Itron / Schneider Demo Setup

Initial Setup

The first step in configuring the demo environment is to ensure that you have an appropriate Fabric Capacity (F16 minimum) and a Workspace configured that you are either an owner of, or have full Write access. Once you have established this, you can follow these steps to configure the demo (these are the high-level steps of the items that you will need, detailed information will be contained later in the document for each item)

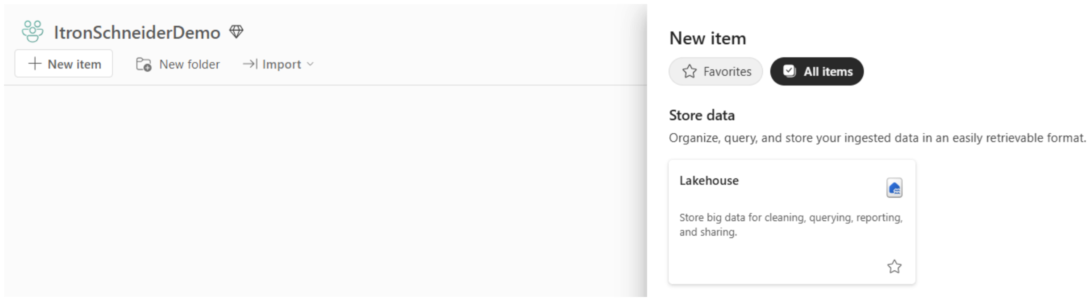
1) Create a Lakehouse
    a. Upload the reference data to the Lakehouse
2) Create an Eventhouse
    a. Within the Eventhouse create a KQL Database and a KQL Queryset
    b. Create a shortcut to the reference tables in the Lakehouse
3) Create an Eventstream
    a. Within the Eventstream, create 2 sinks, one for the KQL Database and one for the Lakehouse
    b. Within the Eventstream, create an Eventhub source
4) Create a Notebook
    a. For the Spark environment that will execute the Notebook, add the azure-servicebus public library
5) Create a Real Time dashboard
6) Create a Power BI Report

Obtain the Required Files

The files that are used throughout this document can be found on GitHub here: https://github.com/temalo/FabricRTIDemo

Create a Lakehouse

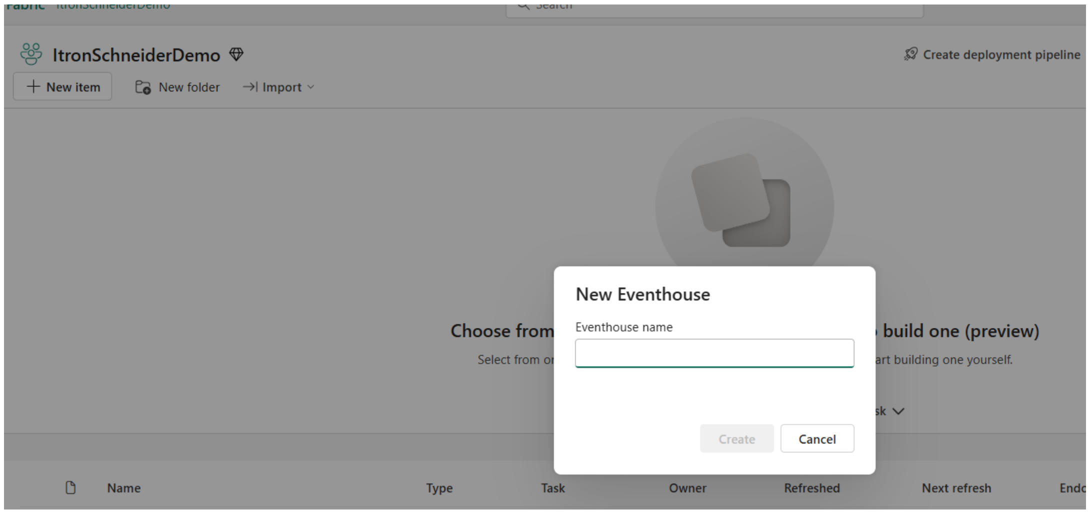
To create a Lakehouse, navigate to the workspace that you created during the initial setup, and select New Item, and then select Lakehouse, and then give the Lakehouse a name and select Create.


ItronSchneiderDemo
New item
New folder
Import
New item
Favorites
All items
Store data
Organize, query, and store your ingested data in an easily retrievable format.
Lakehouse
Store big data for cleaning, querying, reporting, and sharing.

The name of the Lakehouse doesn't matter, but call it something relevant that you will remember.

Create an Eventhouse

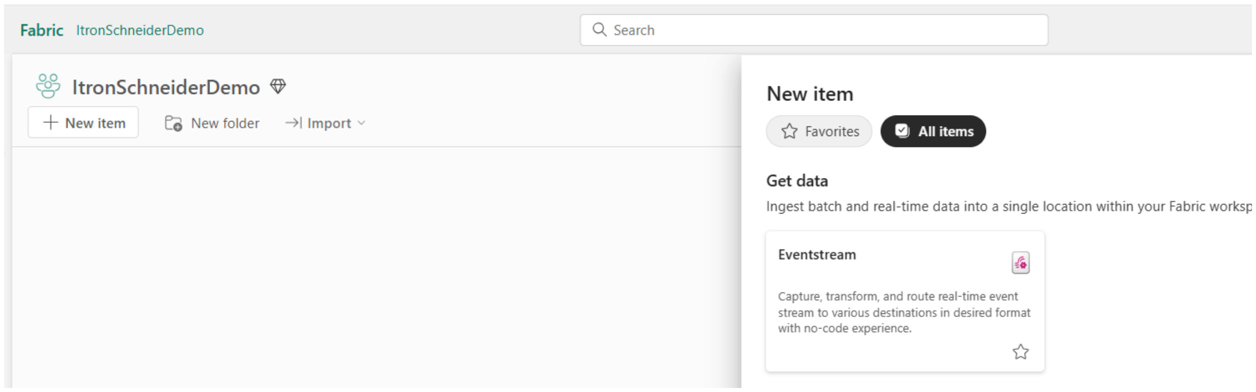
In the Workspace that you are using, select New Item, and then select Eventhouse, and then give the Eventhouse a name, and select Create.



