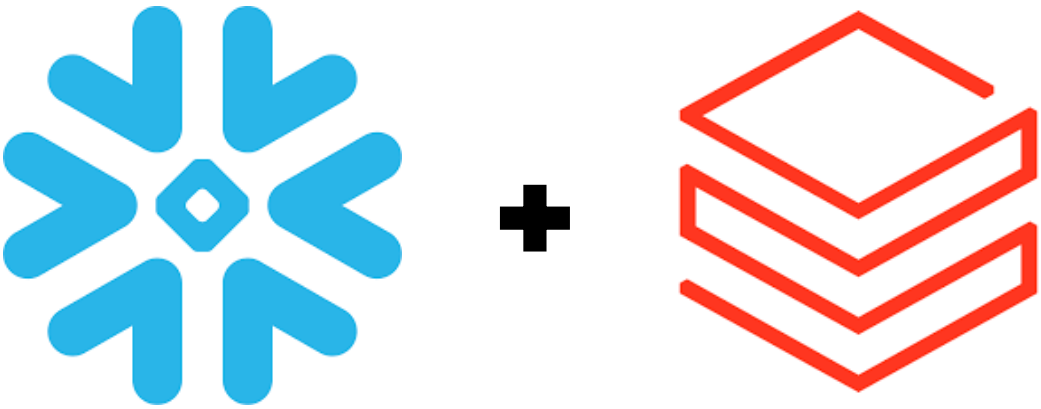
Add Data from Snowflake using Azure Databricks notebook



In this article we are going to use azure Databricks pre-build notebook for Snowflake to:

- Create a table on the fly in Snowflake by using Data Science & Engineering App.
- Read the same table from snowflake and create a Delta table
- Querying the same table by using the Databricks SQL app.

Prerequisite:

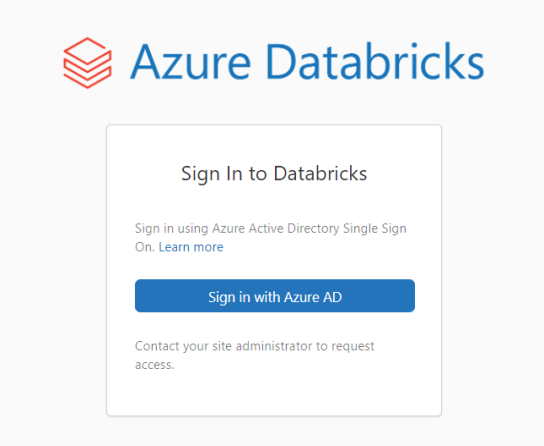
1- You need an Azure Databricks Premium workspace and Snowflake Account on Azure.
2- Access to Azure Key Vault

So, let's see how it's working.

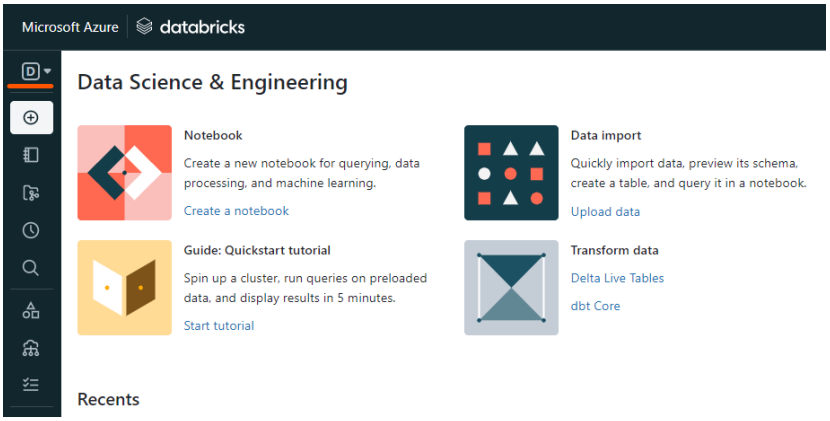
Before we start lets login the Databricks workspace

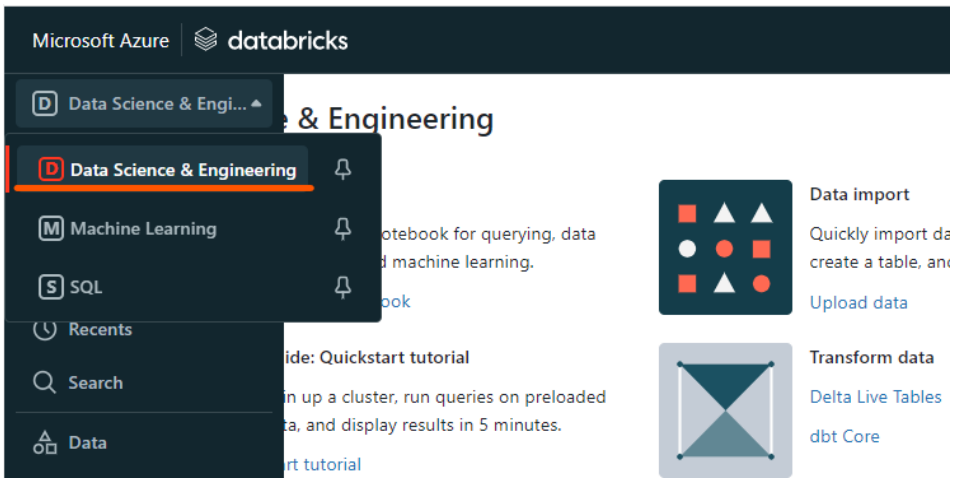
- Create a table on the fly in Snowflake

