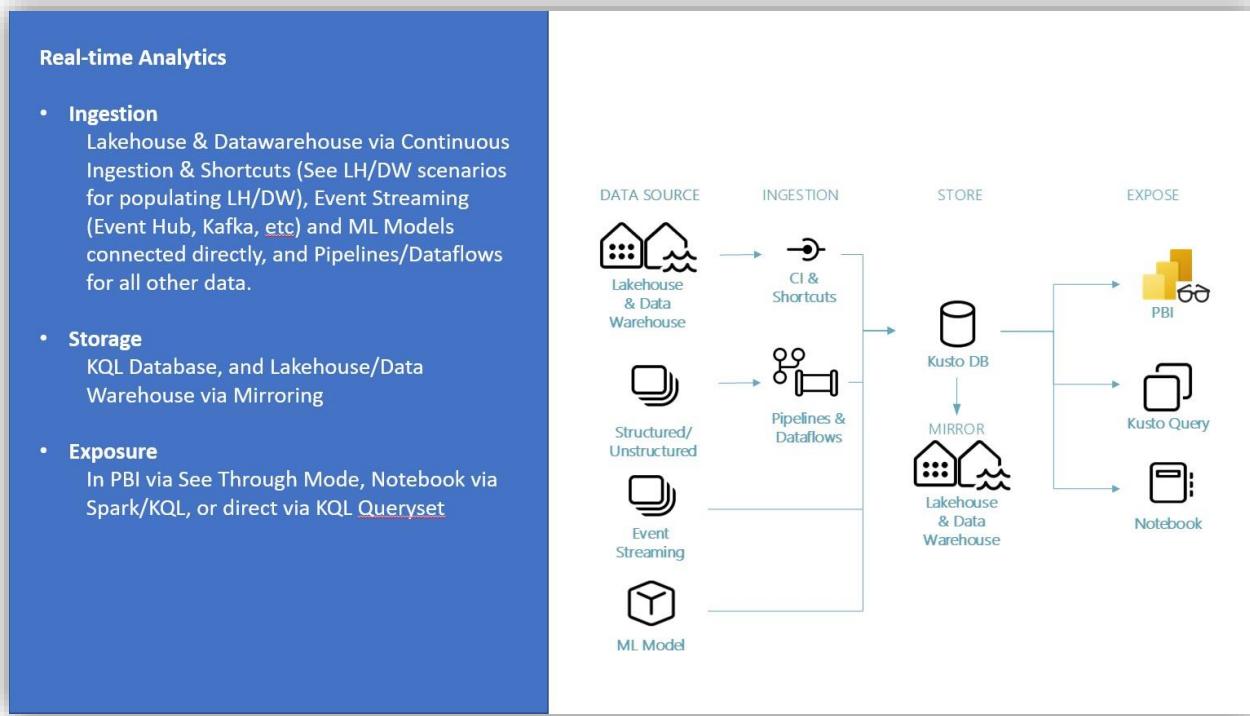


End-to-End Scenario Tutorial

Real-time Analytics

Updated : Feb 2024



Contents

| | |
|--|----|
| Providing feedback..... | 3 |
| Introduction | 5 |
| Real-time Analytics..... | 5 |
| Scenario | 6 |
| Prerequisites | 7 |
| Module 1: Create a Fabric workspace..... | 7 |
| Module 2: Build your first Real-time Analytics solution in Fabric..... | 9 |
| Create a KQL Database | 9 |
| Create an Eventstream | 11 |
| Stream data from Eventstream to KQL Database..... | 13 |
| Explore data and build Power BI report | 20 |
| Module 3: Extending the real-time analytics solution | 22 |
| Get dimension data from Blob Storage | 22 |
| Query data | 26 |
| Explore data further in the KQL Queryset | 27 |
| Build Power BI report | 30 |
| Create OneLake shortcut..... | 33 |
| Module 4: Clean up resources | 36 |

Introduction

What is Fabric?

Fabric provides a one-stop shop for all the analytical needs for every enterprise. It covers the complete spectrum of services including data movement, data lake, data engineering, data integration and data science, real time analytics, and business intelligence. With Fabric, there is no need to stitch together different services from multiple vendors. Instead, the customer enjoys an end-to-end, highly integrated, single comprehensive product that is easy to understand, onboard, create and operate. There is no other product on the market that offers the breadth, depth, and level of integration that Fabric offers. Additionally, Microsoft Purview is included by default in every tenant to meet compliance and governance needs.

To get an overview over the components and concepts of Fabric read [Fabric - Overview and Concepts](#).

The purpose of this tutorial

While many concepts in Fabric may be familiar to data and analytics professionals it can be challenging to apply those concepts in a new environment. This tutorial has been designed to walk step-by-step through an end-to-end scenario from data acquisition to data consumption to build a basic understanding of the Fabric UX, the various workloads and their integration points, and the Fabric professional and citizen developer experiences.

The tutorials are not intended to be a reference architecture, an exhaustive list of features and functionality, or a recommendation of specific best practices.

Real-time Analytics

Real-time Analytics is a portfolio of capabilities that provides end-to-end analytics streaming solution across Fabric experiences. It supplies high velocity, low latency data analysis, and is optimized for time-series data, including automatic partitioning and indexing of any data format and structure, such as structured data, semi-structured (JSON), and free text.

Real-time Analytics delivers high performance when it comes to your increasing volume of data. It accommodates datasets as small as a few gigabytes or as large as several petabytes and allows you to explore data from different sources and a variety of data formats.

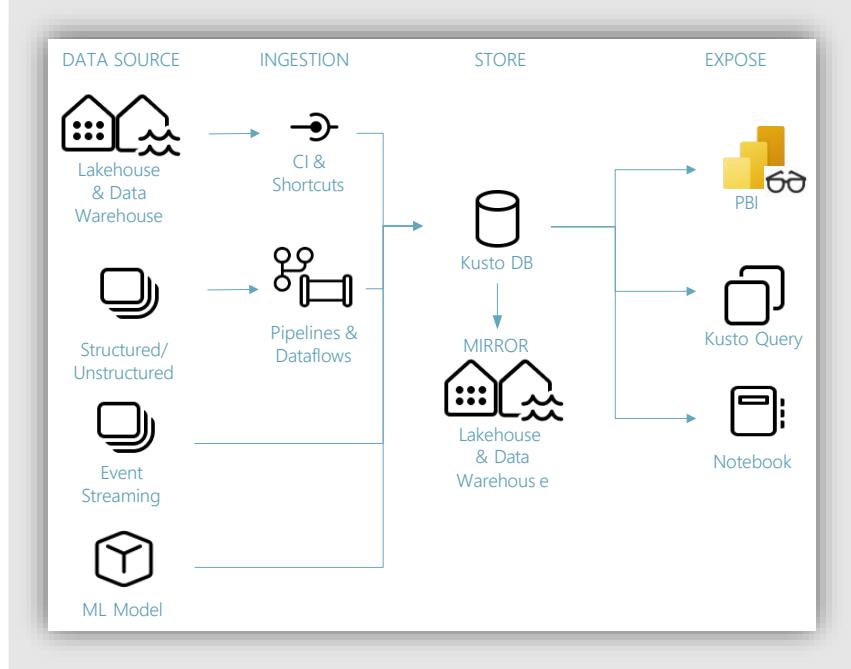


Figure 1: End-to-end Scenario - Real-time Analytics

Real-time Analytics includes integration with other Fabric experiences such as Lakehouse, Data Warehouse, Pipelines, Dataflows and Event Streaming on data sources and ingestion side. On the data exposition and consumption, Real-time analytics integrates with Power BI, and Notebooks.

You can use Real-time Analytics for a range of solutions, such as IoT analytics and log analytics, and in several scenarios including manufacturing operations, oil and gas, automotive, and more.

In this tutorial, you learn how to:

- Create a KQL Database
- Create Eventstream
- Stream data from Eventstream to KQL Database
- Check your data with sample queries
- Save queries as a KQL Queryset
- Create a Power BI report
- Create a OneLake shortcut

Scenario

This tutorial is based on a *sample on New York Yellow Taxi trip data*. The dataset contains trip records of New York's yellow taxis. The yellow taxi trip records include fields capturing pick-up and drop-off dates/times, pick-up and drop-off locations, trip distances, itemized fares, rate types, payment types, and driver-reported passenger counts. You'll use the streaming and query capabilities of Real-time Analytics to answer key questions about the trip statistics, taxi demand in the boroughs of New York and related insights.

Prerequisites

- Power BI Premium subscription. For more information, see [How to purchase Power BI Premium](#).
- Workspace

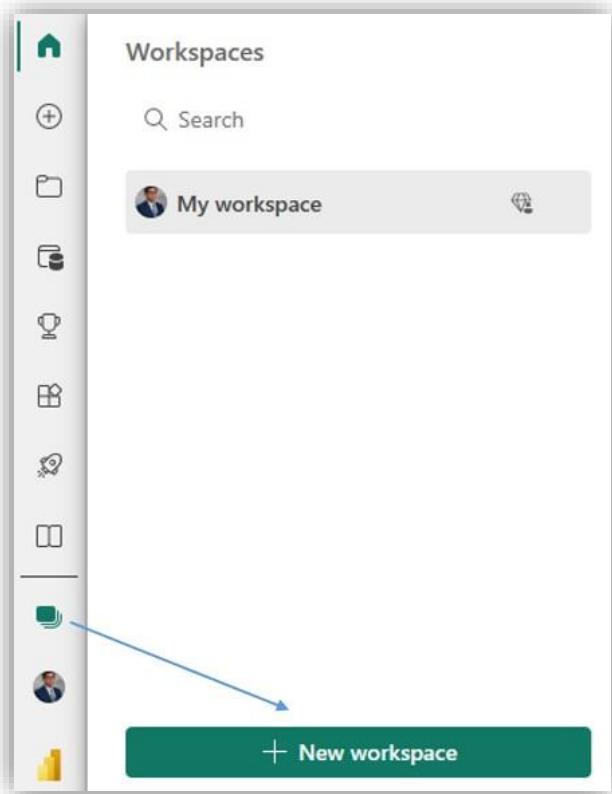
Module 1: Create a Fabric workspace

Before you can begin building the real-time analytics solution, you will need to enable Project Fabric on your Power BI tenant and create a workspace where you will build out the remainder of the tutorial. In this module, you will learn to:

- Create a Fabric workspace

In this step, you create a Fabric workspace in the Power BI service. The workspace will contain all the artifacts needed for real-time analytics including KQL Database, Eventstream, Power BI datasets, and reports.

1. Sign in to [Fabric](#).
2. Select **Workspaces > New Workspace**.



3. Fill out the **Create a workspace** form as follows:

- a. **Name:** Enter *Fabric Real-time Analytics Tutorial*, and some characters for uniqueness.
- b. **Description:** Optionally, enter a description for the workspace.

Create a workspace

Name * This name is available

Description

Domain

Learn more about workspace settings [?](#)

Workspace image

Advanced ^

Contact list *

License mode Pro
 Select Pro to use basic Power BI features and collaborate on reports, dashboards, and scorecards. To access a Pro workspace, users need Pro per-user licenses. [Learn more](#)

Trial
 Select the free trial per-user license to try all the new features and experiences in Microsoft Fabric for 60 days. A Microsoft Fabric trial license allows users to create Microsoft Fabric items and collaborate with others in a Microsoft Fabric trial capacity. Explore new capabilities in Power BI, Data Factory, Data Engineering, and Real-Time Analytics, among others. [Learn more](#)

Premium per-user
 Select Premium per-user to collaborate using Power BI Premium features, including paginated reports, dataflows, and datamarts. To collaborate and share content in a Premium per-user workspace, users need Premium per-user licenses. [Learn more](#)

Premium capacity
 Select premium capacity if the workspace will be hosted in a premium capacity. When you share content from a workspace with Power BI and Microsoft Fabric premium license, anyone who can access this content without needing a Pro or Premium per-user license. [Learn more](#)

Embedded
 Select embedded if the workspace will be hosted in an Azure embedded capacity. GIS and developers use Power BI Embedded to embed visuals and analytics in their applications. [Learn more](#)

Apply **Cancel**

4. Expand the **Advanced** section.
5. Choose **Fabric capacity** in the **License Mode** section.
6. Choose a Fabric capacity you have access to.

Choose an available Premium capacity for this workspace

- ▼

7. Select **Apply**. The workspace will be created and opened.

Module 2: Build your first Real-time Analytics solution in Fabric

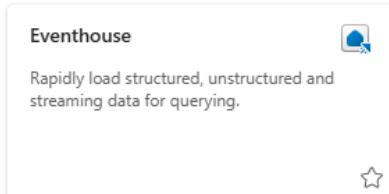
The intent of this module is to quickly build end to end journey of building a real-time analytics solution, ingesting streaming data from Eventstream and then using the KQL Database for creating a real-time refreshing Power BI report.

Create an Eventhouse

1. In the upper left corner of the Fabric Workspace home page, select **New > Eventhouse**

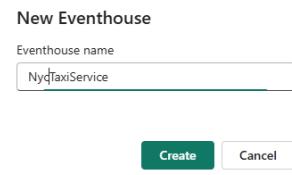
Track data

Monitor your streaming or nearly real-time o|

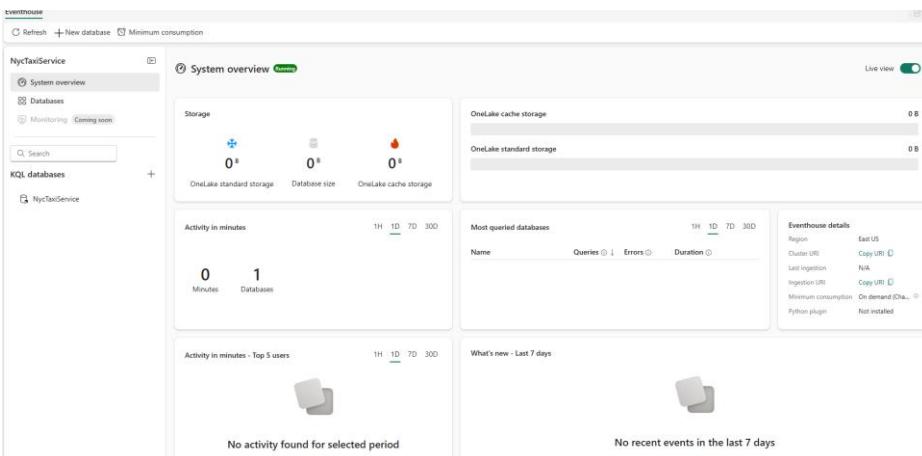


2. On the **New Eventhouse** dialog, enter a **unique name**.

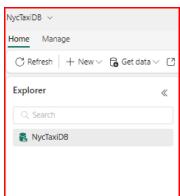
3. Select **Create**.



4. When provisioning is complete the KQL database editor landing page will be shown.



5. Select the **Database** in the Object tree.



6. Select **Explore your data** located in the upper right corner.

NycTaxiService

System overview

Databases

Monitoring Coming soon

Search

KQL databases +

NycTaxiService

Database: NycTaxiService

Database details

Created by: [redacted]
Region: East US
Created on: Today, 2m ago
Last ingestion: -

Query URI Copy URI Inactive

Ingestion URL Copy URI Inactive

OneLake availability Inactive

Size

0B Compressed size 0B Original size 0 Compression ratio

Explore your data

- Enable availability in Onelake (replace **NycTaxiService** with your database's name). In the **Database details** card, select the **pencil** icon.

NycTaxiService

System overview

Databases

Monitoring Coming soon

Search

KQL databases +

NycTaxiService

Database: NycTaxiService

Database details

Created by: [redacted]
Region: East US
Created on: Today, 2m ago
Last ingestion: -

Query URI Copy URI Inactive

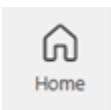
Ingestion URL Copy URI Inactive

OneLake availability Inactive

Toggle the button to **Active** and select **Done**

Upload data

- Return to the Real-Time Analytics home page. The **Home** icon directs you to the home page of the experience you're currently using within Fabric.



- In the **Eventhouse** section, select **Get data**, then select 'Local File' in the drop-down list.

Database Manage Eventhouse

Refresh + New Get data New related item

NycTaxiService

System overview

Databases

Monitoring Coming soon

Search

KQL databases +

NycTaxiService

Database: NycTaxiService

Database details

Created by: [redacted]
Region: East US
Created on: Today, 6m ago
Last ingestion: -

- Under your database, click on '**New table**'.

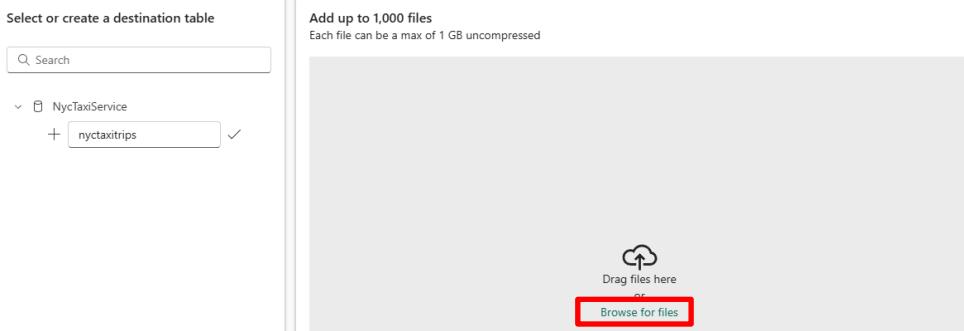
Select or create a destination table

Search

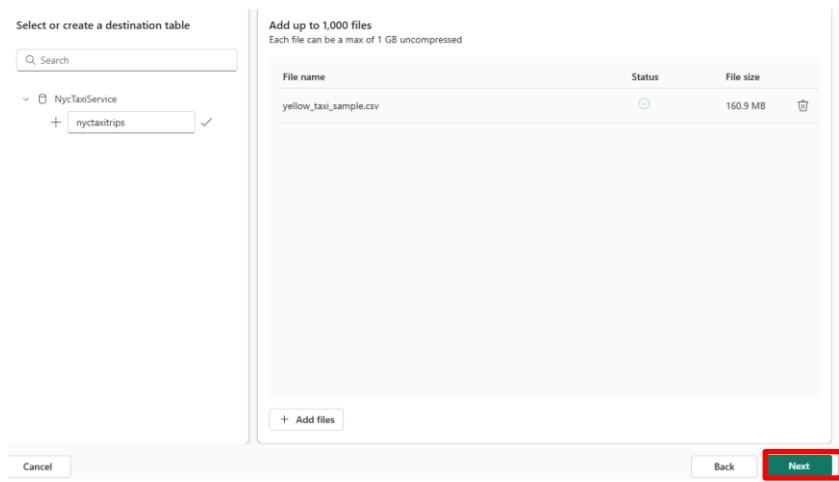
NycTaxiService

+ New table

- Enter 'nyctaxitrips' and then click on 'Browse for files' in the middle of the window. Locate the file 'yellow_taxi_sample.csv' on your computer.



5. Click on 'Next' in the lower right-hand corner once the file is uploaded.



6. In the Inspect tab, choose CSV as the Data format dropdown. Enable First row is column header.

| Preview data | | | | | | | Command viewer | <input checked="" type="checkbox"/> First row is column header | Schema definition file: yellow_t... | Format: CSV | <input type="button"/> Edit columns | <input type="button"/> Advanced |
|---------------------------------|---|--|--|--------------------------------------|-----------------------------------|---------------------------------------|----------------|--|-------------------------------------|-------------|-------------------------------------|---------------------------------|
| <code>\$`VendorID (long)</code> | <code>\$`tpep_pickup_datetime (datetime)</code> | <code>\$`tpep_dropoff_datetime (datetime)</code> | <code>\$`passenger_count (string)</code> | <code>\$`trip_distance (real)</code> | <code>\$`RatecodeID (real)</code> | <code>\$`store_and_fwd_flag (s</code> | | | | | | |
| 1 | 2022-06-04 15:00:01.0000 | 2022-06-04 15:07:58.0000 | 1.5 | 1 | N | | | | | | | |
| 2 | 2022-06-04 15:00:41.0000 | 2022-06-04 16:22:33.0000 | 24.08 | 1 | N | | | | | | | |
| 2 | 2022-06-04 15:00:55.0000 | 2022-06-04 15:51:01.0000 | 18.89 | 2 | N | | | | | | | |
| 1 | 2022-06-04 15:01:31.0000 | 2022-06-04 15:20:54.0000 | 1.5 | 1 | N | | | | | | | |

7. After choosing CSV as the Data format, data preview will refresh and show the data in strongly typed columns.

8. We will change the data types for multiple columns. Click on 'Edit columns'.

- For the column VendorID column name, **Change data type**, and then choose **int**.
- choose datatype as **long** for the columns: passenger_count, PULocationID, DOLocationID
- choose datatype as **real** for the following columns: extra, mta_tax, tolls_amount, improvement_surcharge, congestion_charge, airport_fee, trip_distance, fare_amount, tip_amount, total_amount, payment_type

After changing the datatypes of the above columns, Click on **Apply**

2. Click on Finish

Get data
Inspect the data

Eventstream → NycTaxiDB/nytaxitrips

Preview data Command viewer

Format: JSON Advanced

Data sample found. Fetch more data Discard and fetch new data

23 events found. 23 events match the selected settings. Open for more details.

| VendorID (int) | tripduration (float) | passengercount (long) | tripdistance (m) | RateCodeID (int) | store_and_fwd_flag (string) | PULocationID (long) | DOLocationID (long) | payment_type (int) |
|----------------|----------------------|-----------------------|------------------|------------------|-----------------------------|---------------------|---------------------|--------------------|
| 2 | 2022-09-19 05:25:30 | 2022-09-19 05:25:30 | 1 | N | 162 | 237 | | 1 |
| 2 | 2022-09-19 05:23:00 | 2022-09-19 05:24:11 | 6.67 | 1 | N | 124 | 69 | 1 |
| 2 | 2022-09-19 05:23:00 | 2022-09-19 05:28:14 | 2.38 | 1 | N | 142 | 166 | 1 |
| 2 | 2022-09-19 05:23:00 | 2022-09-19 05:25:00 | 3.59 | 1 | N | 140 | 209 | 1 |
| 1 | 2022-09-19 05:23:00 | 2022-09-19 05:25:00 | 7 | 1 | N | 249 | 81 | 1 |
| 2 | 2022-09-19 05:23:00 | 2022-09-19 05:25:00 | 2.57 | 1 | N | 144 | 66 | 1 |
| 2 | 2022-09-19 05:28:00 | 2022-09-19 05:28:00 | 3.88 | 1 | N | 113 | 163 | 1 |
| 2 | 2022-09-19 05:28:00 | 2022-09-19 05:28:00 | 3.44 | 1 | N | 248 | 201 | 1 |
| 2 | 2022-09-19 05:31:00 | 2022-09-19 05:42:00 | 1.82 | 1 | N | 151 | 162 | 1 |
| 2 | 2022-09-19 05:33:00 | 2022-09-19 05:48:11 | 4.78 | 1 | Y | 148 | 79 | 1 |
| 1 | 2022-09-19 05:33:00 | 2022-09-19 05:45:00 | 1.7 | 1 | N | 239 | 230 | 1 |
| 2 | 2022-09-19 05:38:00 | 2022-09-19 05:40:37 | 0.32 | 1 | N | 260 | 280 | 1 |
| 2 | 2022-09-19 05:38:00 | 2022-09-19 05:39:00 | 4.69 | 1 | N | 231 | 230 | 1 |
| 2 | 2022-09-19 05:41:00 | 2022-09-19 05:41:00 | 19.54 | 2 | N | 152 | 239 | 1 |
| 1 | 2022-09-19 05:41:00 | 2022-09-19 05:47:00 | 6.1 | 1 | N | 234 | 7 | 2 |
| 1 | 2022-09-19 05:45:40 | 2022-09-19 05:52:00 | 4.2 | 1 | N | 250 | 7 | 2 |
| 2 | 2022-09-19 05:51:00 | 2022-09-19 05:51:00 | 0.38 | 1 | N | 79 | 74 | 1 |
| 2 | 2022-09-19 05:58:37 | 2022-09-19 05:59:00 | 2.31 | 1 | N | 163 | 282 | 1 |
| 2 | 2022-09-19 05:58:37 | 2022-09-19 05:58:37 | 8.19 | 1 | N | 254 | 107 | 1 |
| 1 | 2022-09-19 05:58:37 | 2022-09-19 05:58:37 | 3.1 | 1 | N | 150 | 230 | 1 |
| 2 | 2022-09-19 05:58:37 | 2022-09-19 05:58:37 | 1.63 | 1 | N | 80 | 231 | 1 |

Cancel Back Finish

3. In the **Summary** Tab, **Continuous ingestion from local file established** window, all steps will be marked with green check marks when the data connection is successfully created. Click the **Close** button.

Get data
Summary

Local file → NycTaxiDB/nytaxitrips 1 blobs, 1 succeeded, 0 failed ✓

Data Preparation Details

- Create table (nytaxitrips) -
- Create mapping (nytaxitript_mapping) -
- Data queuing -

Blob name Status Details

https://\$virtualdms.blob.core.windows.net/tr... Successfully ingested

What can you do with the data? Now that you've ingested data, you can explore, run queries, and visualize the data using the dashboard.

Explore the results Search, sort, filter, and expand rows to highlight trends or problems.

Undo ingestion Delete the data you've just ingested.

Create new dashboard View your ingested data's key metrics, sample records, and ingestion history.

Explore Delete ingested data Create dashboard

Close

Explore data and build Power BI report

- Click on 'Refresh' at the top left of the screen.
- In the object tree for the KQL database, select the table **nytaxitrips**.

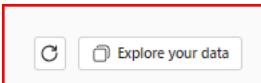
Explorer

Search

NycTaxiDB

- Tables
 - nytaxitrips
- Shortcuts
- Materialized views
- Functions
- Data streams

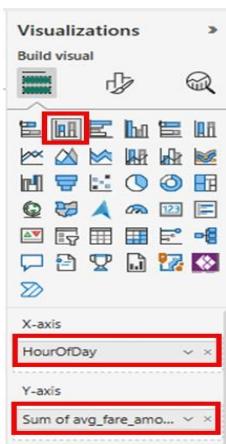
3. In the top right corner, select **Explore your data**.



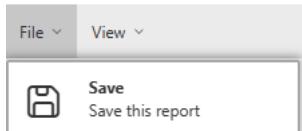
4. Paste the following query in the query editor and select **Run**

```
//Calculate average fare amount by the hour of the day. nyctaxitrips  
| summarize avg(fare_amount) by HourOfDay = hourofday(tpep_dropoff_datetime)
```

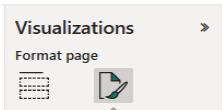
5. Click on **Power BI** in the upper right-hand corner. An empty PowerBI report editing window will open.
6. Select **Stacked Column Chart** in the Visualizations pane. Drag **HourOfDay** field to X-axis and **avg_fare_amount** to Y-axis.



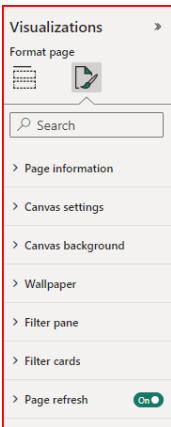
7. Select **File** menu and then **Save** in the top left corner



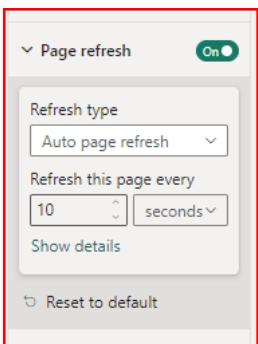
8. Enter **nyctaxitripstats** in the **Name your file in Power BI** field, and choose your workspace.
9. Once the report is saved, click on the link **Open the file in Power BI to view, edit, and get a shareable link**.
10. Click on **Edit** button to edit the Power BI report.
11. Click on an Empty Space in the Canvas, Choose **Format page**.



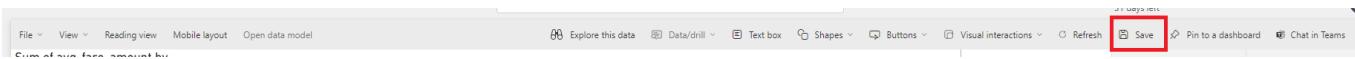
12. Toggle **Page Refresh** to **On**



13. Expand **Page Refresh** and set the refresh interval to 10 seconds. Please note: Refresh interval will be limited by the Admin interval. Refresh interval can only be greater than or equal to the Admin interval.



14. Save your report.



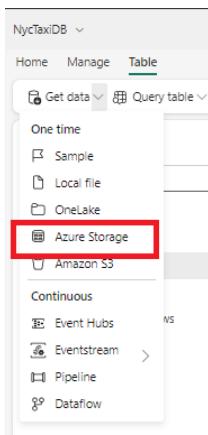
With this tutorial, you have now built an auto-refreshing Power BI report that is querying streaming data arriving in KQL Database from Eventstream.

Module 3: Extending the real-time analytics solution

Get dimension data from Azure Storage

In this module, you are going to ingest Location available in Azure storage. This data contains additional information on the pick-up locations and drop-off locations used in the trips dataset. Real-time analytics reads and ingests data directly from the blob storage without requiring any other intermediary service.

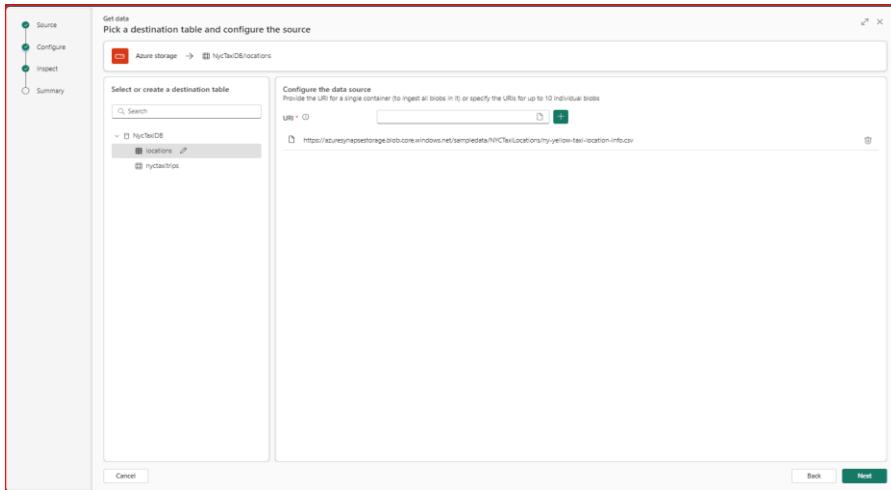
1. Navigate to KQL Database **NycTaxiService** located in your workspace.
2. From within the KQL Database, select **Get Data > Azure Storage**.



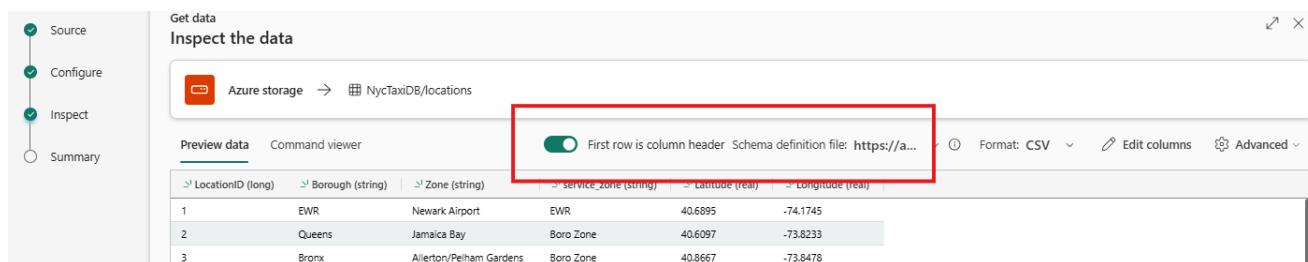
Destination tab

In the section ‘Select or create a destination table’, your **Database** is auto populated with the name of the selected KQL database.

- Under **Table**, make sure that **New table** is selected, and enter **locations** as your table name.



- URI : <https://csarealtimedemo.blob.core.windows.net/sampledata/NYCTaxiLocations/ny-yellow-taxi-location-info.csv> click on + icon and click **Next** :
- Enable ‘First row is column header’
- In the **Inspect tab**, click on **Finish**



Get data

Inspect the data

Azure storage → NyTaxiDB/locations

Preview data Command viewer Schema definition file: https://a... Format: CSV Edit columns Advanced

| | LocationID (long) | Borough (string) | Zone (string) | service_zone (string) | Latitude (real) | Longitude (real) |
|----|-------------------|--------------------------------------|---------------|-----------------------|-----------------|------------------|
| 1 | EWR | Newark Airport | EWR | Boro Zone | 40.6895 | -74.1745 |
| 2 | Queens | Jamaica Bay | Boro Zone | Boro Zone | 40.0097 | -73.8233 |
| 3 | Bronx | Allerton/Pelham Gardens | Boro Zone | Boro Zone | 40.8557 | -73.8478 |
| 4 | Manhattan | Alphabet City | Yellow Zone | Boro Zone | 40.7258 | -73.9795 |
| 5 | Staten Island | Arden Heights | Boro Zone | Boro Zone | 40.5583 | -74.1737 |
| 6 | Staten Island | Brooklyn/Fort Wadsworth | Boro Zone | Boro Zone | 40.5954 | -74.0571 |
| 7 | Queens | Astoria | Boro Zone | Boro Zone | 40.7544 | -73.9335 |
| 8 | Queens | Astoria Park | Boro Zone | Boro Zone | 40.779 | -73.9229 |
| 9 | Queens | Auburndale | Boro Zone | Boro Zone | 40.757 | -73.7898 |
| 10 | Queens | Bayside Park | Boro Zone | Boro Zone | 40.6781 | -73.7055 |
| 11 | Brooklyn | Bath Beach | Boro Zone | Boro Zone | 40.594224 | -74.006121 |
| 12 | Manhattan | Battery Park | Yellow Zone | Boro Zone | 40.705282 | -74.017027 |
| 13 | Manhattan | Battery Park City | Yellow Zone | Boro Zone | 40.711931 | -74.016869 |
| 14 | Brooklyn | Bay Ridge | Boro Zone | Boro Zone | 40.625801 | -74.030621 |
| 15 | Queens | Bay Terrace/Fort Totten | Boro Zone | Boro Zone | 40.791211 | -73.776978 |
| 16 | Queens | Bayside | Boro Zone | Boro Zone | 40.76314 | -73.77125 |
| 17 | Brooklyn | Bedford | Boro Zone | Boro Zone | 40.687216 | -73.941776 |
| 18 | Bronx | Bedford Park | Boro Zone | Boro Zone | 40.8701 | -73.885691 |
| 19 | Queens | Bellrose | Boro Zone | Boro Zone | 40.73301 | -73.722938 |
| 20 | Bronx | Belmont | Boro Zone | Boro Zone | 40.85232 | -73.88801 |
| 21 | Brooklyn | Bensonhurst East | Boro Zone | Boro Zone | 40.610019 | -73.992507 |
| 22 | Brooklyn | Bensonhurst West | Boro Zone | Boro Zone | 40.611011 | -74.002925 |
| 23 | Staten Island | Bloomfield/Emerson Hill | Boro Zone | Boro Zone | 40.055883 | -74.105138 |
| 24 | Manhattan | Bloomindale | Yellow Zone | Boro Zone | 40.801181 | -73.954446 |
| 25 | Brooklyn | Boerum Hill | Boro Zone | Boro Zone | 40.685803 | -73.954401 |
| 26 | Brooklyn | Borough Park | Boro Zone | Boro Zone | 40.633993 | -73.996938 |
| 27 | Queens | Breezy Point/Fort Tilden/Rilis Beach | Boro Zone | Boro Zone | 40.557154 | -73.925042 |

Cancel Finish

7. Summary tab

In the **Data ingestion is in progress** window, all steps will be marked with green check marks when the data has been successfully ingested. The data from Azure Storage will begin streaming automatically into your table.

Get data

Summary

Azure storage → NyTaxiDB/locations

1 blobs, 1 succeeded, 0 failed

| Blob name | Status | Details |
|--|-----------------------|---------|
| https://azuresynapsestorage.blob.core.windows... | Successfully ingested | - |

What can you do with the data?

Now that you've ingested data, you can explore, run queries, and visualize the data using the dashboard

Explore the results

Search, sort, filter, and expand rows to highlight trends or problems.

Explore

Undo ingestion

Delete the data you've just ingested.

Delete Ingested data

Create new dashboard

Create your ingested data's key metrics, sample records, and ingestion history

Create dashboard

Close

Now that you've got data in your database, click **Close**. You're going to check your data with sample queries.

Query data

In the following step, you'll use the advanced data analysis capabilities of Kusto Query language to query your telemetry data.

1. Select **Explore your data** on the right-hand side of your database-editor.

The screenshot shows the Azure Data Explorer interface. On the left, there's a navigation bar with 'Database / Tables / Table: nyctaxitrips'. The main area displays 'Table details' for the 'nyctaxitrips' table, including row count (62,900), last ingestion (Today, this minute), schema information, and file sizes (3.2 MB compressed, 10.97 MB original). To the right, there's a 'Mappings' section with a clipboard icon and a note 'No items to show'. In the top right corner, there's a button labeled 'Explore your data' with a red box around it.

Note: The numbers in the screen capture above may look different in your database editor page.

2. Let's take a look at the data itself. Paste the following query in **Explore your data** window to take 10 random records from your data.

Query to be pasted →

*nyctaxitrips
/ take 10*

The screenshot shows the 'Explore your data' window. On the left, there's a 'Run' button and a 'Save as KQL queryset' button. The main area contains the KQL query: 'nyctaxitrips | take 10'. The results pane shows the first 10 records of the 'nyctaxitrips' table. The entire window is enclosed in a red box.

```
1 //*****  
2 // Here are two articles to help you get started with KQL:  
3 // KQL reference guide - https://aka.ms/kqlguide  
4 // SQL - KQL conversions - https://aka.ms/sqlcheatsheet  
5 //*****  
6  
7 // Use "take" to view a sample number of records in the table and check the data.  
8 nyctaxitrips  
9 | take 10  
10
```

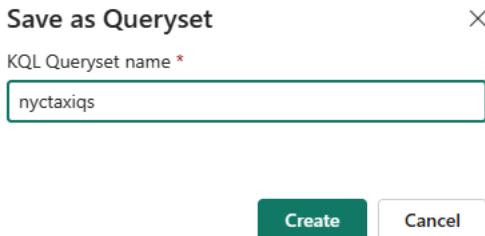
3. Select Run.

