

[Before you start](#)

[Provision Azure resources](#)

[Create a Power BI workspace](#)

[Use Azure Stream Analytics to process streaming data](#)

[Visualize the streaming data in Power BI](#)

[Delete resources](#)

Create a realtime report with Azure Stream Analytics and Microsoft Power BI

Data analytics solutions often include a requirement to ingest and process *streams* of data. Stream processing differs from batch processing in that streams are generally *boundless* - in other words they are continuous sources of data that must be processed perpetually rather than at fixed intervals.

Azure Stream Analytics provides a cloud service that you can use to define a *query* that operates on a stream of data from a streaming source, such as Azure Event Hubs or an Azure IoT Hub. You can use an Azure Stream Analytics query to process a stream of data and send the results directly to Microsoft Power BI for realtime visualization.

In this exercise, you'll use Azure Stream Analytics to process a stream of sales order data, such as might be generated from an online retail application. The order data will be sent to Azure Event Hubs, from where your Azure Stream Analytics job will read and summarize the data before sending it to Power BI, where you will visualize the data in a report.

This exercise should take approximately **45** minutes to complete.

Before you start

You'll need an [Azure subscription](#) in which you have administrative-level access.

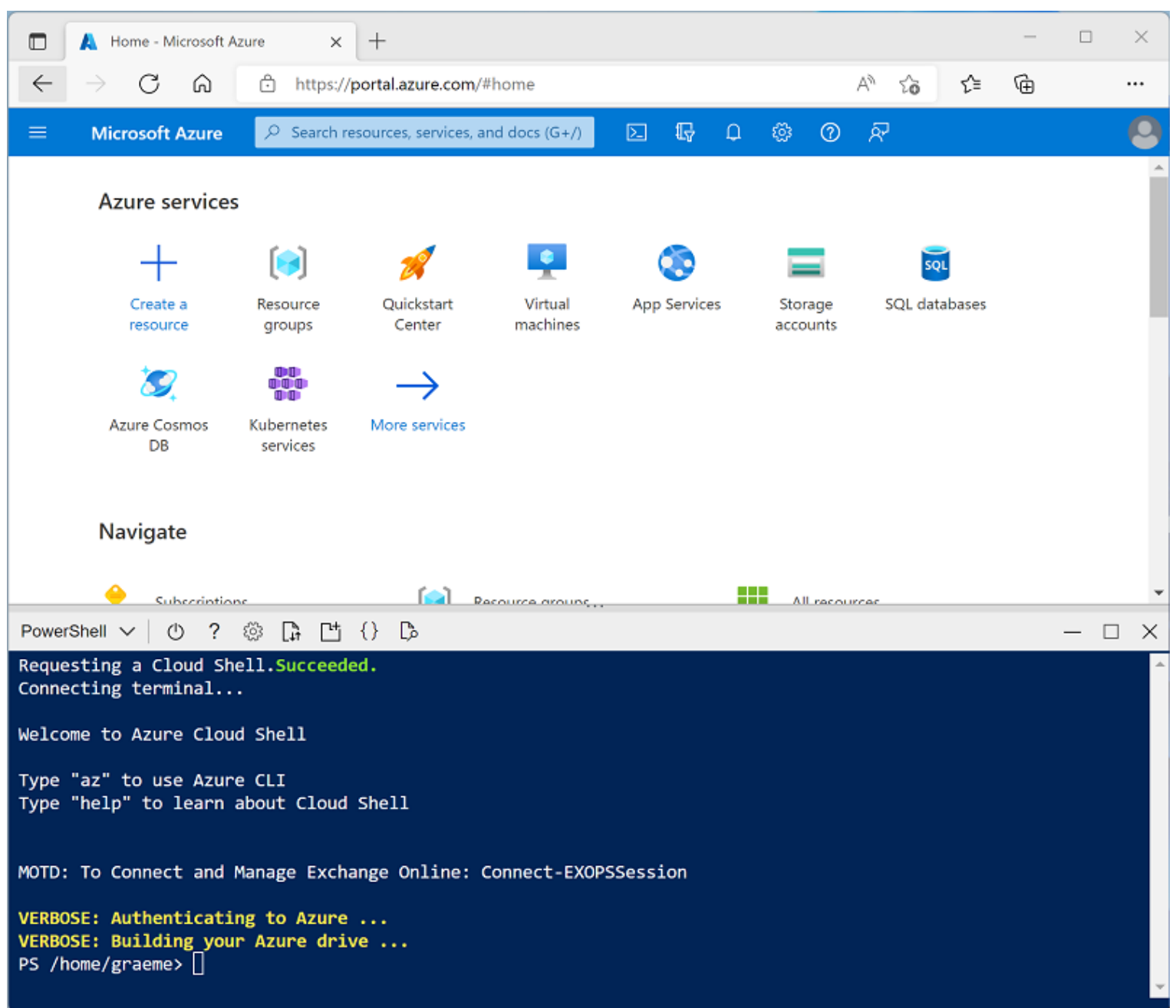
You'll also need access to the Microsoft Power BI service. Your school or organization may already provide this, or you can [sign up for the Power BI service as an individual](#).

