Lab 05: Transferring large data volumes with Fabric Data Factory

Introduction

In this lab, you will ingest data from a Azure SQL Database to the Managed Lakehouse Sales table. The objective is to understand how to connect to this external database and how the Fabric Data Factory Pipeline can scale when having to transfer 10M, 50M and 100M of rows.

Objectives

After completing this lab, you will be better able to:

1. Connect to an Azure SQL Database and read the tables
2. Set up and run the Fabric Data Factory pipeline to transfer data from Azure SQL tables to a table in the managed Lakehouse.

Estimated time to complete this lab

60 minutes

Contents

Lab Prerequisites

- Workspace: Fabric, Power Premium or Fabric trial
- Individual license: Power Pro or Premium Per User account

Information provided by your training provider

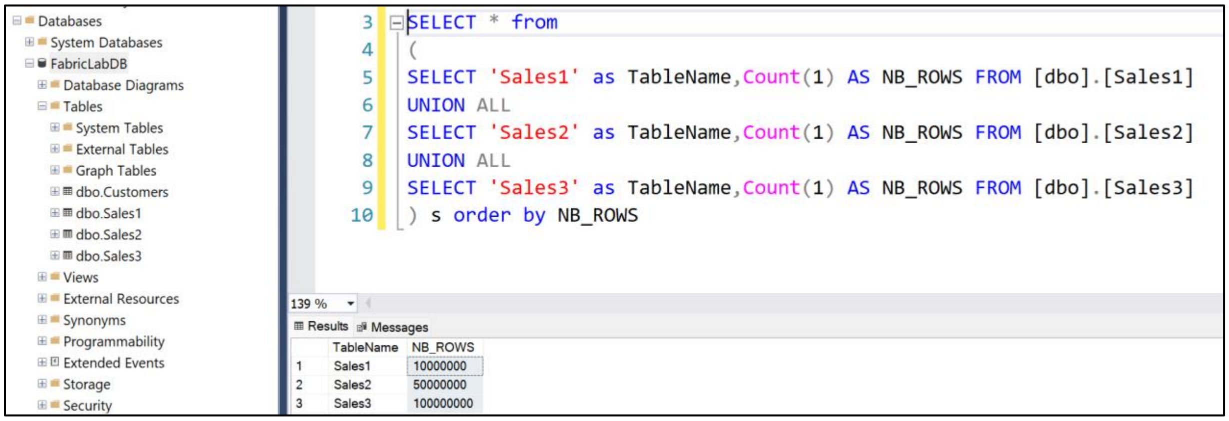
- Trial tenant (if applicable): login & password, workspace to use for the lab.
- Azure SQL Server Database: server name, login and password.

Task 1: Setup the connection to the Azure SQL Database

In this task, you will establish the Linked Service connection to an Azure SQL Database.

This database named FabricLabDB contains 3 tables with the same structure than the managed Sales table in the Lakehouse :

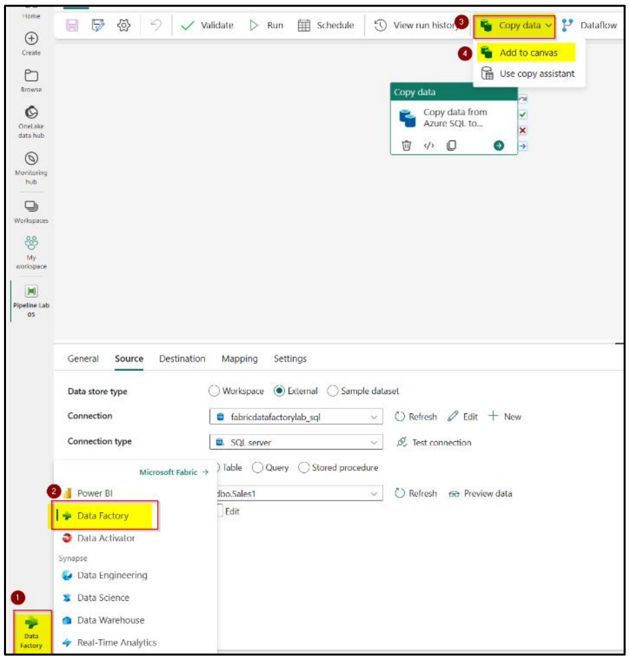
- Sales1 contains 10M rows
- Sales2 contains 50M rows
- Sales3 contains 100M rows



```
SELECT * from
(
SELECT 'Sales1' as TableName,Count(1) AS NB_ROWS FROM [dbo].[Sales1]
UNION ALL
SELECT 'Sales2' as TableName,Count(1) AS NB_ROWS FROM [dbo].[Sales2]
UNION ALL
SELECT 'Sales3' as TableName,Count(1) AS NB_ROWS FROM [dbo].[Sales3]
) s order by NB_ROWS
```

	TableName	NB_ROWS
1	Sales1	10000000
2	Sales2	50000000
3	Sales3	100000000

- On the MS Fabric menu, navigate to Data Factory and click on the Data pipeline. Provide the name Pipeline Lab 05 for this new Lab.
- On the newly created data pipeline, click on “Copy Data” then “Add to canvas”



Data store type: Workspace [✓ External] Sample dataset

Connection: fabricdatafactorylab_sql

Connection type: SQL server

Table Query Stored procedure

dbo.Sales1

- Configure the Data Copy Activity Task
    - Under the **General Tab**, provide the name for the Pipeline: ***Copy data from Azure SQL to Lakehouse***. The other settings remain unchanged.
    -
  - On the **Source Tab** choose the following options:
      - Data store type: External
      - Connection: Click on new and select Azure SQL Server, click on Continue.

      Server: *<database server name>*.database.windows.net (information given by your trainer.

      Database: fabriclabdb

      Connection: Create new connection

      Connection name: azuresqldb

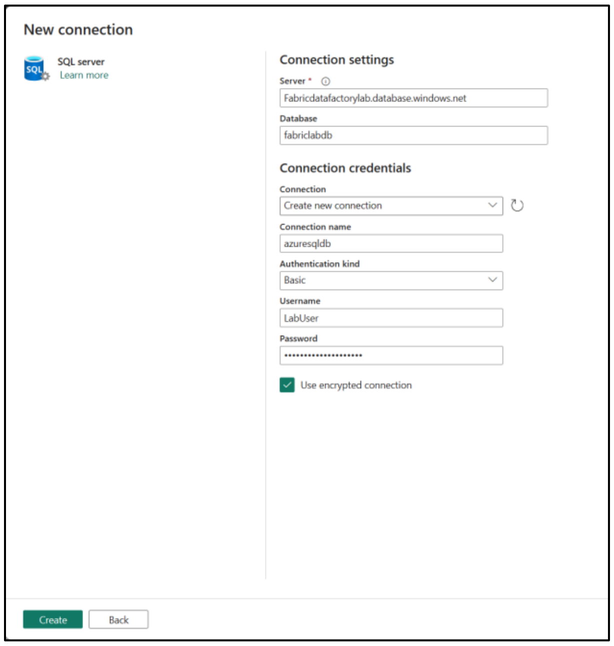
      Authentication kind: Basic

      Username: LabUser (to be confirmed by your trainer)

      Password: MicrosoftFabric@2023 (to be confirmed by your trainer)

      Thick Use **Encrypted Connection**

Make sure that the settings will look like this:


New connection

SQL server
Learn more

Connection settings

Server *: Fabricdatafactorylab.database.windows.net

Database: fabriclabdb

Connection credentials

Connection: Create new connection

Connection name: azuresqldb

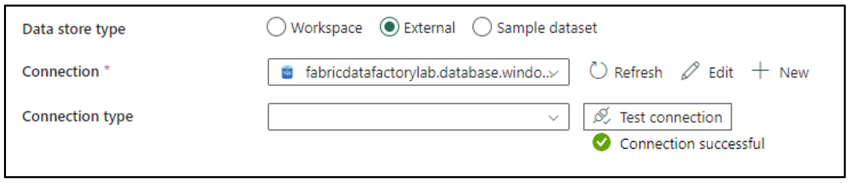
Authentication kind: Basic

Username: LabUser

Password: ••••••••••••••••••••

✓ Use encrypted connection

- Click on Test Connection to make sure that the connection is successful.