The screenshot shows the results of the KQL query 'nyctaxitrips | take 10'. The table has columns: VendorID, tpep_pickup_datetime, tpep_dropoff_datetime, passenger_count, trip_distance, RatecodeID, and store_and_fwd_flag. The data consists of 10 rows, each representing a taxi trip record.

| VendorID | tpep_pickup_datetime | tpep_dropoff_datetime | passenger_count | trip_distance | RatecodeID | store_and_fwd_flag |
|----------|--------------------------|--------------------------|-----------------|---------------|------------|--------------------|
| > 2 | 2022-06-06 18:00:04.0000 | 2022-06-06 18:09:47.0000 | 1 | 2.21 | 1.0 | N |
| > 2 | 2022-06-06 18:05:30.0000 | 2022-06-06 18:25:48.0000 | 1 | 3.46 | 1.0 | N |
| > 2 | 2022-06-06 18:26:37.0000 | 2022-06-06 18:33:04.0000 | 1 | 1.02 | 1.0 | N |
| > 2 | 2022-06-06 18:33:18.0000 | 2022-06-06 18:37:46.0000 | 2 | 0.93 | 1.0 | N |
| > 2 | 2022-06-06 18:35:16.0000 | 2022-06-06 18:55:46.0000 | 1 | 6.42 | 1.0 | N |
| > 2 | 2022-06-06 18:37:57.0000 | 2022-06-06 18:53:17.0000 | 1 | 9.95 | 1.0 | N |
| > 2 | 2022-06-06 18:41:01.0000 | 2022-06-06 18:49:22.0000 | 1 | 1.17 | 1.0 | N |
| > 2 | 2022-06-06 18:51:12.0000 | 2022-06-06 19:04:42.0000 | 1 | 4.35 | 1.0 | N |
| > 2 | 2022-06-06 18:55:50.0000 | 2022-06-06 18:56:59.0000 | 2 | 0.27 | 1.0 | N |
| > 2 | 2022-06-06 18:55:50.0000 | 2022-06-06 18:56:59.0000 | 2 | 0.27 | 1.0 | N |

4. Select **Save as KQL Query Set** to save this and future queries for later use.

- Under **KQL Queryset name**, enter *nyctaxiqs*, then select **Create**.



'Explore your data' enables you to run some quick queries to understand your data. This query can be saved as a KQL Queryset and persisted in the workspace as an item. Query set autosaves the queries as you type them and lets you resume from the point where you had stopped. In the next module, you will work with the KQL Queryset. Saving the quick query as KQL Query Set will automatically open your **KQL Queryset** with the queries that you wrote in the query editor.

Explore data further in the KQL Queryset

In this module, you are going to write queries using *Kusto Query Language* to explore the data that you have ingested from the Event hub and blob storage. Kusto Query Language is a powerful tool to explore your data and discover patterns, identify anomalies and outliers, create statistical modeling, and more. The query uses schema entities that are organized in a hierarchy similar to SQLs: databases, tables, and columns. Kusto query is a read-only request to process data and return results. The request is stated in plain text, using a data-flow model that is easy to read, author, and automate. Kusto queries are made of one or more query statements. You are going to write some simple Kusto queries to get familiar with the language and discover its power and simplicity.

Run the following queries in the new KQL Queryset you have created. Copy/paste each query into your environment, select the query and then select **Run**.

- The following query returns the top 10 pickup locations in New York City for Yellow Taxis.

Query to be pasted →

```
//Top 10 pickup locations
nyctaxitrips
| summarize count() by PULocationID
| top 10 by count_
```

Table 1 Stats

| PULocationID | count_ |
|--------------|--------|
| > 237 | 61,883 |
| > 236 | 55,365 |
| > 132 | 55,057 |
| > 161 | 47,341 |
| > 162 | 40,191 |
| > 142 | 40,128 |
| > 186 | 37,716 |
| > 170 | 37,533 |
| > 230 | 35,370 |

Note: Result of the query may not exactly match the screenshot provided as you are ingesting streaming data.

2. We will run the same query as in the previous step with an addition of looking up the corresponding zones of the top 10 pickup locations by using the 'locations' table.

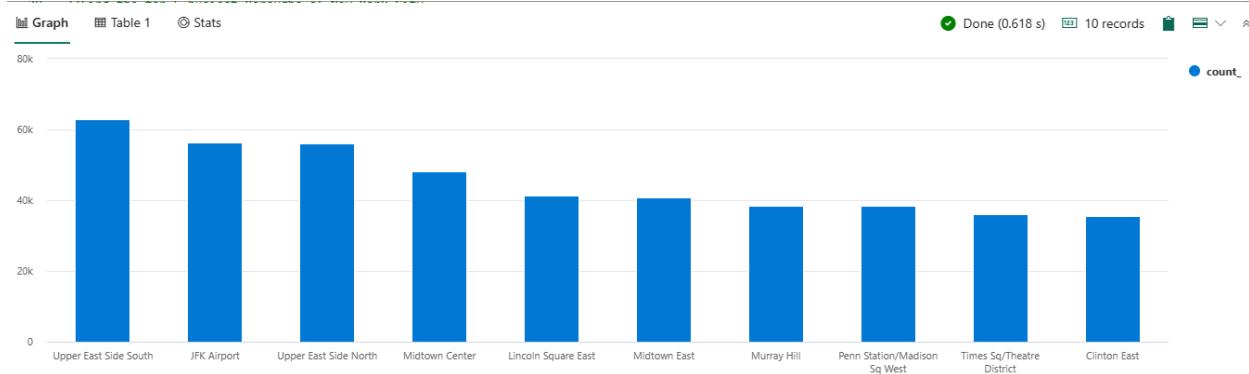
Query to be pasted →

```
//For the same top 10 locations, lookup the NYC zones --> Top 10 zones
nyctaxitrips
| lookup (locations) on $left.PULocationID == $right.LocationID
| summarize count() by Zone
| top 10 by count_
| render columnchart
```

Graph Table 1 Stats

| Zone | count_ |
|------------------------------|--------|
| Upper East Side South | 62,942 |
| JFK Airport | 56,417 |
| Upper East Side North | 56,052 |
| Midtown Center | 48,263 |
| Lincoln Square East | 41,236 |
| Midtown East | 40,895 |
| Murray Hill | 38,479 |
| Penn Station/Madison Sq West | 38,324 |
| Times Sq/Theatre District | 36,167 |
| Clinton East | 35,622 |

Note: Result of the query may not exactly match the screenshot provided as you are ingesting streaming data

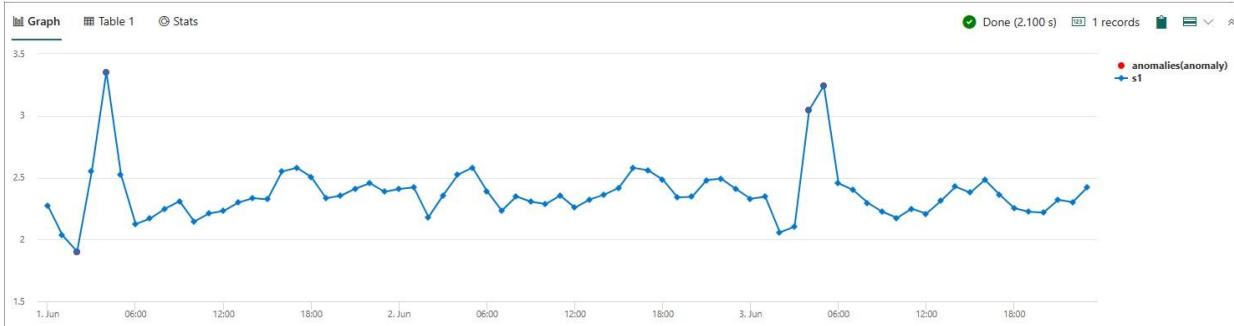


3. Let's check anomalies in the tips that have been given by the customers in the Manhattan borough. Hover over the red dots to see the values.

Query to be pasted →

```
//Find anomalies in the tips given by the customers
nyctaxitrips
| lookup (locations) on $left.PULocationID==$right.LocationID
| where Borough == "Manhattan"
| make-series s1 = avg(tip_amount) on tpep_pickup_datetime from datetime(2022-06-01) to
datetime(2022-06-04) step 1h
| extend anomalies = series_decompose_anomalies(s1)
| render anomalychart with (anomalycolumns=anomalies)
```

Note: Result of the query may not exactly match the screenshot provided as you are ingesting streaming data.

