

Create Real-Time Dashboards with Microsoft Fabric

1. Introduction

<https://learn.microsoft.com/en-us/training/modules/create-real-time-dashboards-microsoft-fabric/1-introduction>

Introduction

Completed

Imagine you're an analyst at a bike sharing company that operates across neighborhoods in a city. As bikes are rented and returned throughout the day, your company needs to monitor current bike availability and dock capacity to make operational decisions like redistributing bikes to high-demand areas.

You can use Real-Time Dashboards in Microsoft Fabric to monitor this changing data. Real-Time Dashboards connect to KQL databases in eventhouses and display visualizations that refresh automatically to show current data. Unlike traditional reports that show historical snapshots, Real-Time Dashboards can be configured to refresh frequently, enabling you to monitor changing conditions and respond to current information.

In this module, you'll learn how to create Real-Time Dashboards in Microsoft Fabric and connect them to KQL databases that contain your data.

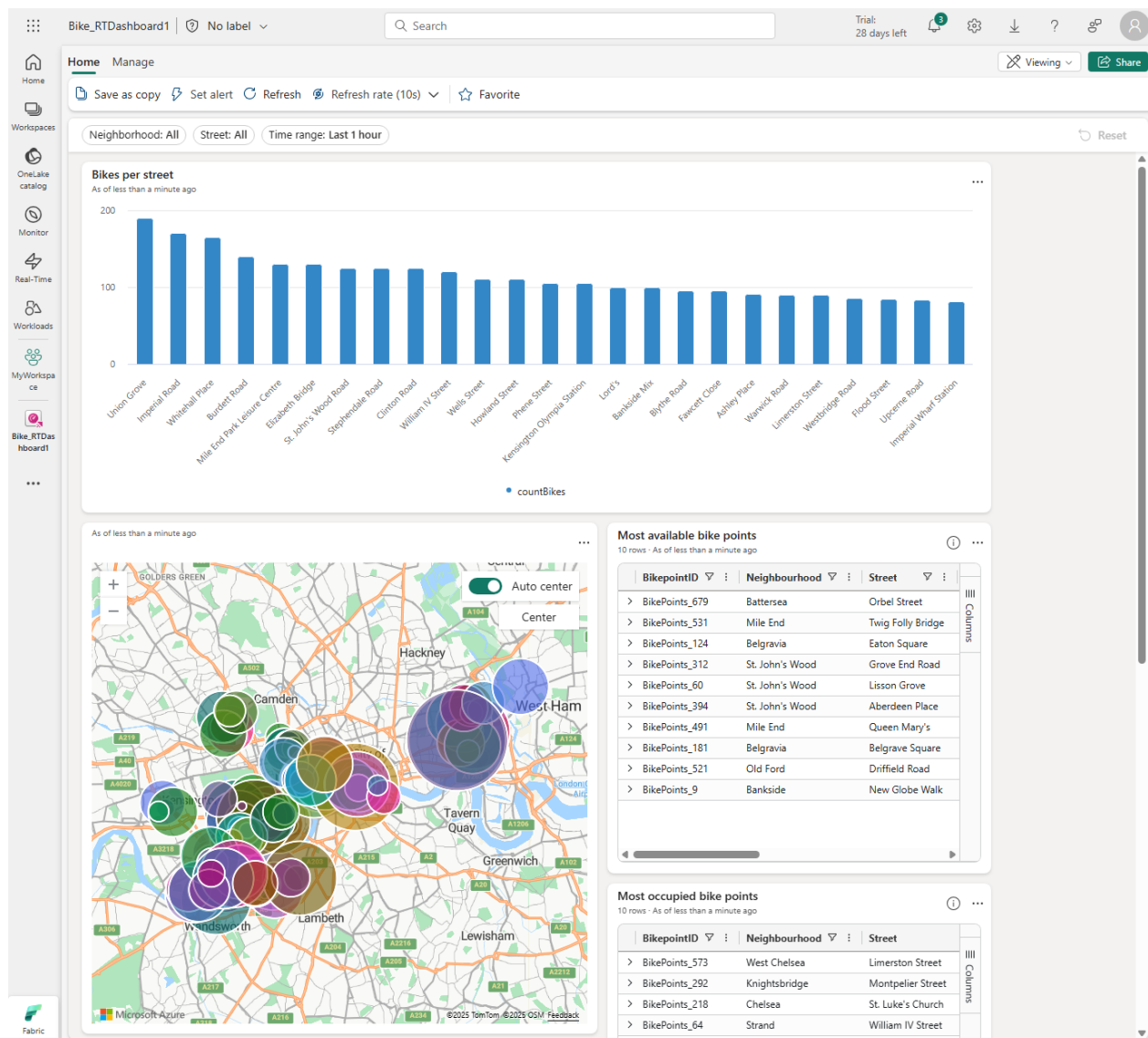
2. Get started with real-time dashboards

<https://learn.microsoft.com/en-us/training/modules/create-real-time-dashboards-microsoft-fabric/2-get-started-with-real-time-dashboards>

Get started with real-time dashboards

Completed

Real-Time Dashboards in Microsoft Fabric are built on real-time streaming data sources. Each dashboard consists of one or more *tiles*, each displaying a real-time data visualization.