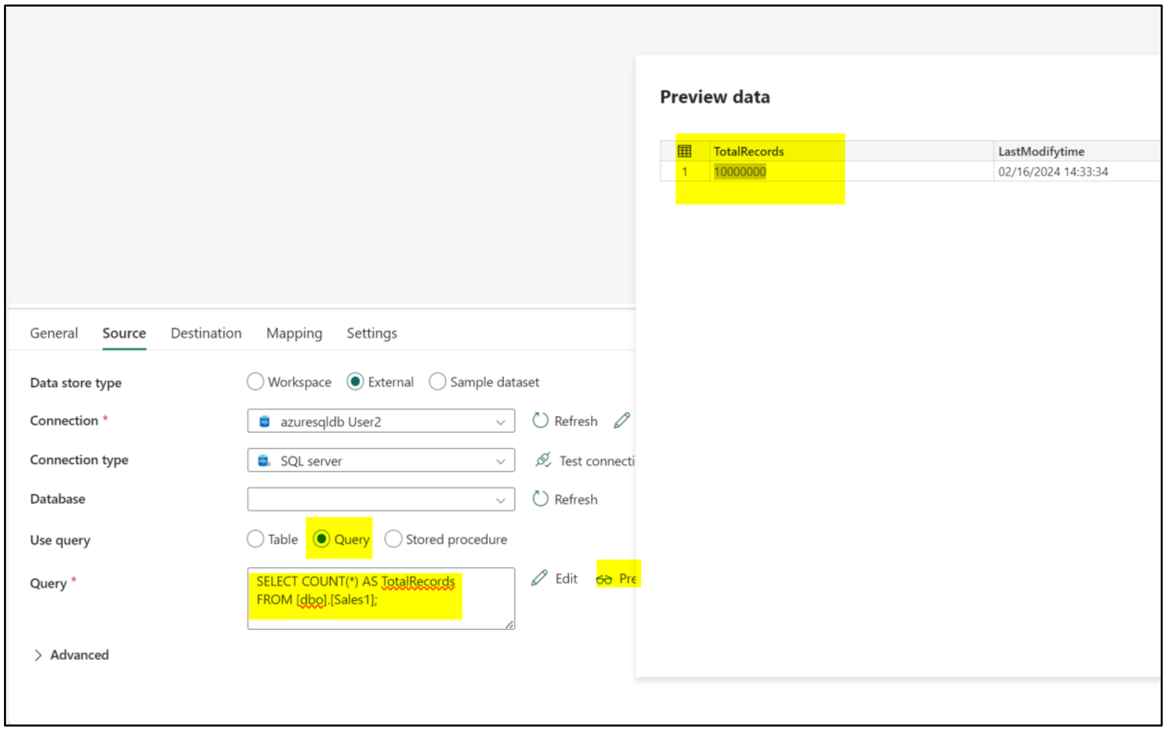


Data store type	Workspace [✓ External] Sample dataset
Connection *	fabricdatafactorylab.database.windo...
Connection type	

Test connection — Connection successful

- Use query: Select Table
- **Table optional**: To find out how many records are on the dbo.Sales1, as a test you can select Query and paste this query:

```
SELECT COUNT(*) AS TotalRecords
FROM [dbo].[Sales1];
```

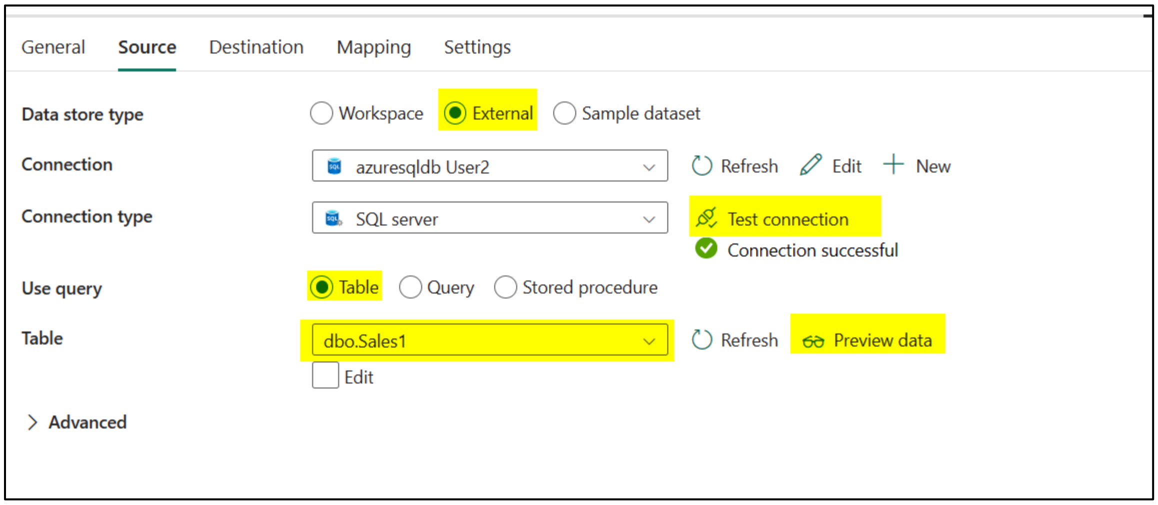


Preview data

TotalRecords	LastModifytime
10000000	02/16/2024 14:33:34

Data store type	Workspace [✓ External] Sample dataset
Connection *	azuresqldb User2
Connection type	SQL server
Database	
Use query	Table [✓ Query] Stored procedure
Query *	SELECT COUNT(*) AS TotalRecords FROM [dbo].[Sales1];