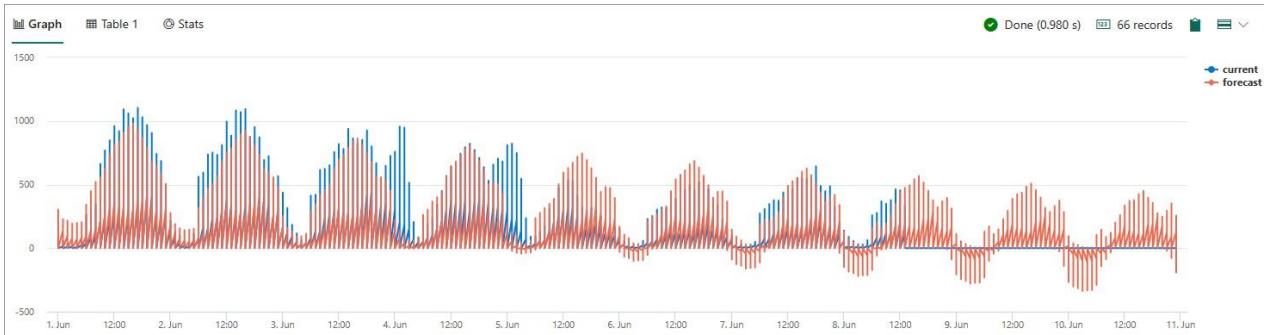


4. To ensure that the sufficient taxis are plying in the Manhattan borough, let's forecast the number of taxis needed per hour.

Query to be pasted →

```
//Forecast the number of trips that will begin from Manhattan to line up the taxis in that borough
nyctaxitrips
| lookup (locations) on $left.PULocationID==$right.LocationID
| where Borough == "Manhattan"
| make-series s1 = count() on tpep_pickup_datetime from datetime(2022-06-01) to datetime(2022-06-08)+3d step 1h by PULocationID
| extend forecast = series_decompose_forecast(s1, 24*3)
| render timechart
```

Note: Result of the query may not exactly match the screenshot provided below as you are ingesting streaming data.



Build Power BI report

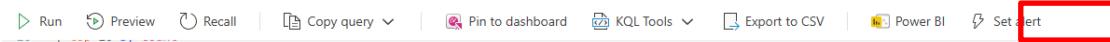
A Power BI report is a multi-perspective view into a dataset, with visuals that represent findings and insights from that dataset.

1. Continuing the in same queryset, paste the following query. The output of this query will be used as the dataset for building the Power BI report.

Query to be pasted →

```
//Find the total number of trips that started and ended at the same location
nyctaxitrips
| where PULocationID == DOLocationID
| lookup (locations) on $left.PULocationID==$right.LocationID
| summarize count() by Borough, Zone, Latitude, Longitude
```

2. Select the query and then select **Power BI** on the top ribbon.



```

17 | render columnchart
18
19
20 //Find anomalies in the tips given by the customers
21 nyctaxitrips
22 | lookup (locations) on $left.PULocationID==$right.LocationID
23 | where Borough == "Manhattan"
24 | make-series s1 = avg(tip_amount) on tpep_pickup_datetime from datetime(2022-06-01) to datetime(2022-06- 04) step 1h
25 | extend anomalies = series_decompose_anomalies(s1)
26 | render anomalychart with (anomalycolumns=anomalies)
27
28 //Find the total number of trips that started and ended at the same location
29 nyctaxitrips
30 | where PULocationID == DOLocationID
31 | lookup (locations) on $left.PULocationID==$right.LocationID
32 | summarize count() by Borough, Zone, Latitude, Longitude

```

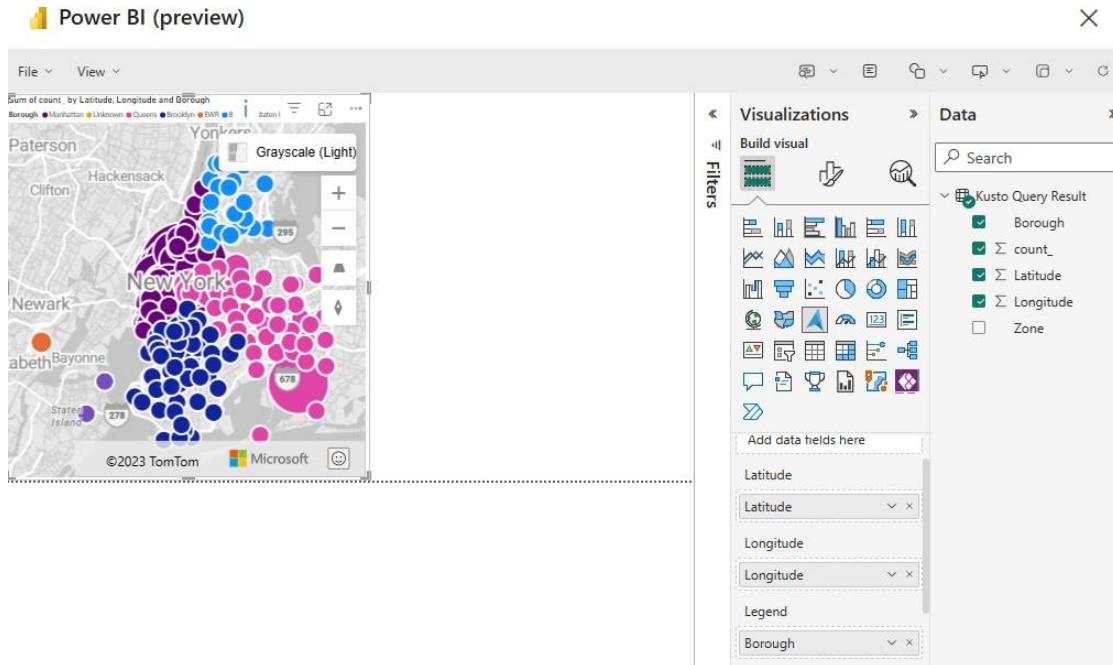
Power BI report editor will open with the result of the query available as a table with the name Kusto Query Result.

Note:

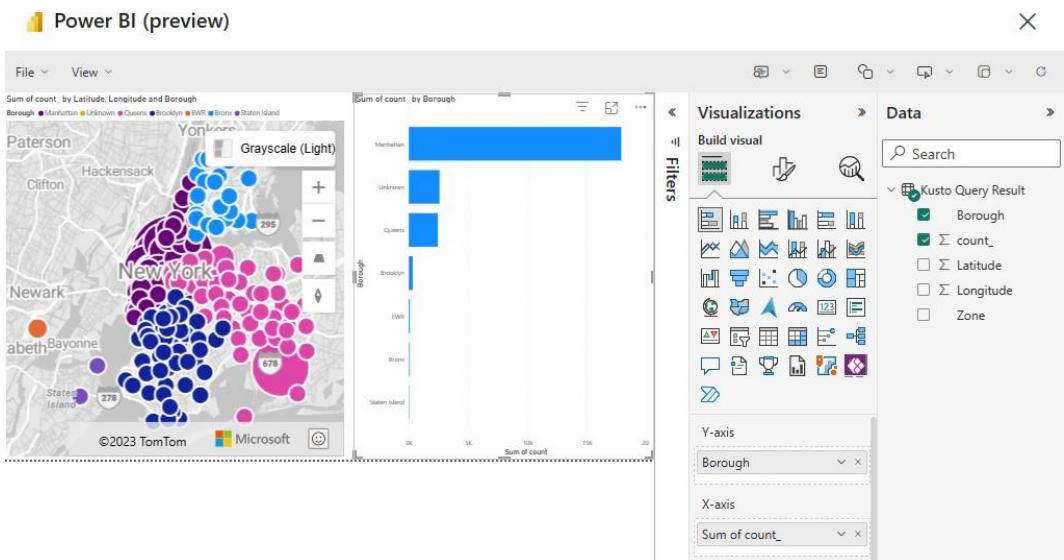
- (a) When you build a report, a dataset is created and saved in your workspace. You can create multiple reports from a single dataset. (b) Result of the query may not exactly match the screenshot provided below as you are ingesting streaming data.