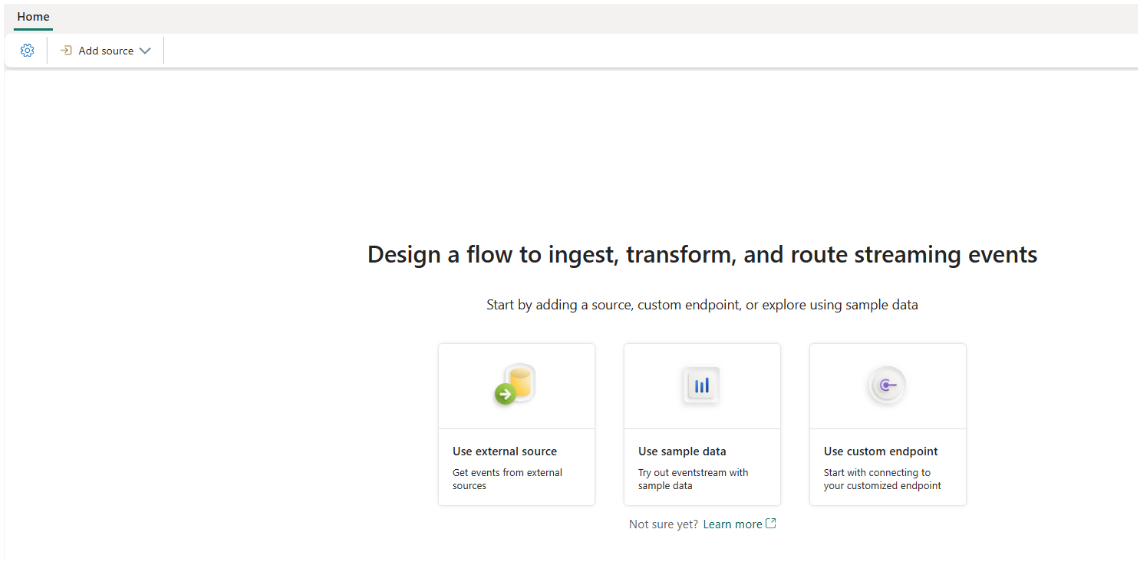
The name of the Eventhouse doesn't matter, but make it something relevant that you will remember.

Create an Eventstream

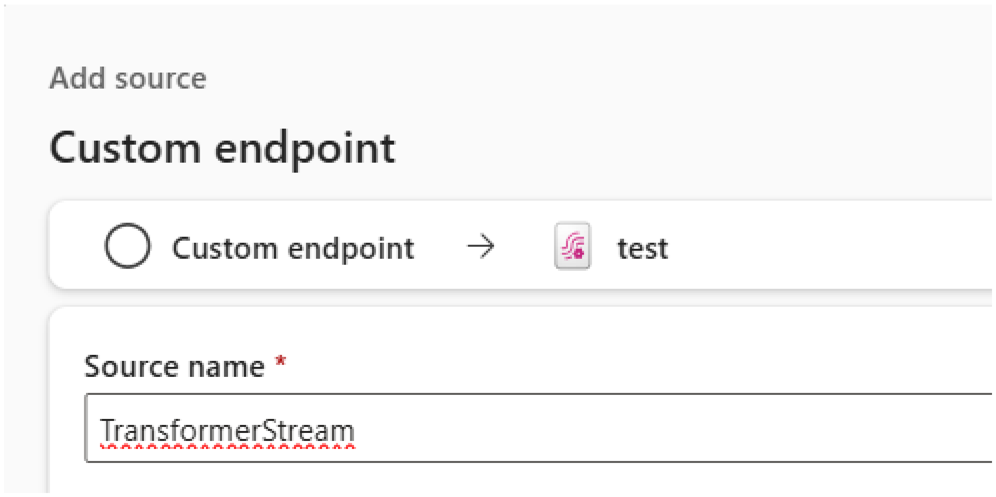
Navigate to the workspace that you are using, select New Item, and then select Eventstream. Name the Eventstream and select Create


Fabric
ItronSchneiderDemo
Search
ItronSchneiderDemo
New item
New folder
Import
New item
Favorites
All items
Get data
Ingest batch and real-time data into a single location within your Fabric worksp
Eventstream
Capture, transform, and route real-time event stream to various destinations in desired format with no-code experience.

Once the Eventstream is created, you will be presented with the blank canvas. Choose Custom Endpoint:


Home
Add source
Design a flow to ingest, transform, and route streaming events
Start by adding a source, custom endpoint, or explore using sample data
Use external source
Get events from external sources
Use sample data
Try out eventstream with sample data
Use custom endpoint
Start with connecting to your customized endpoint
Not sure yet? Learn more

In the Custom Endpoint dialog, name the source TransformerStream and click Add:


Add source

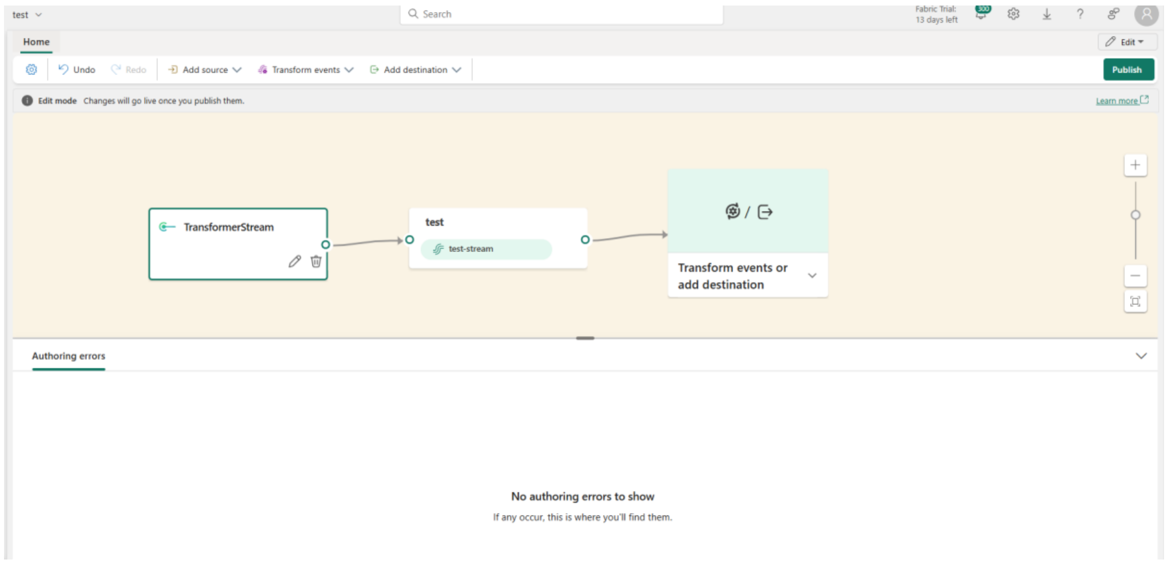
Custom endpoint

Custom endpoint → test

Source name *

TransformerStream

This will open the authoring panel.

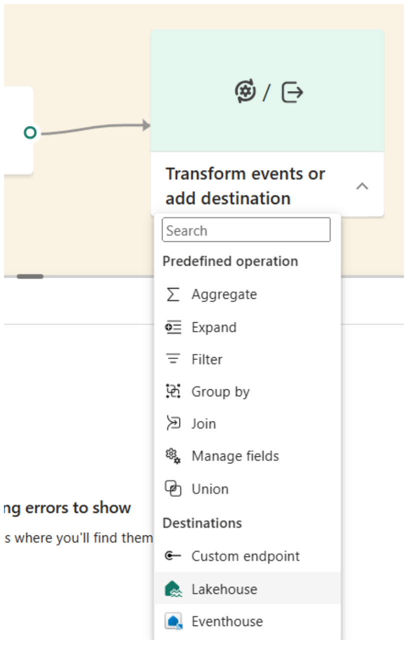


Edit mode Changes will go live once you publish them.

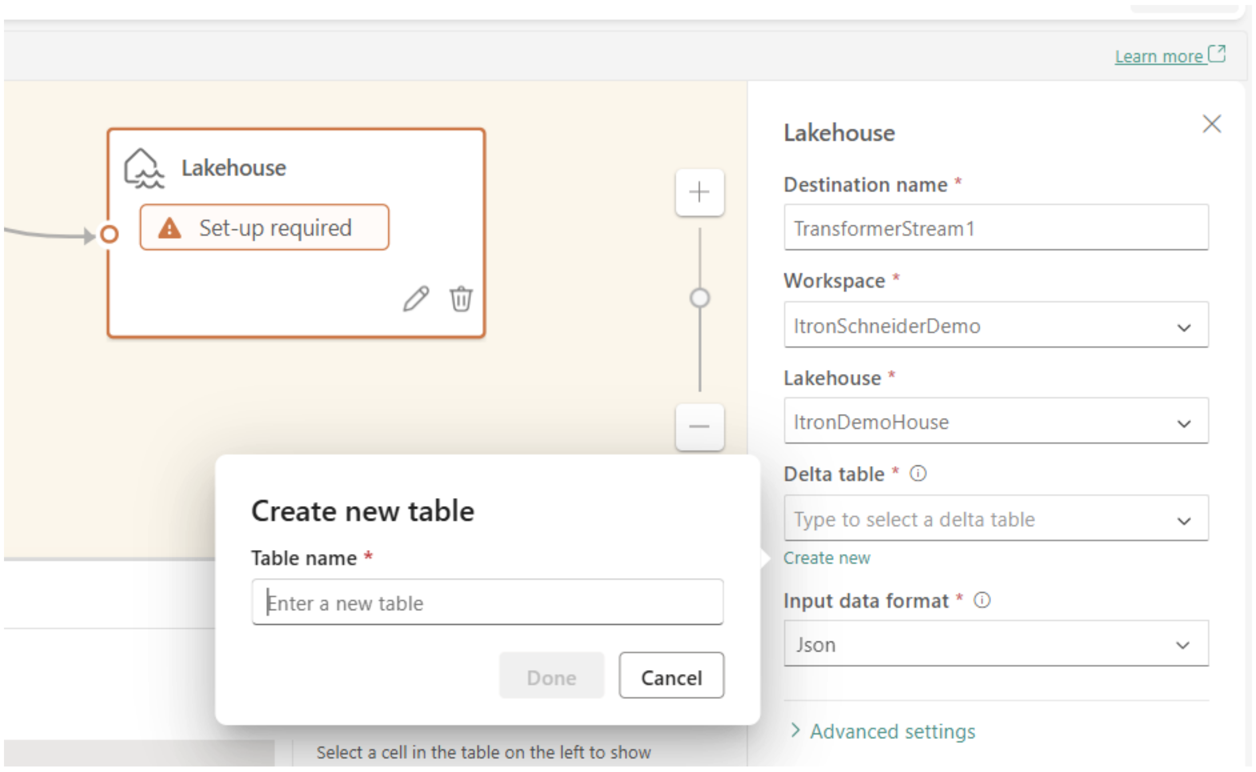
No authoring errors to show

If any occur, this is where you'll find them.

Select Transform events, and then select Lakehouse as the destination:



Name the destination TransformerStream and then select the Lakehouse that you created above. Select New Table, and add a name for the Table that will reside in the Lakehouse (For simplicity, you can name the table TransformerStream) and select Done



Lakehouse

Destination name *: TransformerStream1

Workspace *: ItronSchneiderDemo

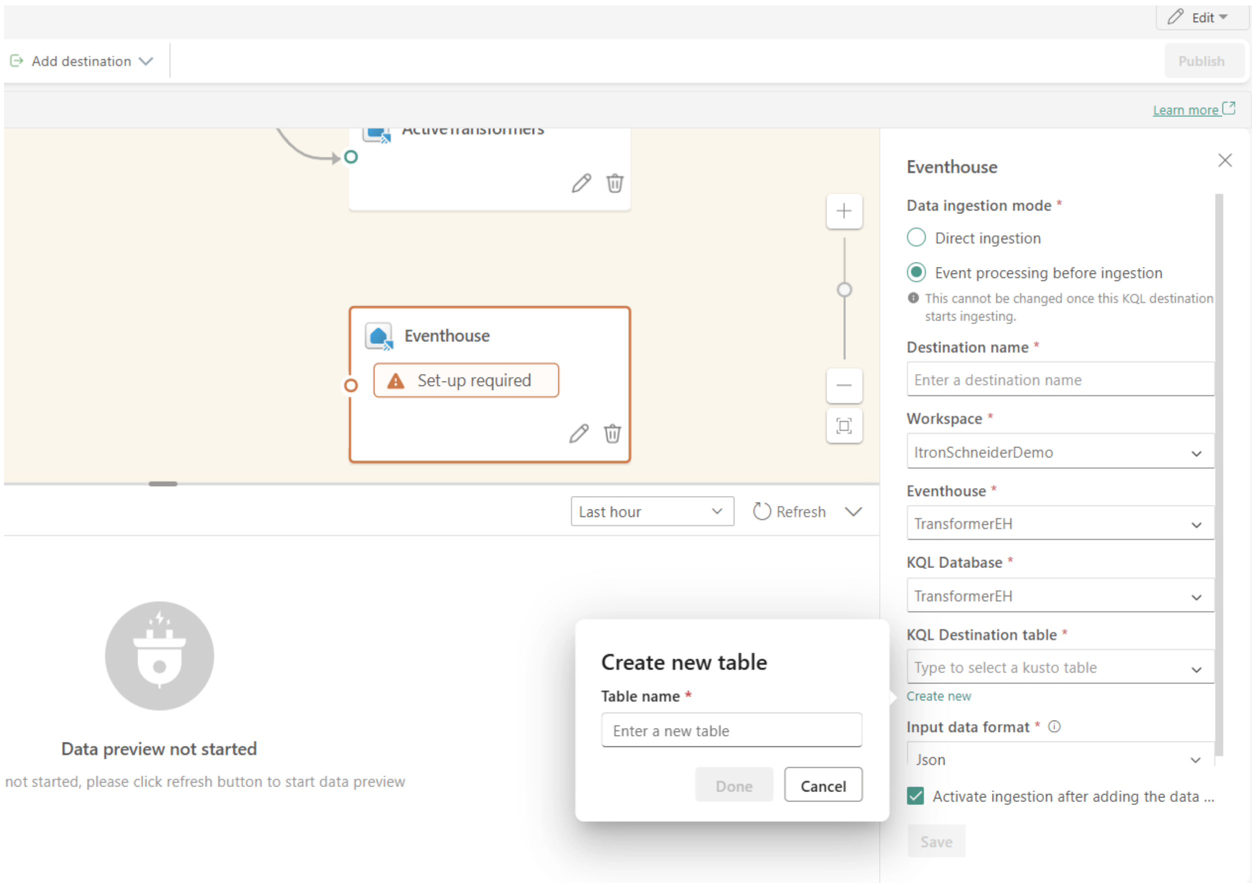
Lakehouse *: ItronDemoHouse

Delta table *: Type to select a delta table

Input data format *: Json

Table name *: Enter a new table

Once the destination is saved, go back to the authoring panel and add a second destination for the KQL Database in your Eventhouse by selecting Add Destination, then name the destination TransformerStreamEH , then fill out the form referencing the Eventhouse you created earlier, the KQL Database in that Eventhouse, and create a new Table in the KQL Database. Name the table ActiveTransformers and select Done and then save the destination. Draw a line between the stream and the new destination to connect them. This will ensure that the data flows the stream directly to both the Lakehouse table and the KQL database table.



Create new table

Table name *

Enter a new table

Done Cancel

Eventhouse

Data ingestion mode *

Direct ingestion

Event processing before ingestion

This cannot be changed once this KQL destination starts ingesting.

Destination name *

Enter a destination name

Workspace *

ItronSchneiderDemo

Eventhouse *

TransformerEH

KQL Database *

TransformerEH

KQL Destination table *

Type to select a kusto table

Create new

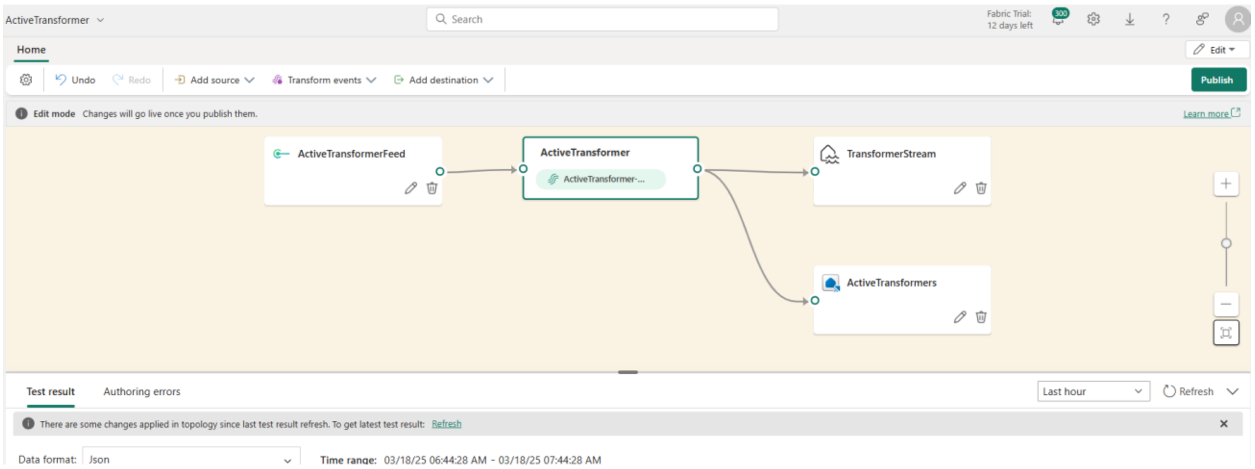
Input data format *

Json

Activate ingestion after adding the data ...

Save

Once the Eventstream is complete, select Publish to save the changes and active the Eventstream.



There are some changes applied in topology since last test result refresh. To get latest test result: Refresh

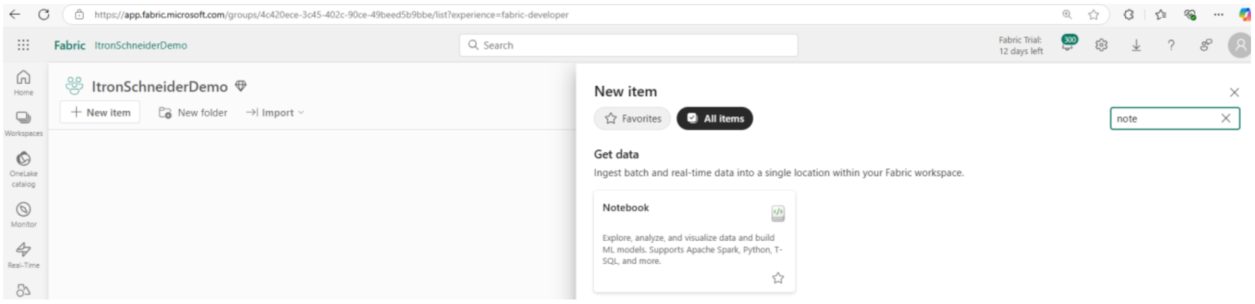
Data format: Json

Time range: 03/18/25 06:44:28 AM - 03/18/25 07:44:28 AM

Once the Eventstream is published, select the input node on the left side of the canvas and then select the Event Hub tab and select the SAS Key Authentication option. Copy and save the Connection String Primary Key value as you will need this in the Notebook you create in the next section.

Create a Notebook

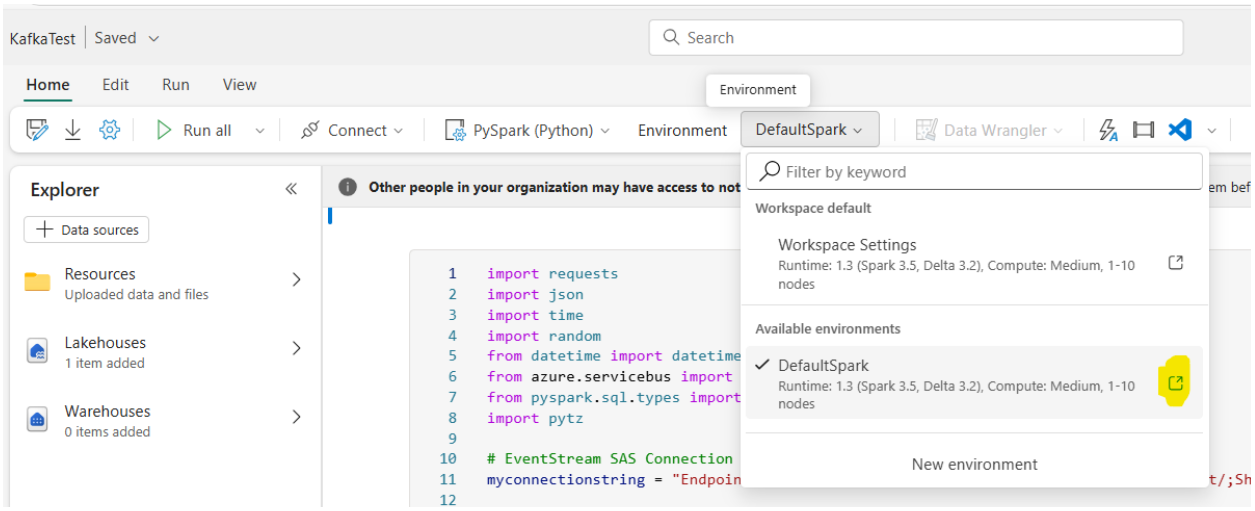
Navigate to the Workspace that you are using and select New Item, and then select Notebook.



Ingest batch and real-time data into a single location within your Fabric workspace.

Explore, analyze, and visualize data and build ML models. Supports Apache Spark, Python, T-SQL, and more.

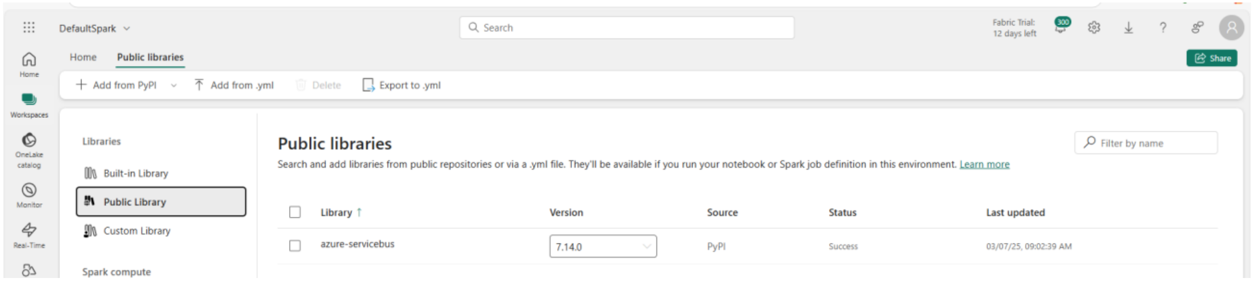
Give the Notebook a name and then select the Environment option. Select the Notebook Environment and select the Edit button to the right of the Environment name:



Other people in your organization may have access to not ... em bef

```
import requests
import json
import time
import random
from datetime import datetime
from azure.servicebus import
from pyspark.sql.types import
import pytz

# EventStream SAS Connection
myconnectionstring = "Endpoin                                t/;Sh
```

Select the Public Library Option, and then select the Add From PyPI option, and then select the azure-servicebus library.



Public libraries

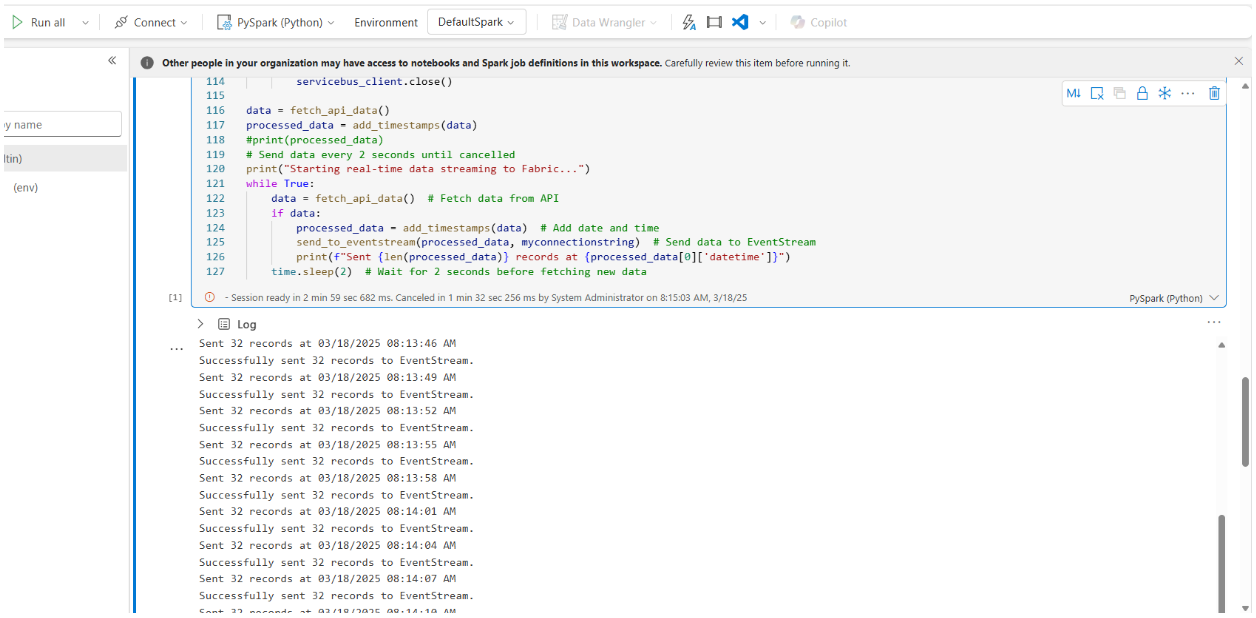
Search and add libraries from public repositories or via a .yml file. They'll be available if you run your notebook or Spark job definition in this environment. Learn more

Library ↑	Version	Source	Status	Last updated
azure-servicebus	7.14.0	PyPI	Success	03/07/25, 09:02:39 AM

Navigate to the Notebook and paste the code from the Notebook example text file into the Notebook you created. Locate the section that includes the Service Bus connection String and replace the text with the text you copied in the previous step:

```
# EventStream SAS Connection String
# This is not a best practice, these secrets should be stored in Azure Key Vault or some other secure mechanism
myconnectionstring = "<Insert Connection String Here for the Eventhouse Servicebus Connection>"
```

Click the Run All button at the top of the Notebook, and once the Spark Session starts, you should see output showing that records are being sent to the Eventstream



Other people in your organization may have access to notebooks and Spark job definitions in this workspace. Carefully review this item before running it.