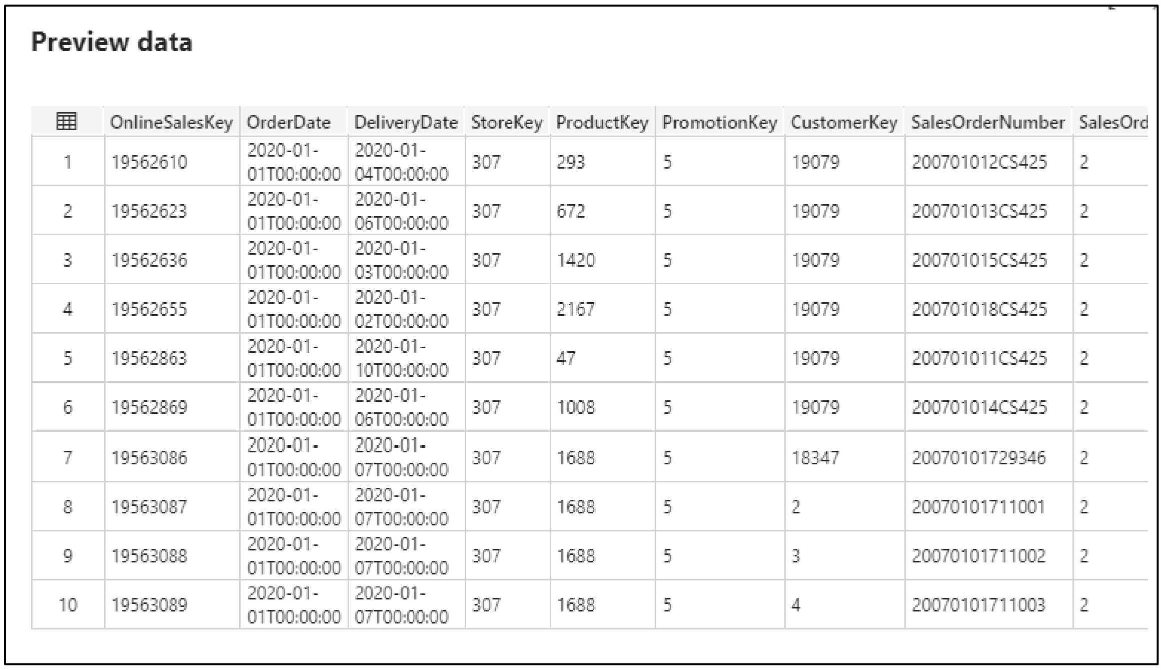
- As you can see, the table dbo.Sales1 contains 10M of rows.
- **Table**: select **dbo.Sales1**. Click on Preview Data to make sure that you can read the data from the Database



General | **Source** | Destination | Mapping | Settings

Data store type	Workspace [✓ External] Sample dataset
Connection	azuresqldb User2 — Refresh, Edit, New
Connection type	SQL server — Test connection: Connection successful
Use query	[✓ Table] Query Stored procedure
Table	dbo.Sales1 — Refresh, Preview data; Edit

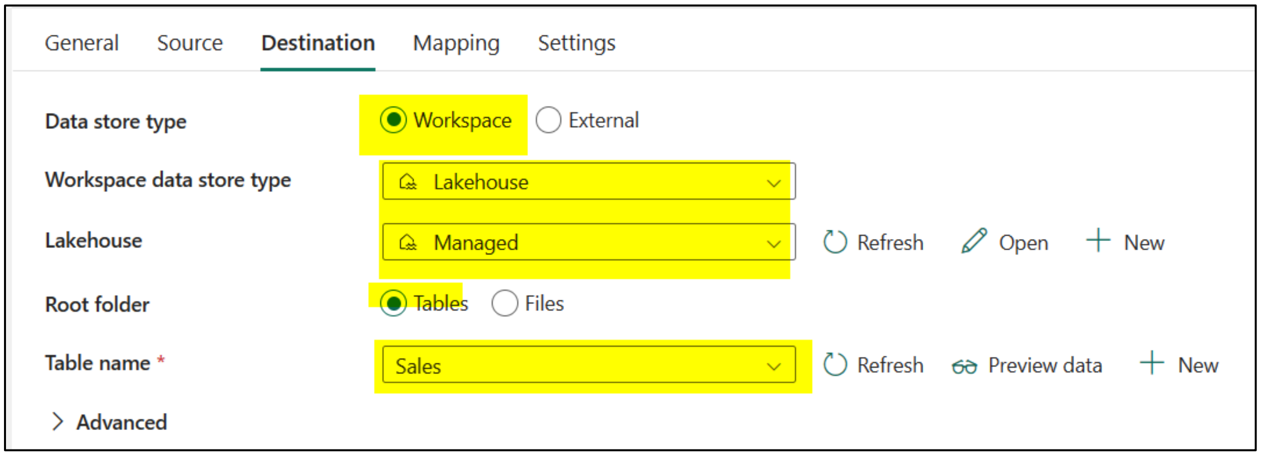
> Advanced



Preview data

	OnlineSalesKey	OrderDate	DeliveryDate	StoreKey	ProductKey	PromotionKey	CustomerKey	SalesOrderNumber	SalesOrd
1	19562610	2020-01-01T00:00:00	2020-01-04T00:00:00	307	293	5	19079	200701012CS425	2
2	19562623	2020-01-01T00:00:00	2020-01-06T00:00:00	307	672	5	19079	200701013CS425	2
3	19562636	2020-01-01T00:00:00	2020-01-03T00:00:00	307	1420	5	19079	200701015CS425	2
4	19562655	2020-01-01T00:00:00	2020-01-02T00:00:00	307	2167	5	19079	200701018CS425	2
5	19562863	2020-01-01T00:00:00	2020-01-10T00:00:00	307	47	5	19079	200701011CS425	2
6	19562869	2020-01-01T00:00:00	2020-01-06T00:00:00	307	1008	5	19079	200701014CS425	2
7	19563086	2020-01-01T00:00:00	2020-01-07T00:00:00	307	1688	5	18347	20070101729346	2
8	19563087	2020-01-01T00:00:00	2020-01-07T00:00:00	307	1688	5	2	20070101711001	2
9	19563088	2020-01-01T00:00:00	2020-01-07T00:00:00	307	1688	5	3	20070101711002	2
10	19563089	2020-01-01T00:00:00	2020-01-07T00:00:00	307	1688	5	4	20070101711003	2

- Under the **Destination** tab, select the following options:
  - Data store type: Workspace
  - Workspace data store type: Lakehouse
  - Lakehouse: Contoso (your managed Lakehouse created previously)
  - Root folder: Tables
  - Table name: Sales
  - Make sure that the settings will look like this:

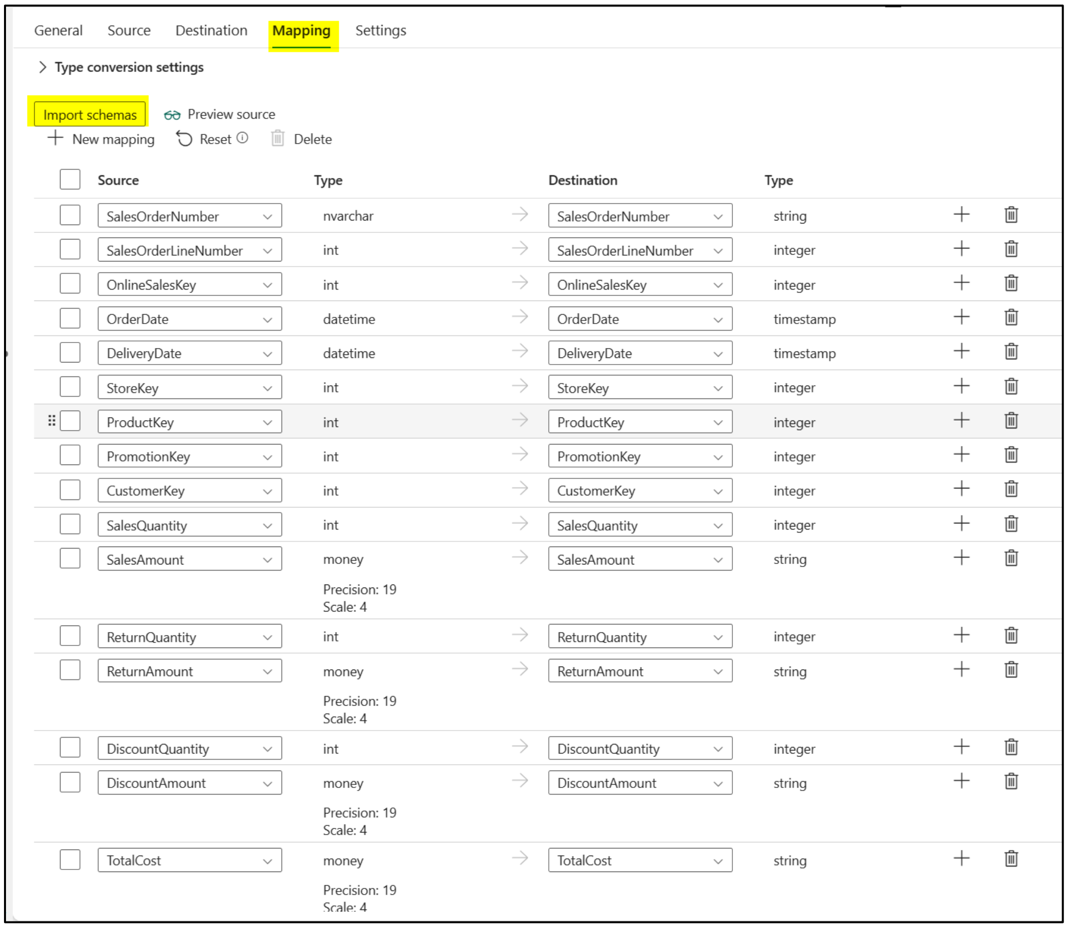


General | Source | **Destination** | Mapping | Settings

Data store type	[✓ Workspace] External
Workspace data store type	Lakehouse
Lakehouse	Managed
Root folder	[✓ Tables] Files
Table name *	Sales

Advanced

- On the **Mapping** tab select Import schemas:



General | Source | Destination | **Mapping** | Settings

Type conversion settings

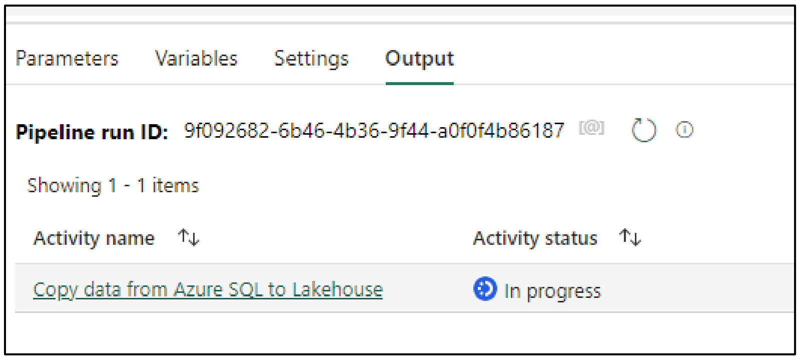
Import schemas | Preview source | New mapping | Reset | Delete

Source	Type	Destination	Type
SalesOrderNumber	nvarchar	SalesOrderNumber	string
SalesOrderLineNumber	int	SalesOrderLineNumber	integer
OnlineSalesKey	int	OnlineSalesKey	integer
OrderDate	datetime	OrderDate	timestamp
DeliveryDate	datetime	DeliveryDate	timestamp
StoreKey	int	StoreKey	integer
ProductKey	int	ProductKey	integer
PromotionKey	int	PromotionKey	integer
CustomerKey	int	CustomerKey	integer
SalesQuantity	int	SalesQuantity	integer
SalesAmount	money Precision: 19 Scale: 4	SalesAmount	string
ReturnQuantity	int	ReturnQuantity	integer
ReturnAmount	money Precision: 19 Scale: 4	ReturnAmount	string
DiscountQuantity	int	DiscountQuantity	integer
DiscountAmount	money Precision: 19 Scale: 4	DiscountAmount	string
TotalCost	money Precision: 19 Scale: 4	TotalCost	string

- On the **Settings** tab, no changes are required
- Validate and Save the Pipeline

- You can now **Run** your pipeline

- During the pipeline execution, you can monitor the progression by clicking on the Activity in the Output section

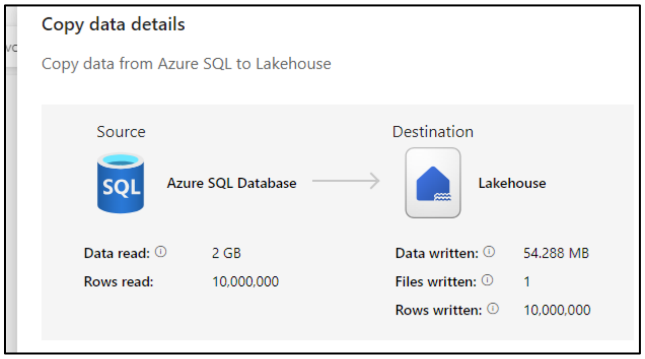


Pipeline run ID: 9f092682-6b46-4b36-9f44-a0f0f4b86187

Showing 1 - 1 items

Activity name	Activity status
Copy data from Azure SQL to Lakehouse	In progress

- Details should appear, and 10M rows should we be written.

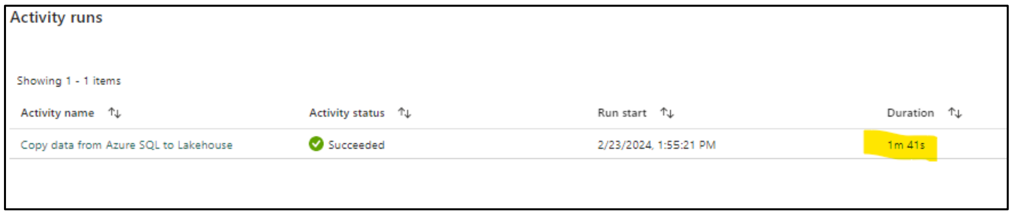


Copy data details

Copy data from Azure SQL to Lakehouse

Data read:	2 GB
Rows read:	10,000,000
Data written:	54.288 MB
Files written:	1
Rows written:	10,000,000

- Note the time spent to execute the pipeline to transfer 10M of rows

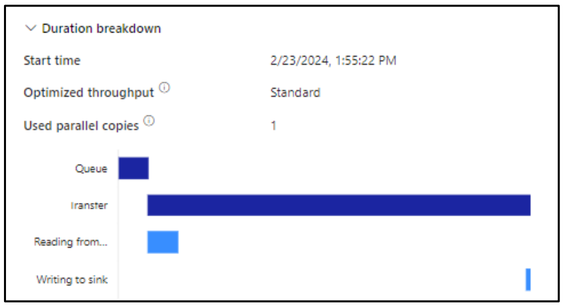


Activity runs

Showing 1 - 1 items

Activity name	Activity status	Run start	Duration
Copy data from Azure SQL to Lakehouse	Succeeded	2/23/2024, 1:55:21 PM	1m 41s