Create a Real-Time Dashboard

To create a Real-Time Dashboard, you'll need a source of real-time data; such as a KQL database in an eventhouse. You can then create a Real-Time Dashboard with a *data source* that references the KQL database.

Configure authorization for data sources

When connecting the dashboard to the data source, you can specify who has permission to access the data:

- **Pass-through identity** - Each person viewing the dashboard accesses data using their own permissions.

- **Dashboard editor's identity** - Everyone viewing the dashboard accesses data using the dashboard creator's permissions.

Create tiles

A dashboard contains at least one tile, in which the results of a KQL query are displayed.

Specify a query

When you first add a tile, you can enter and test the query you want to use to query the underlying data source. For example, you might use a KQL query similar to the following example to query a table named **bikes** and retrieve details of available bikes and empty bike parking docks for a bike rental system in a city:

```
bikes
| where ingestion_time() between (ago(30min) .. now())
| summarize latest_observation = arg_max(ingestion_time(), *) by Neighbourhood
| project Neighbourhood, latest_observation, No_Bikes, No_Empty_Docks
| order by Neighbourhood asc
```

The query in the example retrieves the most recent observation (the latest record ingested into the source table) within the last 30 minutes for each neighborhood.

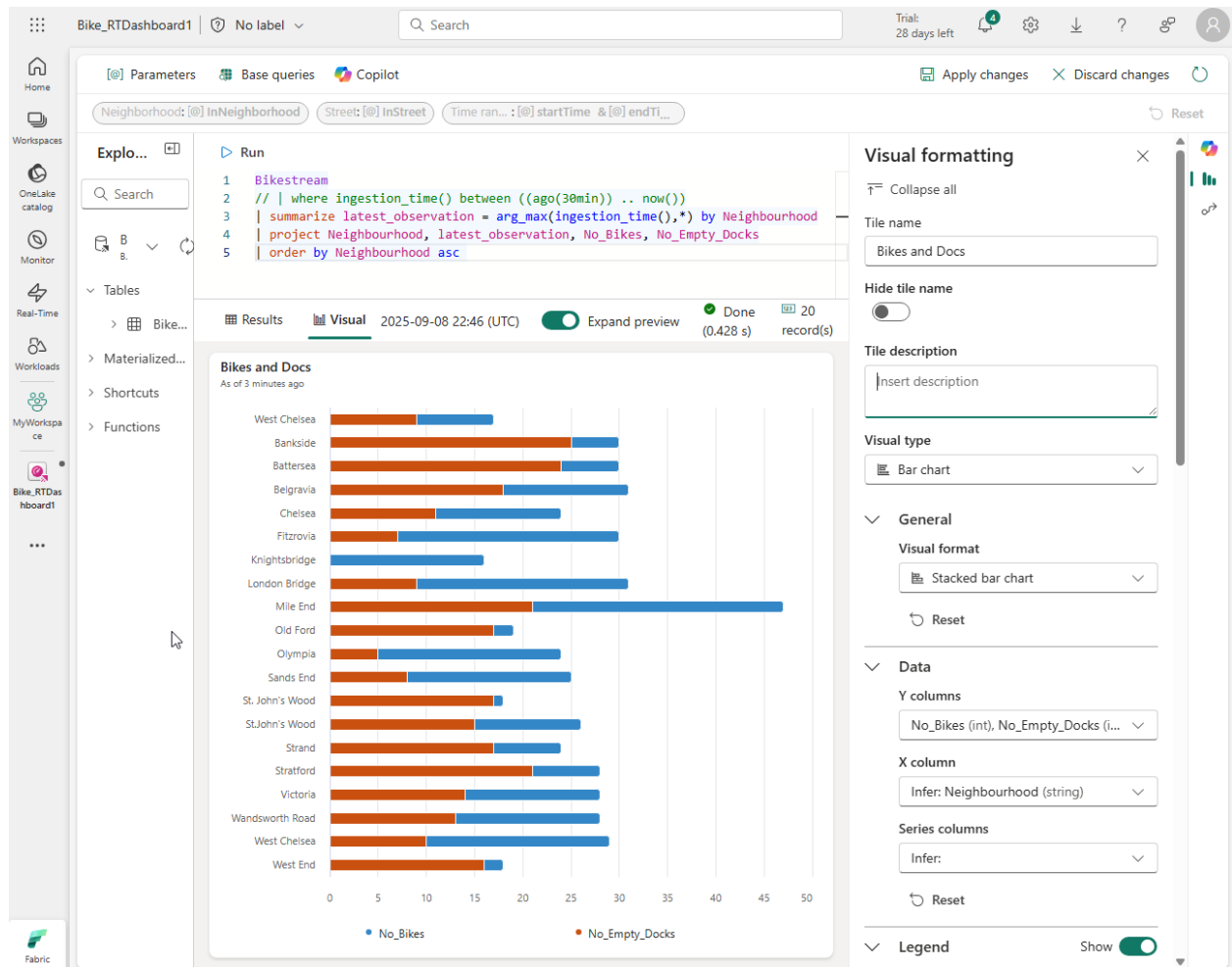
Initially, the tile displays the results of the query as a table as shown in the following image.

The screenshot shows the Microsoft Fabric dashboard editor interface. The top navigation bar includes 'Home', 'Manage', and a search bar. The main workspace displays a 'New tile' with a table of bike counts by street. The table has two columns: 'Street' and 'countBikes'. The data is sorted by 'countBikes' in descending order. The table shows 25 rows of data, with the first 10 rows visible. The interface also includes a sidebar with navigation options like 'Home', 'Workspaces', 'OneLake catalog', 'Monitor', 'Real-Time', 'Workloads', 'MyWorkspace', and 'Bike_RTDashboard1'.

Street	countBikes
> Union Grove	532
> Imperial Road	476
> Whitehall Place	458
> Burdett Road	408
> Elizabeth Bridge	373
> Mile End Park Leisure Centre	364
> Stephendale Road	350
> Clinton Road	350
> Bankside Mix	346
> William IV Street	344
> Wells Street	325
> St. John's Wood Road	325
> Howland Street	319

Visualize the data

After creating a tile, you can edit it to define a visual in which the data is represented as a chart, map, or other data visualization. For example, it might make more sense to display the rental bike data as a bar chart that shows the number of bikes and empty parking docks in each neighborhood.



You can add multiple tiles to a dashboard and arrange them to organize the way the data is visualized. You can also add *text tiles* to provide additional information.

3. Organize and filter dashboard data

<https://learn.microsoft.com/en-us/training/modules/create-real-time-dashboards-microsoft-fabric/3-advanced-features>

Organize and filter dashboard data

Completed

Real-Time Dashboards include several features that help you organize and interact with your data more effectively. You can organize your dashboard using multiple pages, add interactive parameters to filter data, and set up automatic data refresh.

Pages

By default, a Real-Time Dashboard has one page. However, you can add more pages as containers for more tiles. This helps you organize related content into logical groups. For example, you might create separate pages for different data sources, or subject areas.

Parameters

Parameters add flexibility to your dashboard by allowing users to filter the data displayed in tiles. For example, users might want to view data for only a specific time period or focus on a particular subset of the data.

You can create multiple parameters for a dashboard. Parameter values can be based on specific text or on the results of a query. Each parameter has a variable name that you can reference in your tile queries using an underscore prefix.

For example, in a bike rental dashboard, if you created a parameter that lets users filter by neighborhood and named it `_selected_neighbourhoods`, you would reference it in your tile queries like this:

```
bikes
| where ingestion_time() between (ago(30min) .. now())
    and (isempty(_selected_neighbourhoods) or Neighbourhood in (_selected_neighbourhoods))
| summarize latest_observation = arg_max(ingestion_time(), *) by Neighbourhood
```

This query includes a filter that checks whether the `_selected_neighbourhoods` parameter is empty (which happens when no neighborhoods are selected, showing all neighborhoods) or contains specific values (showing only those neighborhoods).

Auto refresh

Auto refresh automatically updates your dashboard data without needing to manually reload the page. Dashboard editors can set a default refresh rate and viewers can adjust this rate during their session. However, editors can also set a minimum refresh rate to manage system performance and prevent users from refreshing too frequently.

