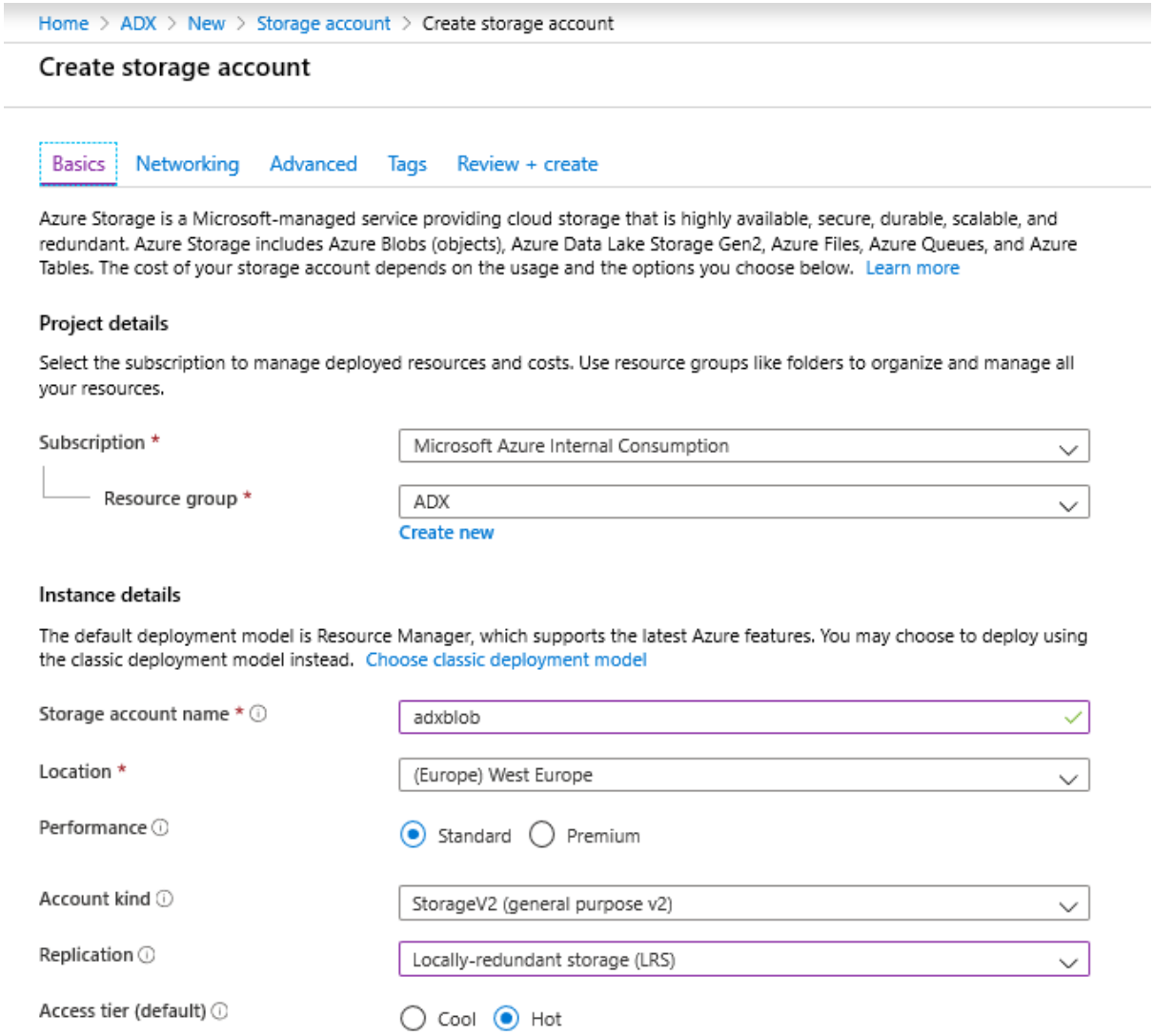


3. On the Storage Account blade, select **Create**:



4. On the **Create storage account** blade, complete the Basics blade (see below):

5.

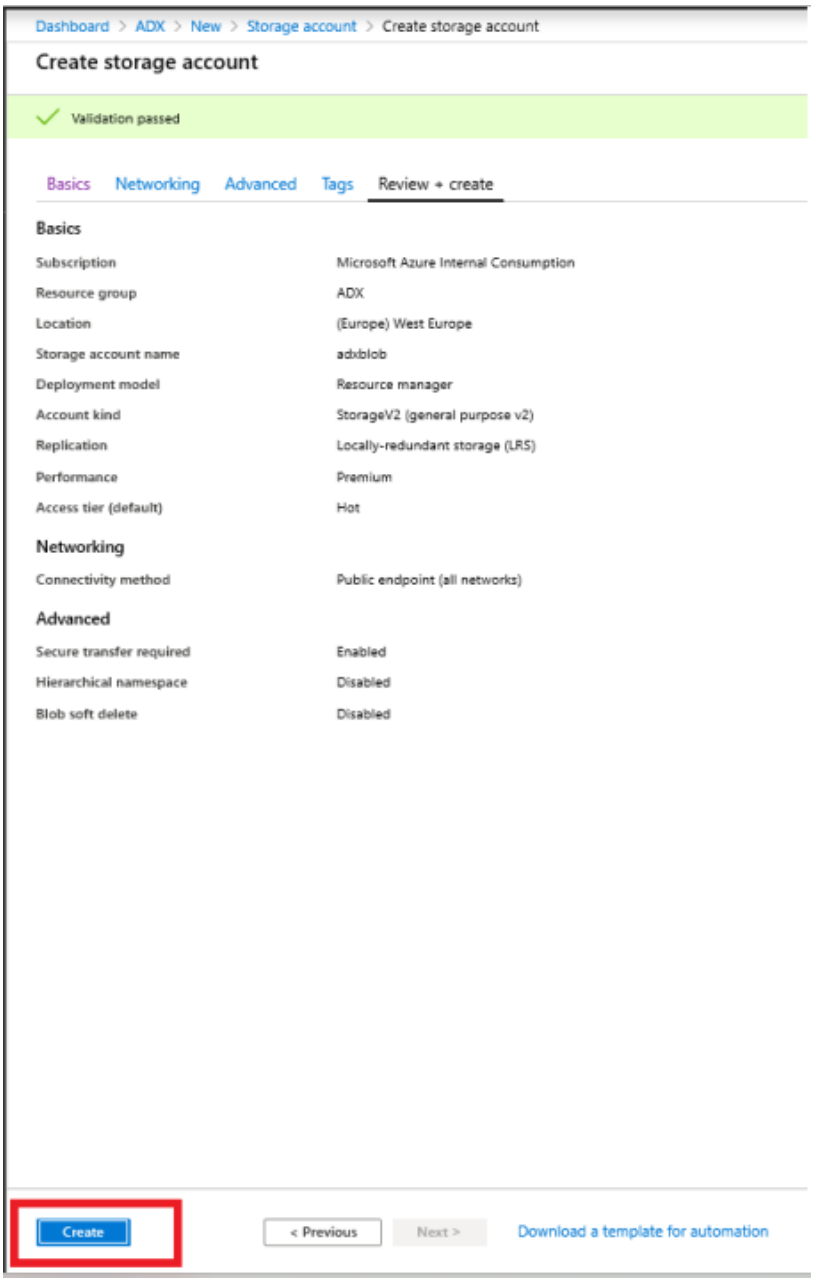


Complete the blade by entering the details as below:

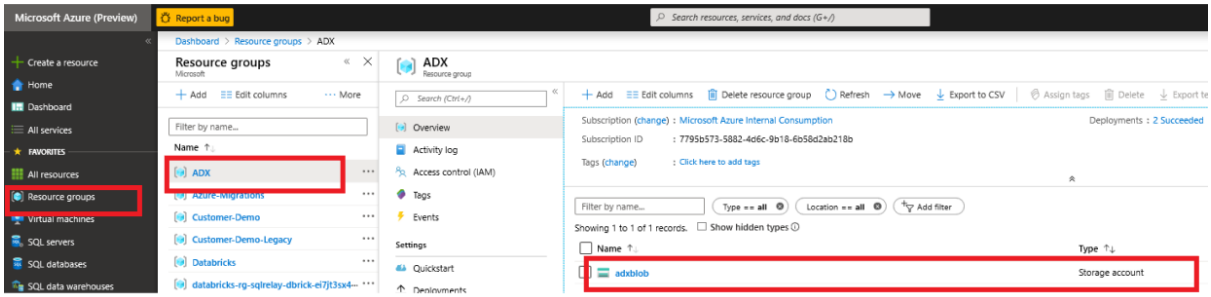
Property	Description	Required
Subscription	The value must be set to the subscription the ADX resource group is created in.	Yes
Resource Group	The name of the workshop resource group. Enter **ADX**.	Yes
Storage account name	Enter **adxblob**	Yes
Location	Select your chosen region to match the resource group ADX you have created.	Yes

Property	Description	Required
Performance	Enter **Standard**	Yes
Account Kind	Select **Storage v2 (general purpose v2)**	Yes
Replication	Select **Locally redundant storage (LRS)**	Yes

5. Select **Review + Create**. The blade below will appear, select **Create**:



After a short wait you will be notified that the blob storage has been created. You can go to the resource to review its setup or select **Resource Groups**, then select **ADX**, to see the ***adxblob*** resource you have just created:



This resource template can be found in the Git Repository as 'ExportedTemplate_adxblob.zip'

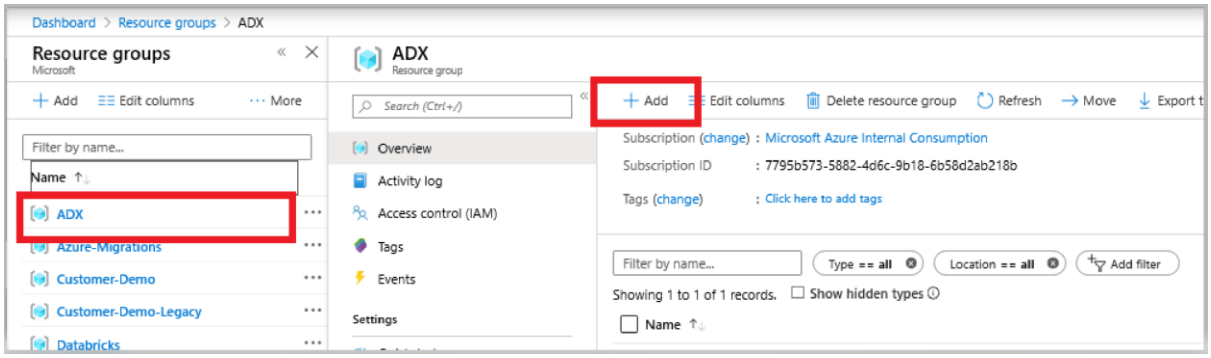
Activity 1.3 - Create the data lake storage account

The data lake storage account will serve as the *data lake* for both this and all the other labs within this workshop.

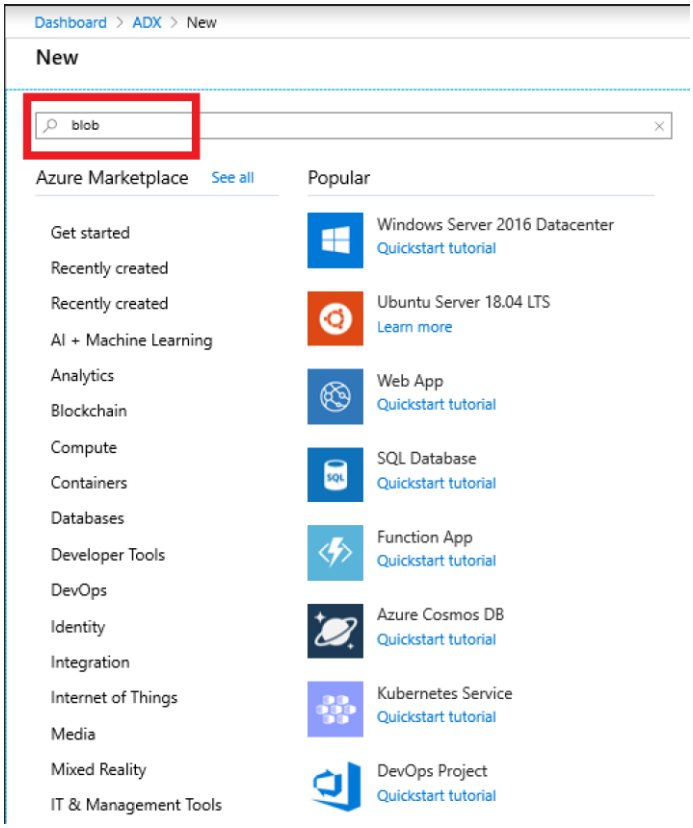
An explanation of Azure Data Lake (ADLS) resource is here [link].

NOTE: *the ADLS storage account is created via the same steps as the **adxblob** blob storage account. For completeness all the steps are included to below although there is some repetition to 'Activity 1.2 - Create the Blob storage account'.*

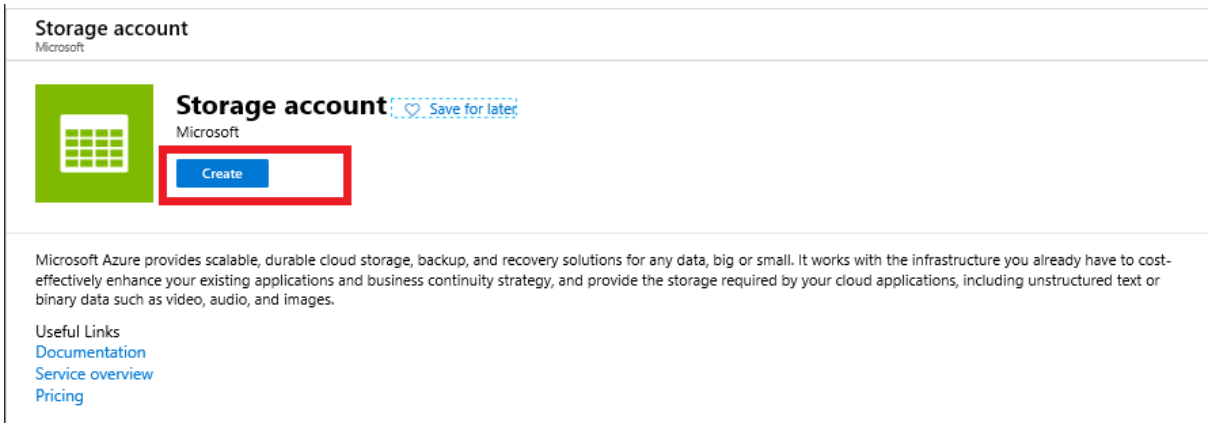
1. Within the ADX resource Group, select **+ Add** or **Create Resource**



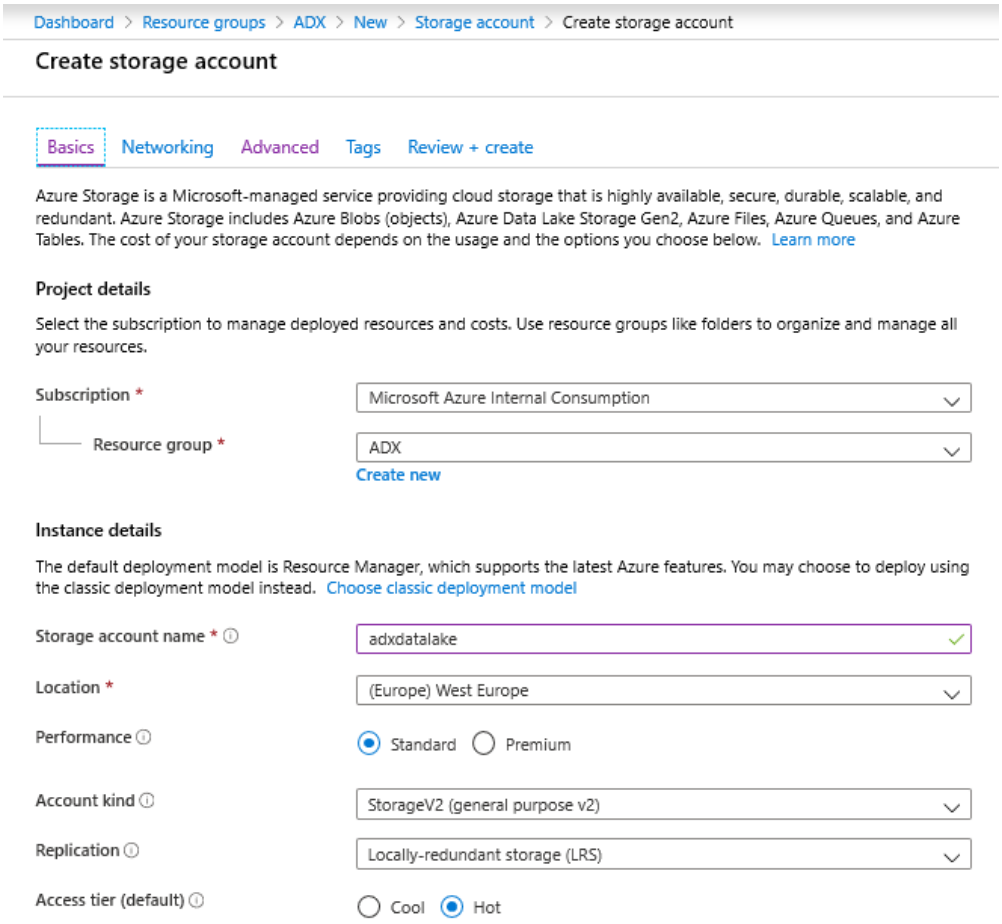
2. Type **blob** or **storage** in the search window (see below) and then select **Storage Account** from the results displayed:



3. On the Storage Account blade, select **Create**:



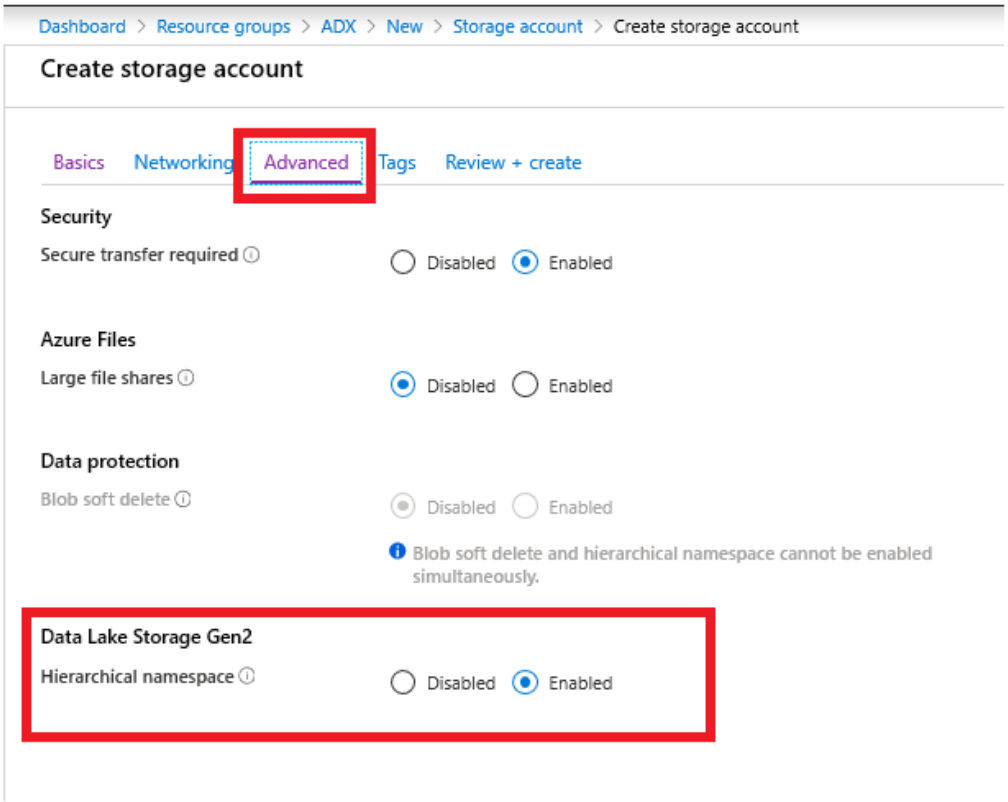
4. On the **Create storage account** blade, complete the Basics blade (see below)