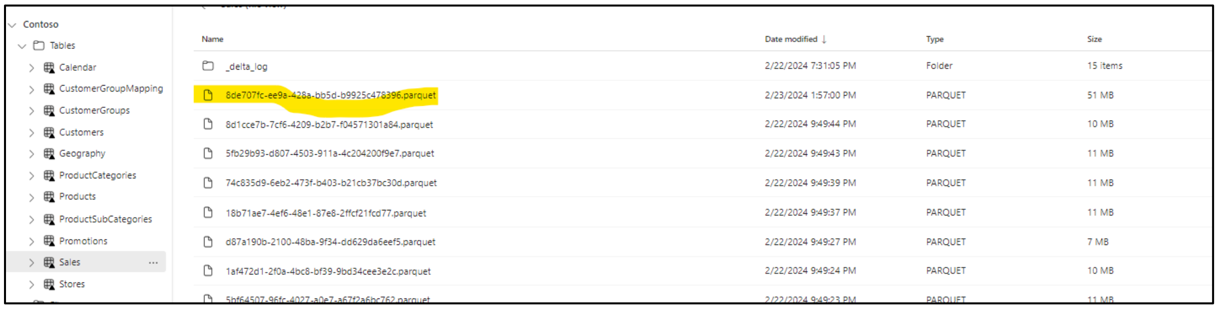
- As demonstrated in the lab 3, you can use the Duration Breakdown to understand where the execution time is mostly consumed – depending on the status of the Azure SQL DB used during the lab, the Reading from source (Light Blue) might be bigger.



Duration breakdown

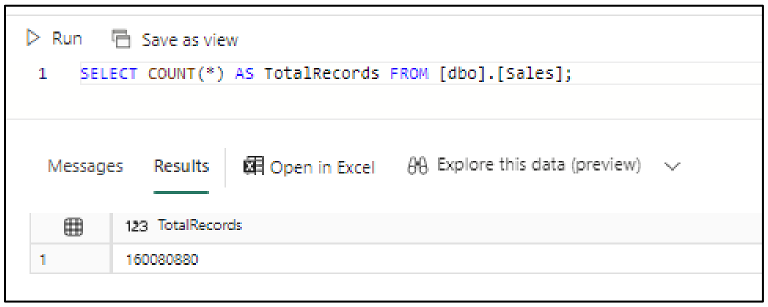
Start time	2/23/2024, 1:55:22 PM
Optimized throughput	Standard
Used parallel copies	1

- Using the Lakehouse explorer, you can also see the last Parquet file written during the last pipeline execution. This parquet file weights 50MB and contains 10M of records.



Name	Date modified ↓	Type	Size
_delta_log	2/22/2024 7:31:05 PM	Folder	15 items
8de707fc-ee9a-428a-bb5d-b9925c478396.parquet	2/23/2024 1:57:00 PM	PARQUET	51 MB
8d1cce7b-7cf6-4209-b2b7-f04571301a84.parquet	2/22/2024 9:49:44 PM	PARQUET	10 MB
5fb29b93-d807-4503-911a-4c204200f9e7.parquet	2/22/2024 9:49:43 PM	PARQUET	11 MB
74c835d9-6eb2-473f-b403-b21cb37bc30d.parquet	2/22/2024 9:49:39 PM	PARQUET	11 MB
18b71ae7-4ef6-48e1-87e8-2ffcf21fcd77.parquet	2/22/2024 9:49:37 PM	PARQUET	11 MB
d87a190b-2100-48ba-9f34-dd629da6eef5.parquet	2/22/2024 9:49:27 PM	PARQUET	7 MB
1af472d1-2f0a-4bc8-bf39-9bd34cee3e2c.parquet	2/22/2024 9:49:24 PM	PARQUET	10 MB
5bf64507-96fc-4027-a0e7-a67f2a6bc762.parquet	2/22/2024 9:49:23 PM	PARQUET	11 MB

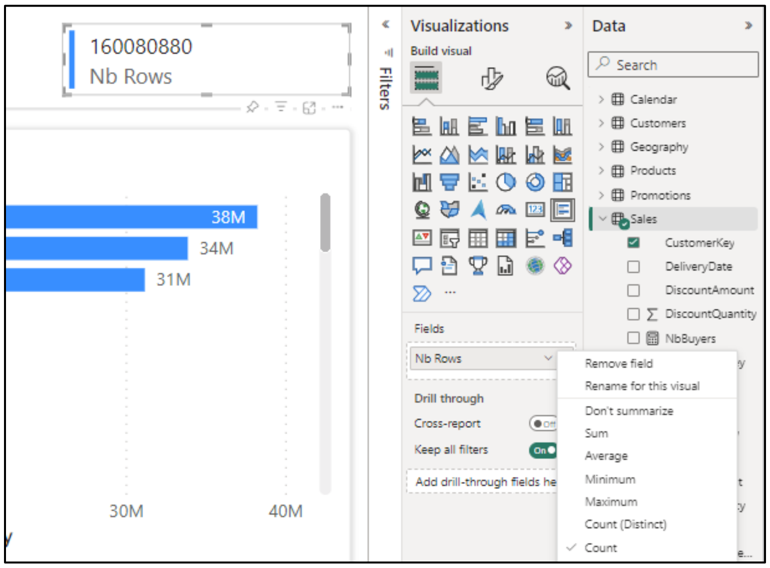
- It is also recommended to use the available analytics tools in Fabric to see the new ingested in the Sales Tables
  - **The SQL Analytics Endpoint:**



```sql
SELECT COUNT(*) AS TotalRecords FROM [dbo].[Sales];
```

	TotalRecords
1	160080880

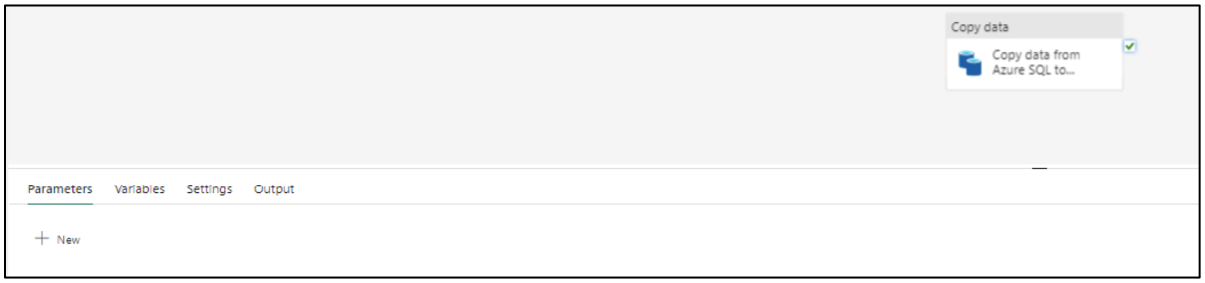
  - **The Power BI report**
    - You can add a simple visual card to count the number of rows in the Sales tables, and you don't have to refresh the report or the semantic model, thanks to the **Direct Lakehouse Mode**