1- From Azure Databricks, click sign in with Azure AD to the workspace

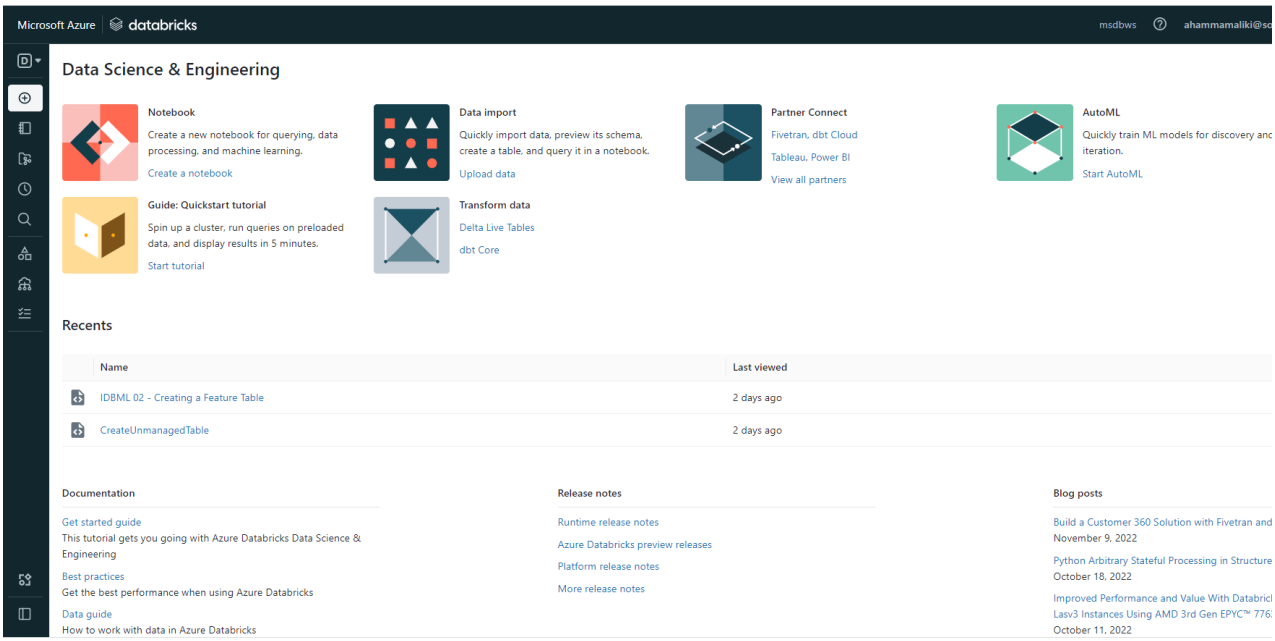


2- At the left side bar menu select the app switch to select the app you need to use.



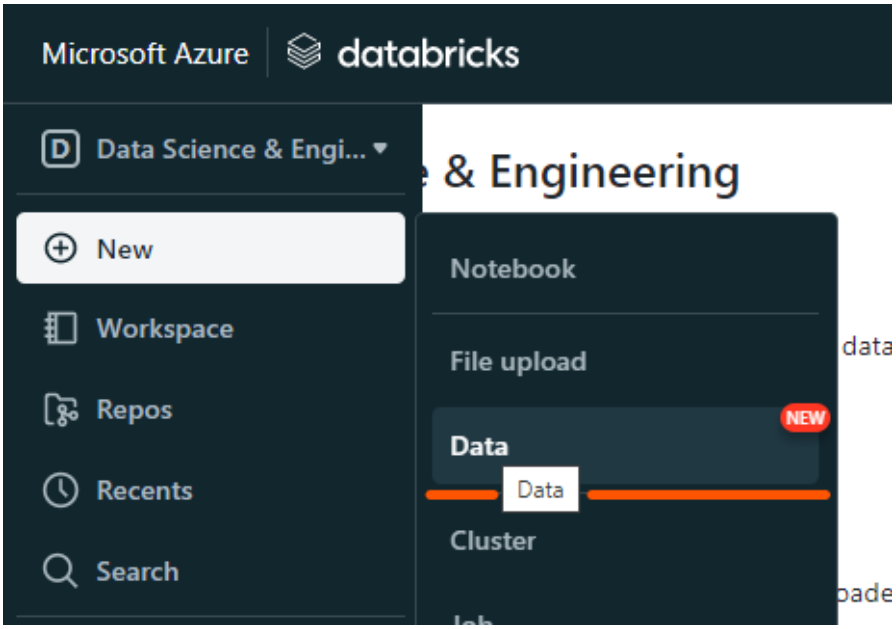


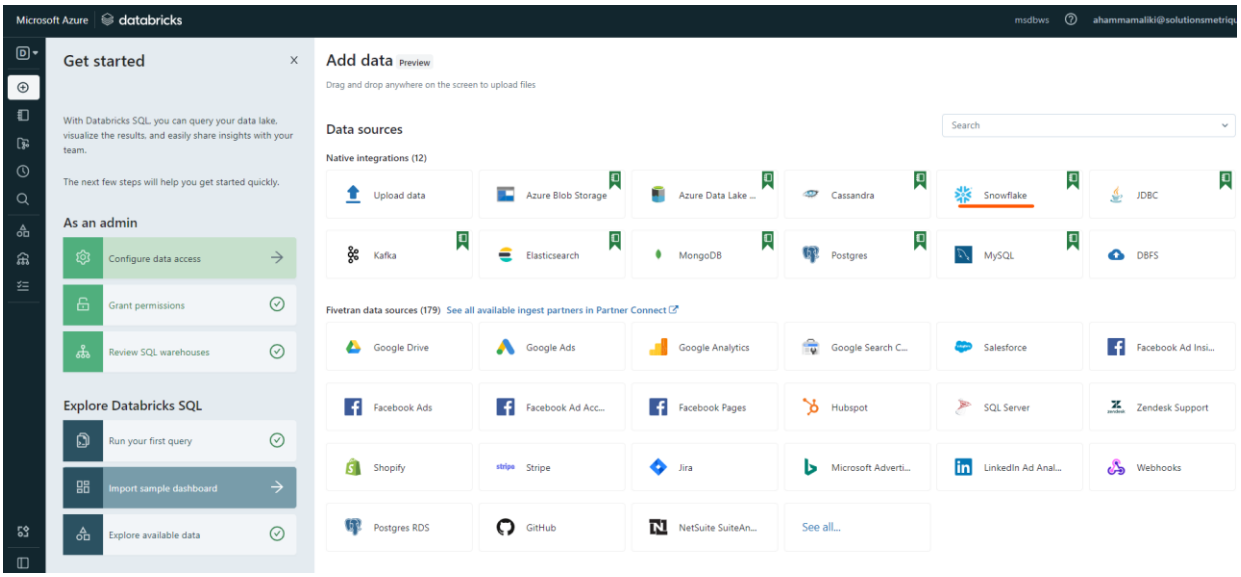
3- By default, when you login the workspace to are directly landing to the Data Science & Engineering App page