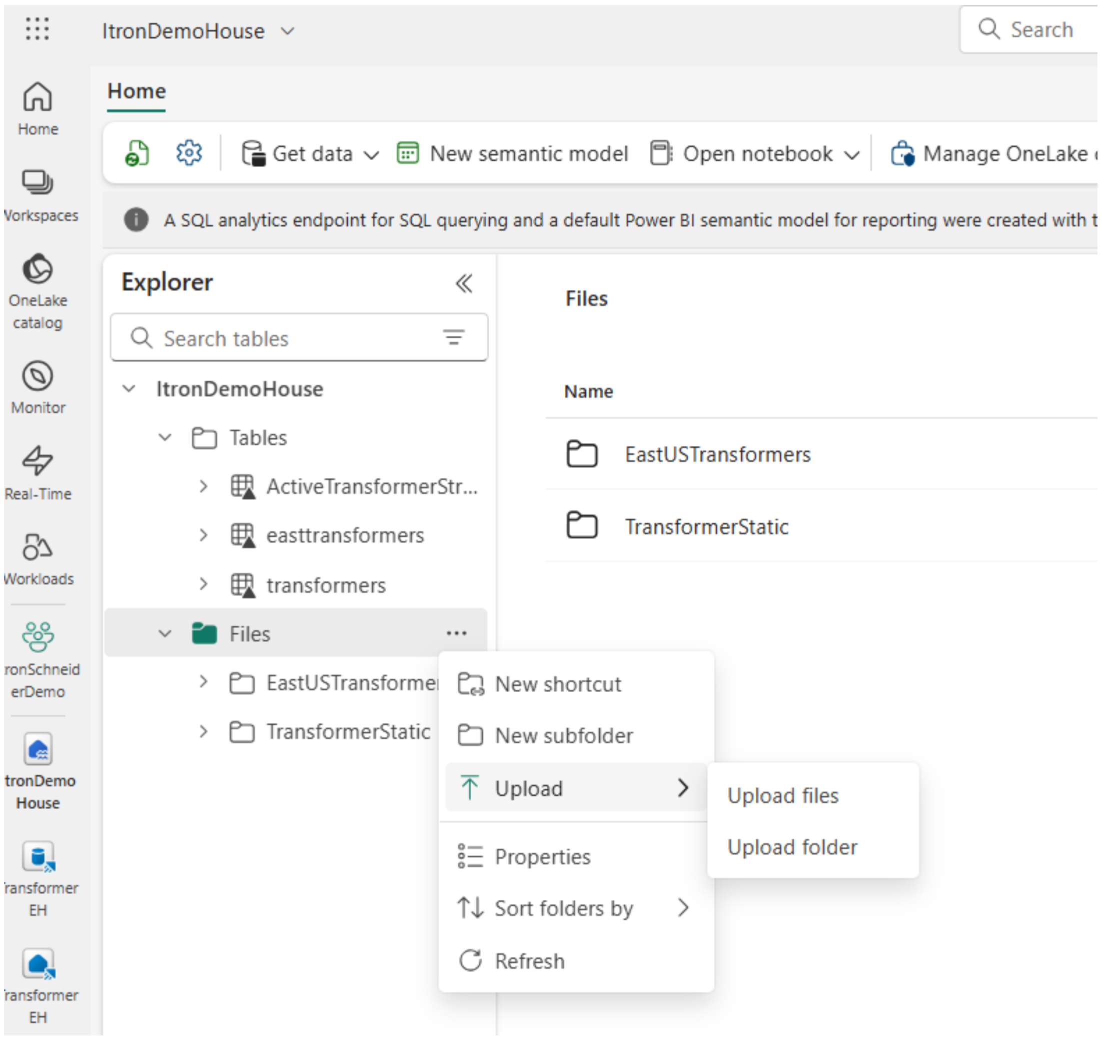
```
            servicebus_client.close()

    data = fetch_api_data()
    processed_data = add_timestamps(data)
    #print(processed_data)
    # Send data every 2 seconds until cancelled
    print("Starting real-time data streaming to Fabric...")
    while True:
        data = fetch_api_data()  # Fetch data from API
        if data:
            processed_data = add_timestamps(data)  # Add date and time
            send_to_eventstream(processed_data, myconnectionstring)  # Send data to EventStream
            print(f"Sent {len(processed_data)} records at {processed_data[0]['datetime']}")
        time.sleep(2)  # Wait for 2 seconds before fetching new data
```

[1] - Session ready in 2 min 59 sec 682 ms. Canceled in 1 min 32 sec 256 ms by System Administrator on 8:15:03 AM, 3/18/25

```
Sent 32 records at 03/18/2025 08:13:46 AM
Successfully sent 32 records to EventStream.
Sent 32 records at 03/18/2025 08:13:49 AM
Successfully sent 32 records to EventStream.
Sent 32 records at 03/18/2025 08:13:52 AM
Successfully sent 32 records to EventStream.
Sent 32 records at 03/18/2025 08:13:55 AM
Successfully sent 32 records to EventStream.
Sent 32 records at 03/18/2025 08:13:58 AM
Successfully sent 32 records to EventStream.
Sent 32 records at 03/18/2025 08:14:01 AM
Successfully sent 32 records to EventStream.
Sent 32 records at 03/18/2025 08:14:04 AM
Successfully sent 32 records to EventStream.
Sent 32 records at 03/18/2025 08:14:07 AM
Successfully sent 32 records to EventStream.
```

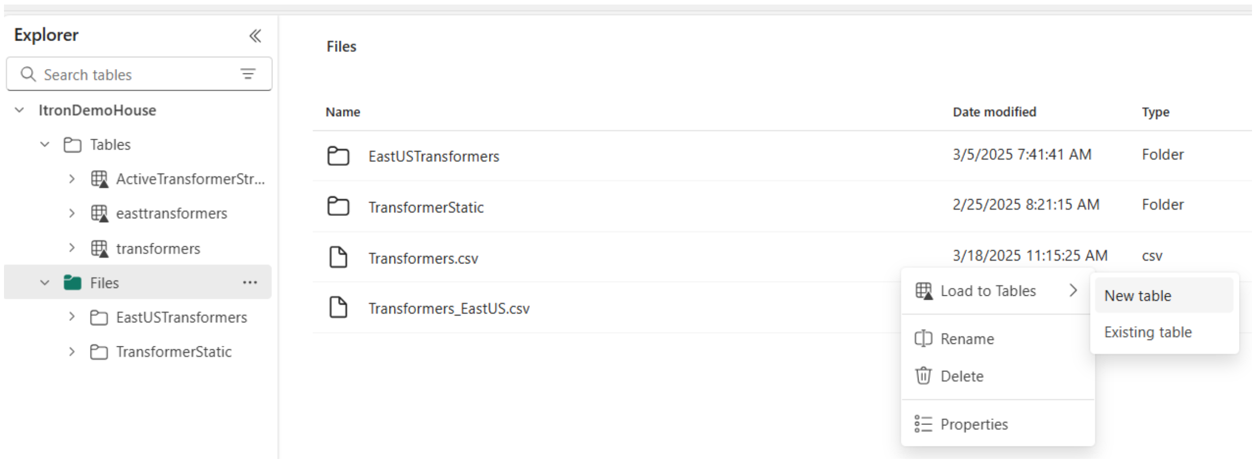
Upload Reference Files to the Lakehouse

Navigate to the Lakehouse that you created above, select the Files section, and from the ellipse menu select Upload Files. Select the 2 reference data files and upload them.



A SQL analytics endpoint for SQL querying and a default Power BI semantic model for reporting were created with t

Once the files are uploaded, select the ellipse next to the file and select Load to Table/New Table, and name the Table the same as the file (Transformers and EastTransformers).

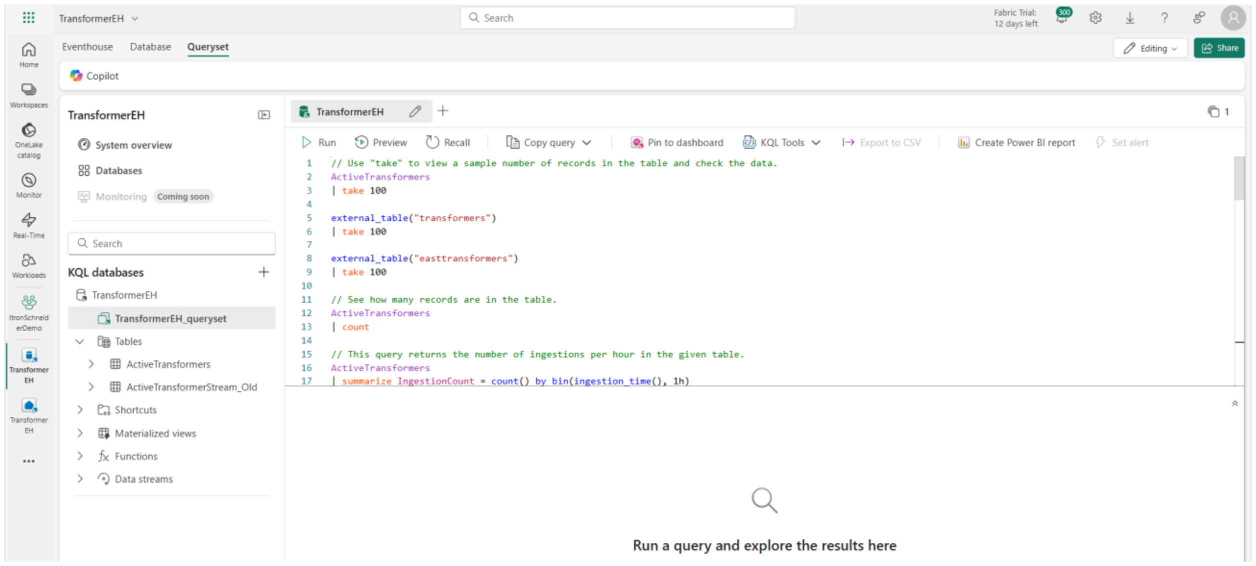


Name	Date modified	Type
EastUSTransformers	3/5/2025 7:41:41 AM	Folder
TransformerStatic	2/25/2025 8:21:15 AM	Folder
Transformers.csv	3/18/2025 11:15:25 AM	csv
Transformers_EastUS.csv		

This will create 2 tables in the Lakehouse that contain the reference data that you will use in the following section.

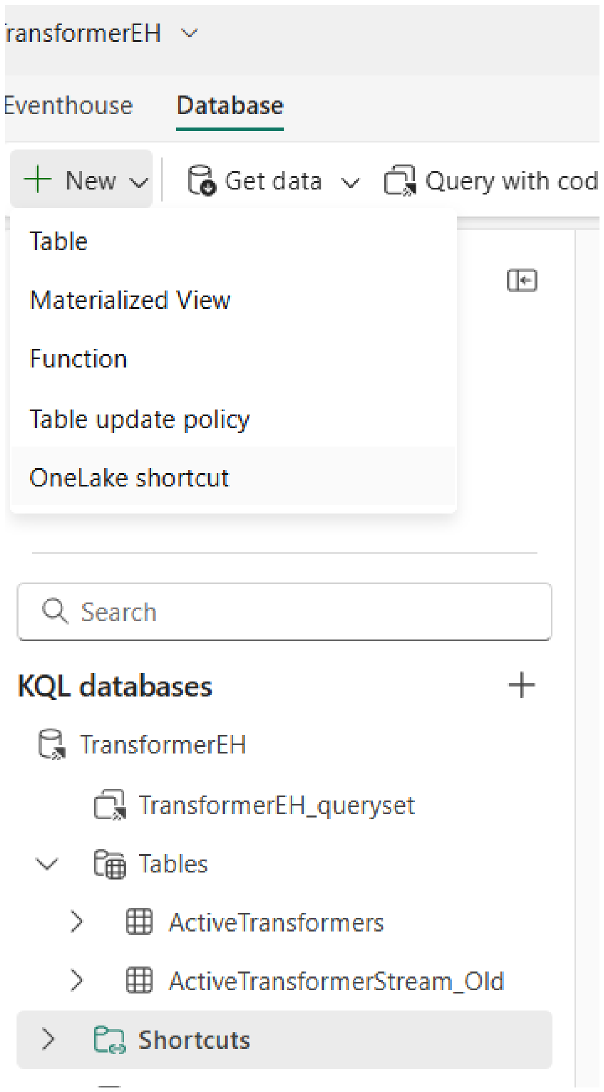
Populate a Queryset

Navigate to the Eventhouse that you created above and select the KQL Database and then the Queryset. Replace the text in the Queryset with the text from the provided Queryset file.