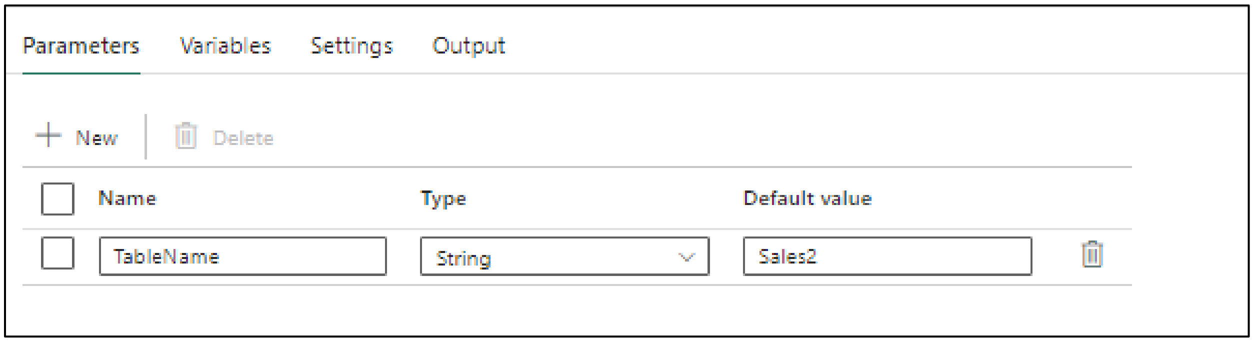
Task 2: Use Data Pipeline Parameters to change the source table

In this task, you will declare a Parameter to be used in the Pipeline. Then you will load data from the second table (50M of records)

- Open the Fabric Data Factory Pipeline, and go to the **Parameters** section

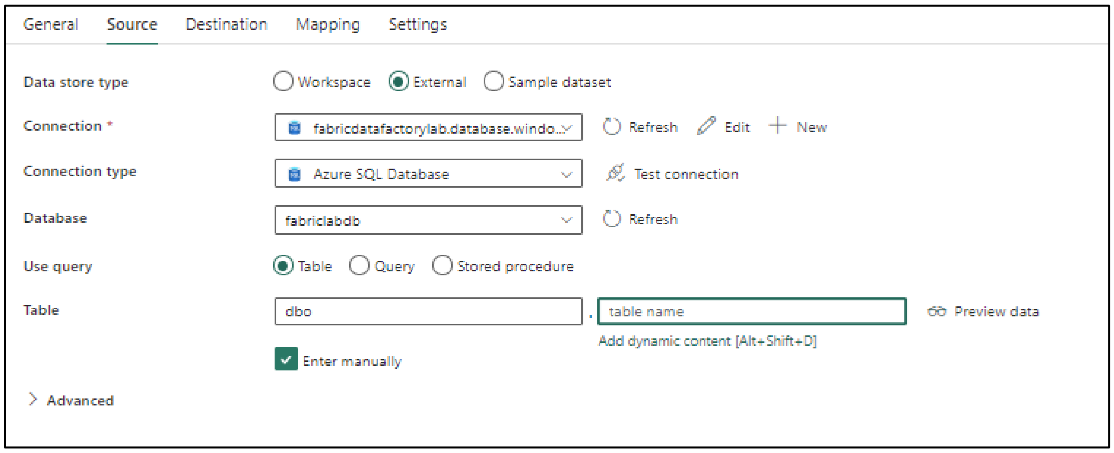


- Define the following parameter:
    - Name : **TableName**
    - Type : **String**
    - DefaultValue : **Sales2**



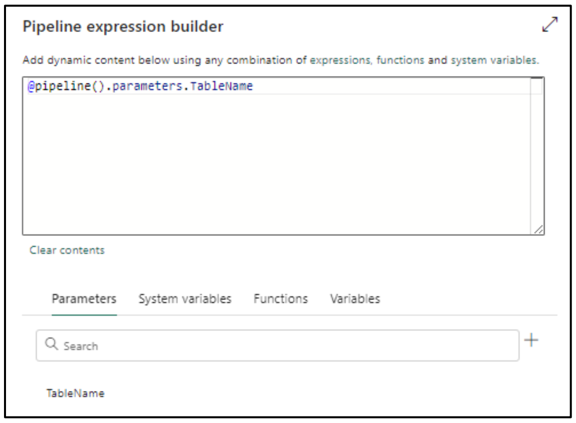
Name	Type	Default value
TableName	String	Sales2

- Select the **Copy Data Activity** in the Pipeline, go to the **Source**, and enable the **Enter manually** check box
- Click on the **Add dynamic content** link to access to the Pipeline expression builder



Data store type	Workspace [✓ External] Sample dataset
Connection *	fabricdatafactorylab.database.windo...
Connection type	Azure SQL Database
Database	fabriclabdb
Use query	[✓ Table] Query Stored procedure
Table	dbo . table name
	Add dynamic content [Alt+Shift+D]
	✓ Enter manually

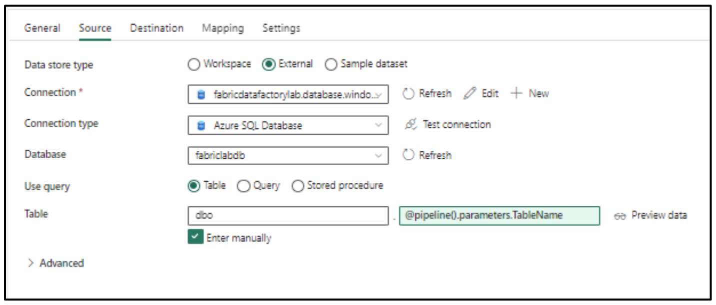
- From the Pipeline Expression Builder, drag and drop the TableName parameter, and the following expression should appear. Click on OK.


Pipeline expression builder

Add dynamic content below using any combination of expressions, functions and system variables.

```
@pipeline().parameters.TableName
```

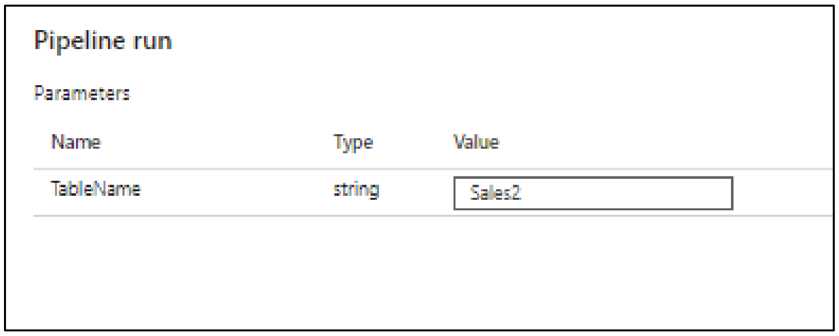
- The expression should be now visible in the table name section.



Data store type	Workspace [✓ External] Sample dataset
Connection *	fabricdatafactorylab.database.windo...
Connection type	Azure SQL Database
Database	fabriclabdb
Use query	[✓ Table] Query Stored procedure
Table	dbo . @pipeline().parameters.TableName
	✓ Enter manually

- The expression should be now visible in the table name section.

- Save the pipeline and click on Run.
- As the Pipeline now contains a parameter, you will have the choice to keep the default parameter value, or change it before the execution. Keep the default parameter value for this execution to get data from the table Sales2.