Complete the blade by entering the details as below:

Property	Description	Required
Subscription	The value must be set to the subscription the ADX resource group is created in.	Yes
Resource Group	The name of the workshop resource group. Enter **ADX**.	Yes
Storage account name	Enter **adxblob**	Yes
Location	Select your chosen region to match the resource group *ADX* you have created.	Yes
Performance	Enter **Standard**	Yes
Account Kind	Select **Storage v2 (general purpose v2)**	Yes
Replication	Select **Locally-redundant storage (LRS)**	Yes

5. Select **Advanced**, this is where the blob storage account had the ADLS data lake attributes enabled:

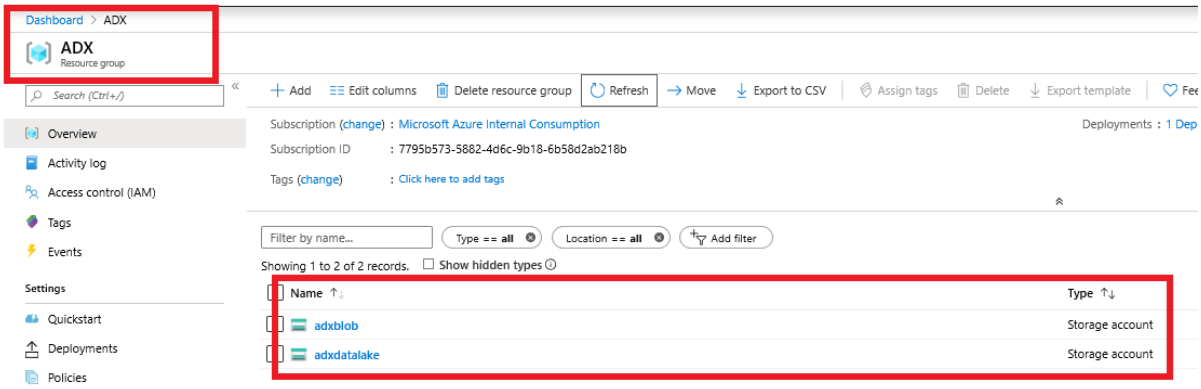


Complete the blade by entering the details as below:

Property	Description	Required
Security	Select **Enabled**	Yes
Azure Files	Select **Disabled.** The purpose of this Lab is to create an architecture to handle files landing on the ***adxblob*** storage account (100's per second if necessary). As a result, these should be relatively small files (KBs to 1MB). Large files should be ingested via different architectural patterns.	Yes
Data Lake Storage Gen2	Enter **adxblob**	Yes

6. Select **Review + Create**. The blade below will appear, select **Create**:

After a short wait you will be notified that the adls storage has been created. You can go to the resource to review its setup or select **Resource Groups**, then select **ADX**, to see the ***adxdatalake*** resource you have just created:



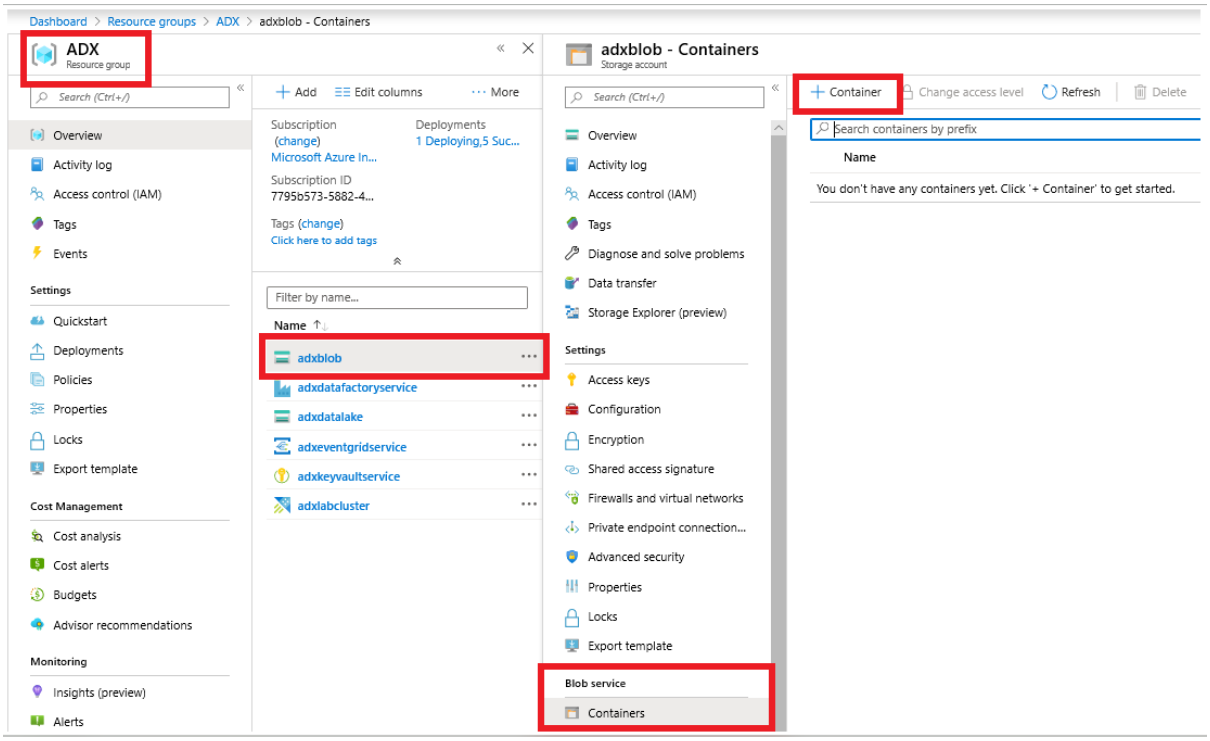
Notice that both ***adxblob*** and ***adxdatalake*** have the Type: Storage Account. Make certain you are familiar with the Azure Data Lake (ADLS) and Azure Blob services.

This ***adxdatalake*** resource template can be found in the Git Repository as 'ExportedTemplate_adxdatalake.zip'.

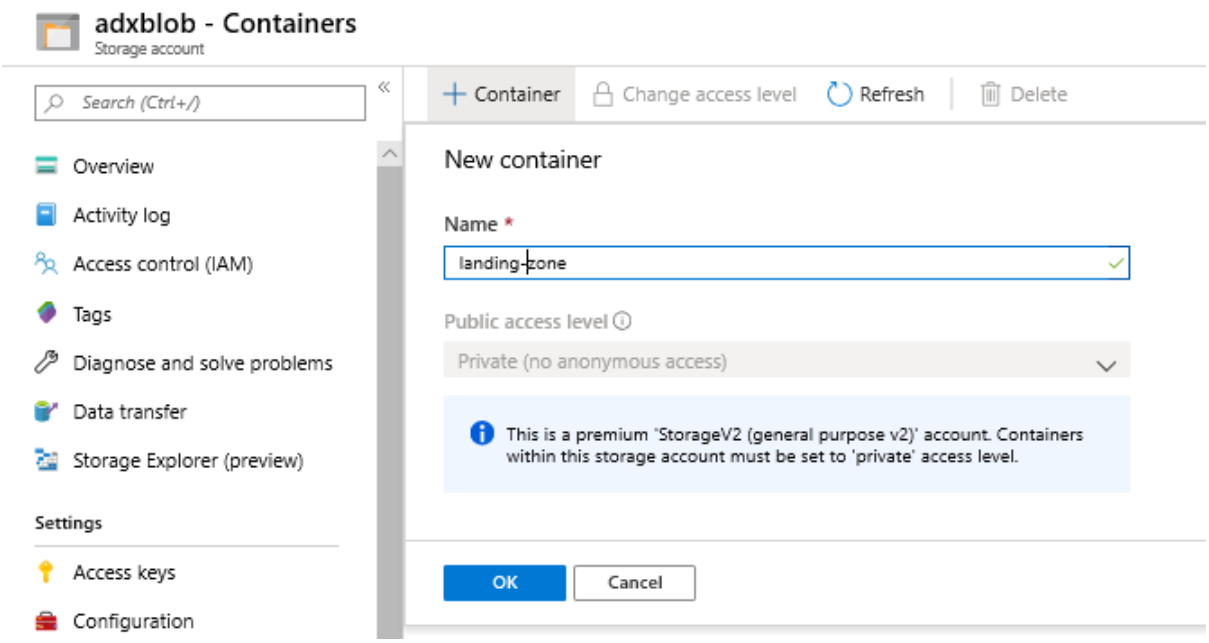
Activity 1.4: Configure the Blob storage

A container is required for the files to be landed to from upstream systems and services. The Azure Event Grid will generate *blob created* events that the Azure Data Factory will utilise during it pipeline execution.

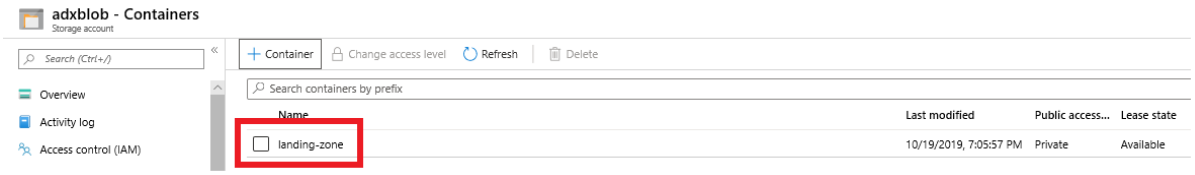
1. Select the **ADX** resource group in the Azure portal
2. Select ***adxblob*** storage account
3. Select **Containers** underneath *Blob Service*
4. Select **+ Container** to add a new container, as below:



5. Enter **landing-zone** and select **Ok:**



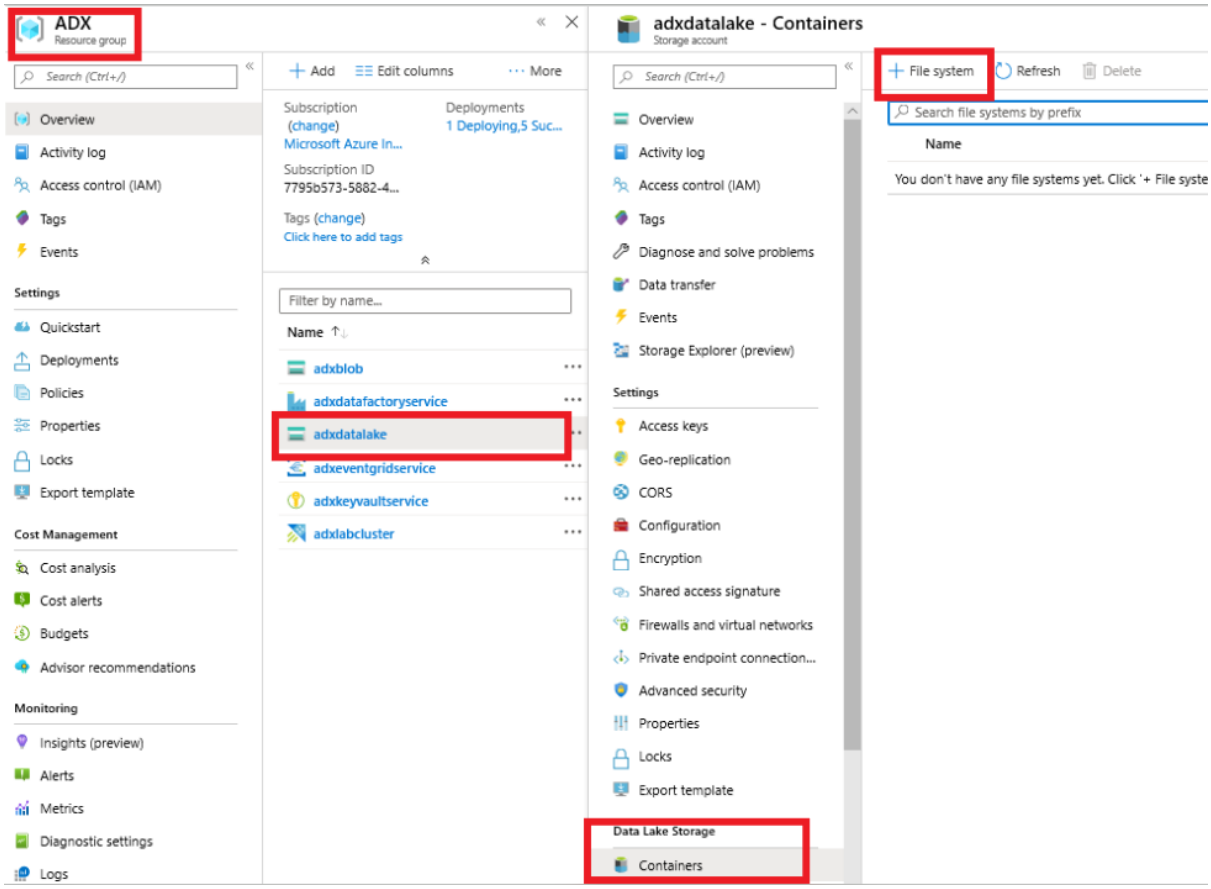
The *landing-zone* container will be created:



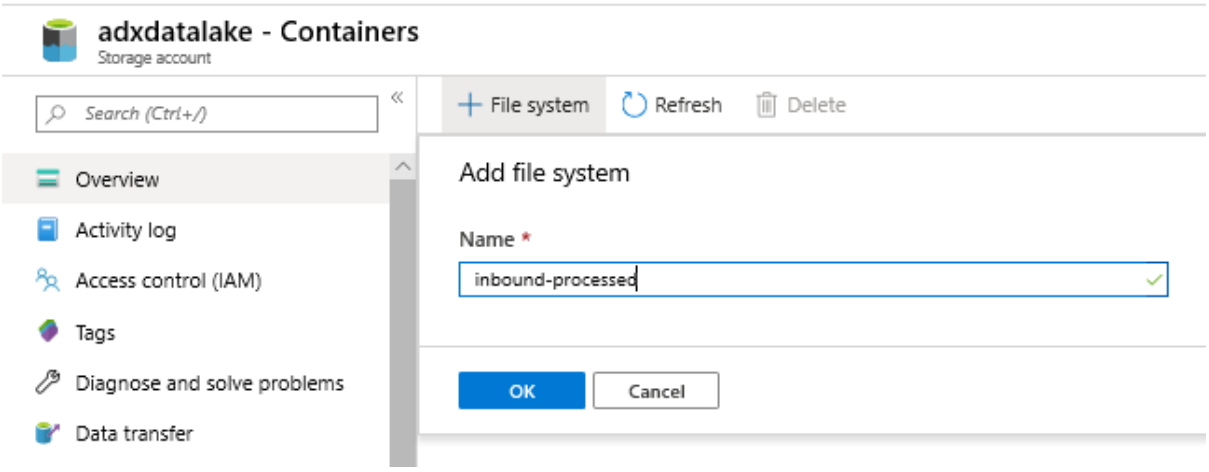
Activity 1.5: Configure the ADLS storage

A root container is required for files to be landed to in a hierarchical structure. Files will be landed to ADLS via an Azure Data Factory Pipeline activity.

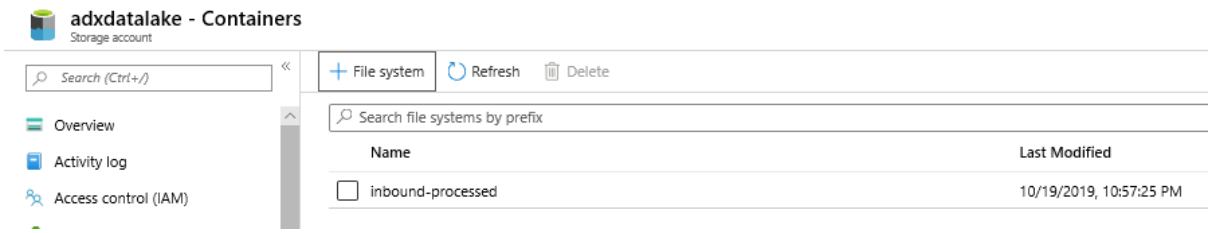
1. Select the **ADX** resource group in the Azure portal
2. Select ***adxdatalake*** storage account
3. Select **Containers** underneath *Blob Service*
4. Select **+ File System** to add a new container, as below:



5. Enter **inbound-processed** and select **OK:**



The inbound-processed container will be created:



Activity 2 – Key Vault, Event Grid

As per the diagram "Lab1 - Activity 2 – Key Vault – Event Grid.pdf" we are creating an Azure Key Vault to securely store access account and credential information. Additionally, we are creating an Azure Event Grid to process a file(s) that lands on the blob storage ***adxblob***.

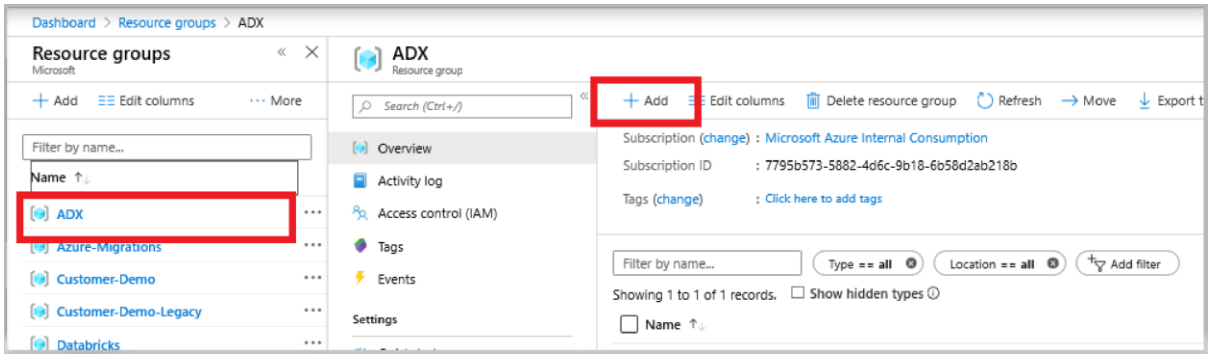
An explanation of Azure Key Vault is here [link]

An explanation of Azure Event Grid is here [link]

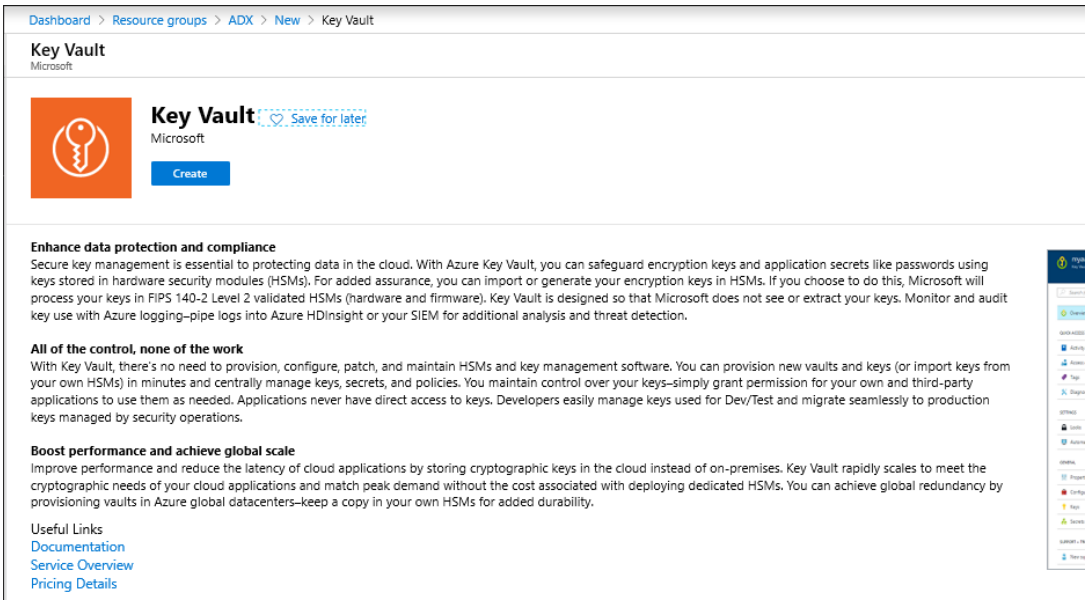
Activity 2.1 – Create the Azure Key Vault service

We will create an Azure KeyVault for use in future labs.

