IoT Enablement Workshop

`Internet of Things

Hands-on Partner Lab

October 2018 v0.01

Information in this document, including URL and other Internet Web site references, is subject to change without notice. Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, e-mail address, logo, person, place or event is intended or should be inferred. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

The names of manufacturers, products, or URLs are provided for informational purposes only and Microsoft makes no representations and warranties, either expressed, implied, or statutory, regarding these manufacturers or the use of the products with any Microsoft technologies. The inclusion of a manufacturer or product does not imply endorsement of Microsoft of the manufacturer or product. Links may be provided to third party sites. Such sites are not under the control of Microsoft and Microsoft is not responsible for the contents of any linked site or any link contained in a linked site, or any changes or updates to such sites. Microsoft is not responsible for webcasting or any other form of transmission received from any linked site. Microsoft is providing these links to you only as a convenience, and the inclusion of any link does not imply endorsement of Microsoft of the site or the products contained therein.

© 2017 Microsoft Corporation. All rights reserved.

Microsoft and the trademarks listed at https://www.microsoft.com/en-us/legal/intellectualproperty/Trademarks/Usage/General.aspx are trademarks of the Microsoft group of companies. All other trademarks are property of their respective owners.

Contents

Internet of Things Enablement Workshop 1

Abstract and learning objectives 1

Overview ..... **Error! Bookmark not defined.**

Requirements 1

Before the hands-on lab 2

Task 1: Provision Power BI 2

Lab 1 – Hot Path 3

Summary and Objectives 3

Exercise 1 – Provisioning IoT Hub 4

Create Device in IoT Hub 7

Connect Simulated Device 8

Exercise 2 – