

# 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

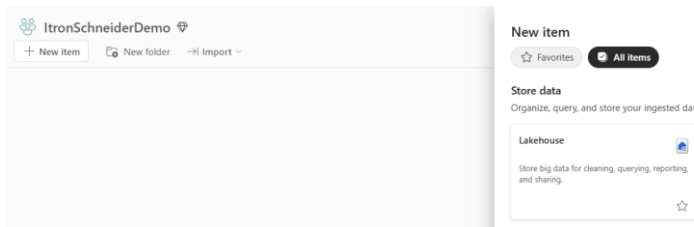
## Obtain the Required Files

The files that are used throughout this document can be found on GitHub here:

<https://github.com/temalo/FabricRTIDemo>

## Create a Lakehouse

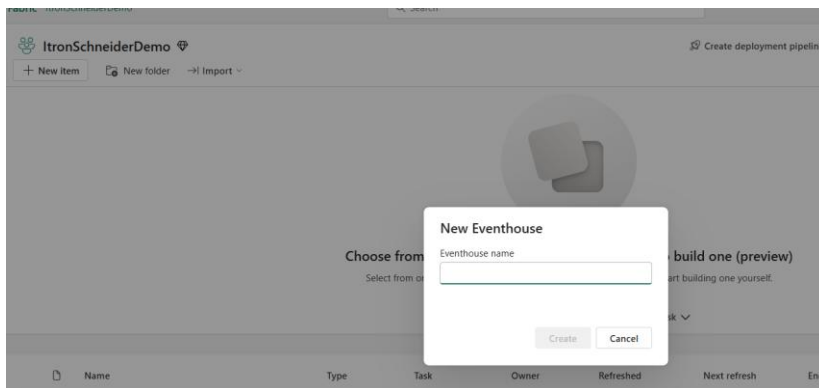
To create a Lakehouse, navigate to the workspace that you created during the initial setup, and select New Item, and then select Lakehouse, and then give the Lakehouse a name and select Create.



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

## Create an Eventhouse

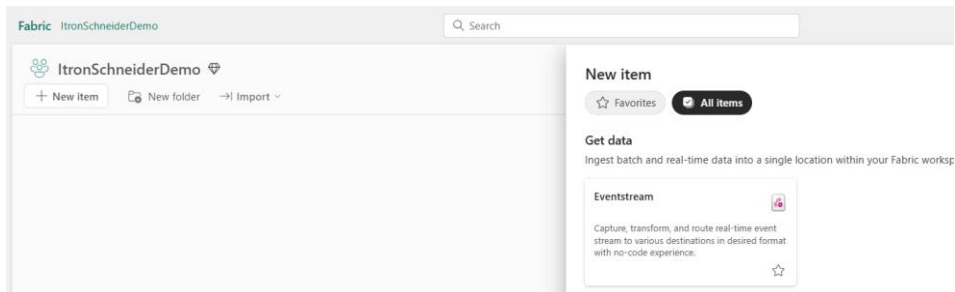
In the Workspace that you are using, select New Item, and then select Eventhouse, and then give the Eventhouse a name, and select Create.



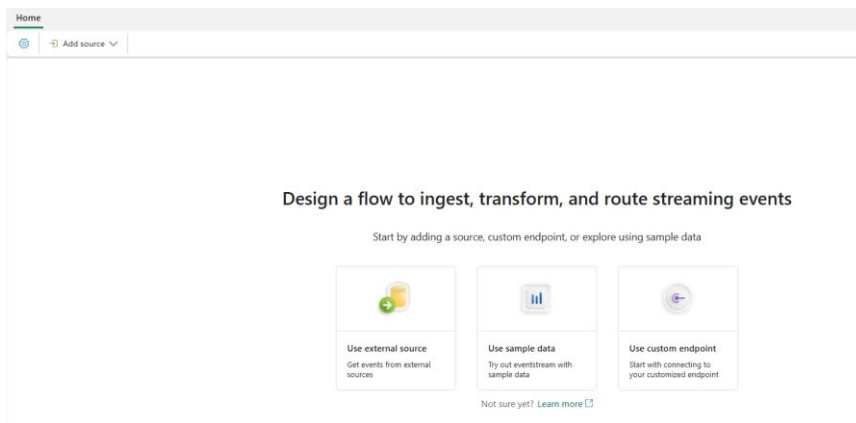
The name of the Eventhouse doesn't matter, but make it something relevant that you will remember.