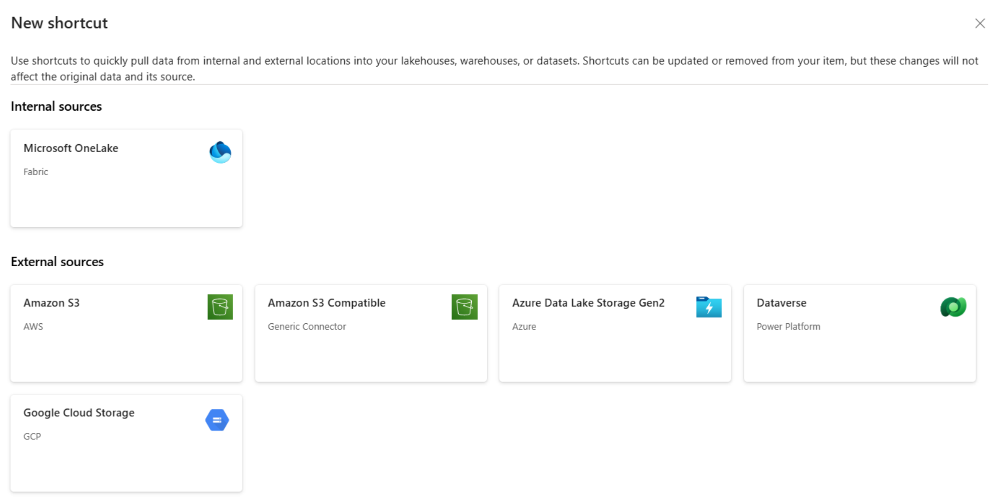


```
// Use "take" to view a sample number of records in the table and check the data.
ActiveTransformers
| take 100

external_table("transformers")
| take 100

external_table("easttransformers")
| take 100

// See how many records are in the table.
ActiveTransformers
| count

// This query returns the number of ingestions per hour in the given table.
ActiveTransformers
| summarize IngestionCount = count() by bin(ingestion_time(), 1h)
```

Since you will be using external tables in some of the queries, you will need to create shortcuts to reference the tables in your Lakehouse. To do this, navigate to the Shortcuts icon in the KQL database, and then in upper left select New and then select Onelake Shortcut.



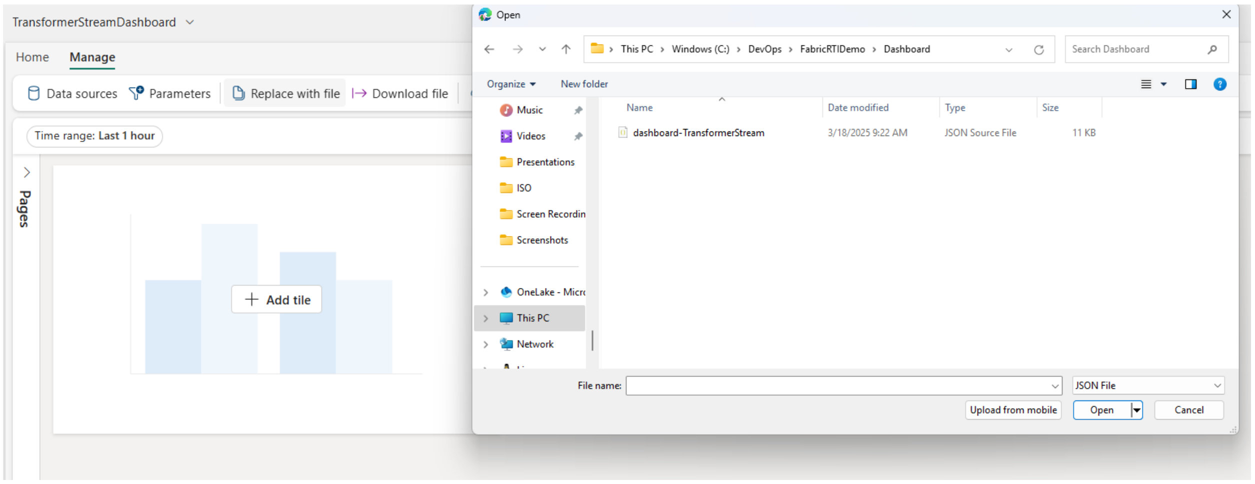
Select Microsoft Onelake, and then navigate to the Lakehouse you created above and select the 2 tables that were created when you uploaded the reference files.


New shortcut
Use shortcuts to quickly pull data from internal and external locations into your lakehouses, warehouses, or datasets. Shortcuts can be updated or removed from your item, but these changes will not affect the original data and its source.
Internal sources
Microsoft OneLake
Fabric
External sources
Amazon S3
AWS
Amazon S3 Compatible
Generic Connector
Azure Data Lake Storage Gen2
Azure
Dataverse
Power Platform
Google Cloud Storage
GCP

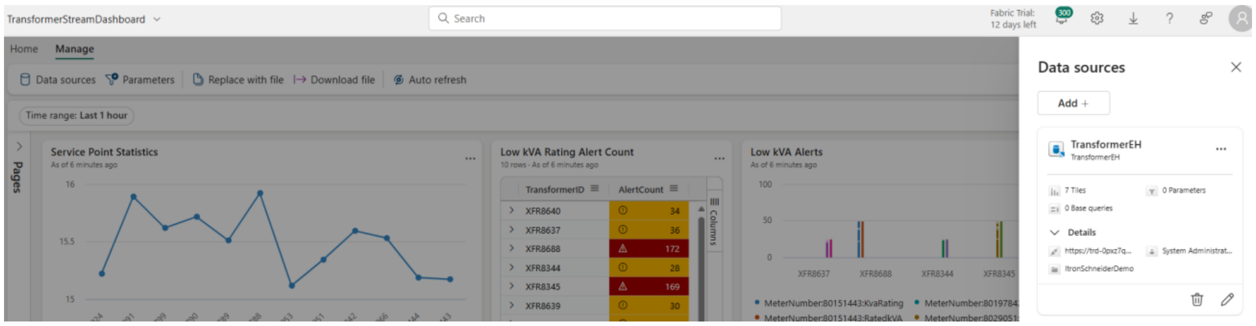
The Queryset is the source of queries that will populate the elements on the Real Time Dashboard. You can test the queries by highlighting each query element and selecting Run or hit shift-enter on the keyboard.

Create a Real Time Dashboard

Navigate to the workspace that you are using and select New Item and then select Real-Time Dashboard. Name the Dashboard TransformerStreamDashboard and when it loads, select the Manage tab, then select Replace with File, and then select the dashboard-TransformerStream JSON file.



When you replace the dashboard with the included file, you will likely need to change the Data Source to the Eventhouse and KQL database that you created above. To accomplish this, you need to select the Data sources button and replace the Data source on the right side of the screen.


TransformerStreamDashboard
Search
Fabric Trial: 12 days left
Home
Manage
Data sources
Parameters
Replace with file
Download file
Auto refresh
Time range: Last 1 hour
Service Point Statistics
Low kVA Rating Alert Count
Low kVA Alerts
Data sources
Add +
TransformerEH
7 Tiles
0 Parameters
0 Base queries
Details
ItronSchneiderDemo