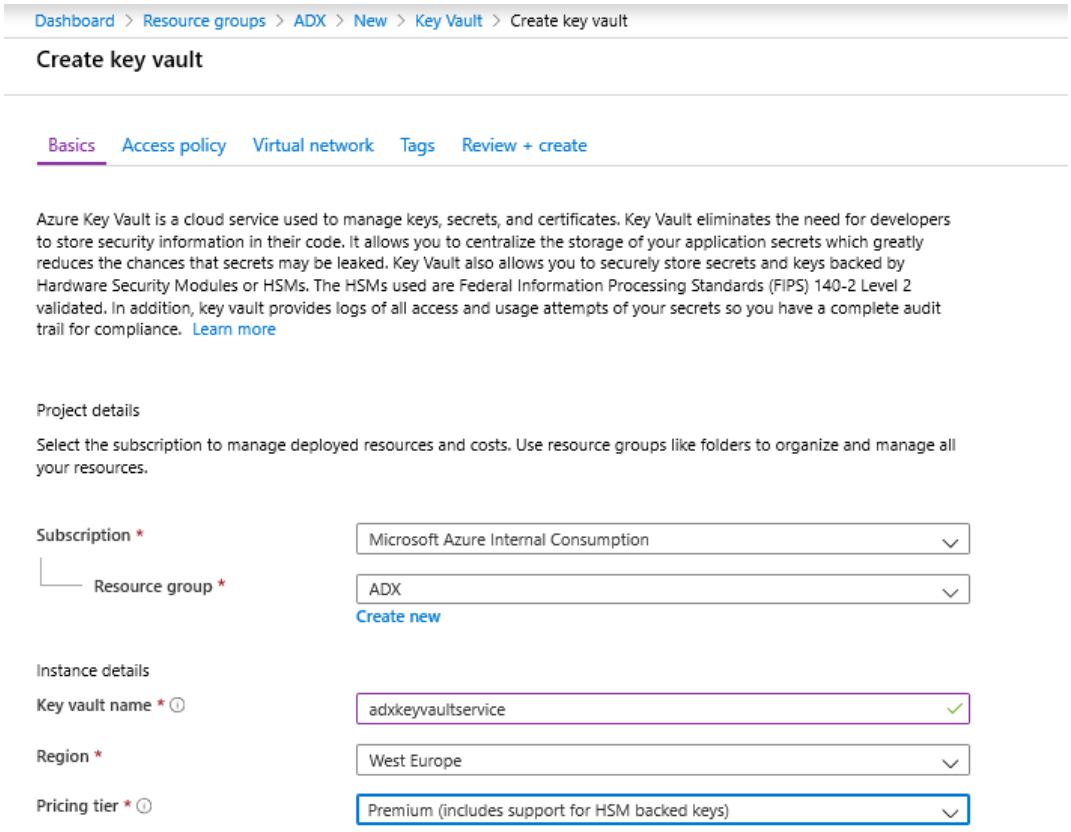
1. Within the ADX resource Group, select **+ Add** or **Create Resource**



2. Type **keyvault** in the search window (see below) and then select **Key Vault** from the results displayed. The following blade will appear:



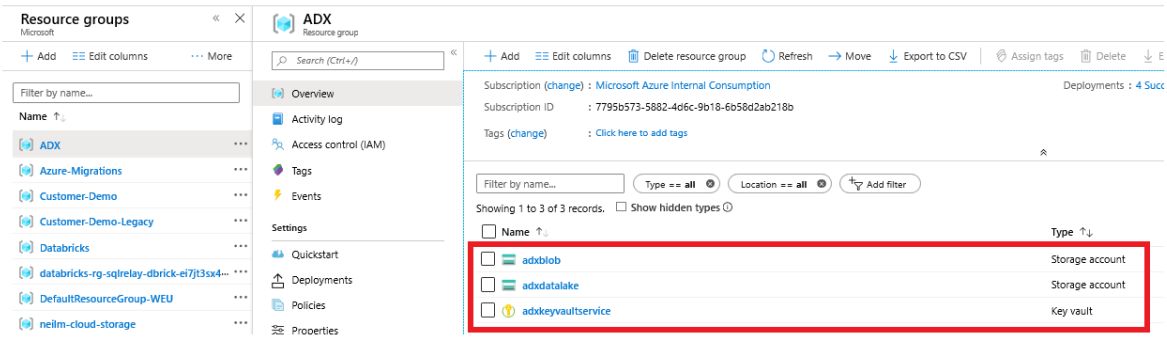
3. Select **Create**
4. On the Basics tab, enter details as below:



Property	Description	Required
Subscription	The value must be set to the subscription the ADX resource group is created in.	Yes
Resource Group	The name of the workshop resource group. Enter **ADX**.	Yes
Key vault name	Enter **adxkeyvaultservice**	Yes
Region	Select your chosen region to match the resource group *ADX* you have created.	Yes
Pricing Tier	Enter **Premium**	Yes

5. Select **Review + Create**. Then select **Create**:

After a short wait you will be notified that the Azure Key Vault has been created. You can go to the resource to review its setup or select **Resource Groups**, then select **ADX**, to see the **adxkeyvaultservice** resource you have just created:



This **adxkeyvaultservice** resource template can be found in the Git Repository as 'ExportedTemplate_adxkeyvaultservice.zip'.

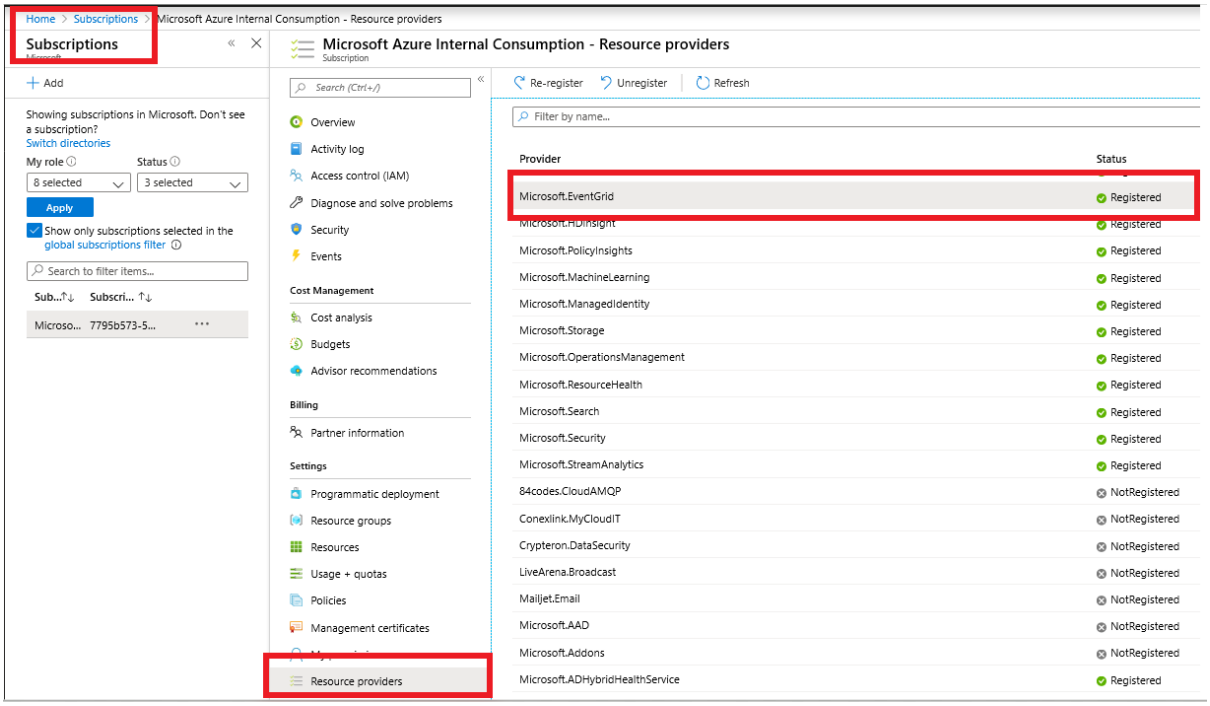
Activity 2.2 – Create the Azure Event Grid service

If you haven't previously used Event Grid in your Azure subscription, you may need to register the Event Grid resource provider.

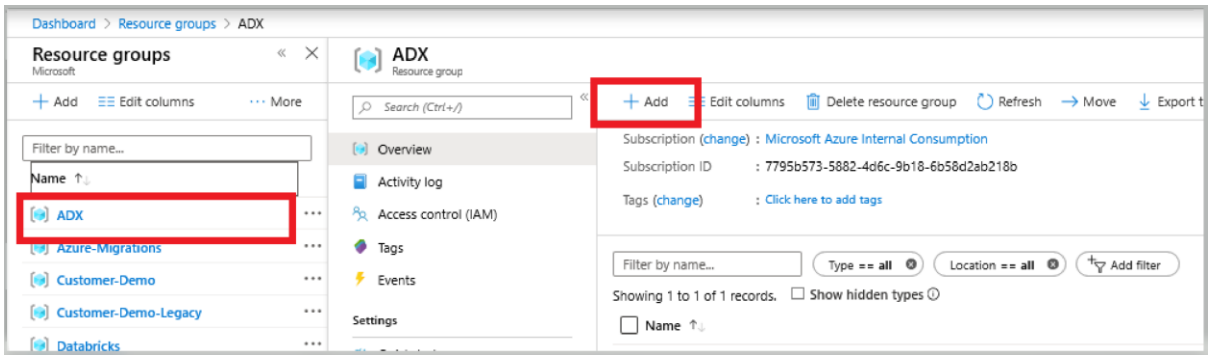
In the Azure portal:

1. Select **All Services**
2. Select **Subscriptions**
3. Select the subscription you're using for this workshop/lab
4. Select **Resource providers**.
5. Find **Microsoft.EventGrid**.
6. If not registered, select **Register**.

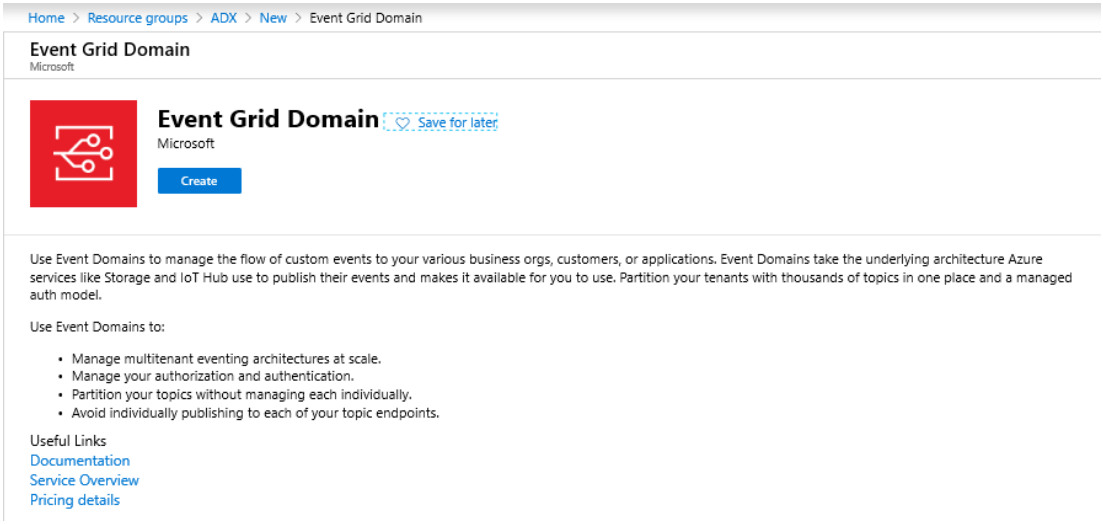
It may take a moment for the registration to finish. Select **Refresh** to update the status. When **Status** is **Registered**, you're ready to continue.



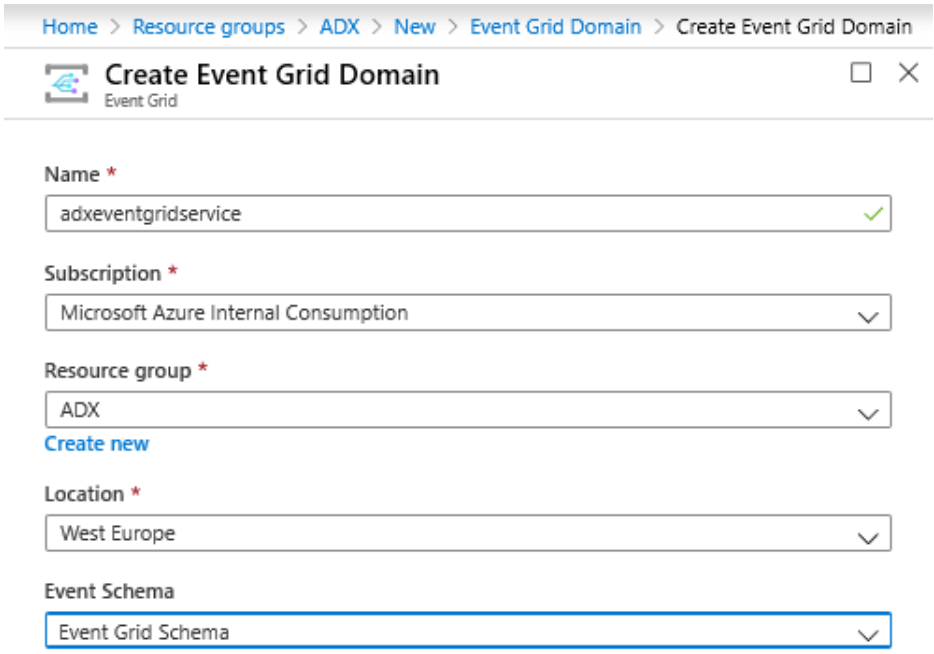
6. Within the ADX resource Group, select **+ Add** or **Create Resource**



7. Type **event grid domain** in the search window (see below) and then select **Event Grid** Domain from the results displayed. The following blade will appear:



8. Select **Create** and the following blade will appear:

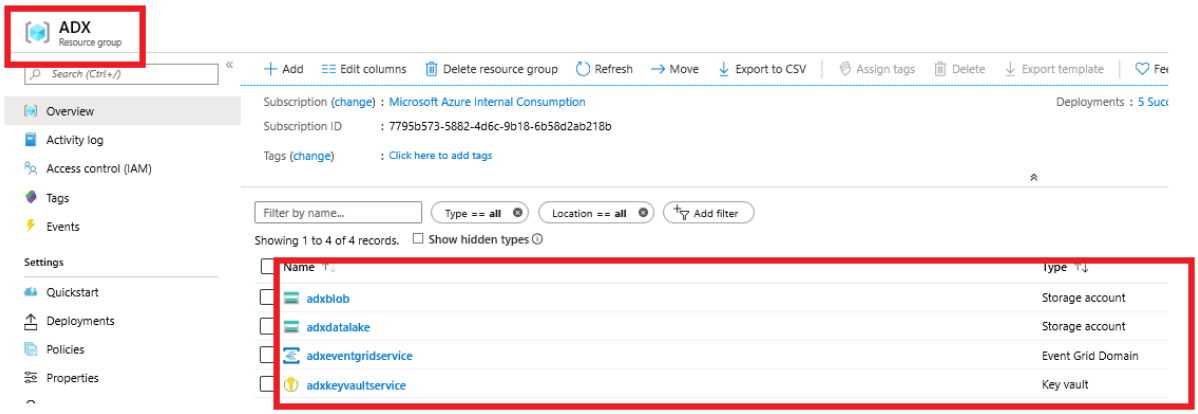


Complete the blade by entering the details as below:

Property	Description	Required
Name	Enter adxeventgridservice	Yes
Subscription	The value must be set to the subscription the ADX resource group is created in.	Yes

Property	Description	Required
Resource Group	The name of the workshop resource group. Enter **ADX**.	Yes
Region	Select your chosen region to match the resource group *ADX* you have created.	Yes
Event Schema	Enter **Event Grid Schema**	Yes

9. Select **Create.** After a short wait you will be notified that the Azure Event Grid has been created. You can go to the resource to review its setup or select **Resource Groups**, then select **ADX**, to see the ***adxeventgridservice*** resource you have just created:



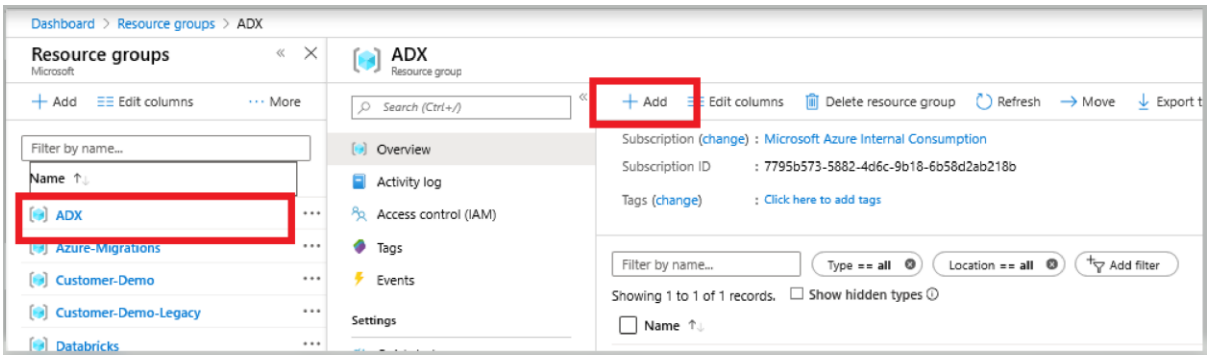
This **adxeventgridservice** resource template can be found in the Git Repository as 'ExportedTemplate_adxeventgridservice.zip'.

Activity 3 – Data Factory

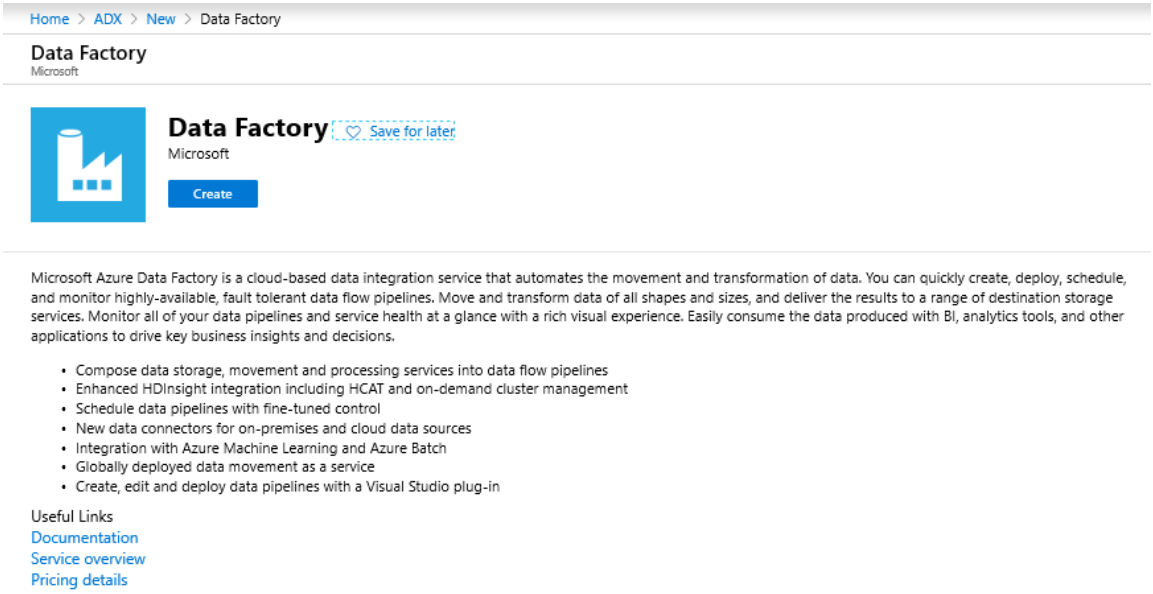
As per the diagram "Lab1 - Activity 3 – Data Factory.pdf" we are creating an Azure Data Factory (ADF). The ADF activity will be triggered by a file creation on the ***adxblob*** which will be detected via the Azure Event Grid. The ADF pipeline will move the file to the ***adxdatalake*** and deposit it in a hierarchical structure based on the date of the file creation.

An explanation of Azure Data Factory (ADF) is here [link].

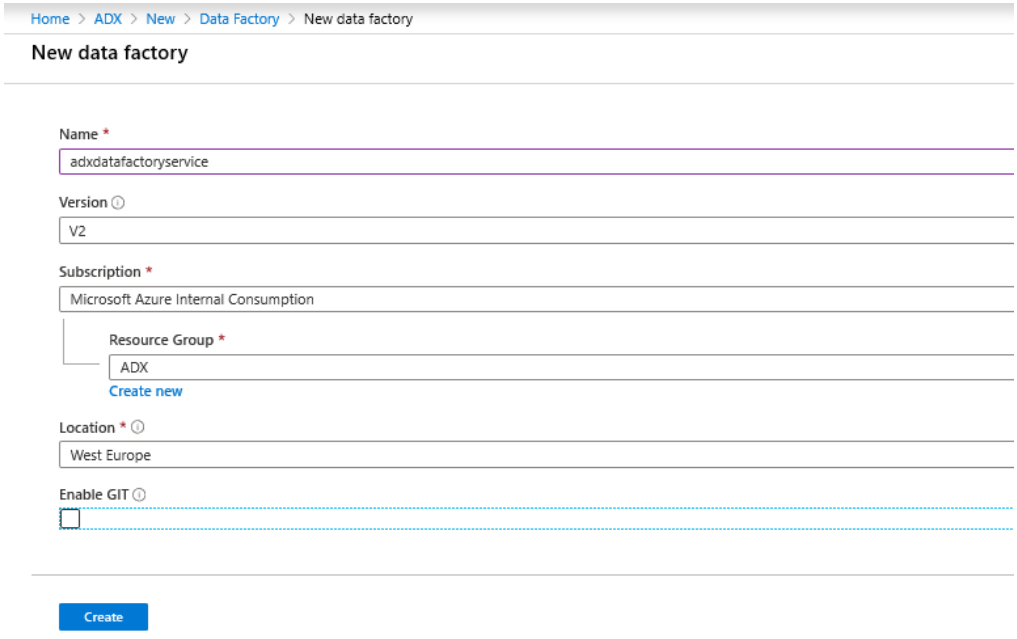
1. Within the ADX resource Group, select **+ Add** or **Create Resource**



2. Type **data factory** in the search window (see below) and then select **Data Factory** from the results displayed. The following blade should be displayed:



3. Select **Create**, the **New data factory** blade will appear:



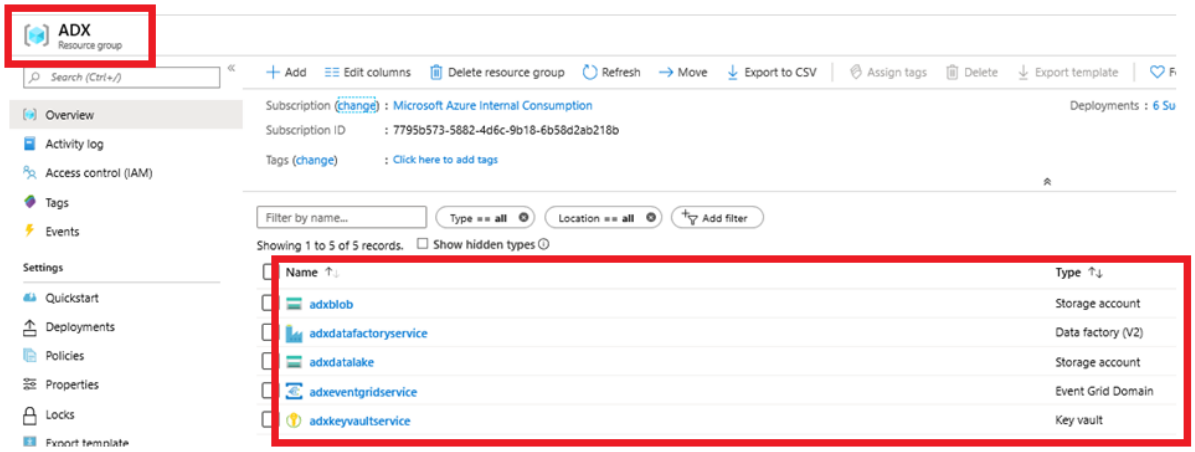
Complete the blade by entering the details as below:

Property	Description	Required
Name	Enter **adxdatafactoryservice**	Yes
Version	Enter **V2**	Yes
Subscription	The value must be set to the subscription the ADX resource group is created in.	Yes
Resource Group	The name of the workshop resource group. Enter **ADX**.	Yes
Region	Select your chosen region to match the resource group *ADX* you have created.	Yes
Enable GIT	Uncheck this box. GIT integration allows you to utilise in a DevOps CI/CD environment. This integration is beyond the scope of this workshop, so will not be configured. Further details can be found here if this is of interest. https://docs.microsoft.com/en-us/azure/data-factory/continuous-integration-deployment	Yes

NOTE: *GIT integration allows you to utilise in a DevOps CI/CD environment. This integration is beyond the scope of this workshop, so will not be configured. Further details can be found here - https://docs.microsoft.com/en-us/azure/data-factory/continuous-integration-deployment.*

GIT integration maybe the subject of a future lab within this workshop.

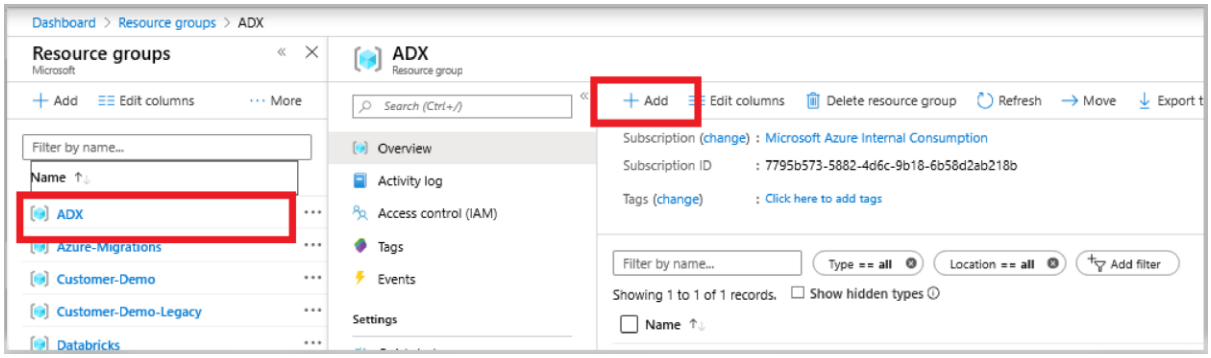
4. Select **Create** to build the ADF. After a short wait you will be notified that the Azure Data Factory has been created. You can go to the resource to review its setup or select **Resource Groups**, then select **ADX**, to see the ***adxdatafactoryservice*** resource you have just created:



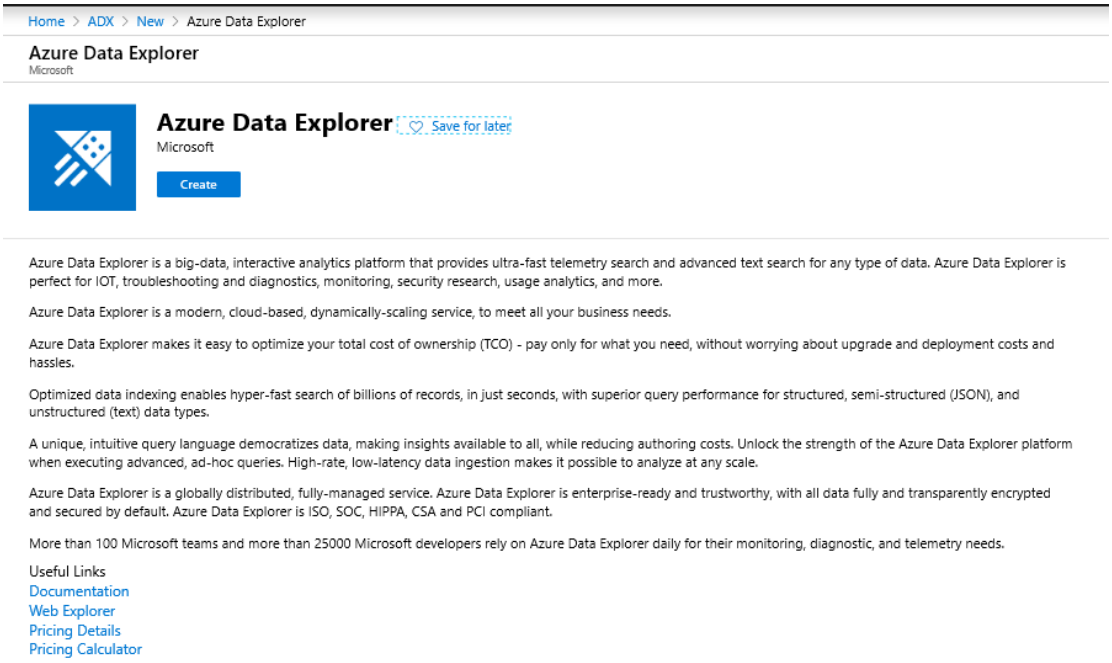
Activity 4 – Azure Data Explorer (ADX)

As per the diagram "Lab1 - Activity 4 – Azure Data Explorer.pdf" we are creating the ADX cluster.

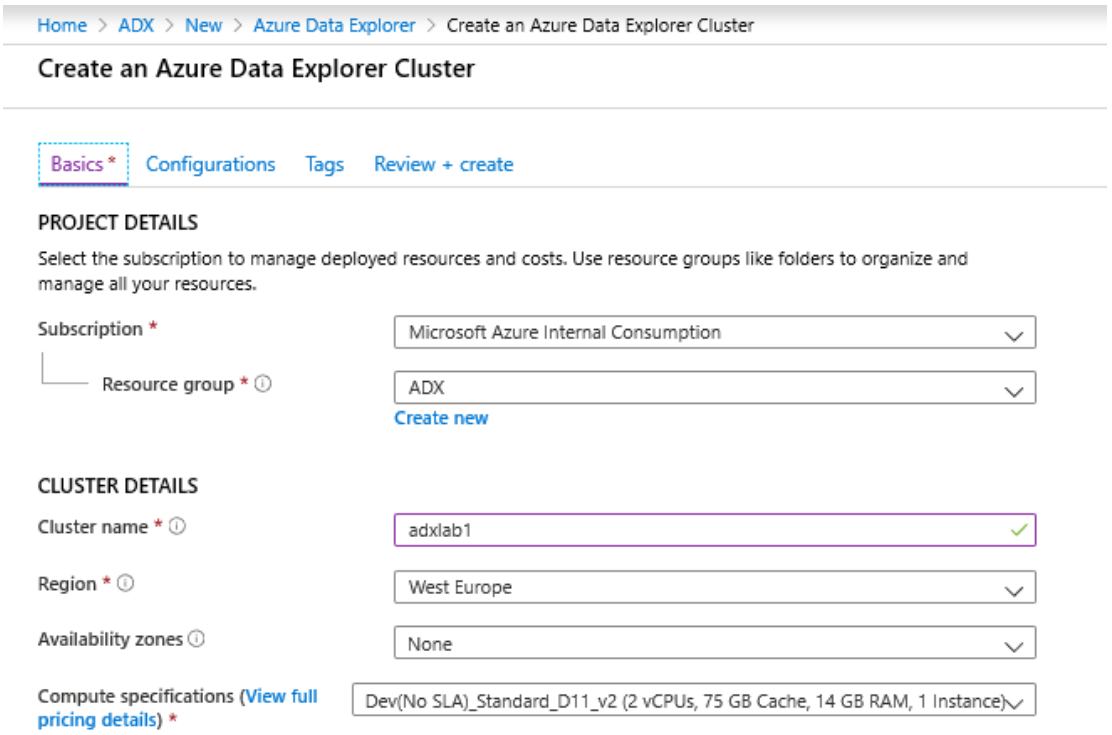
1. Within the ADX resource Group, select **+ Add** or **Create Resource**



2. Type **data explorer** in the search window (see below) and then select **Azure Data Explorer** from the results displayed. The following blade will appear:



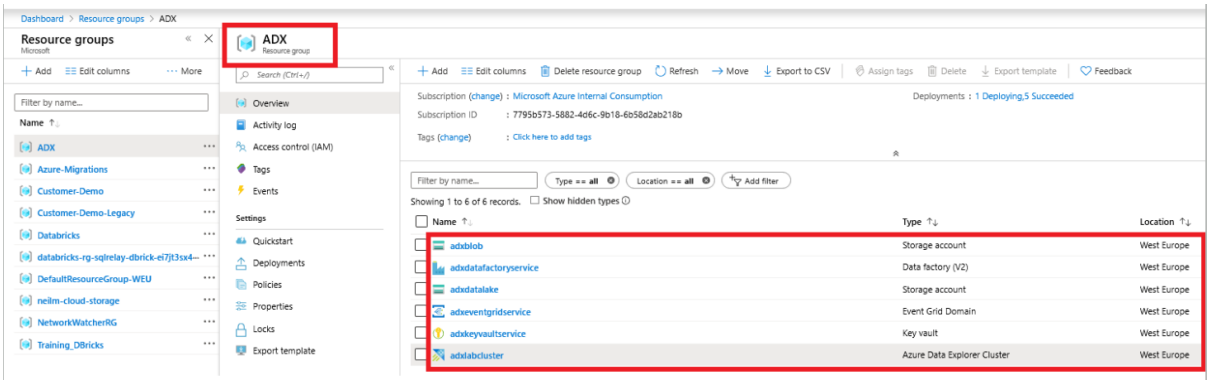
3. Select **Create** and the following blade will appear:



Complete the **Basics** blade by entering the details as below:

Property	Description	Required
Subscription	The value must be set to the subscription the ADX resource group is created in.	Yes
Resource Group	The name of the workshop resource group. Enter **ADX**.	Yes
Name	Enter **adxlabcluster**	Yes
Region	Select your chosen region to match the resource group *ADX* you have created.	Yes
Availability zones	Select **None**	Yes
Compute specifications	The options available in this listbox will depend upon you **Region** selection. As this is a workshop, select a low spec configuration to work with. For a PoC, MVP or production installations you will need a higher specification. Select '**Dev(No SLA)**' or the smallest specification in your listbox.	Yes

4. Select **Review + Create**, then select **Create** to build the ADX cluster. After a short wait you will be notified that the ADX cluster has been created. You can go to the resource to review its setup or select **Resource Groups**, then select **ADX**, to see the ***adxlabcluster*** resource you have just created:



This ***adxlabcluster*** resource template can be found in the Git Repository as 'ExportedTemplate_adxlabcluster.zip'.

Activity 5 – Security & Access

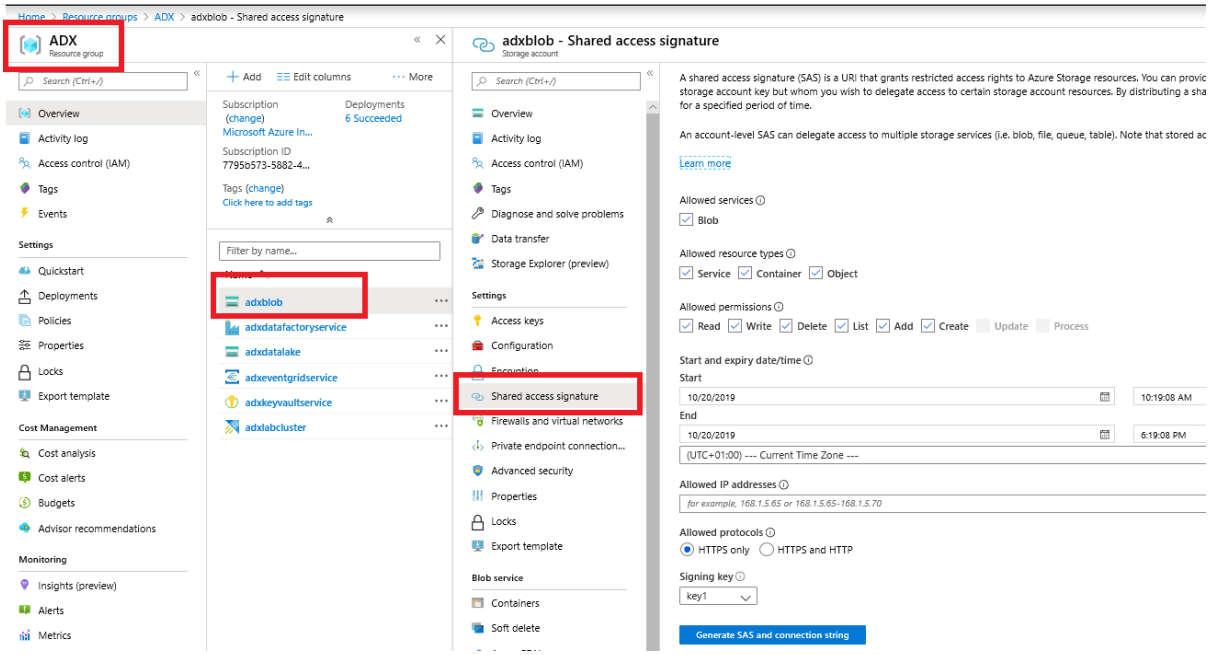
Now we have the core Azure services we need to start to configure these, in terms of security and access, in order to be able to configure the Azure Event Grid and build the Azure Data Factory (ADF) pipeline to process files.