Provision Azure resources

In this exercise, you'll need an Azure Synapse Analytics workspace with access to data lake storage and a dedicated SQL pool. You'll also need an Azure Event Hubs namespace to which the streaming order data can be sent.

You'll use a combination of a PowerShell script and an ARM template to provision these resources.

1. Sign into the [Azure portal](#) at `https://portal.azure.com`.
2. Use the [**>**] button to the right of the search bar at the top of the page to create a new Cloud Shell in the Azure portal, selecting a **PowerShell** environment and creating storage if prompted. The cloud shell provides a command line interface in a pane at the bottom of the Azure portal, as shown here:



! **Note:** If you have previously created a cloud shell that uses a *Bash* environment, use the the drop-down menu at the top left of the cloud shell pane to change it to **PowerShell**.

- Note that you can resize the cloud shell by dragging the separator bar at the top of the pane, or by using the —, □, and X icons at the top right of the pane to minimize, maximize, and close the pane. For more information about using the Azure Cloud Shell, see the [Azure Cloud Shell documentation](#).
- In the PowerShell pane, enter the following commands to clone the repo containing this exercise:

Code	Copy
<pre>rm -r dp-203 -f git clone https://github.com/MicrosoftLearning/dp-203-azure-data-engineer dp-203</pre>	

- After the repo has been cloned, enter the following commands to change to the folder for this exercise and run the **setup.ps1** script it contains:

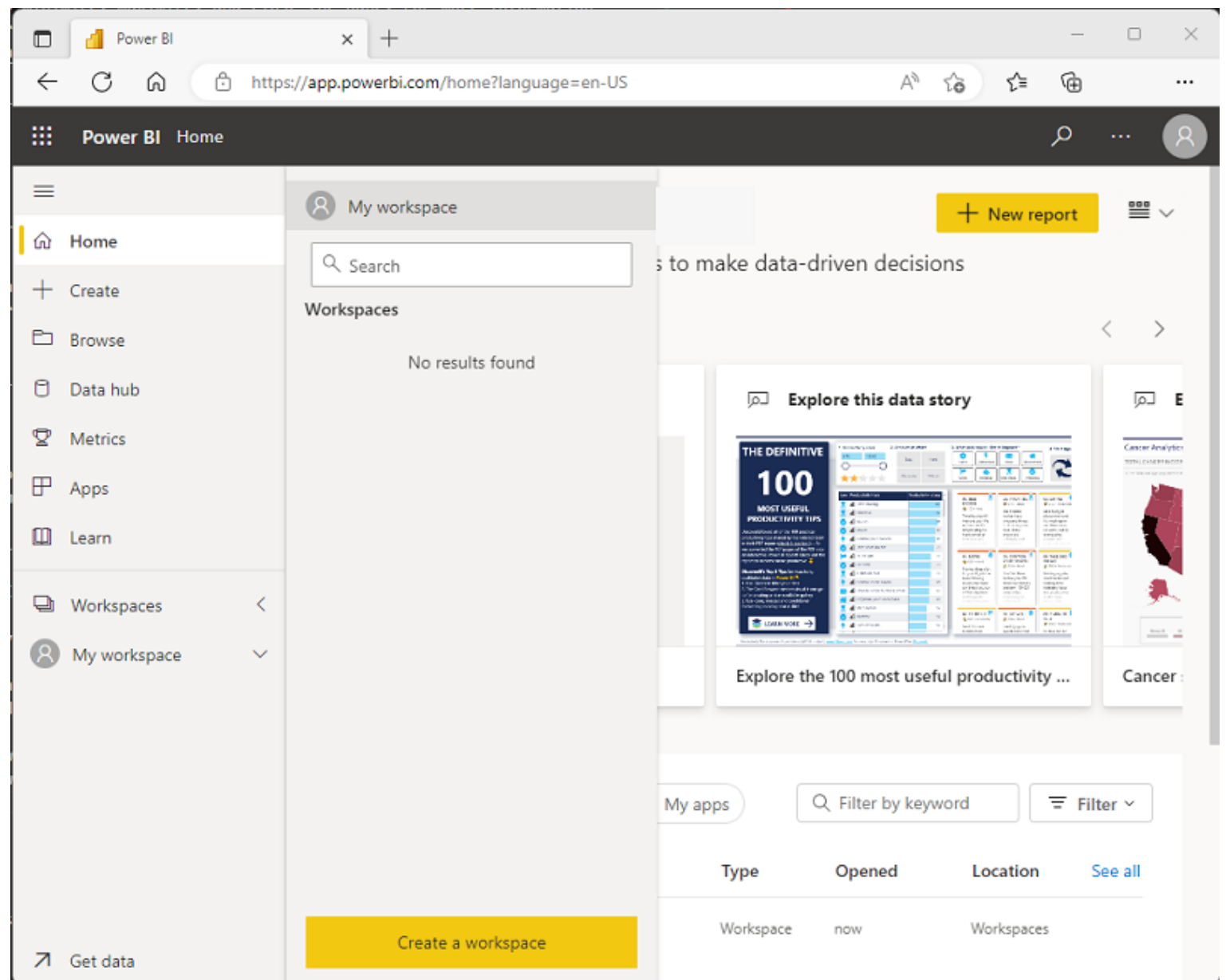
Code	Copy
<pre>cd dp-203/Allfiles/labs/19 ./setup.ps1</pre>	

- If prompted, choose which subscription you want to use (this will only happen if you have access to multiple Azure subscriptions).
- While you are waiting for the script to complete, continue with the next task.

Create a Power BI workspace

In the Power BI service, you organize datasets, reports, and other resources in *workspaces*. Every Power BI user has a default workspace named **My Workspace**, which you can use in this exercise; but it's generally good practice to create a workspace for each discrete reporting solution you want to manage.

1. Sign into the Power BI service at <https://app.powerbi.com/> using your Power BI service credentials.
2. On the **Workspaces** page, select **Create a workspace**:



! **Note:** If you are using a trial account, you may need to enable additional trial features.

3. Create a new workspace with a meaningful name (for example, *mslearn-streaming*).
4. When viewing your workspace, note its globally unique identifier (GUID) in the page URL (which should be similar to <https://app.powerbi.com/groups/<GUID>/list>). You will need this GUID later.

Use Azure Stream Analytics to process streaming data

An Azure Stream Analytics job defines a perpetual query that operates on streaming data from one or more inputs and sends the results to one or more outputs.

Create a Stream Analytics job

1. Switch back to the browser tab containing the Azure portal, and when the script has finished, note the region where your **db000-xxxxxxx** resource group was provisioned.
2. On the **Home** page of the Azure portal, select **+ Create a resource** and search for [Stream Analytics job](#). Then create a **Stream Analytics job** with the following properties:

- **Subscription:** Your Azure subscription
- **Resource group:** Select the existing **dp203-xxxxxxx** resource group.
- **Name:** [stream-orders](#)
- **Region:** Select the region where your Synapse Analytics workspace is provisioned.
- **Hosting environment:** Cloud
- **Streaming units:** 1

3. Wait for deployment to complete and then go to the deployed Stream Analytics job resource.

Create an input for the event data stream

- On the **stream-orders** overview page, select **Add input**. Then on the **Inputs** page, use the **Add stream input** menu to add an **Event Hub** input with the following properties:
 - Input alias:** `orders`
 - Select Event Hub from your subscriptions:** Selected
 - Subscription:** Your Azure subscription
 - Event Hub namespace:** Select the **eventsxxxxxxx** Event Hubs namespace
 - Event Hub name:** Select the existing **eventhubxxxxxxx** event hub.
 - Event Hub consumer group:** Select the existing **\$Default** consumer group
 - Authentication mode:** Create system assigned managed identity
 - Partition key:** *Leave blank*
 - Event serialization format:** JSON
 - Encoding:** UTF-8
- Save the input and wait while it is created. You will see several notifications. Wait for a **Successful connection test** notification.