If you delete the dataset, your reports will also be removed.

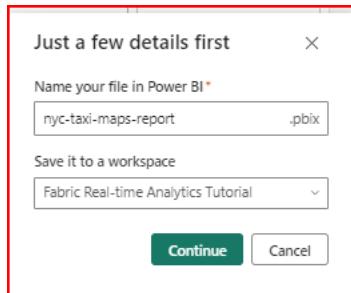
3. In the report editor, choose **Azure Maps** as the visual, drag **Latitude** field to Latitude, **Longitude** field to Longitude, **Borough** field to Legend, and **count_** field to Size.



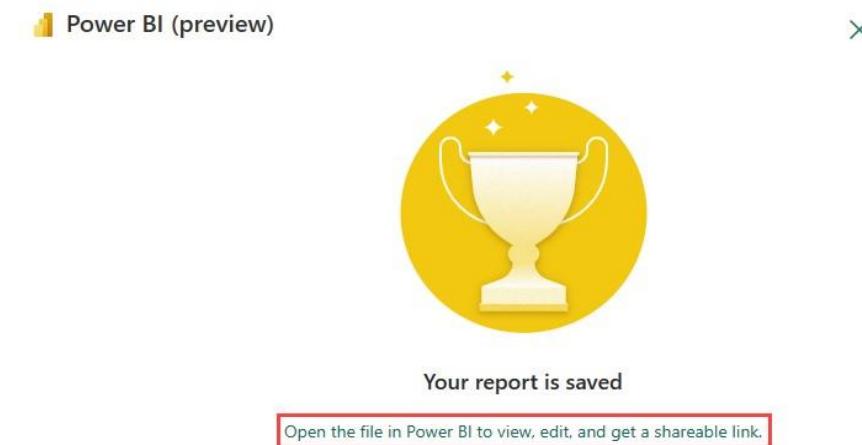
4. Add a **Stacked Bar Chart** to the canvas. Drag **Borough** field to Y-Axis and **count_** to the X-axis



5. Click File > Save
6. Under **Name your file in Power BI**, enter *nyc-taxi-maps-report*.



7. Select the workspace in which you want to save this report. The report can be saved in a different workspace than the one you started in.
8. Select **Open the file in Power BI to view, edit, and get a shareable link** to view and edit your report.



Create OneLake shortcut

Now that you have finished exploring your data, you may want to access the underlying data from other Fabric experiences.

OneLake is a single, unified, logical data lake for Fabric to store lakehouses, warehouses and other items. Shortcuts are embedded references within OneLake that point to other files' store locations. The embedded reference makes it appear as though the files and folders are stored locally but in reality; they exist in another storage location. Once you create a shortcut, you can access your data in all of Fabric's experiences. Shortcuts can be updated or removed from your items, but these changes will not affect the original data and its source.

1. Select **Create** in the **Navigation pane**.



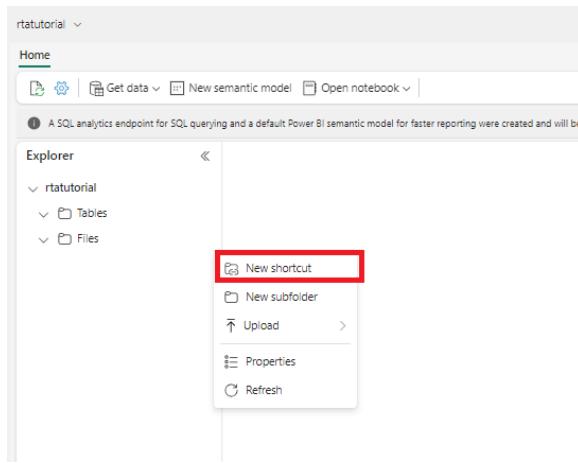
2. Under **Data engineering**, select **Lakehouse**.

A screenshot of the Data Engineering section. It contains four cards: 1. Lakehouse: "Store big data for cleaning, querying, reporting, and sharing." 2. Notebook: "Explore data and build machine learning solutions with Apache Spark applications." 3. Environment (Preview): "Set up shared libraries, Spark compute settings, and resources for notebooks and Spark job definitions." 4. Spark Job Definition: "Define, schedule, and manage your Apache Spark jobs for big data processing." The "Lakehouse" card is highlighted with a red box.

3. Enter *rtatutorial* as your Lakehouse name, then select **Create**.

A screenshot of the "New lakehouse" dialog box. It has a title bar "New lakehouse" and a close button "X". Below the title is a "Name *" label followed by a text input field containing "rtatutorial". At the bottom are two buttons: a blue "Create" button with a red border and a white "Cancel" button.

1. In Lakehouse Explorer, navigate to **Files**, select "...", select **New Shortcut** from the menu.



1. Under Internal sources, select OneLake.

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

Azure Data Lake Storage Gen2 (Azure)

Amazon S3 (File)

2. In Select a data source type, Filter KQL Database and select **your KQL Database**, then select **Next** to connect the data to your shortcut.

Select a data source type

When accessing this shortcut using a dataset or T-SQL, the identity of the calling item's owner is used to authorize access rather than the user's identity. [Learn more](#)

rtutorial is located in the region West US 3. Any data sourced through this shortcut will be processed in the same region.

Find and connect to the data you want to use with your shortcut.

All My data Endorsed in your org Favorites

Filter by keyword Filter (1)

| Name | Type | Capacity region | Owner | Location | Endorsement | Sensitivity |
|-----------|--------------|-----------------|-------------------|-------------------------|-------------|-------------|
| NyCTaxiDB | KQL Database | West US 3 | MOD Administrator | Fabric Real-time Ana... | - | - |

Previous Next Cancel

3. To connect the table with the data from Eventstream, select > to expand the tables in the left-hand pane, then select the table titled **nyctaxitrips**.

New shortcut

⚠️ When accessing this shortcut using a dataset or T-SQL, the identity of the calling item's owner is used to authorize access rather than the user's identity. [Learn more](#)

rtutorial is located in the region West US 3. Any data sourced through this shortcut will be processed in the same region.

Find and connect to the data you want to use with your shortcut.

OneLake

NycTaxiDB

Tables

locations

nyctaxitrips

Select the table and files that you want to include in your shortcut.

Previous Next Cancel

4. Select **Create** to create the shortcut. The Lakehouse will automatically refresh. Click on Close.

New shortcut

Review the folders and tables, and then Create. Each table and folder selected will appear as a unique shortcut in the location path.

Shortcut Location
rtutorial > Files

| Shortcut Name | Source | Actions | Status |
|---------------|-----------------------------------|---------|---------|
| nyctaxitrips | NycTaxiDB > Tables > nyctaxitrips | | Success |

Close

The Lakehouse shortcut has been created. Without additional management, you now have one logical copy of your data in the Lakehouse that you can use in other Fabric experiences, such as Notebooks and Spark jobs.

Module 4: Clean up resources

You can delete individual reports, eventstreams, KQL databases, KQL querysets, and other items or remove the entire workspace.

1. Select Fabric Real-time Analytics Solution Tutorial in the left-hand navigation menu to return to the workspace artifact view



2. Below the workspace name and description at the top of the workspace header, select Workspace settings.

Fabric Real-time Analytics Tutorial

+ New Upload Create deployment pipeline Create app Manage access Workspace settings

3. Select Other > Remove this workspace

Workspace settings X

Delete this workspace and all data and items in it?

This workspace will be permanently deleted after a retention period of 7 days, as defined by your Fabric administrator. Contact your Fabric administrator if you need to restore the workspace and recover the items in it during this retention period.

[Remove this workspace](#)

About

Premium

Azure connections

System storage

Git integration

OneLake

Other

Power BI

Data Engineering/Science

4. Select Delete on the warning.