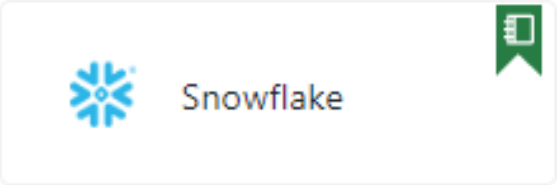
To add a data in Databricks we have several ways to do that. In this article we are going to use the **Add data** page from Data option in the Create menu.

4- Click on New on **+** sign and in the drop list, select Data.
Automatically I have been taking to the Add Data page where I have multiple data sources Native integrations prebuild notebooks and data ingestion for partners prebuild apps.





5- Select Snowflake prebuild notebook



6- A notebook named Import from Snowflake - Date is created

Microsoft Azure | databricks

Import from Snowflake - 2022-11-13 Python

File Edit View Run Help Last edit was 1 minute ago Give feedback

Cmd 1

```python
# Use dbutils secrets to get Snowflake credentials.
user = dbutils.secrets.get("data-warehouse", "<snowflake-user>")
password = dbutils.secrets.get("data-warehouse", "<snowflake-password>")

options = {
  "sfUrl": "<snowflake-url>",
  "sfUser": user,
  "sfPassword": password,
  "sfDatabase": "<snowflake-database>",
  "sfSchema": "<snowflake-schema>",
  "sfWarehouse": "<snowflake-cluster>"
}
```

Cmd 2

```python
# Generate a simple dataset containing five values and write the dataset to Snowflake.
spark.range(5).write \
  .format("snowflake") \
  .options(**options) \
  .option("dbtable", "table_name") \
  .save()
```

Cmd 3

```python
# Read the data written by the previous cell back.
df = spark.read \
  .format("snowflake") \
  .options(options) \
  .option("dbtable", "table_name") \
  .load()

display(df)
```

We have 4 commands in this notebook.

Let see each of them.

The first command allows us to setup connection between Azure Databricks workspace and Snowflake.

To successfully execute this command, you need to have:

Snowflake sysadmin credentials,

information about the Snowflake Datawarehouse like the URL, User, Password, Database name, Schema.

You also need to create a secret scope for Azure Databricks.

Cmd 1

```python
# Use dbutils secrets to get Snowflake credentials.
user = dbutils.secrets.get("snowflake", "snow-user")
password = dbutils.secrets.get("snowflake", "snow-password")

options = {
  "sfUrl": "https:/[illegible]canada-central.azure.snowflakecomputing.com/",
  "sfUser": user,
  "sfPassword": password,
  "sfDatabase": "DATABRICKS",
  "sfSchema": "public",
  "sfWarehouse": "COMPUTE_WH"
}
```

Command took 2.35 seconds -- by ahammamaliki@solutionsmetriques.ca at 11/12/2022, 8:01:48 PM on Spark

The second command generate a simple table with 5 values and write the data into Snowflake. The table is created automatically, and the data loaded for you

Cmd 2

```python
# Generate a simple dataset containing five values and write the dataset to Snowflake.
spark.range(5).write \
  .format("snowflake") \
  .options(**options) \
  .option("dbtable", "PUBLIC.Databricks_table") \
  .save()
```

The third one read the Data written by the previous command and display the content

Cmd 3

```python
# Read the data written by the previous cell back.
df = spark.read \
  .format("snowflake") \
  .options(**options) \
  .option("dbtable", "Databricks_table") \
  .load()
```

The fourth command write the data from the Spark Data frame and save if as Delta table