## Create an Eventstream

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




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



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

Add source

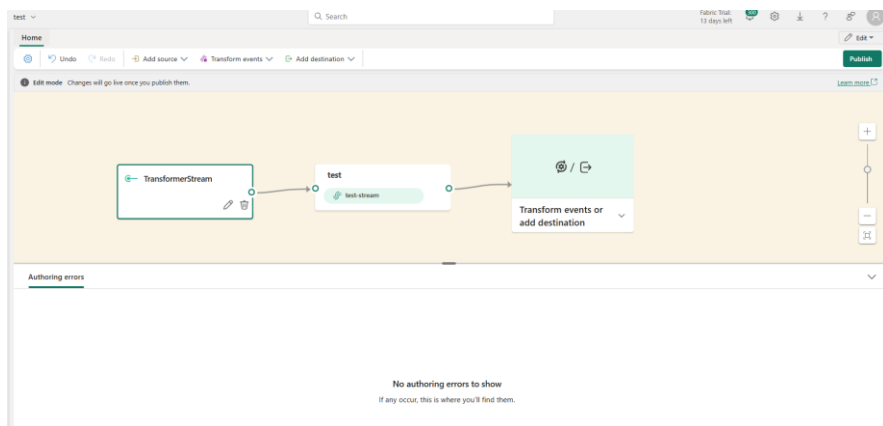
## Custom endpoint

☐ Custom endpoint →  test

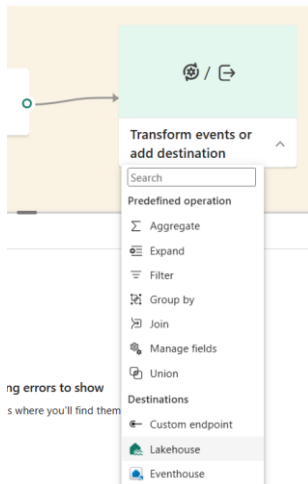
Source name \*

TransformerStream

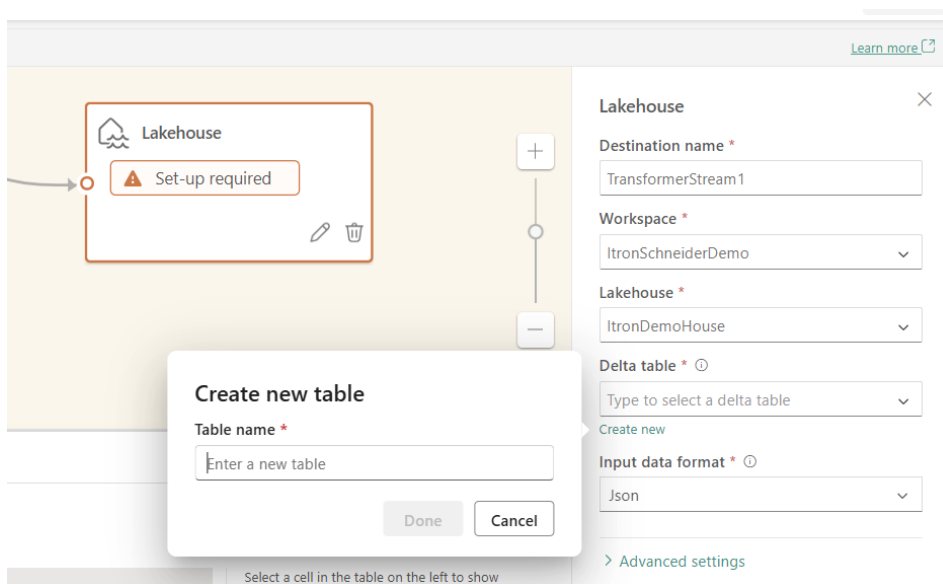
This will open the authoring panel.