Activity 5.1 – Create blob SAS key

We need a SAS key to be able to access the blob ***adxblob*** storage. In the Azure portal:

1. Select the **ADX** resource group
2. Select the ***adxblob*** storage account
3. Select **Share Access Signature**

The **adxblob – Shared access signature** blade will appear:



Enter values in this blade to generate a SAS key that is valid for 12 months from the date you are running this lab:

Property	Description	Required
Allowed Services	Leave all options selected	Yes
Allowed Resource Types	Leave all options selected	Yes
Allowed Permissions	Leave all options selected	Yes
Start and expiry date/time - Start	Leave as current date/time	Yes

Property	Description	Required
Start and expiry date/time - End	Select your chosen region to match the resource group *ADX* you have created. Increment the year value by 1, e.g: - If 2019, make 2020 - If 2020, make 2021	Yes
Allowed IP address	Leave blank	Yes

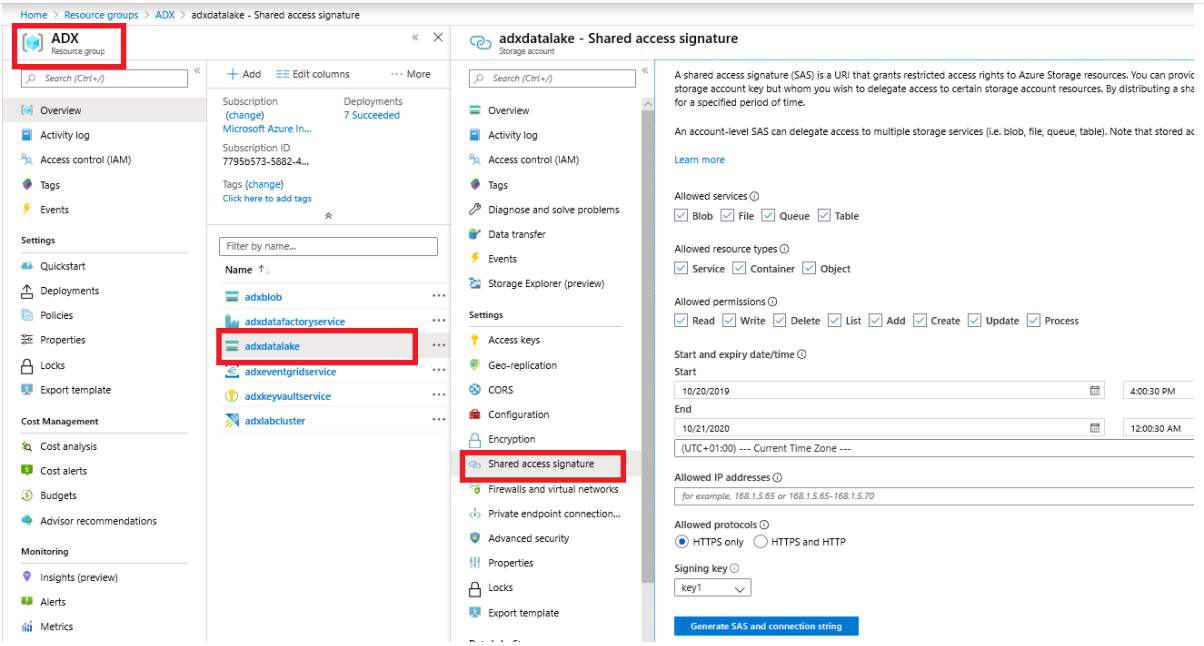
4. Select **Generate SAS and connection string**. You need to make a note of these values for future labs:
   - Connection string – save this in a text file for future reference
   - SAS token – save this in a text file for future reference
   - Blob service SAS URL – save this in a text file for future reference

Activity 5.2 – Create ADLS SAS key

We need a SAS key to be able to access the ADLS blob ***adxdatalake*** storage. In the Azure portal:

5. Select the **ADX** resource group
6. Select the ***adxdatalake*** storage account
7. Select **Share Access Signature**

The **adxdatalake – Shared access signature** blade will appear:



Enter values in this blade to generate a SAS key that is valid for 12 months from the date you are running this lab:

Property	Description	Required
Allowed Services	Leave all options selected	Yes
Allowed Resource Types	Leave all options selected	Yes
Allowed Permissions	Leave all options selected	Yes
Start and expiry date/time - Start	Leave as current date/time	Yes
Start and expiry date/time - End	Select your chosen region to match the resource group *ADX* you have created. Increment the year value by 1, e.g:- If 2019, make 2020 - If 2020, make 2021	Yes
Allowed IP address	Leave blank	Yes

8. Select **Generate SAS and connection string**. You need to make a note of these values for future labs:
   - Connection string – save this in a text file for future reference
   - SAS token – save this in a text file for future reference
   - Blob service SAS URL – save this in a text file for future reference

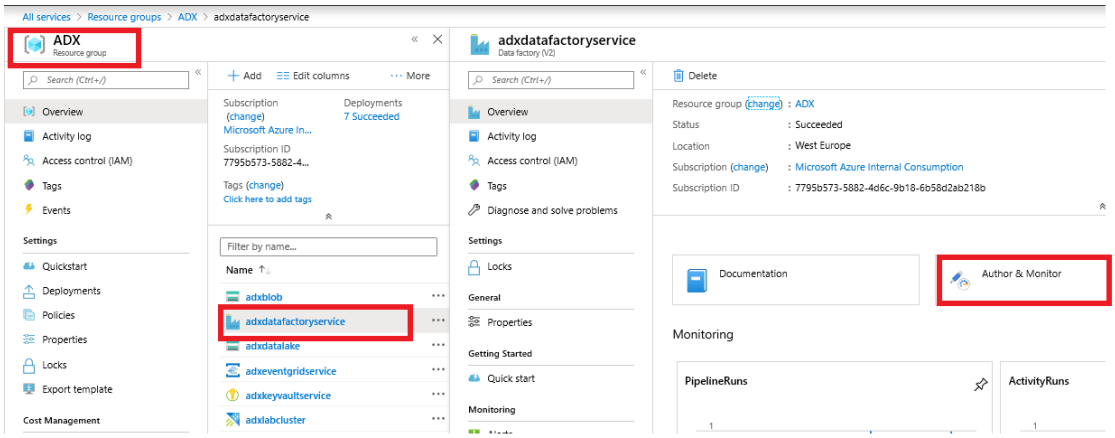
Activity 6 – ADF Pipeline

We will now create the ADF pipeline that will move the files from the blob **landing-zone** container to the ADLS **inbound-processed** container.

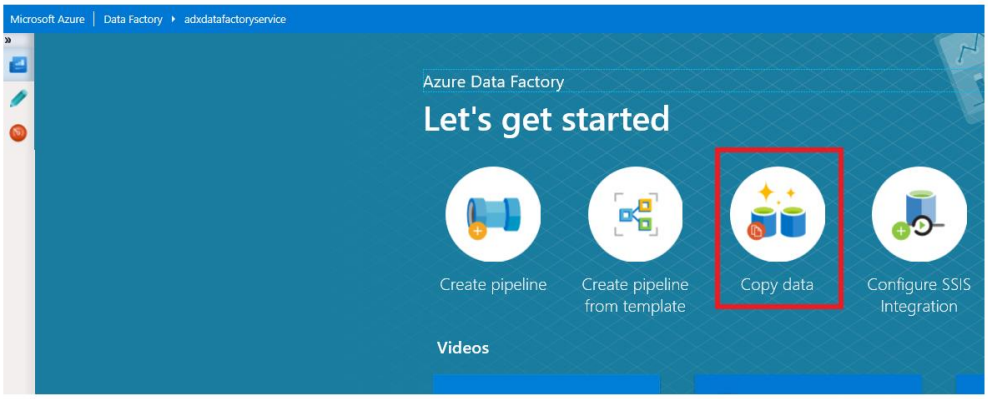
NOTE: *you may find it easier to have one of the JSON data files loaded into the **landing-zone** container on **adxblob**. The sample data is in the compressed file "sample_quote_data.zip".*

Activity 6.1 - Build the ADF Pipeline

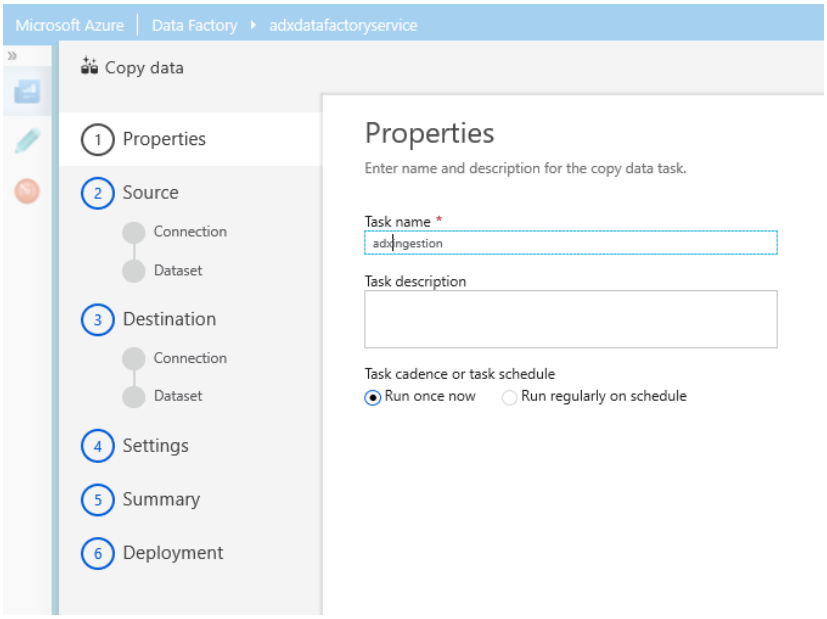
1. Select the **ADX** resource group
2. Select ***adxdatafactoryservice***
3. Select **Author & Monitor**



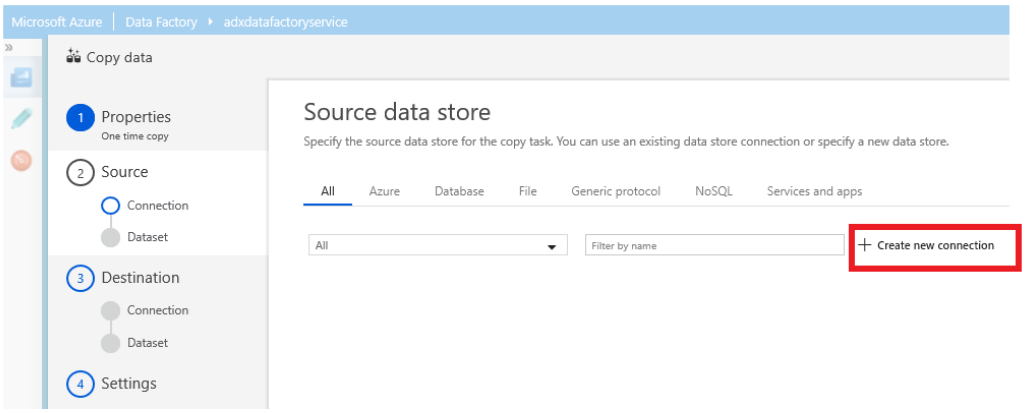
4. From the landing page, select **Copy data:**



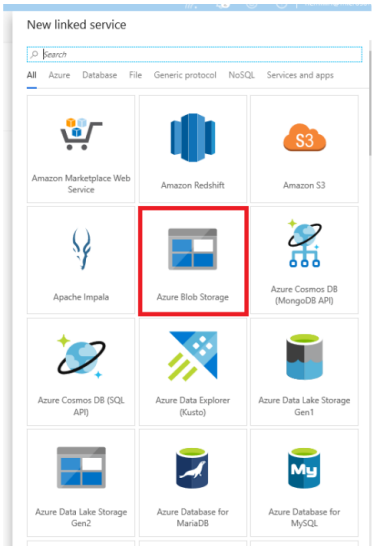
5. On the **Properties** blade, in **Task name**, enter ***adxingestion***



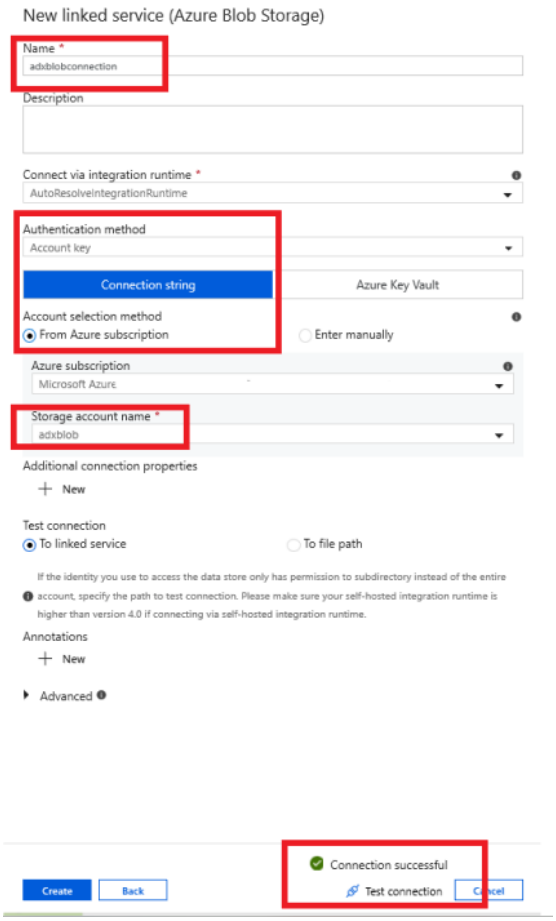
6. On the **Source data store** blade, select **+ Create new connection**



7. On the **New linked service** blade, select **Azure Blob** Storage and select **Continue**



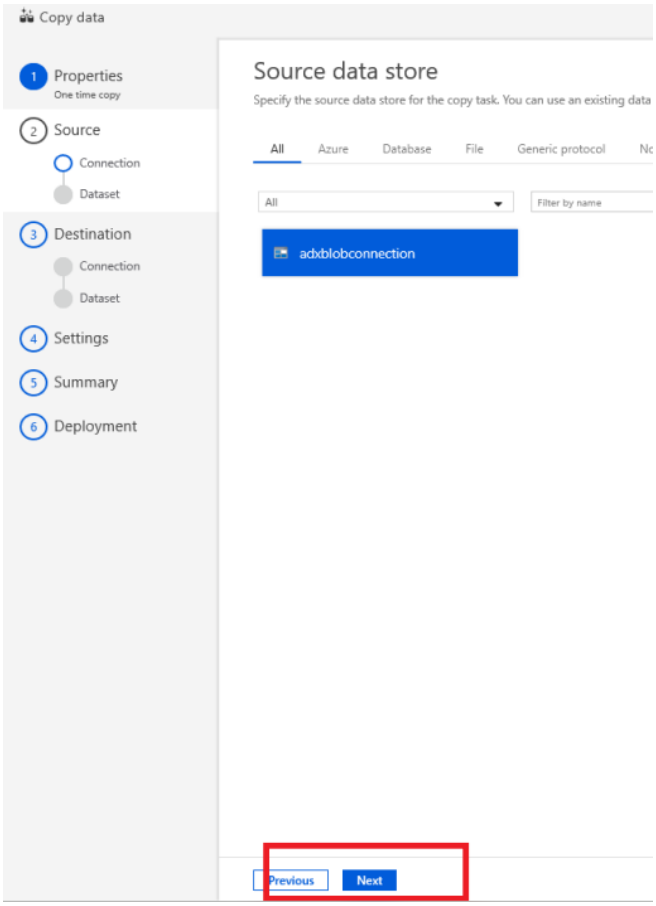
8. On the New linked service (Azure Blob Storage) blade:



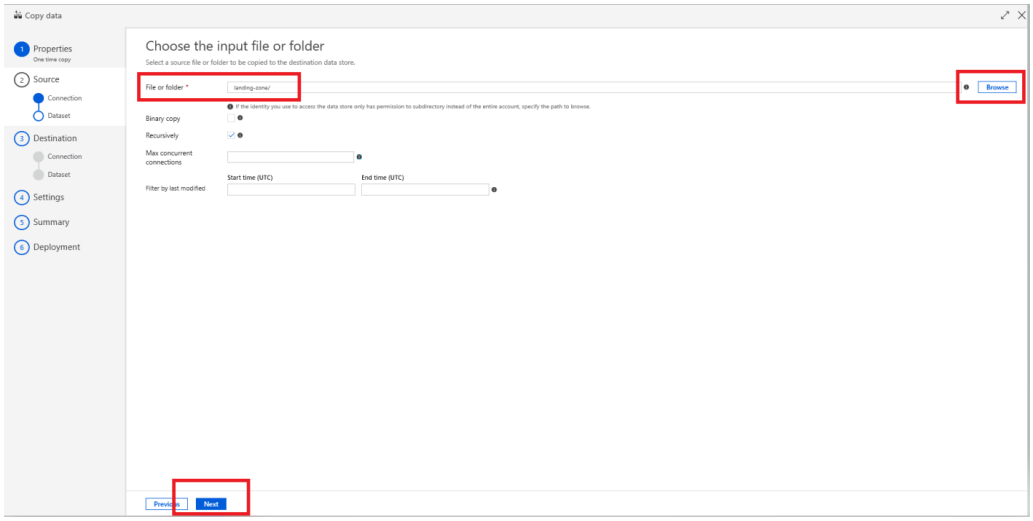
- Complete the blade using the values below:

Property	Value	Required
Name	**adxblobconnection**	Yes
Authentication Method	**Account Key**	Yes
Azure Subscription	The Azure subscription the **ADX** resource group has been created in	Yes
Storage account name	**adxblob**	Yes
Allowed IP address	Leave blank	Yes

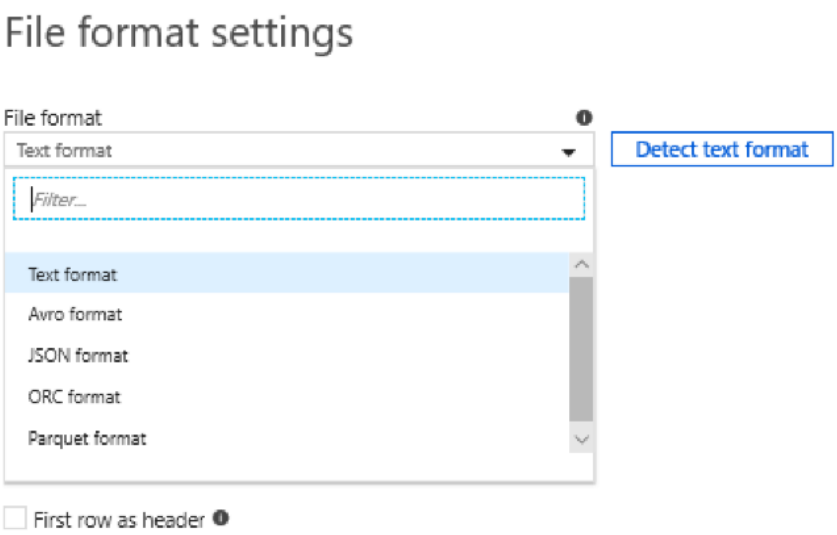
9. Select T**est Connection**
10. Assuming “Connection successful appears”, select **Create**
11. On the **Source data store** blade, select **Next**:



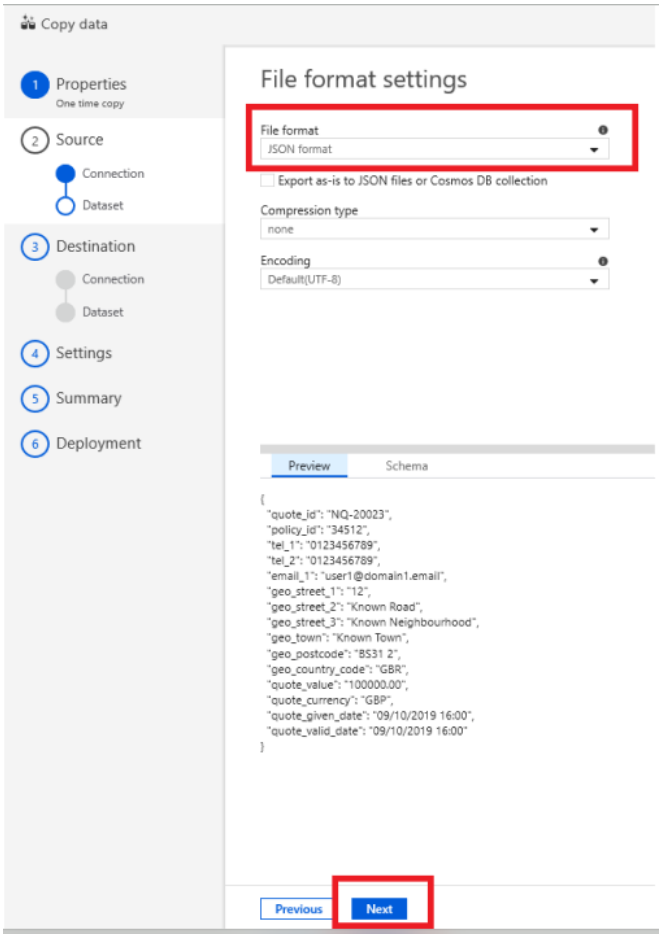
12. On the **Choose the input file or folder**:
13. Browse and select the **landing-zone** container on ***adxblob*** in the **File or folder**