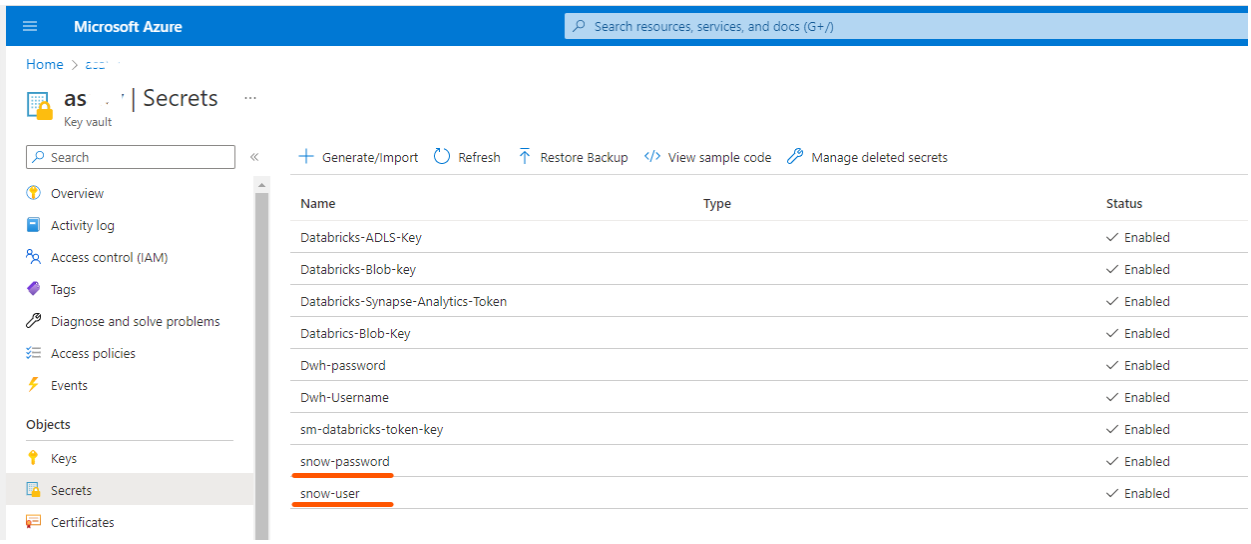
Cmd 4

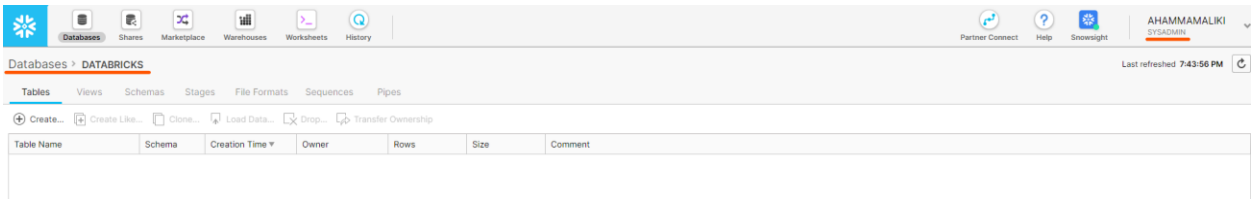
```python
# Write the data to a Delta table

df.write.format("delta").saveAsTable("sf_ingest_snowflake_table")
```

Note: for the purpose of this article, I already created a Database named DATABRICKS in snowflake, also create secret scope for Databricks and Azure key Vault secrets for Snowflake and create Cluster for the notebooks. You need to be at minimum SYSADMIN role in Snowflake.

You can learn how to create a scope for Databricks here: https://learn.microsoft.com/en-us/azure/databricks/security/secrets/secret-scopes

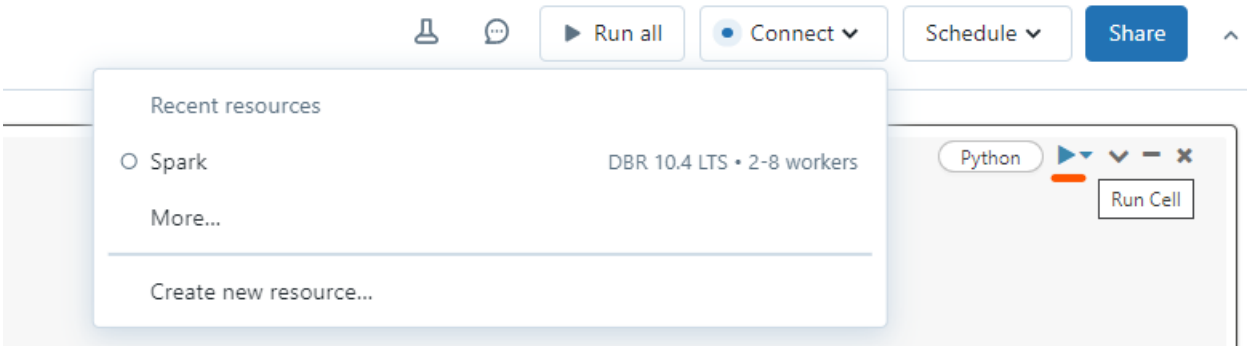
And Azure Key Vault Secret here: https://learn.microsoft.com/en-us/azure/key-vault/general/quick-create-portal?source=recommendations



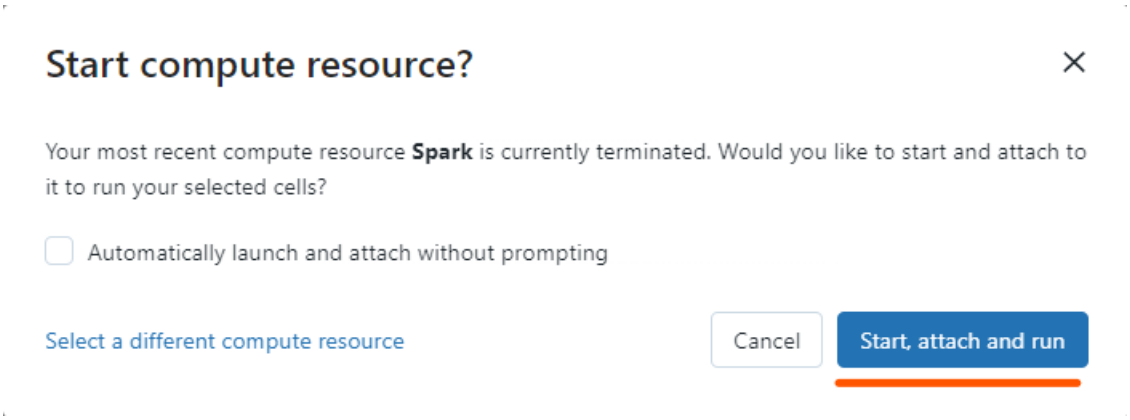


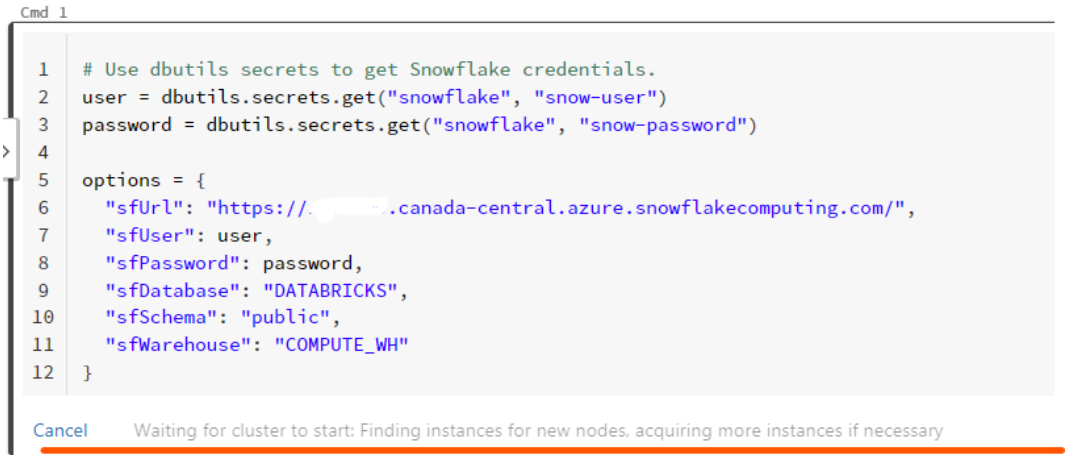
Now let's execute the first command

1- Select the run cell button to fire up the cluster and execute the first command

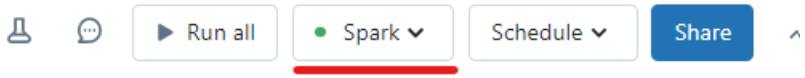


2- Click Start, attach, and run





The cluster is starting at this stage and try to scale out if needed.