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

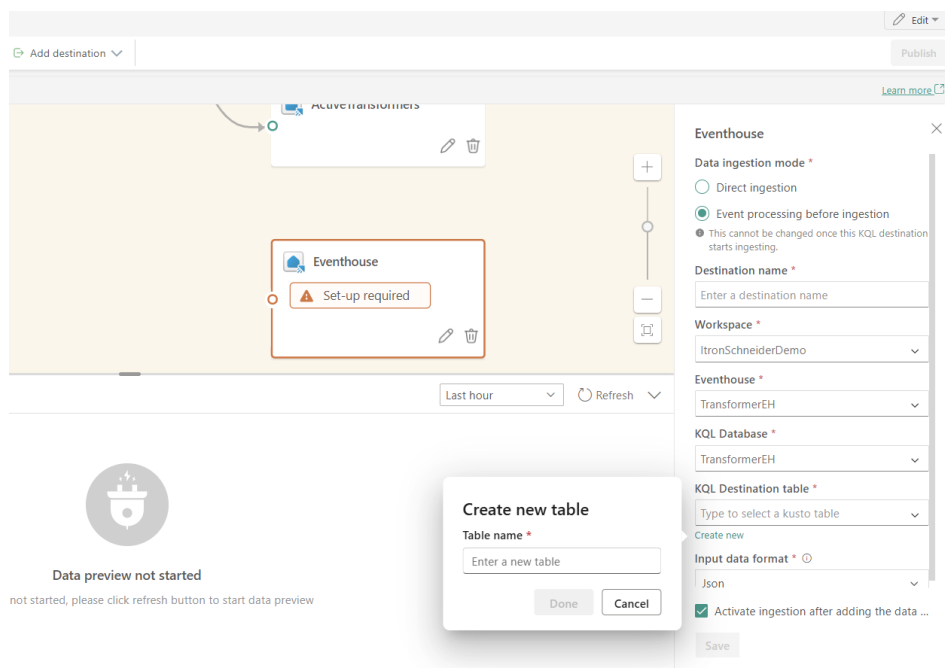


Name the destination `TransformerStream` and then select the Lakehouse that you created above. Select **New Table**, and add a name for the Table that will reside in the Lakehouse (For simplicity, you can name the table `TransformerStream`) and select **Done**

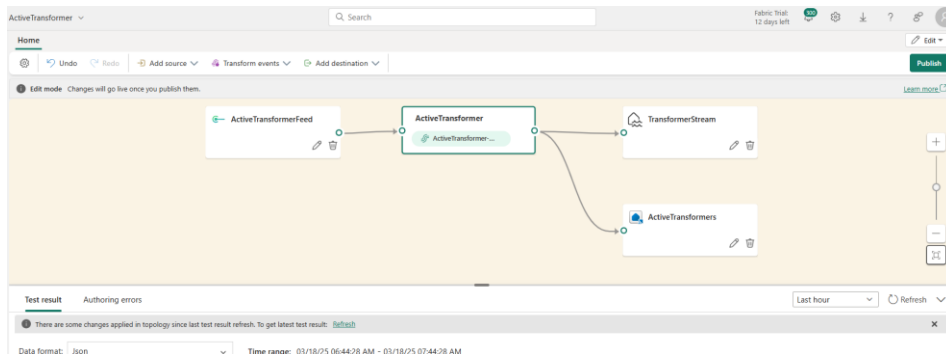


Once the destination is saved, go back to the authoring panel and add a second destination for the KQL Database in your Eventhouse by selecting Add Destination, then name the destination TransformerStreamEH, then fill out the form referencing the Eventhouse you created earlier, the KQL Database in that 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.

**Commented [MD1]:** Just a suggestion, but it will help people put the concepts together if you provide a suggestion for the name on the second destination. Just helps with the Why am I doing this



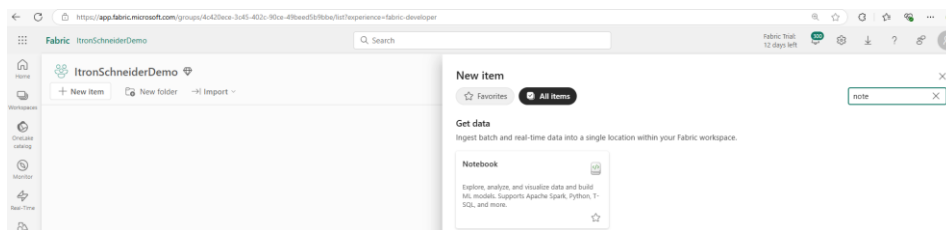
Once the Eventstream is complete, select Publish to save the changes and active the Eventstream.