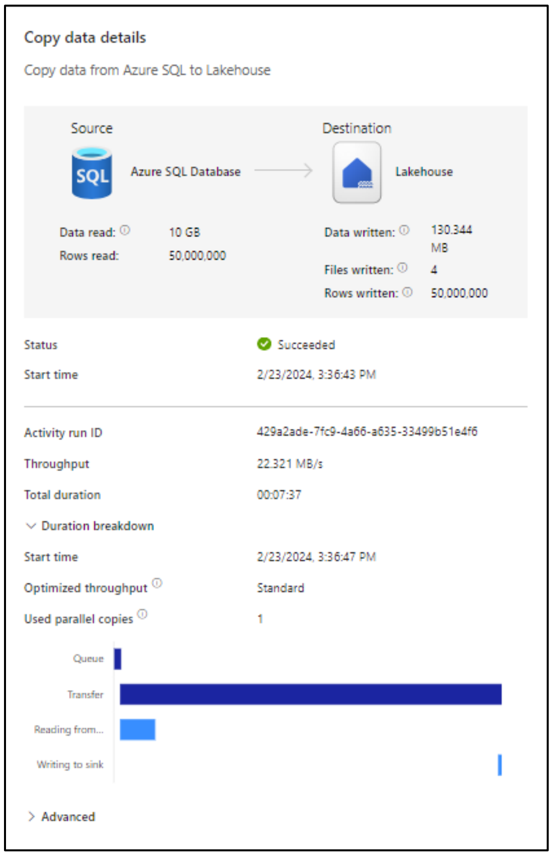


Pipeline run

Parameters

Name	Type	Value
TableName	string	Sales2

- At the end of the pipeline execution, you can observe that :
  - 50M of records were loaded
  - 4 files were written
  - The parallelism was 1 (in the next lab you will configure the parallelism using a parameter).



Copy data details

Copy data from Azure SQL to Lakehouse

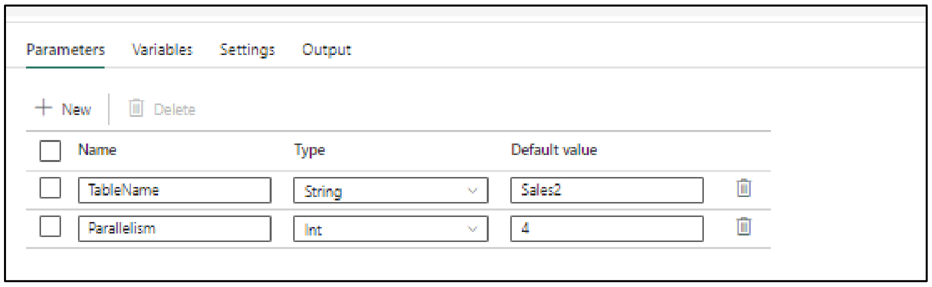
Source: Azure SQL Database → Destination: Lakehouse

Data read:	10 GB
Rows read:	50,000,000
Data written:	130.344 MB
Files written:	4
Rows written:	50,000,000
Status	Succeeded
Start time	2/23/2024, 3:36:43 PM
Activity run ID	429a2ade-7fc9-4a66-a635-33499b51e4f6
Throughput	22.321 MB/s
Total duration	00:07:37
Duration breakdown	
Start time	2/23/2024, 3:36:47 PM
Optimized throughput	Standard
Used parallel copies	1

Task 3: Optimize Data Pipeline performance with parallelism and partitions

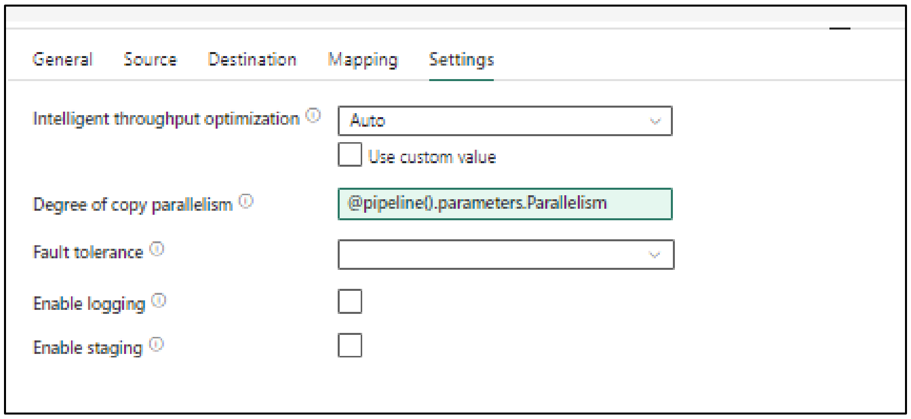
In this task, you will declare a second Parameter to be used as the level of parallelism in order to accelerate the data transfer and optimize performance by reducing the execution duration.

- Create a second parameter with the following settings:
  - Name: Parallelism
  - Type: Int
  - Default Value: 4



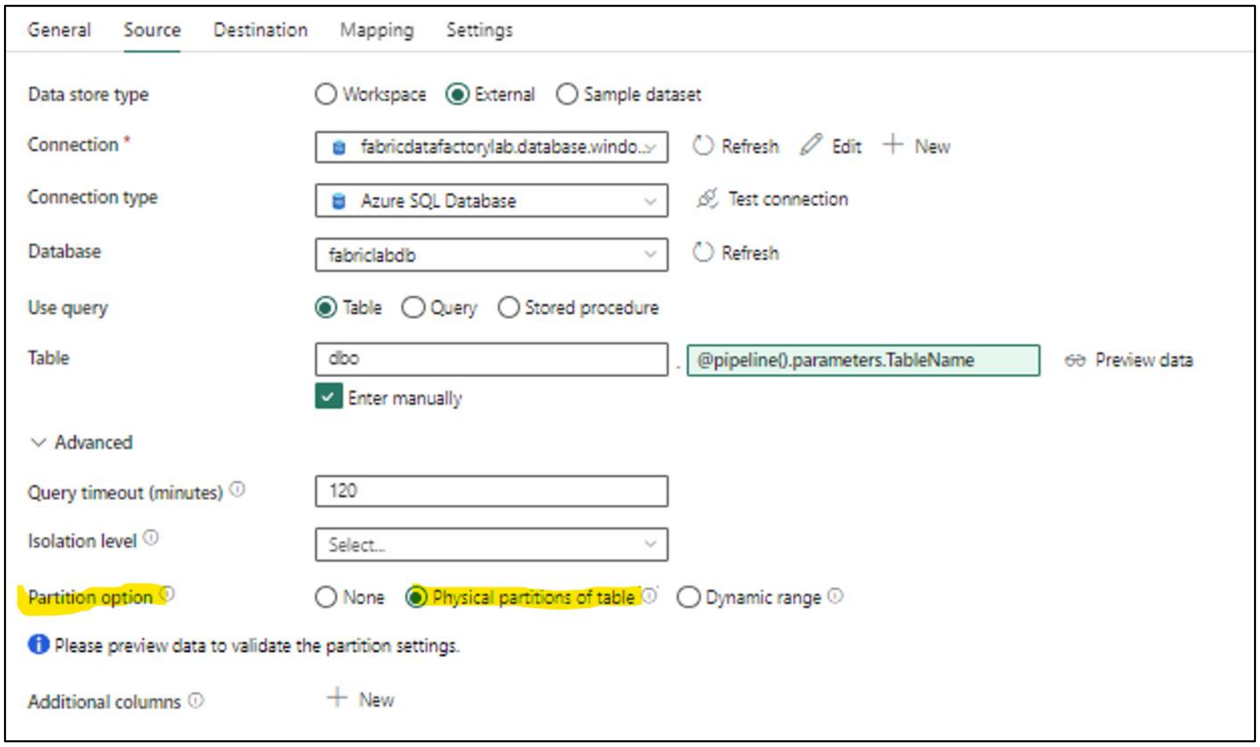
Name	Type	Default value
TableName	String	Sales2
Parallelism	Int	4

- Use this parameter to dynamically configure the Degree of copy parallelism (in the settings section of the Copy activity)



Intelligent throughput optimization: Auto
Use custom value
Degree of copy parallelism: @pipeline().parameters.Parallelism
Fault tolerance
Enable logging
Enable staging

- On the Source section, enable the partitioning by selecting **Physical partitions** of table in the **Partition option** item.



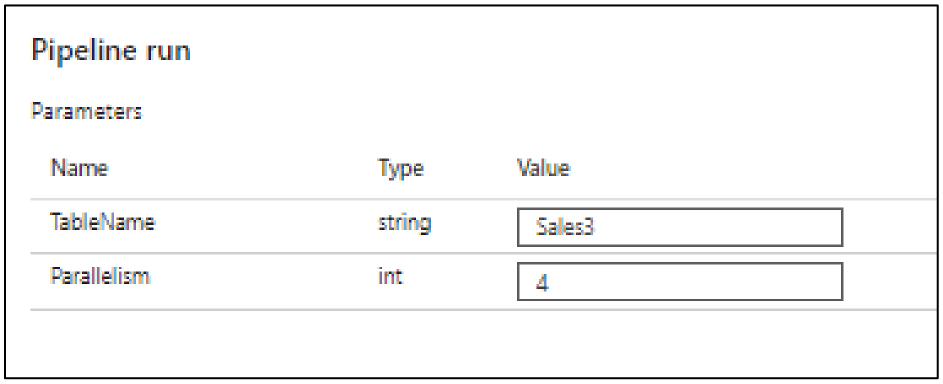
General | Source | Destination | Mapping | Settings

Data store type	Workspace [✓ External] Sample dataset
Connection *	fabricdatafactorylab.database.windo...
Connection type	Azure SQL Database
Database	fabriclabdb
Use query	[✓ Table] Query Stored procedure
Table	dbo . @pipeline().parameters.TableName
	[✓ Enter manually]
Query timeout (minutes)	120
Isolation level	Select...
Partition option	None [✓ Physical partitions of table] Dynamic range

Please preview data to validate the partition settings.

Additional columns + New

- Save the pipeline and run it with the following configuration:
    - TableName : Sales3 (100M of rows)
    - Parralelism : 4

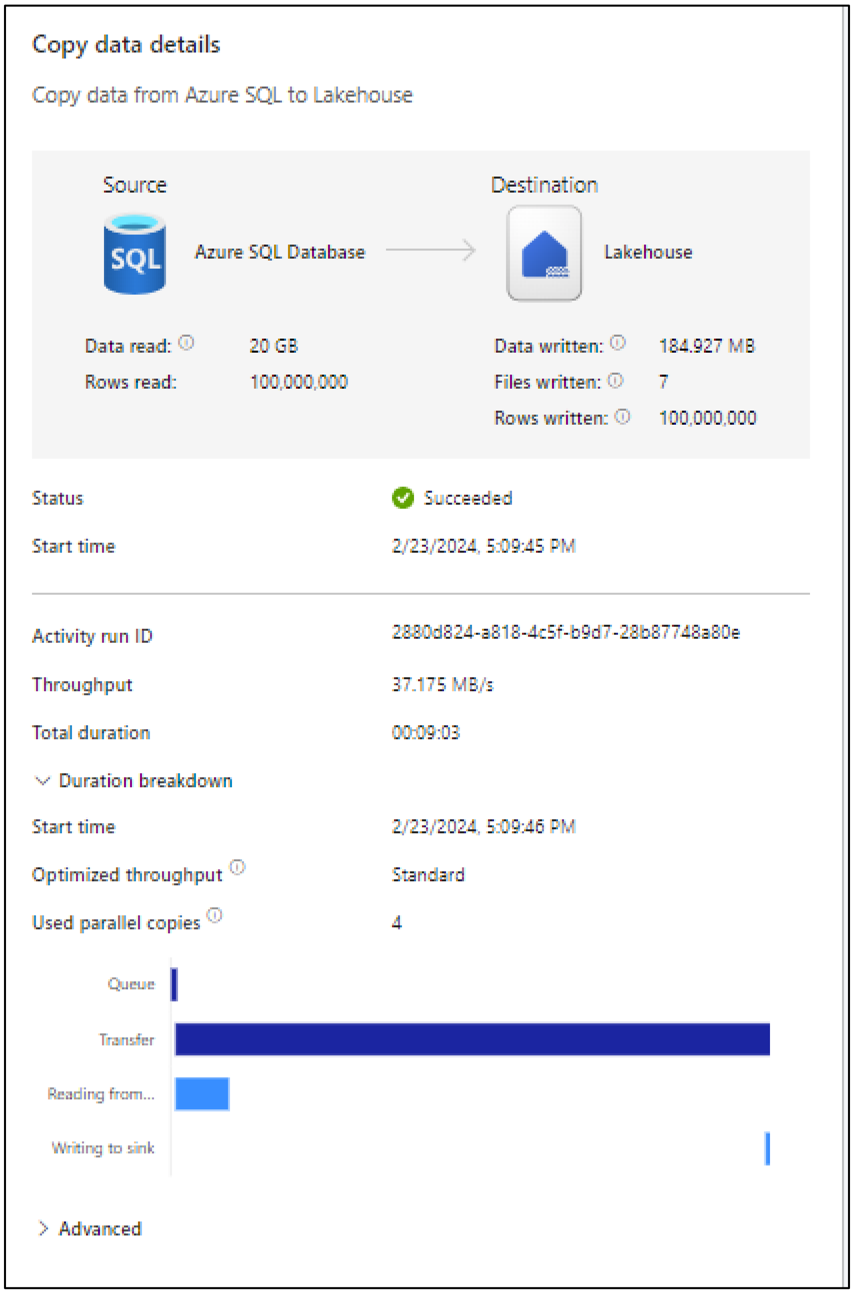


Pipeline run

Parameters

Name	Type	Value
TableName	string	Sales3
Parallelism	int	4

- At the end of the pipeline execution you can observe :
  - A better throughput when transferring data
  - Good performance (only 2mn more when moving from 50M to 100M of rows)



Copy data details

Copy data from Azure SQL to Lakehouse

Source: Azure SQL Database → Destination: Lakehouse

Data read:	20 GB	Data written:	184.927 MB
Rows read:	100,000,000	Files written:	7
		Rows written:	100,000,000

Status	Succeeded
Start time	2/23/2024, 5:09:45 PM
Activity run ID	2880d824-a818-4c5f-b9d7-28b87748a80e
Throughput	37.175 MB/s
Total duration	00:09:03

Duration breakdown

Start time	2/23/2024, 5:09:46 PM
Optimized throughput	Standard
Used parallel copies	4

Advanced