Cmd 1

```python
# Use dbutils secrets to get Snowflake credentials.
user = dbutils.secrets.get("snowflake", "snow-user")
password = dbutils.secrets.get("snowflake", "snow-password")

options = {
  "sfUrl": "https:/[illegible].canada-central.azure.snowflakecomputing.com/",
  "sfUser": user,
  "sfPassword": password,
  "sfDatabase": "DATABRICKS",
  "sfSchema": "public",
  "sfWarehouse": "COMPUTE_WH"
}
```

Command took 2.35 seconds -- by ahammamaliki@solutionsmetriques.ca at 11/12/2022, 8:01:48 PM on Spark

The cluster is fire up and the command successfully executed.

3- Same way, click run cell for the second command

Cmd 2

```python
# Generate a simple dataset containing five values and write the dataset to Snowflake.
spark.range(5).write \
  .format("snowflake") \
  .options(**options) \
  .option("dbtable", "PUBLIC.Databricks_table") \
  .save()
```

(1) Spark Jobs

Job 0 View (Stages: 1/1)

Stage 0: 4/4

Command took 11.02 seconds -- by ahammamaliki@solutionsmetriques.ca at 11/12/2022, 8:08:58 PM on Spark

The second command executed successfully.

Now let check in the Snowflake Database the result of this execution. We should have a table named **Databricks_table** under **PUBLIC** schema.

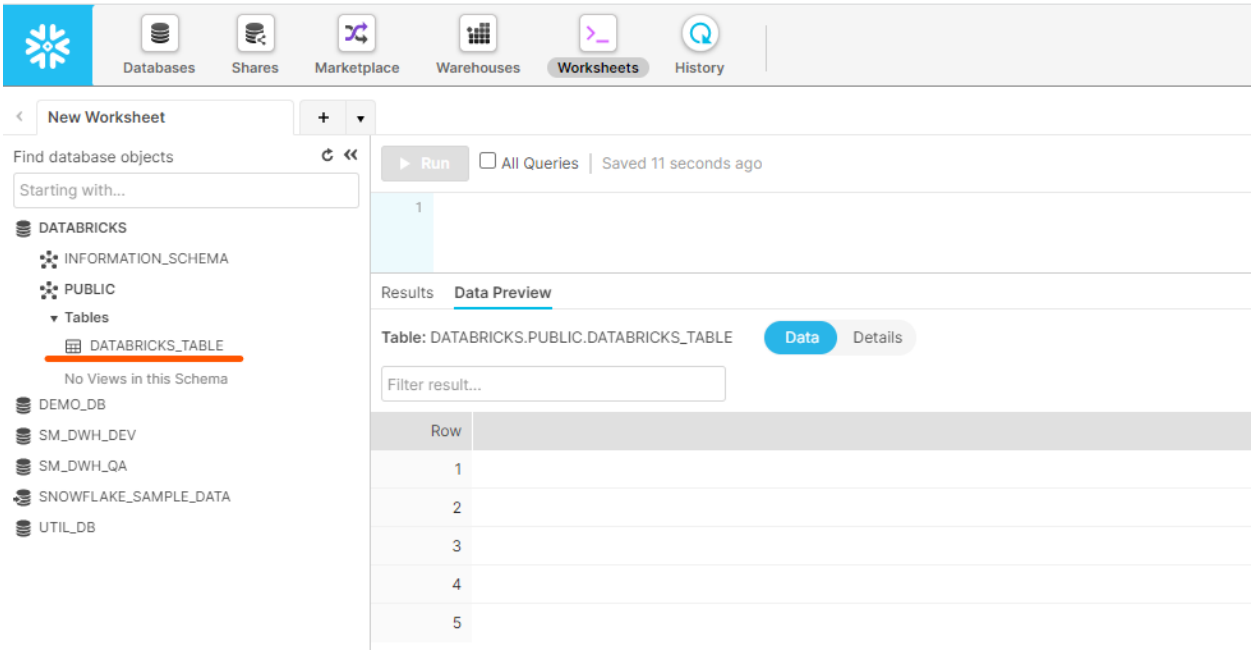
Databases Shares Marketplace Warehouses Worksheets History

Databases › DATABRICKS

Tables Views Schemas Stages File Formats Sequences Pipes

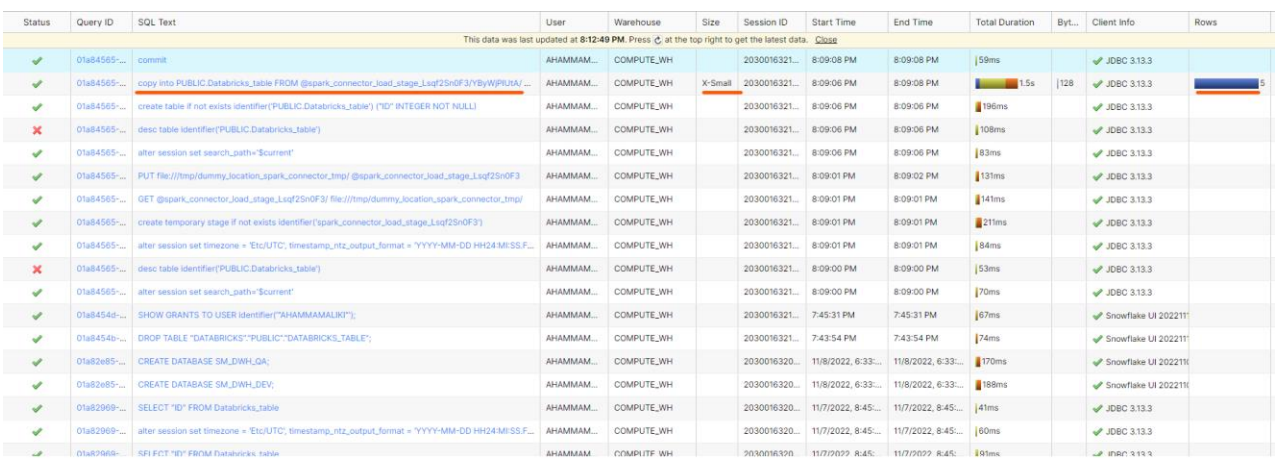
Create... Create Like... Clone... Load Data... Drop... Transfer Ownership