4. Dashboard management and optimization

Dashboard management and optimization

Completed

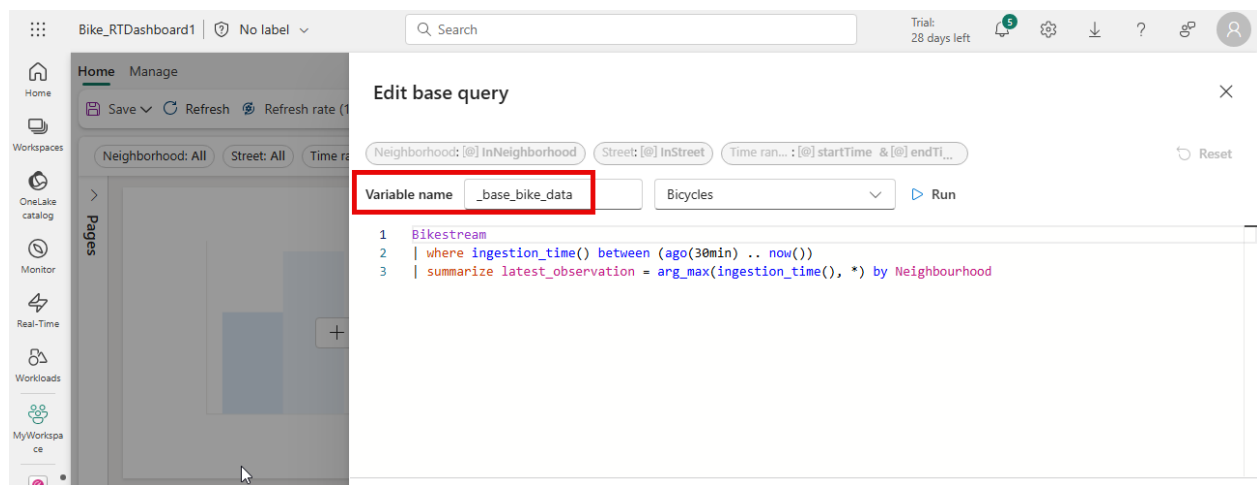
When multiple tiles in a dashboard use related data, you can improve maintainability by defining *base queries* that retrieve a general set of records relevant for multiple tiles. Tile-specific queries then filter or group this data for particular visualizations.

Base queries help you avoid duplicating the same logic across multiple tiles. Instead of writing similar queries for each tile, you create one base query and reference it from multiple tiles.

For example, in a bike rental dashboard, you could create a base query through the dashboard's **Base queries** menu. When creating the base query, you would:

1. Assign it the variable name `_base_bike_data`
2. Define a query that returns all data fields in each neighborhood within the last 30 minutes:

```
bikes
| where ingestion_time() between (ago(30min) .. now())
| summarize latest_observation = arg_max(ingestion_time(), *) by Neighbourhood
```



Then, you can create individual tile queries that reference this base query by using its variable name:

```
_base_bike_data
| project Neighbourhood, latest_observation, No_Bikes, No_Empty_Docks
| order by Neighbourhood asc
```

5. Exercise - Get started with real-time dashboards

<https://learn.microsoft.com/en-us/training/modules/create-real-time-dashboards-microsoft-fabric/5-exercise>

Exercise - Get started with real-time dashboards

Completed

Now it's your opportunity to create a Real-Time Dashboard using bike-sharing data from an eventstream. In this exercise, you build visualizations, configure parameters, and set up auto refresh functionality.

This exercise takes approximately **30** minutes to complete.

Important

You need a **Microsoft Fabric tenant** to complete this exercise. See [Getting started with Fabric](#) for details.

Launch the exercise and follow the instructions.

[Launch Exercise](#)

6. Module assessment

<https://learn.microsoft.com/en-us/training/modules/create-real-time-dashboards-microsoft-fabric/6-knowledge-check>

Module assessment

Completed

7. Summary

<https://learn.microsoft.com/en-us/training/modules/create-real-time-dashboards-microsoft-fabric/7-summary>

Summary

Completed

In this module, you learned how to create Real-Time Dashboards in Microsoft Fabric and connect them to data sources. You also learned to add and configure tiles to visualize real-time data using KQL queries and organize dashboard content using multiple pages for logical grouping.

You also discovered how to add interactive parameters to dynamically filter dashboard data and configure auto refresh to keep data current automatically. Finally, you learned to use base queries to improve maintainability when multiple tiles share related data. Real-Time Dashboards are a key component of Microsoft Fabric's Real-Time Intelligence capabilities, enabling you to provide users with current data visualizations that support time-sensitive decision making as events happen.

Learn more

- [Create a Real-Time Dashboard](#)
- [Add parameters to a Real-Time Dashboard](#)