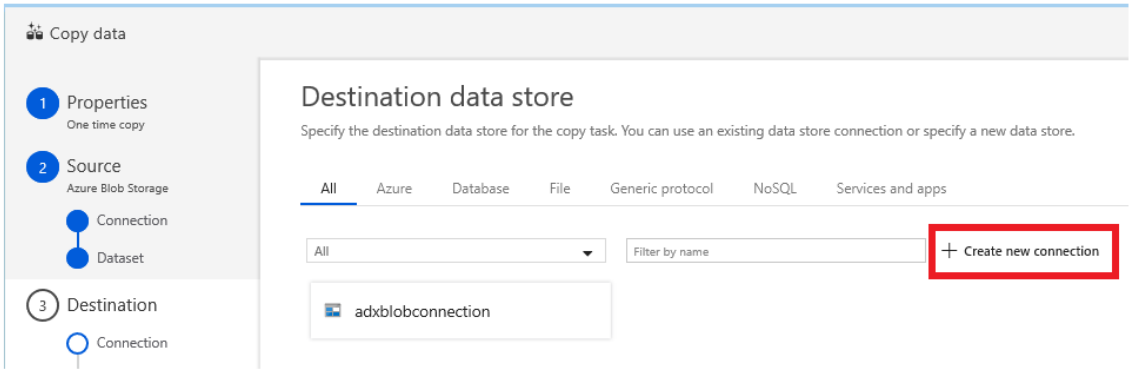
14. On the **File format settings** blade, various formats are supported:



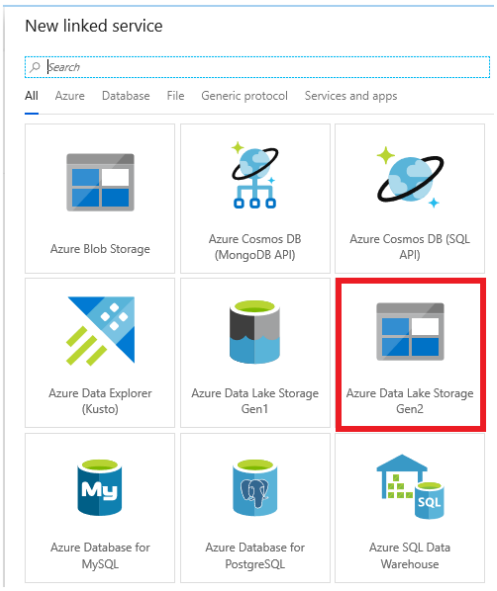
15. Select **JSON** as the **File Format** *(the sample dataset is in JSON)*
16. Select **Next**



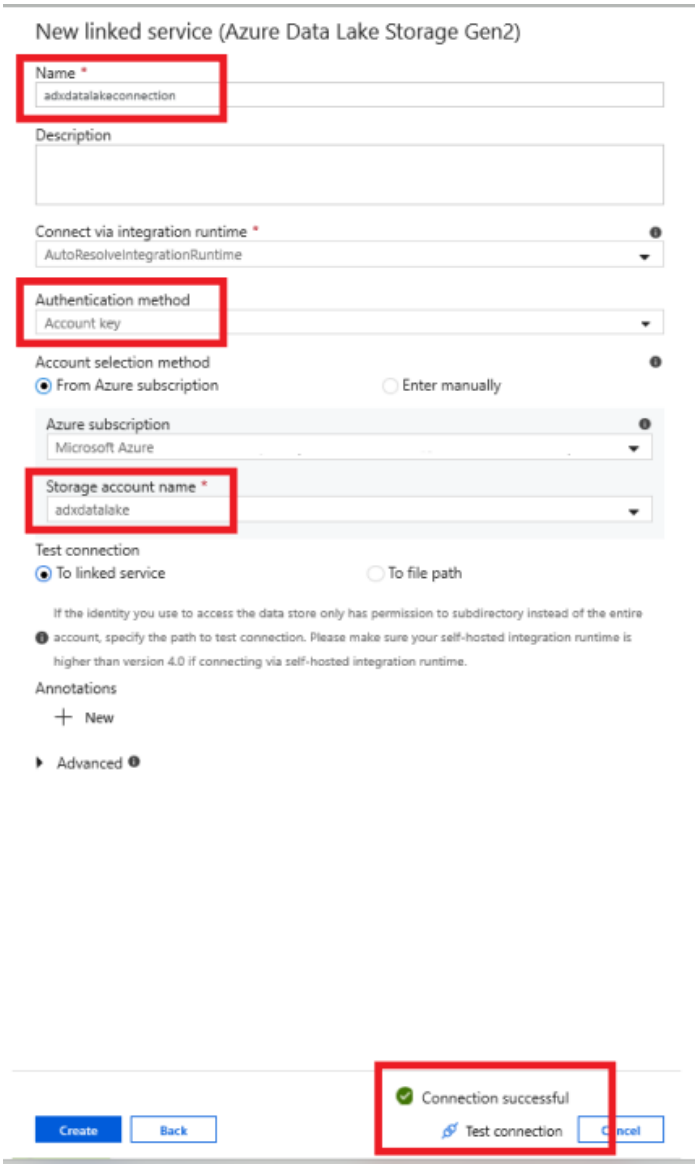
17. On the **Destination data store** blade, select **+ Create new connection**:



18. On the **New linked service** blade, select **Azure Data Lake Storage Gen2**:



19. On the New linked service (Azure Blob Storage) blade:

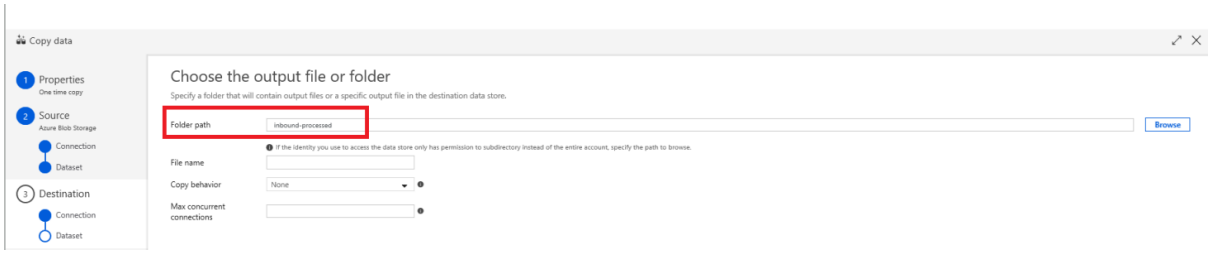


- Complete the blade using the values below:

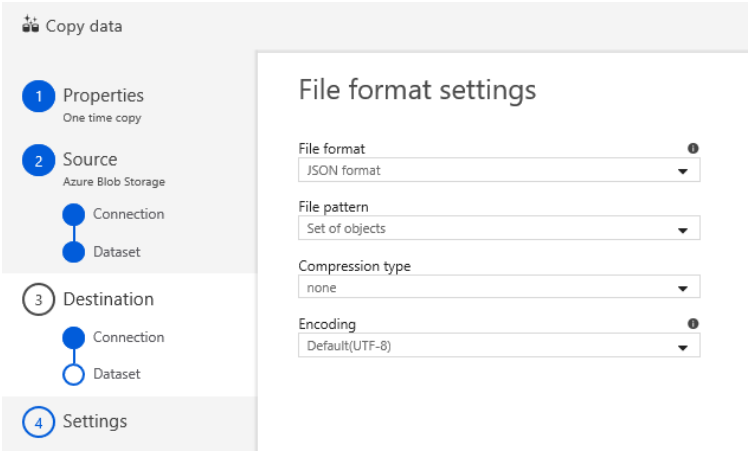
Property	Value	Required
Name	**adxdatalakeconnection**	Yes
Authentication Method	**Account Key**	Yes
Azure Subscription	The Azure subscription the **ADX** resource group has been created in	Yes
Storage account name	**adxdatalake**	Yes

20. Select T**est Connection**
21. Assuming “Connection successful appears”, select **Create**

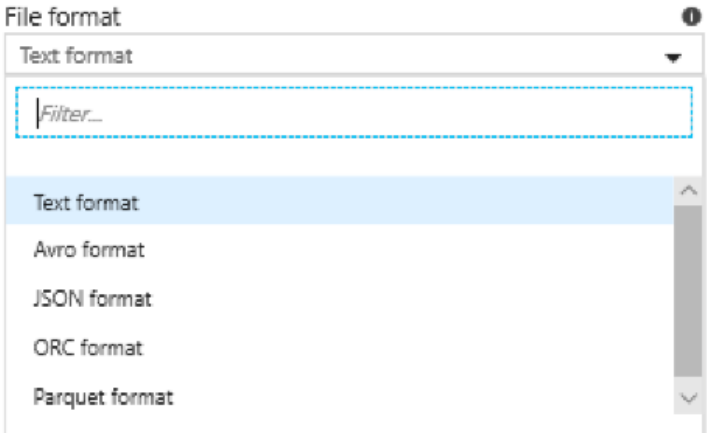
22. On the **Destination data store** blade, select **Next**
23. On the **Choose the output file or folder**:
24. Browse and select the **inbound-processed** container on ***adxdatalake*** in the **File or folder**



25. On the **File format settings** blade, select **JSON**
26. Select **Set of objects** as the **File pattern**
27. Select **Next**

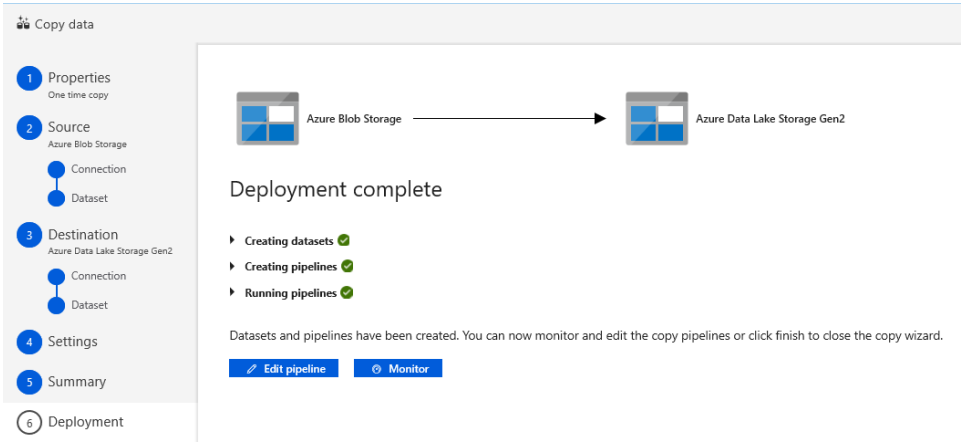


NOTE: On the **File format** list-box the various formats supported:



28. On **Schema mapping** blade, select **Next** (make no alterations to the defaults)
29. On **Settings** blade, select **Next** (make no alterations to the defaults)
30. On **Summary** blade, select **Next** (make no alterations to the defaults)

31. On **Deployment** blade, the ADF pipeline will deploy. The following display should be achieved:

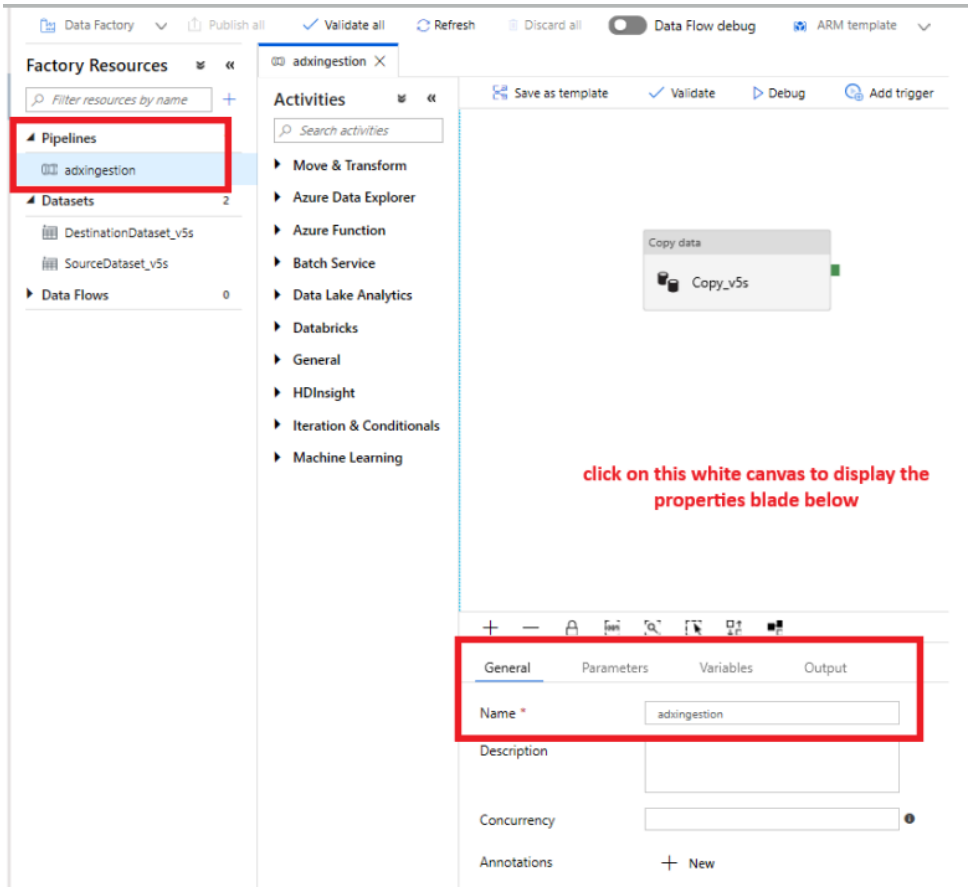


32. Select **Finish**

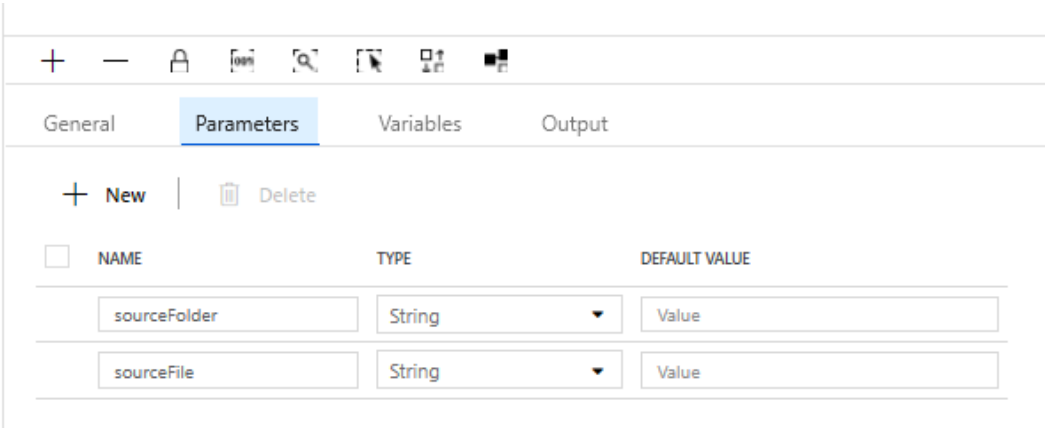
Activity 6.2 – Create the pipeline parameters

The ADF pipeline will be parameter driven to accept the blob filename and filepath that will be passed from the Event Grid.

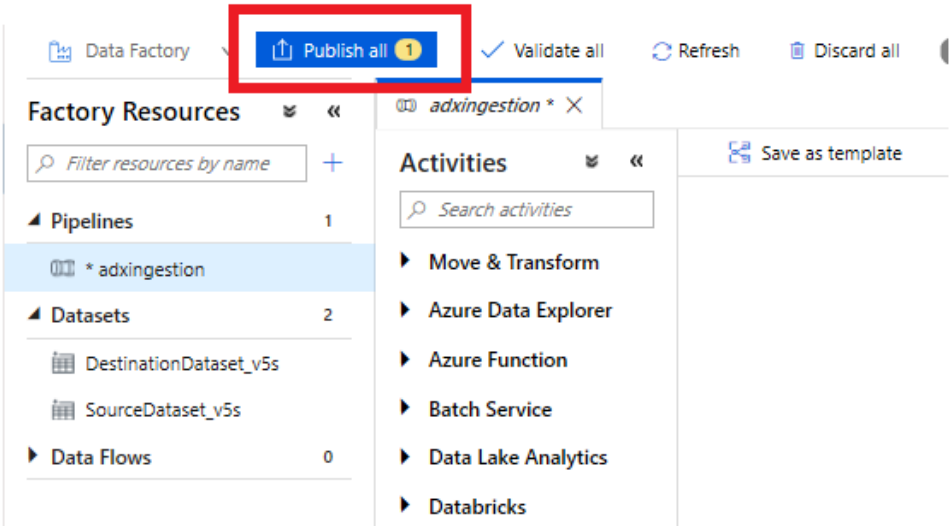
1. Select the **ADX** resource group
2. Select the ***adxdatafactoryservice***
3. Select **Author & Monitor**
4. Select the ***adxingestion*** pipeline
5. Select the ***adxingestion*** canvas to display the properties blade:



6. Select **Parameters**
7. Using the **+ New**, enter the following two parameters, **sourceFolder** and **sourceFile**



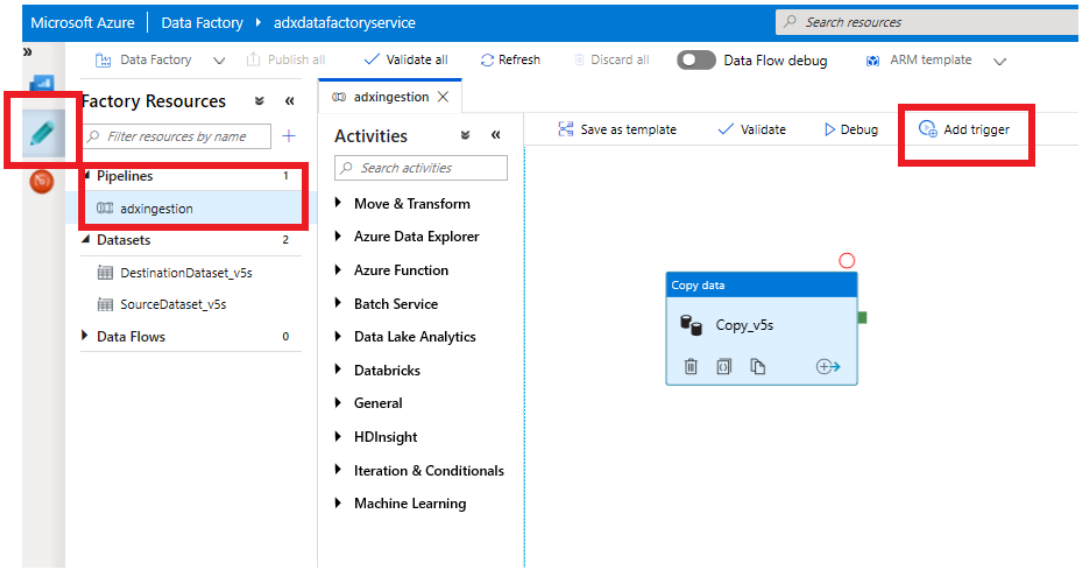
8. Select **Publish all** to publish the changes to ADF:



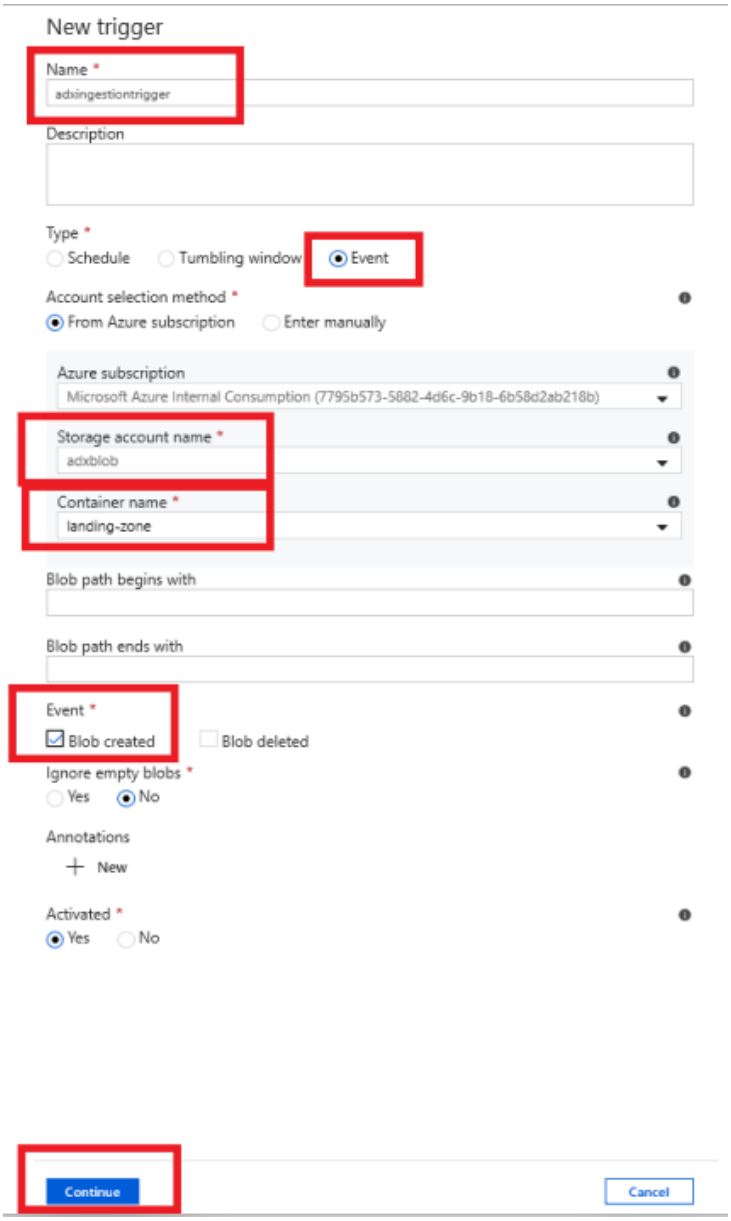
Activity 6.3 – Create the pipeline trigger

We now have the basic ADF pipeline however it needs to be triggered via files being created in the **landing-zone** container.

1. Select the **ADX** resource group
2. Select the ***adxdatafactoryservice***
3. Select **Author & Monitor**
4. Select the author icon:



5. Select the ***adxingestion*** pipeline (see above)
6. Select **Add trigger** followed by **New/Edit** in the drop-down
7. On the **Add triggesr** blade, in the drop-down, select **+ New**

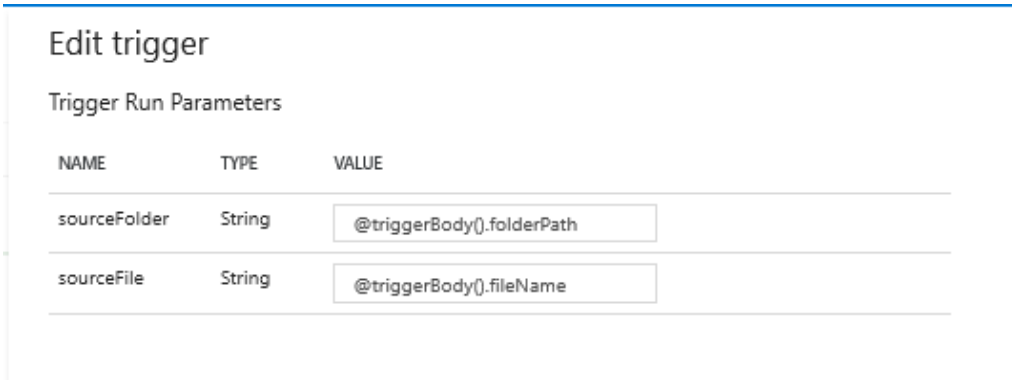


- Complete the blade using the values below:

Property	Value	Required
Name	**adxingestiontrigger**	Yes
Type	**Event**	Yes
Azure Subscription	The Azure subscription the **ADX** resource group has been created in	Yes
Storage account name	**adxblob**	Yes

Property	Value	Required
Container name	**Landing-zone**	Yes
Event	**Blob created**	Yes

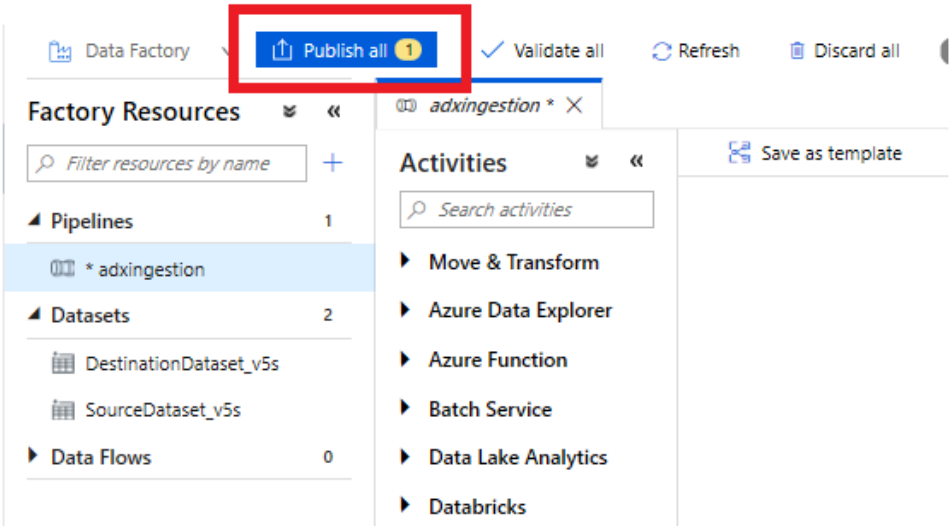
8. Select **Continue**
9. On New trigger blade enter the Event Grid parameters that will be passed to the pipeline parameters:



- Complete the blade using the values below:

Property	Value	Required
sourceFolder	**@triggerBody().folderPath**	Yes
SourceFile	**@triggerBody().fileName**	Yes

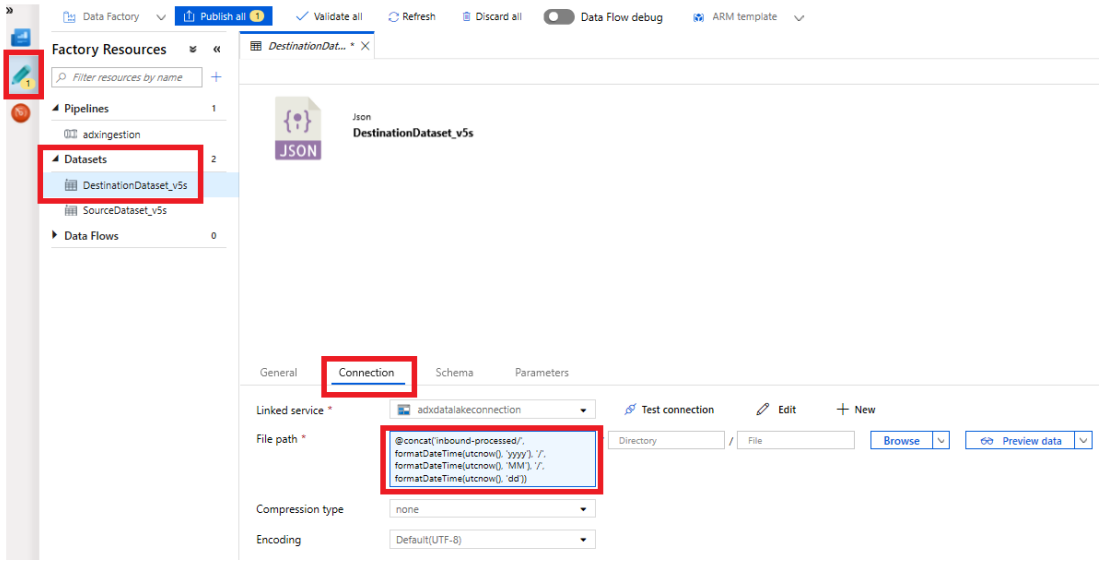
10. Select **OK**
11. Select **Publish all** to publish the changes to ADF:



Activity 6.4 – Parameterise the ADLS hierarchy

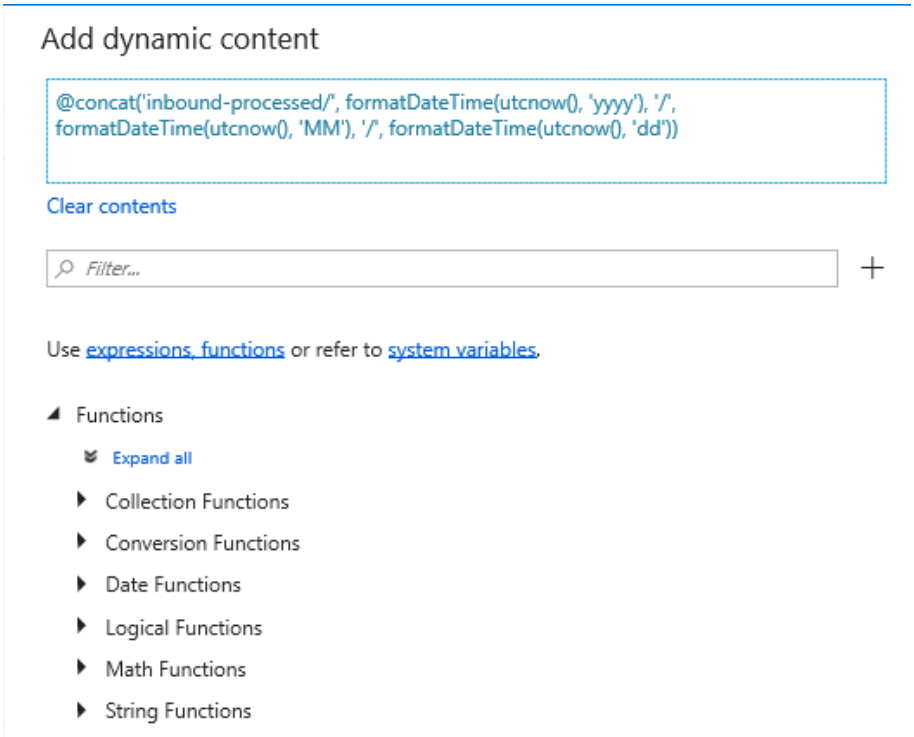
The hierarchy on ***adxdatalake*** needs to be parameterised so that files are landed in a YYYY/MM/DD structure. We will edit the ADF destination dataset to achieve this:

1. Select the **ADX** resource group
2. Select the ***adxdatafactoryservice***
3. Select **Author & Monitor**
4. Select the ***adxingestion*** pipeline
5. Select the **DestinationDataset_xxx** (**xxx** is the value specific to your Dataset)
6. Select **Connection** in the **DestinationDataset_xxx** properties blade
7. Select the **File path** field showing ***inbound-processed***

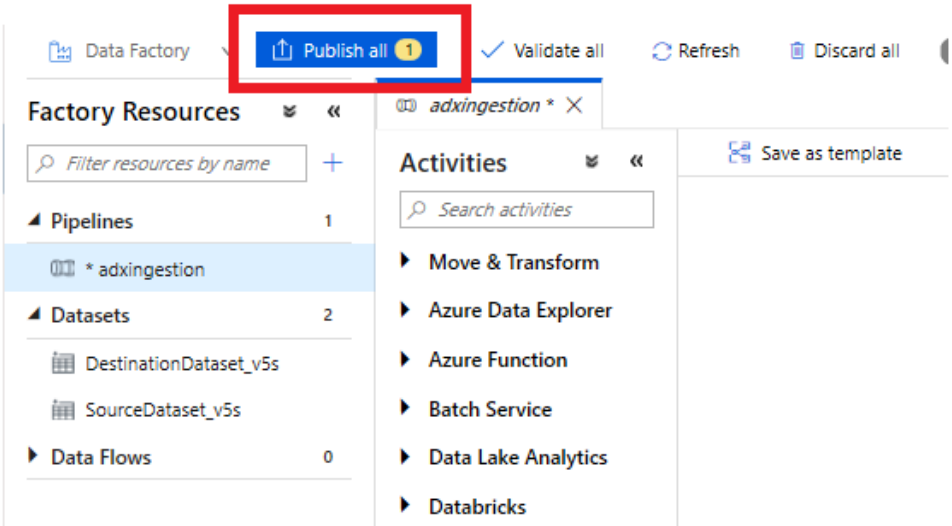


8. In the **Add dynamic content [Alt + p]** blade, enter:

   @concat('inbound-processed/', formatDateTime(utcnow(), 'yyyy'), '/', formatDateTime(utcnow(), 'MM'), '/', formatDateTime(utcnow(), 'dd'))



9. Select **Finish**
10. Select **Publish all** to publish the changes to ADF:



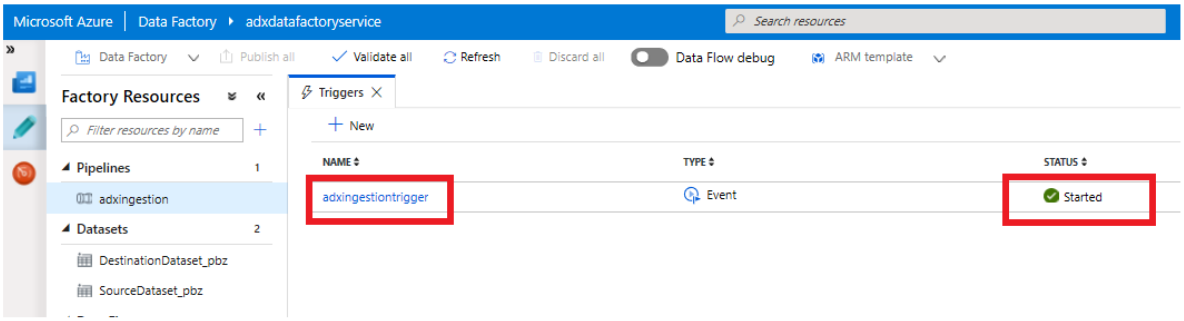
Activity 7 – Execute ADF Pipeline

We can now run the ADF ***adxingestiontrigger*** to demonstrate multiple files landing on blob and being ingested through to ***adxdatalake*** in a date orientated hierarchy (inbound-processed/YYYY/MM/DD). Repeating this process on the consecutive day will produce the same results but into inbound-processed/YYYY/MM/DD **+ 1** where + 1 is the next/consecutive day in the month.

Activity 7.1 – Start/activate the ADF trigger

To test the ADF pipeline, and generate data for Activity 8, we need to start the ADF trigger.

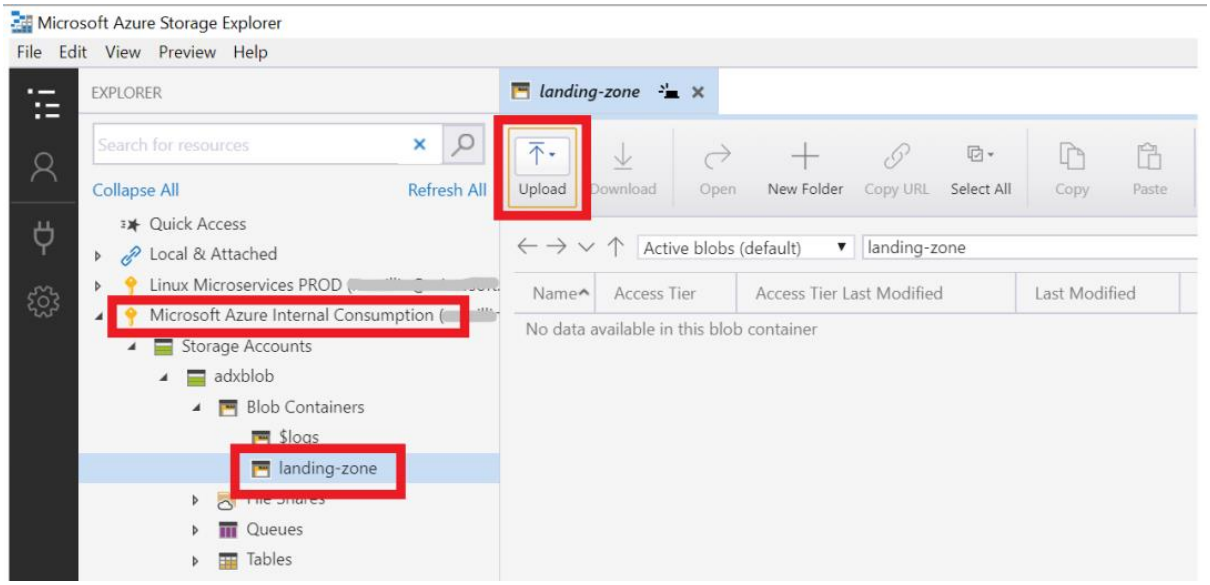
1. Select the **ADX** resource group
2. Select the ***adxdatafactoryservice***
3. Select **Author & Monitor**
4. Select the ***adxingestion*** pipeline
5. Select **Triggers**
6. Ensure ***adxingestiontrigger*** Status is **Started**:

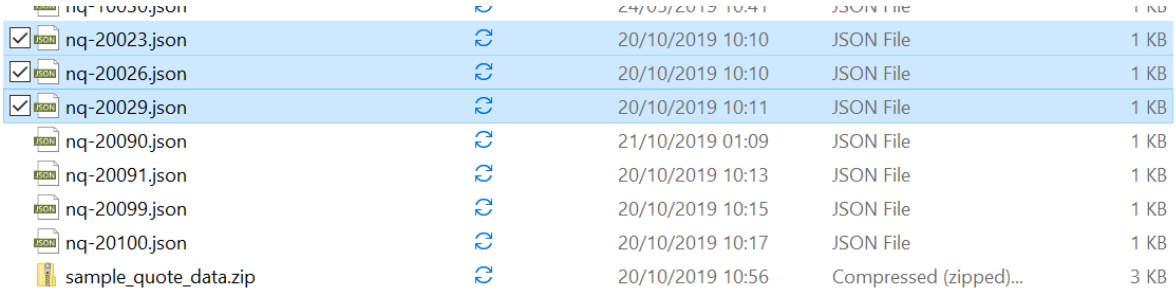


Activity 7.2 – Ingest demo data

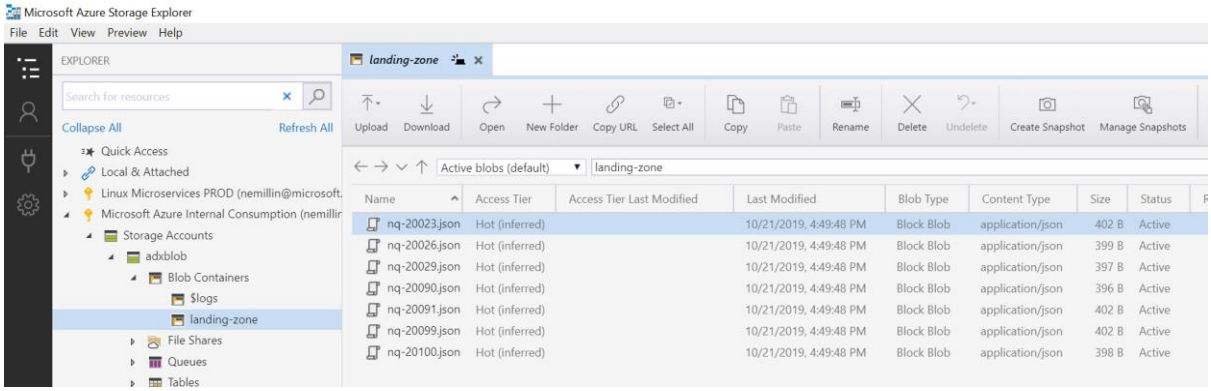
The demo data can be found in **sampledata/sample_quote_data.zip**. This represents JSON structures for simplistic quote data. Other formats can be used according to your pipeline artefacts.