Create an output for the Power BI workspace

- View the **Outputs** page for the **stream-orders** Stream Analytics job. Then use the **Add** menu to add an **Power BI** output with the following properties:
 - Output alias:** `powerbi-dataset`
 - Select Power BI settings manually:** Selected
 - Group workspace:** *The GUID for your workspace*
 - Authentication mode:** *Select **User token** and then use the **Authorize** button at the bottom to sign into your Power BI account*
 - Dataset name:** `realtime-data`
 - Table name:** `orders`
- Save the output and wait while it is created. You will see several notifications. Wait for a **Successful connection test** notification.

Create a query to summarize the event stream

- View the **Query** page for the **stream-orders** Stream Analytics job.
- Modify the default query as follows:

Code Copy

```
SELECT
    DateAdd(second,-5,System.Timestamp) AS StartTime,
    System.Timestamp AS EndTime,
    ProductID,
    SUM(Quantity) AS Orders
INTO
    [powerbi-dataset]
FROM
    [orders] TIMESTAMP BY EventEnqueuedUtcTime
GROUP BY ProductID, TumblingWindow(second, 5)
HAVING COUNT(*) > 1
```

Observe that this query uses the **System.Timestamp** (based on the **EventEnqueuedUtcTime** field) to define the start and end of each 5 second *tumbling* (non-overlapping sequential) window in which the total quantity for each product ID is calculated.

3. Save the query.

Run the streaming job to process order data

1. View the **Overview** page for the **stream-orders** Stream Analytics job, and on the **Properties** tab review the **Inputs**, **Query**, **Outputs**, and **Functions** for the job. If the number of **Inputs** and **Outputs** is 0, use the **Refresh** button on the **Overview** page to display the **orders** input and **powerbi-dataset** output.
2. Select the **Start** button, and start the streaming job now. Wait until you are notified that the streaming job started successfully.
3. Re-open the cloud shell pane and run the following command to submit 100 orders.