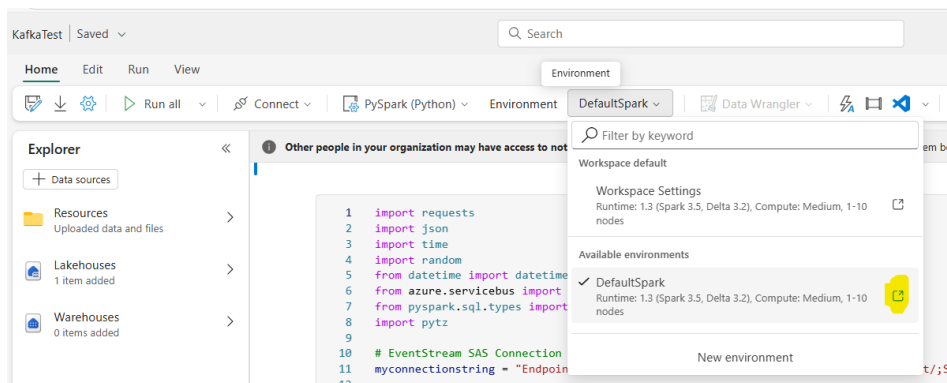
Once the Eventstream is published, select the input node on the left side of the canvas and then select the Event Hub tab and select the SAS Key Authentication option. Copy and save the Connection String Primary Key value as you will need this in the Notebook you create in the next section.

## Create a Notebook

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

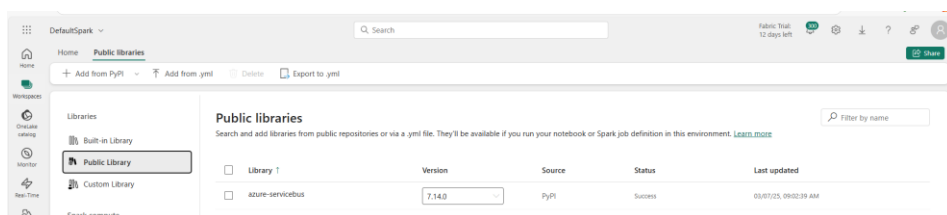


Give the Notebook a name and then select the Environment option. Select the Notebook Environment and select the Edit button to the right of the Environment name:



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

**Commented [MD2]:** Might be worth commenting that this can take some time to publish to the spark env. In my case it took 5+minutes to publish



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



Run allConnectPySpark (Python)EnvironmentDefaultSparkData WranglerCopilot

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

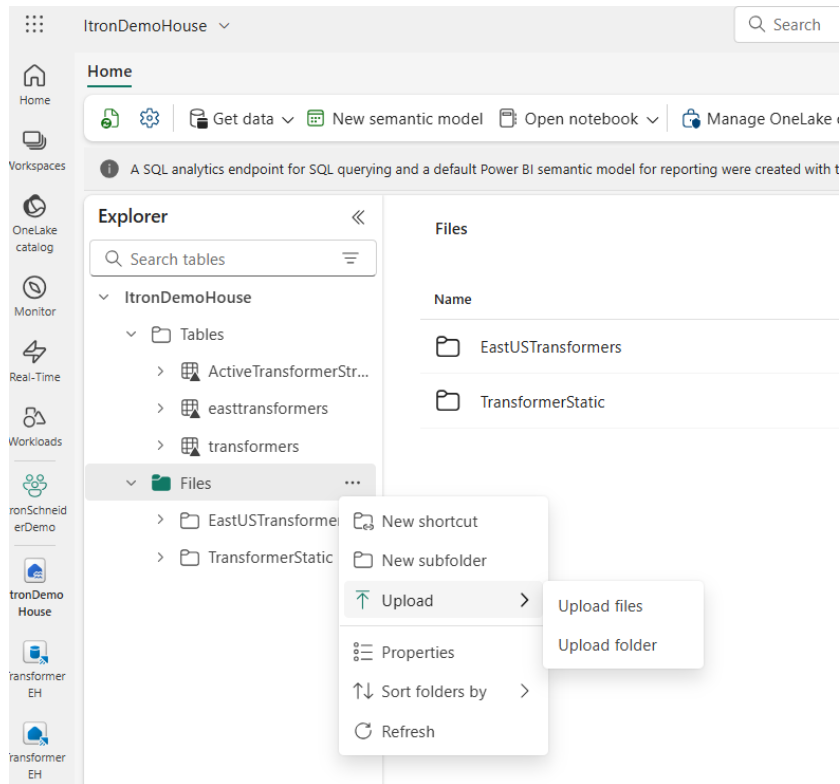
114servicebus\_client.close()  
115  
116data = fetch\_api\_data()  
117processed\_data = add\_timestamps(data)  
118#print(processed\_data)  
119# Send data every 2 seconds until cancelled  
120print("Starting real-time data streaming to Fabric...")  
121while True:  
122 data = fetch\_api\_data() # Fetch data from API  
123 if data:  
124 processed\_data = add\_timestamps(data) # Add date and time  
125 send\_to\_eventstream(processed\_data, myconnectionstring) # Send data to EventStream  
126 print(f"Sent {len(processed\_data)} records at {processed\_data[0]['datetime']}")  
127 time.sleep(2) # Wait for 2 seconds before fetching new data

111- Session ready in 2 min 59 sec 682 ms. Cancelled in 1 min 32 sec 256 ms by System Administrator on 8/19/25 AM, 3/18/25PySpark (Python)

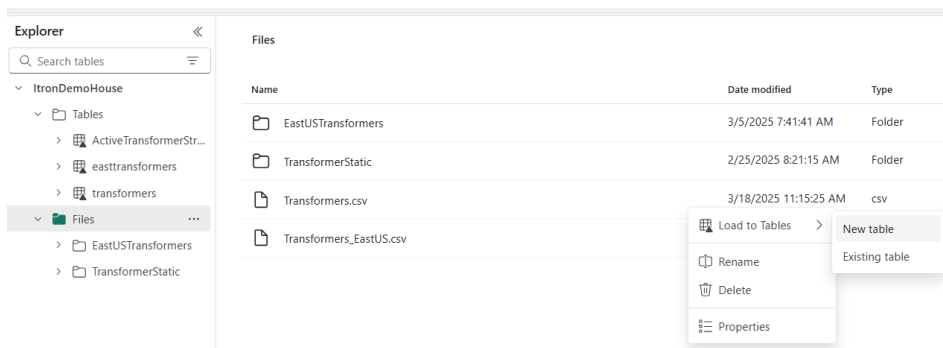
> log  
...  
Sent 32 records at 8/18/2025 08:13:46 AM  
Successfully sent 32 records to EventStream.  
Sent 32 records at 8/18/2025 08:13:49 AM  
Successfully sent 32 records to EventStream.  
Sent 32 records at 8/18/2025 08:13:52 AM  
Successfully sent 32 records to EventStream.  
Sent 32 records at 8/18/2025 08:13:55 AM  
Successfully sent 32 records to EventStream.  
Sent 32 records at 8/18/2025 08:13:58 AM  
Successfully sent 32 records to EventStream.  
Sent 32 records at 8/18/2025 08:14:01 AM  
Successfully sent 32 records to EventStream.  
Sent 32 records at 8/18/2025 08:14:04 AM  
Successfully sent 32 records to EventStream.  
Sent 32 records at 8/18/2025 08:14:07 AM  
Successfully sent 32 records to EventStream.  
C:\> ... 8/18/2025 08:14:10 AM

## 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.



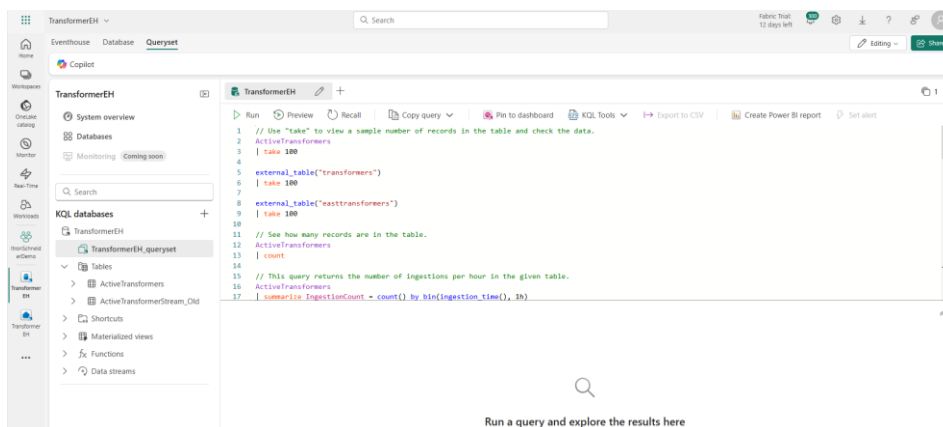
Once the files are uploaded, select the ellipse next to the file and select Load to Table/New Table, and name the Table the same as the file (Transformers and EastTransformers).