1. Unzip **sample_quote_data.zip**
2. Using either the Azure Portal or **Azure Storage Explorer**, upload these **files** (not the whole folder) to the blob ***adxblob*****:**





When the files have been uploaded the view should look like this:

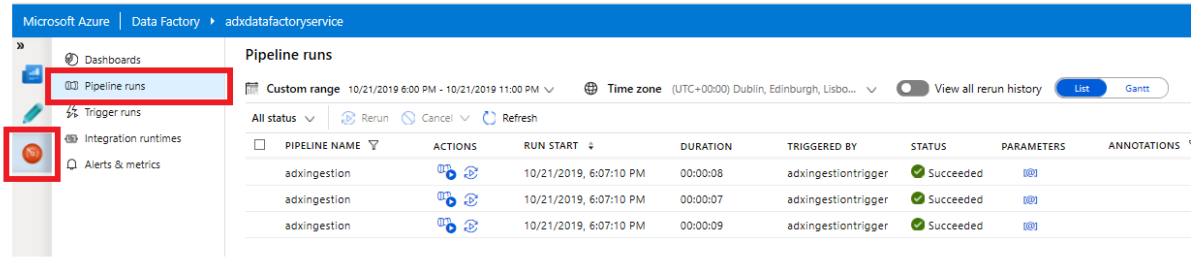


Activity 7.3 – Monitor ADF pipeline execution

Uploading the files (Activity 7.2 – Ingest demo data) to ***adxblob*** will have generated blob create events that, via Azure Event Grid ***adxeventgridservice***, will have triggered the ADF ***adxingestiontrigger*** for each individual blob created.

We can monitor the progress and status of the ADF pipeline:

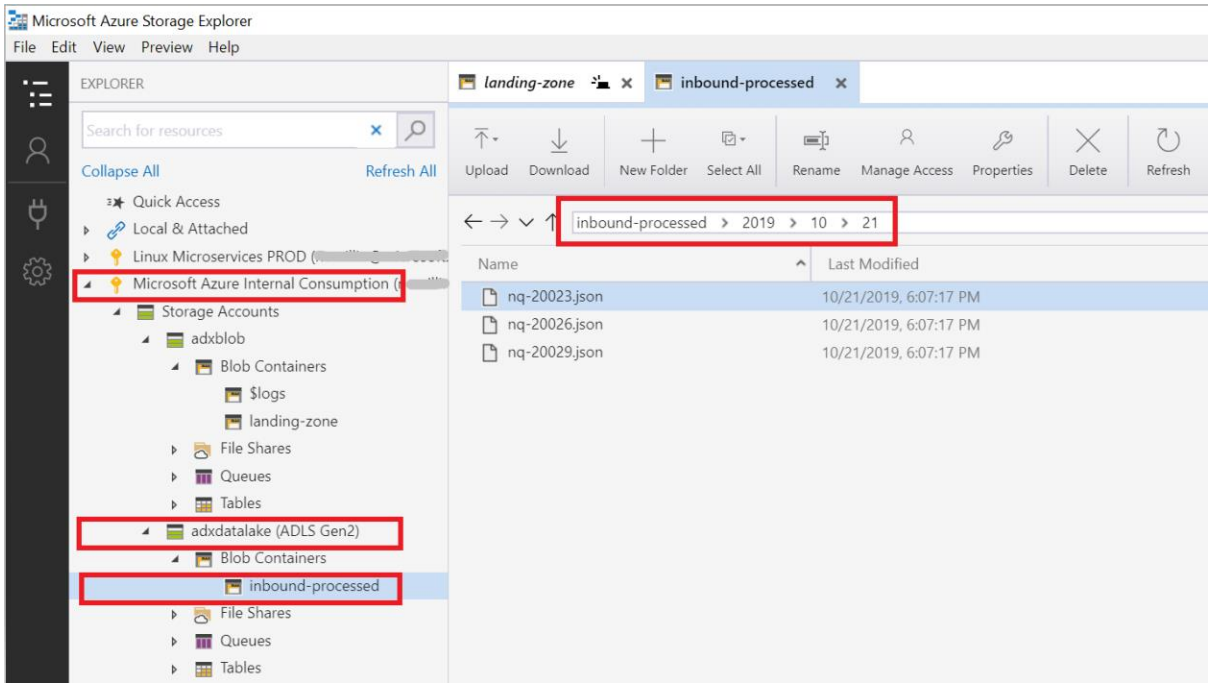
1. Select the **ADX** resource group
2. Select the ***adxdatafactoryservice***
3. Select **Author & Monitor**
4. Select **Pipeline runs:**



Notice that there are three pipelines runs, corresponding with the three files I uploaded.

Familiarise yourself with monitoring, and reviewing, ADF pipeline activity.

5. Using either the Azure Portal or **Azure Storage Explorer**, navigate to ***adxdatalake*:**



Notice that the three files have been placed in a dynamic hierarchy:

Inbound-processed/YYYY/MM/DD, which [in this example] is

Inbound-processed/2019/10/21

Activity 8 – Query data on Data Lake

We can now use ADX to query data ingested [via ADF] to **adxdatalake.**

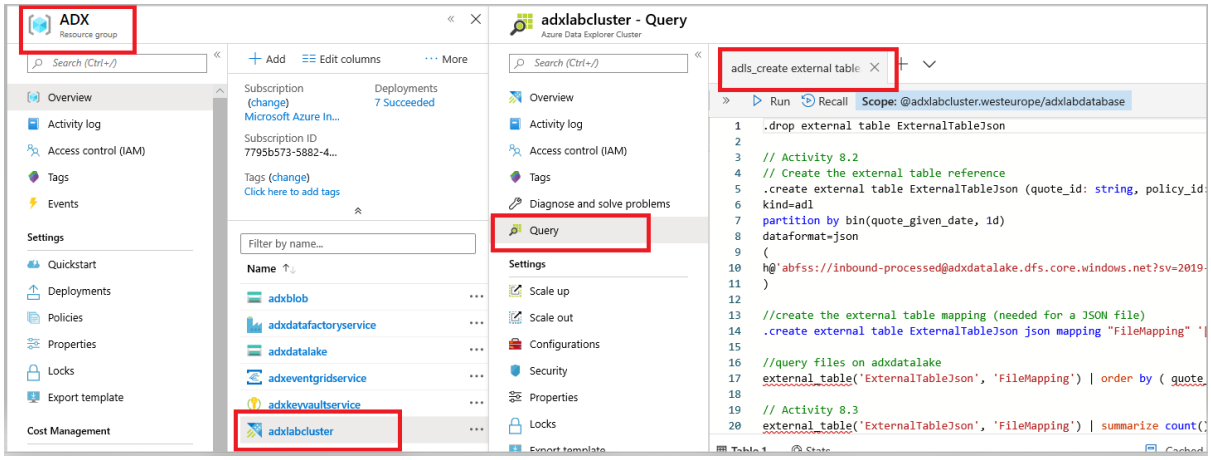
Useful links for ADX external tables are:

- Querying ADLS with ADX
- Table Management
- ADX //Build blog

Activity 8.1 – Open ADX query

We can use the Azure Portal ADX Query for this activity:

1. Select the **ADX** resource group
2. Select the **adxlabcluster**
3. Select **Query**
4. Open the KQL query file **adls_create_external_table.kql**, located in the **KQL** subfolder, in the query window



We will now step through the KQL queries:

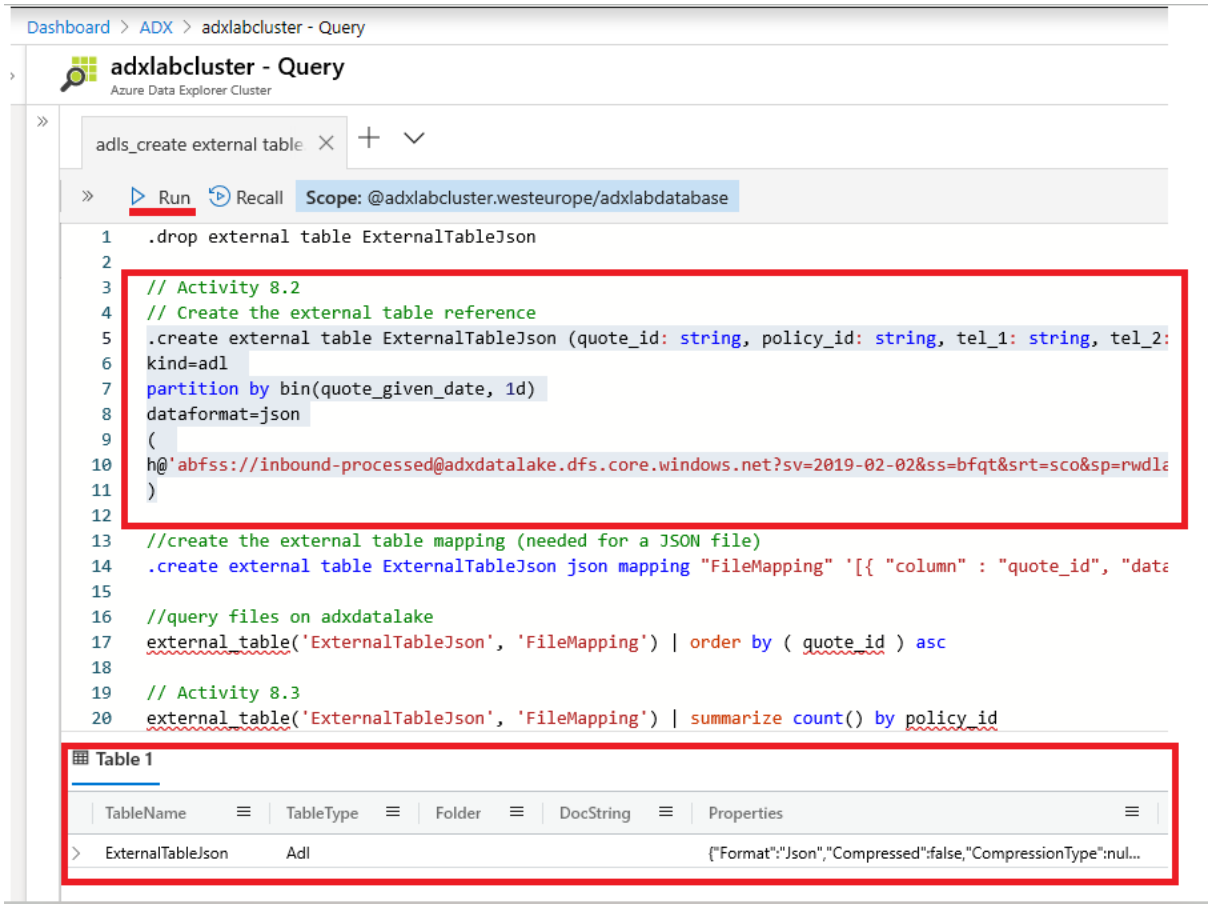
Activity 8.2 – Create External Table

We create the external table reference **ExternalTableJson**.

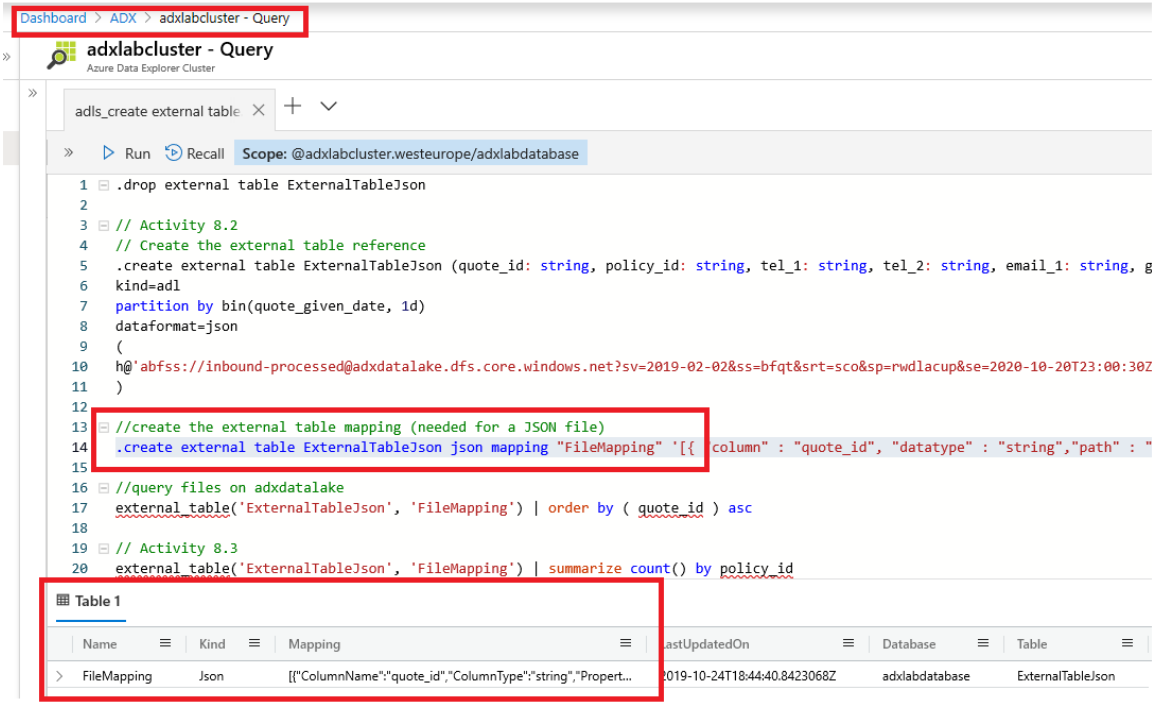
1. In the query window, in section 8.2 Substitute the **adxdatalake** SAS key you generated earlier where the comment appears “<substitute your adxdatalake SAS key here>”. The resulting paste should have no spaces between the leading and trailing “ ‘ ”, as below:

```
// Activity 8.2
// Create the external table reference
.create external table ExternalTableJson (quote_id: string, policy_id: string, tel_1: str
kind=adl
partition by bin(quote_given_date, 1d)
dataformat=json
(
h@'abfss://inbound-processed@adxdatalake.dfs.core.windows.net?sv=2019-02-02&ss=bfqt&srt=s
)
```

2. **Run** the KQL command. No errors should appear in the results:



3. The lab sample data is JSON format. A mapping definition is required for this format of data. Highlight and **Run** the KQL code under **"//create the external table mapping (needed for a JSON file)**". No errors should appear in the results:



Activity 8.3 – Query files on adxdatalake

We can now query data stored on **adxdatalake** that has been automatically ingested via ADF without having to ingest it into ADX.

1. Highlight the KQL under "// Activity 8.2 - query files on adxdatalake"
2. **Run** the command and see the three JSON files that were uploaded to **adxblob** and ingested to **adxdatalake** via the **adxdatafactoryservice** ADF:

```
13 //create the external table mapping (needed for a JSON file)
14 .create external table ExternalTableJson json mapping "FileMapping" '[{ "column" : "quote_id", "datat
15
16 // Activity 8.3 - query files on adxdatalake
17 external_table('ExternalTableJson', 'FileMapping') | order by ( quote_id ) asc
18
19 // Activity 8.4 - query automatically ingested files on adxdatalake
20 external_table('ExternalTableJson', 'FileMapping') | order by ( quote_id ) asc
21 external_table('ExternalTableJson', 'FileMapping') | summarize count() by policy_id
22
```

Table 1 | Stats

quote_id	policy_id	tel_1	tel_2	email_1	geo_street_1	geo_street_2
NQ-20023	34512	0123456789	0123456789	user1@domain1.email	12	Known Road
NQ-20026	34599	0123456789	0123456789	user2@domain1.email	12	Known Road
NQ-20029	10002	0123456789	0123456789	user3@domain.email	1	Known Road
NQ-20090	34444	0123456789	0123456789	user4@domain.email	1	Known Road
NQ-20091	34512	0123456789	0123456789	user1@domain1.email	12	Known Road
NQ-20099	34512	0123456789	0123456789	user1@domain1.email	12	Known Road

Activity 8.4 – Query automatically ingested files on adxdatalake

The use of ADX external tables mapping onto ADLS (**adxdatalake**) warrants that an executed query always has the current data to query. We can demonstrate this by uploading files to **adxblob** and running a KQL query against **adxdatalake**. The results of that query will have the recently uploaded JSON files included in the resultset.

1. From the JSON data located in **/sampledata** upload the following files to **adblob** (use Azure Storage Explorer or the Azure Portal)

   Nq-20090.json

   Nq-20091,json

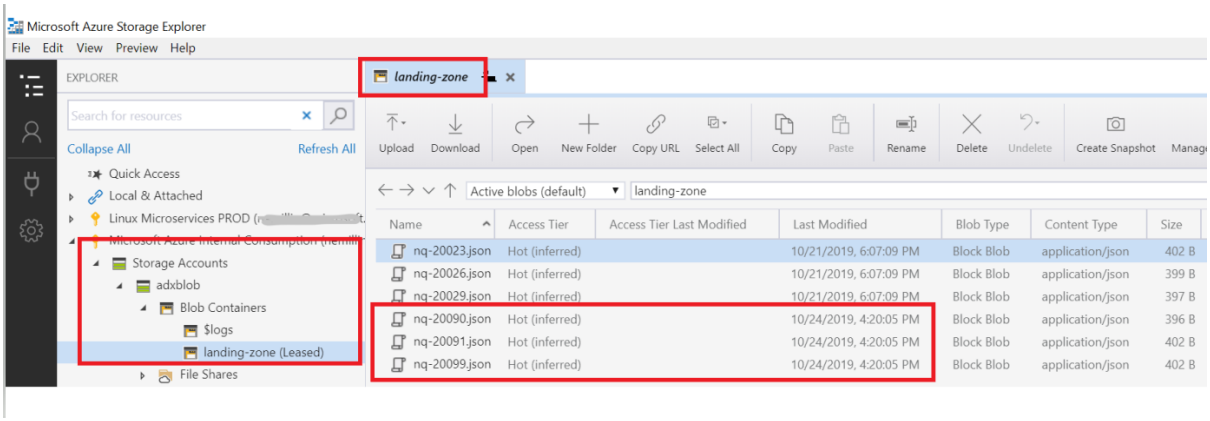
   Nq-20099.json

   Reminder: These files will create 'blob created' events on **adxblob**. The evets will be captured by the **adxeventgridservice** that will, in turn, trigger the ADF pipeline **adxingestion.** The ADF pipeline automatically ingests the files to the **adxdatalake** data lake, storing them in a YYYY/MM/DD hierarchy dependent upon the data the pipeline executed.

2. Highlight the first line of KQL under "// Activity 8.4 - query automatically ingested files on adxdatalake" and **Run** this.

```
external_table('ExternalTableJson', 'FileMapping') | order by ( quote_id ) asc
```

Notice the three JSON files uploaded in (1.) appear in the results:



3. Highlight the second line of KQL under "// Activity 8.4 - query automatically ingested files on adxdatalake" and **Run** this.

```
external_table('ExternalTableJson', 'FileMapping') | order by ( quote_id ) asc
| summarize count() by policy_id
```

```
18
19  // Activity 8.4 - query automatically ingested files on adxdatalake
20  external_table('ExternalTableJson', 'FileMapping') | order by ( quote_id ) asc
21  external_table('ExternalTableJson', 'FileMapping') | order by ( quote_id ) asc | summarize count() by policy_id
22
```

Table 1 Stats

policy_id	count_
34512	3
10002	1
34599	1
34444	1

Notice that policy_id 34512 now has three files in the results.

This demonstrated the automatic file ingestion with resulting dynamic querying.

Activity 8.5 – Enhancements

You have completed lab 1 now try to enhance the activities in section 8 as follows:

- Try different file formats (csv) or larger volumes of files
- Experiment with KQL to perform richer queries

Other labs will show other ADX data ingestion techniques. Check back for updates.

Activity 8.6 – Clean-up

Remember to stop the cluster **adxlabcluster** to prevent Azure charges being incurred unnecessarily.

Remember to pause the ADF trigger.