Code

node ~/dp-203/Allfiles/labs/19/orderclient

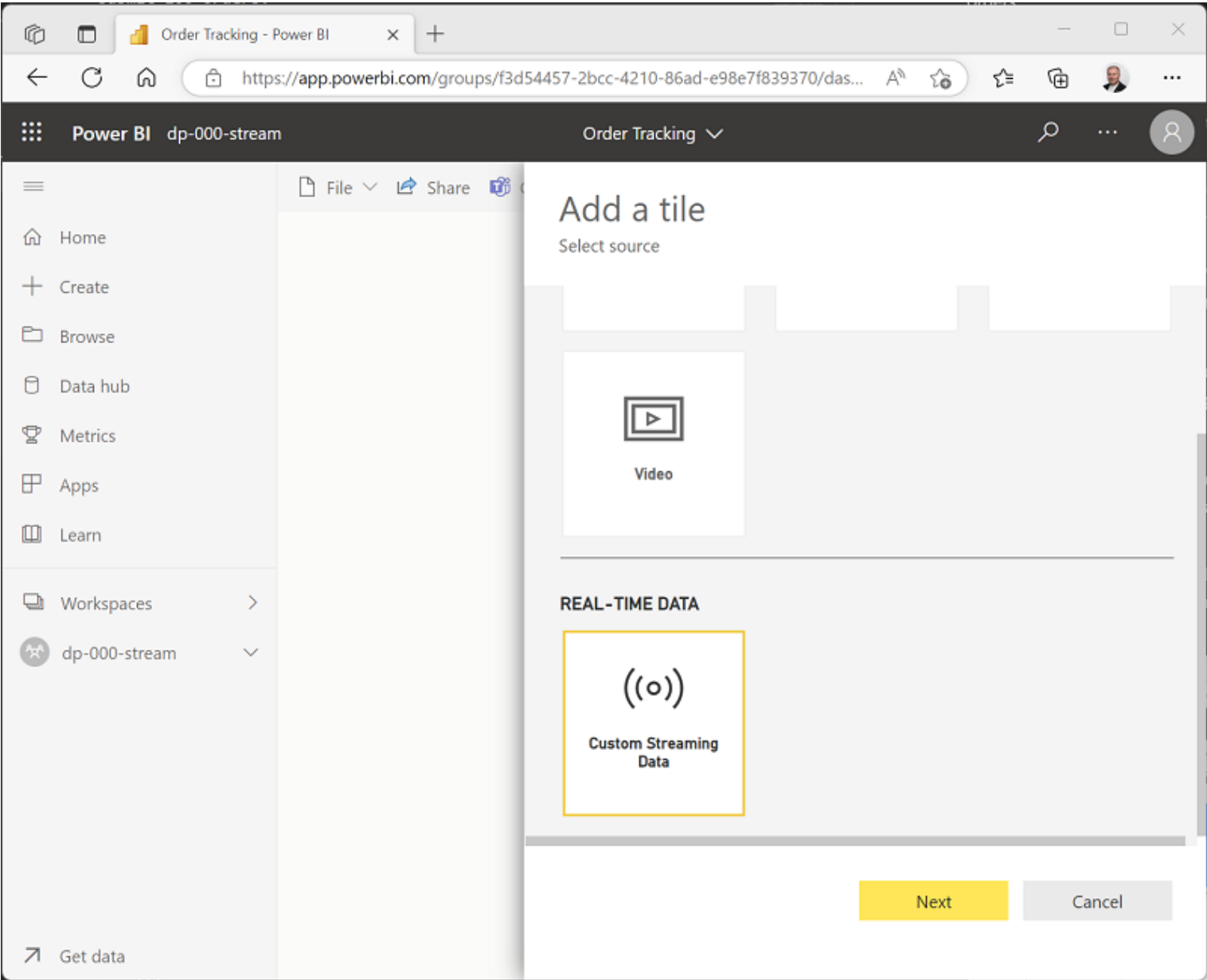
Copy

4. While the order client app is running, switch to the Power BI app browser tab and view your workspace.
5. Refresh the Power BI app page until you see the **realtime-data** dataset in your workspace. This dataset is generated by the Azure Stream Analytics job.

Visualize the streaming data in Power BI

Now that you have a dataset for the streaming order data, you can create a Power BI dashboard that represents it visually.

1. In the **+ New** drop-down menu for your workspace, select **Dashboard**, and create a new dashboard named **Order Tracking**.
2. In the **Edit** menu for the **Order Tracking** dashboard, select **Add a tile**. Then in the **Add a tile** pane, select **Custom Streaming Data** and click **Next**:



3. In the **Add a custom streaming data tile** pane, under **Your datasets**, select the **realtime-data** dataset, and click **Next**.
4. Change the default visualization type to **Line chart**. Then set the following properties and click **Next**:
 - **Axis**: EndTime
 - **Value**: Orders
 - **Time window to display**: 1 Minute