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

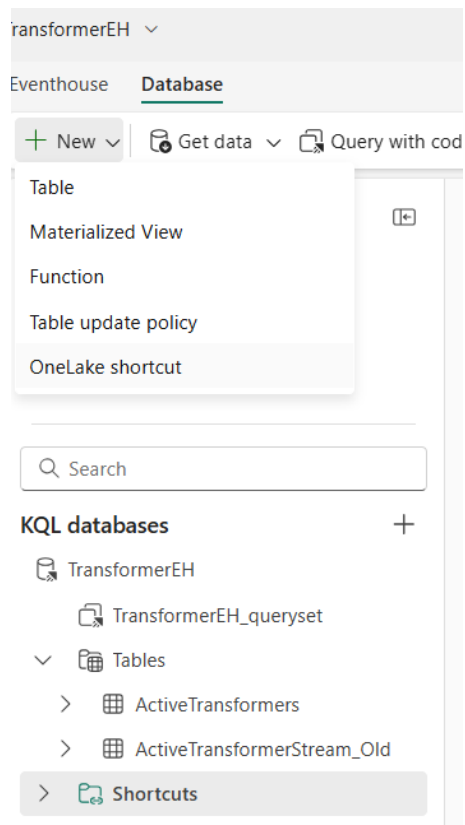
## Populate a Queryset

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.

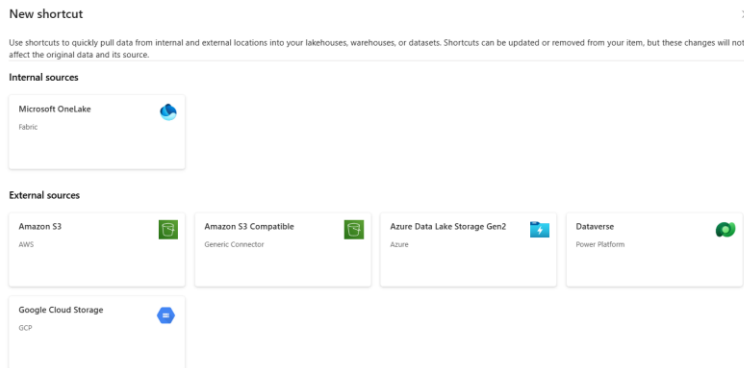


Since you will be using external tables in some of the queries, you will need to create shortcuts to reference the tables in your Lakehouse. To do this, navigate to the Shortcuts

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.

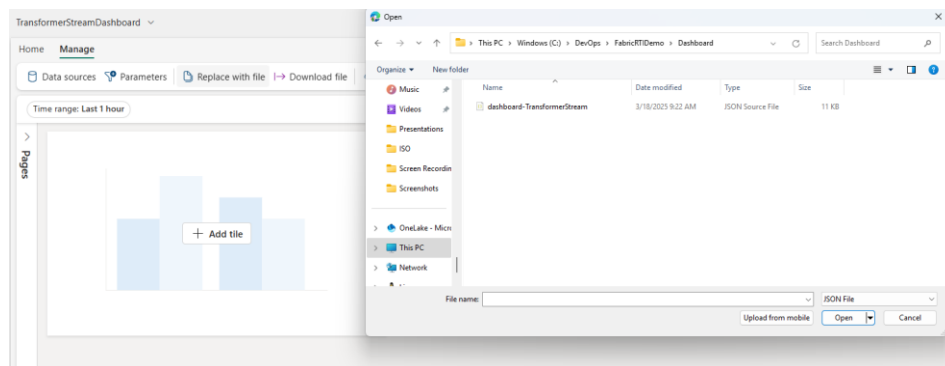


The Queryset is the source of queries that will populate the elements on the Real Time Dashboard. You can test the queries by highlighting each query element and selecting Run or hit shift-enter on the keyboard.

**Commented [MD3]:** or shift+enter, also your query set is using your name for the KQP Table, so above just tell the user to use ActiveTransformers

## 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.