Table Name	Schema	Creation Time	Owner	Rows	Size	Comment
DATABRICKS_TABLE	PUBLIC	8:09:06 PM	SYSADMIN	5	1KB	



The table is created with **5 rows** under **PUBLIC** schema

We could see from the Snowflake History tab the detail about the operation



- Read the same table from snowflake and create a Delta table

In this section we are going to reload the table we have created in snowflake and save it as Delta table into Azure Databricks Lakehouse.

1- From the notebook let's execute the third command

Cmd 3

```python
# Read the data written by the previous cell back.
df = spark.read \
  .format("snowflake") \
  .options(**options) \
  .option("dbtable", "Databricks_table") \
  .load()
display(df)
```

(1) Spark Jobs

df: pyspark.sql.dataframe.DataFrame = [ID: decimal(38,0)]

Table +

	ID
1	1
2	0
3	2
4	3
5	4

Showing all 5 rows. | 3.46 seconds runtime

Command took 3.46 seconds -- by ahammamaliki@solutionsmetriques.ca at 11/12/2022, 8:21:50 PM on Spark

The command executed successfully with the 5 rows read from Snowflake table.

2- Execute the last command to save the data frame as a delta table

Cmd 4

```python
# Write the data to a Delta table

df.write.format("delta").saveAsTable("sf_ingest_snowflake_table")
```

(4) Spark Jobs

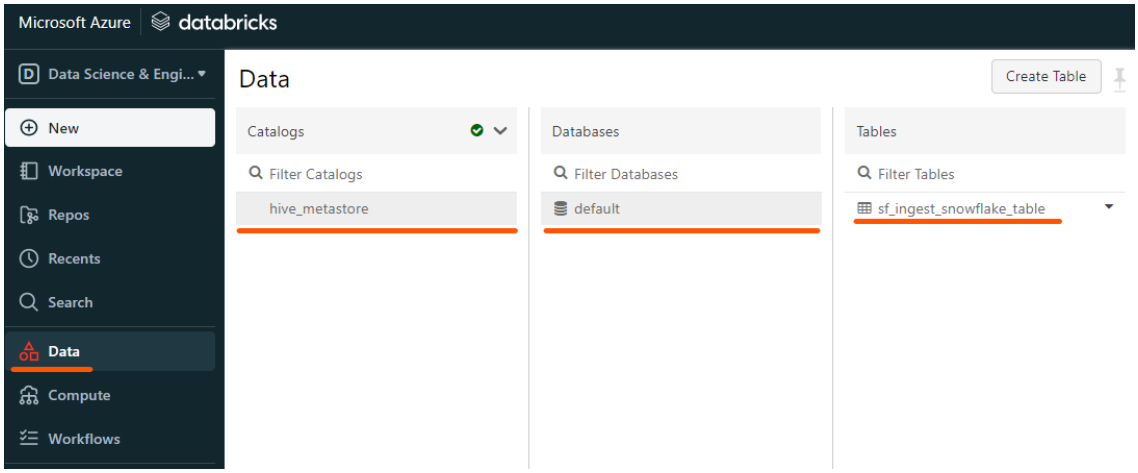
- Job 2 View (Stages: 1/1)
- Job 3 View (Stages: 1/1)
- Job 4 View (Stages: 1/1, 1 skipped)
- Job 5 View (Stages: 1/1, 2 skipped)

Command took 15.71 seconds -- by ahammamaliki@solutionsmetriques.ca at 11/12/2022, 8:26:43 PM on Spark

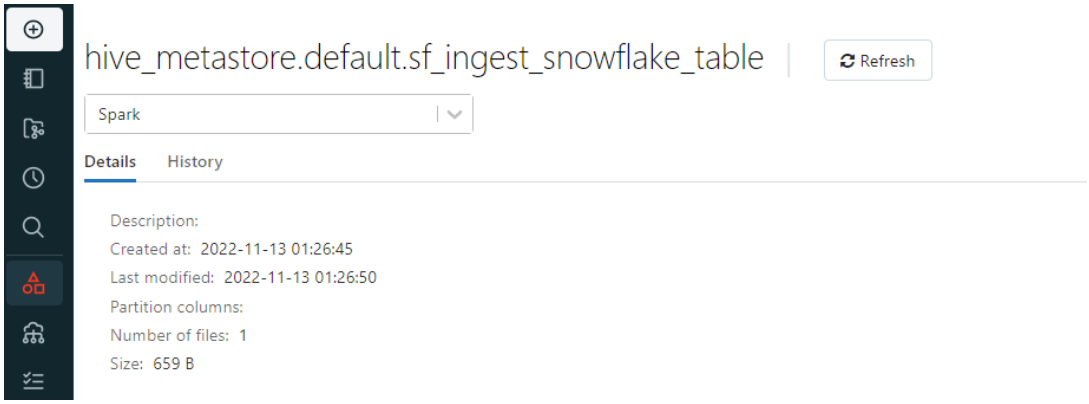
The command executed.

Now that we have our table created and saved as delta table, we can access to it through the Data menu or share it with the data analyst through and SQL endpoint.