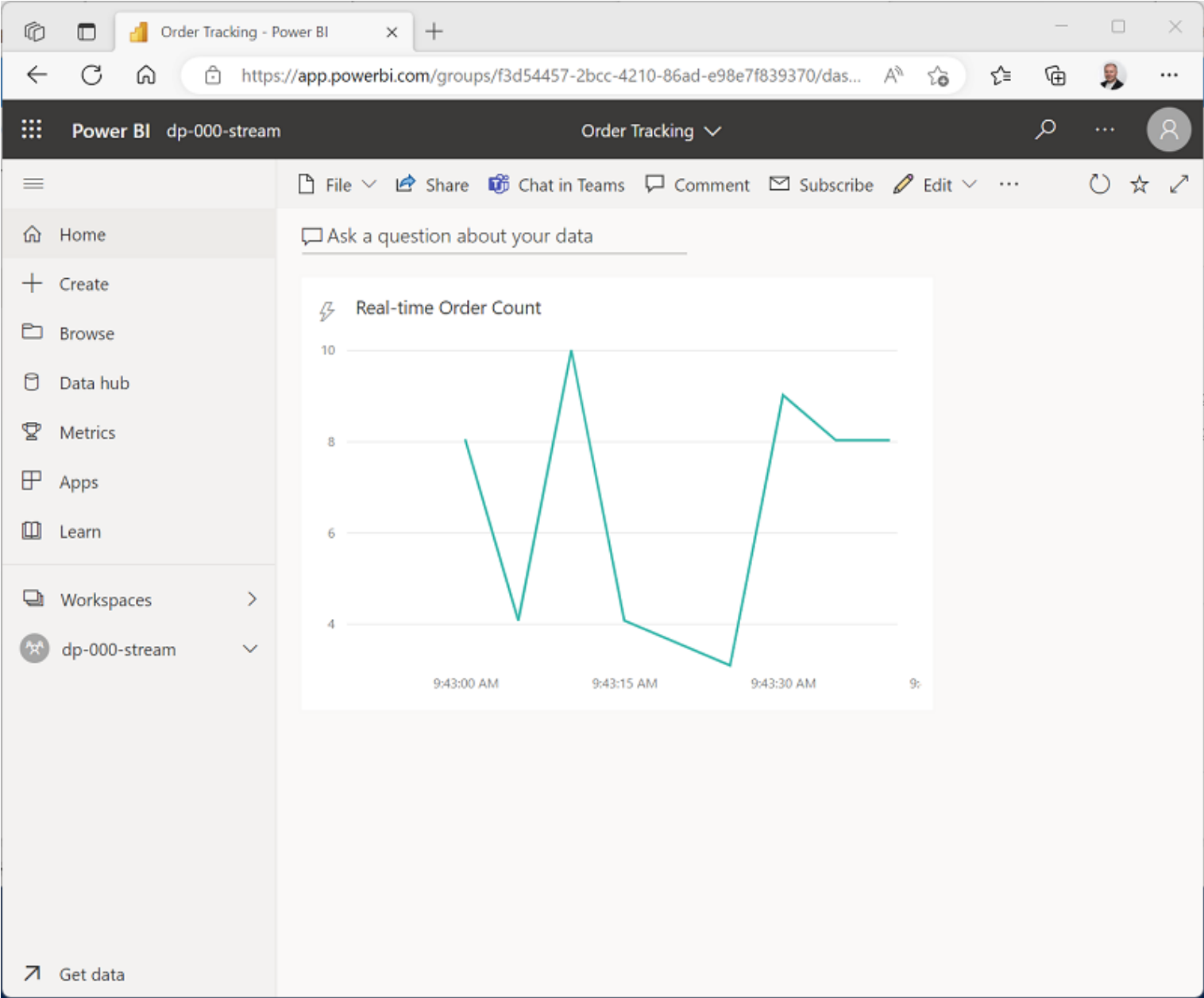
5. On the **Tile details** pane, set the **Title** to **Real-time Order Count** and click **Apply**.
6. Switch back to the browser tab containing the Azure portal, and if necessary, re-open the cloud shell pane. Then re-run the following command to submit another 100 orders.

Code

Copy

```
node ~/dp-203/Allfiles/labs/19/orderclient
```

7. While the order submission script is running, switch back to the browser tab containing the **Order Tracking** Power BI dashboard and observe that the visualization updates to reflect the new order data as it is processed by the Stream Analytics job (which should still be running).



You can re-run the **orderclient** script and observe the data being captured in the real-time dashboard.

Delete resources

If you've finished exploring Azure Stream Analytics and Power BI, you should delete the resources you've created to avoid unnecessary Azure costs.

1. Close the browser tab containing the Power BI report. Then in the **Workspaces** pane, in the **:** menu for your workspace select **Workspace settings** and delete the workspace.
2. Return to the browser tab containing the Azure Portal, close the cloud shell pane, and use the **Stop** button to stop the Stream Analytics job. Wait for the notification that the Stream Analytics job has stopped successfully.
3. On the Azure portal, on the **Home** page, select **Resource groups**.
4. Select the **dp203-xxxxxxx** resource group containing your Azure Event Hub and Stream Analytics resources.
5. At the top of the **Overview** page for your resource group, select **Delete resource group**.
6. Enter the **dp203-xxxxxxx** resource group name to confirm you want to delete it, and select **Delete**.

After a few minutes, the resources created in this exercise will be deleted.