3- Navigate and select Data menu bouton



4- Click on the table to access the table Details about the schema, Sample Data, History



Schema:

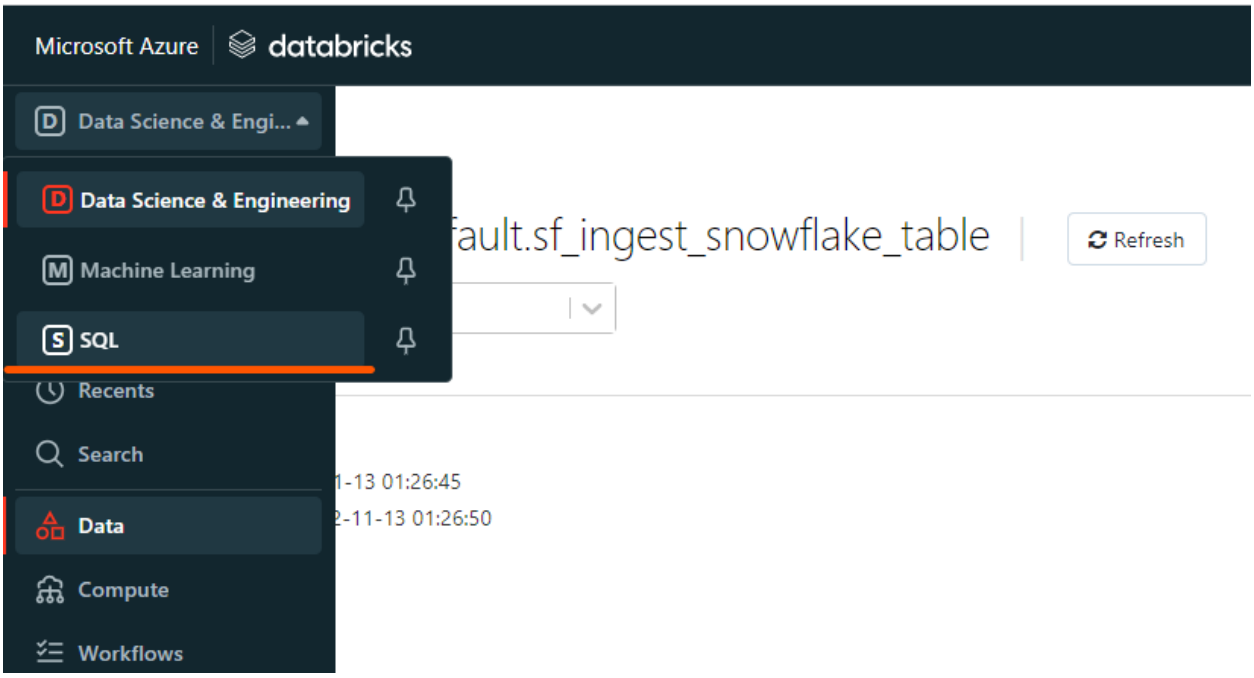
	col_name	data_type	comment
1	ID	decimal(38,0)	
2			
3	# Partitioning		
4	Not partitioned		

Sample Data:

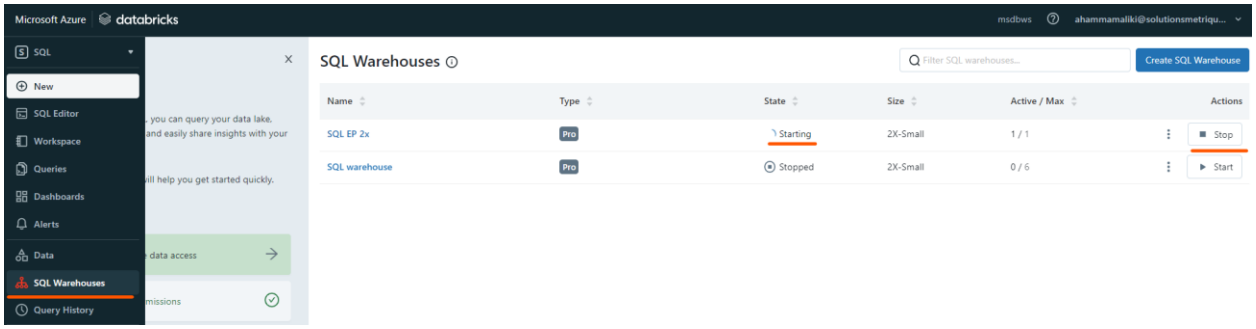
	ID
1	1
2	0
3	2
4	3
5	4

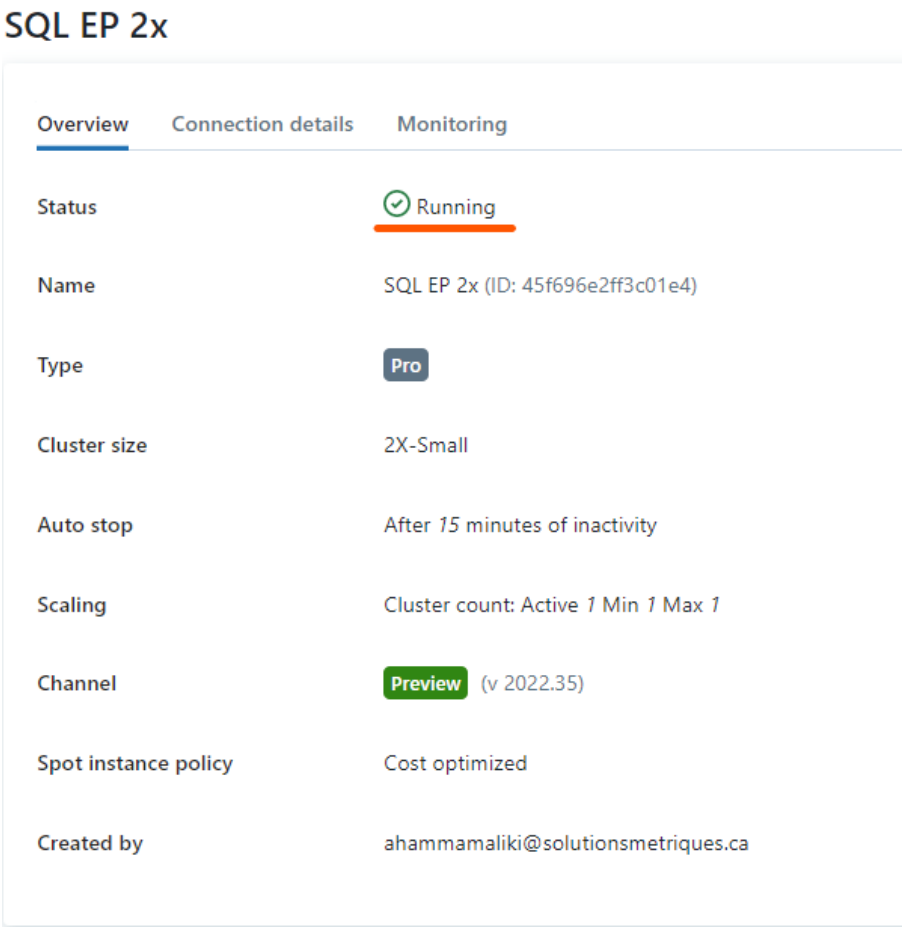
- Querying the same table by using the Databricks SQL app.

1- Use the App switcher and select **Databricks SQL app**



2- From the Azure Databricks SQL App page click on SQL Warehouses to start the cluster

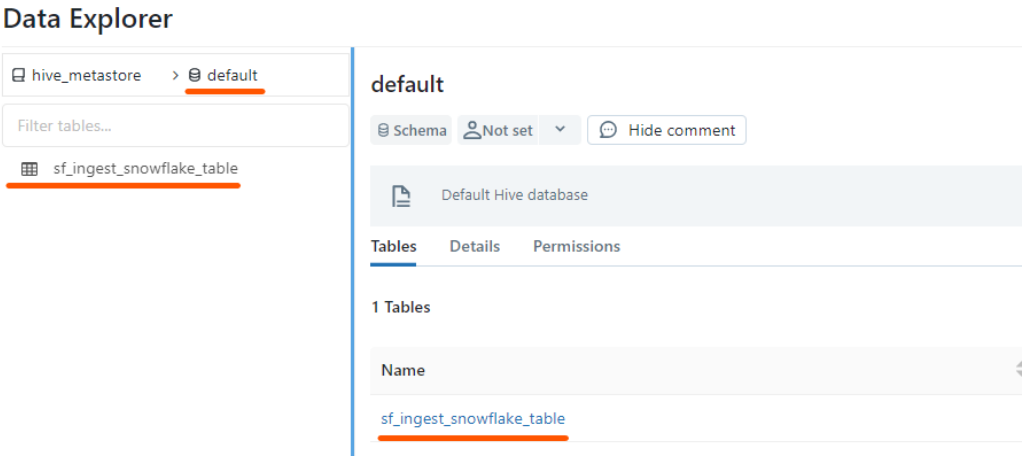




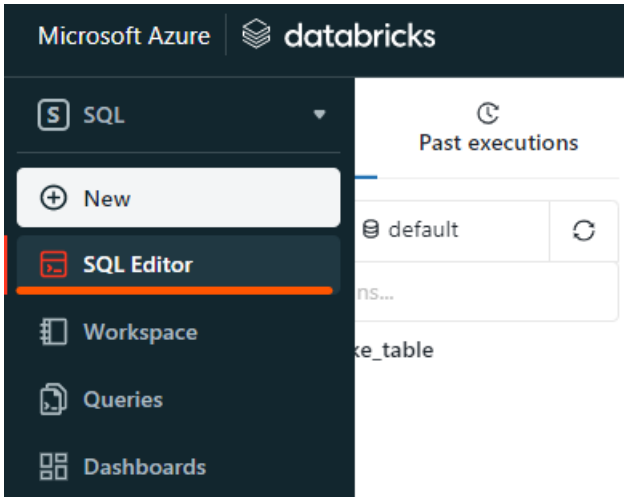
Now that we the confirmation lets switch to the Data Explorer page.

3- From the Azure Databricks SQL App page click on Data menu.

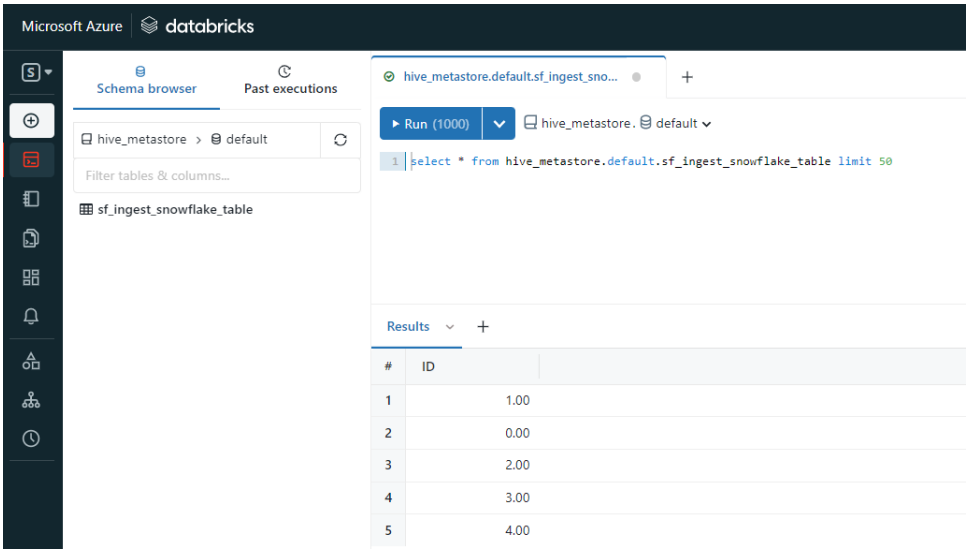
The table below was created through a notebook and store as delta table.



4- Now select the SQL Editor menu



5- Run the Query



Conclusion:

Azure Databricks **Add data** (mode preview) feature page have pre-build notebooks and pre-build data ingestion apps that you can use to fast Ingest, Extract Transform and Load data into your Organisation Data platform. It is also helps you applying the best practices and having a easy hands on to the Azure Databricks workspace.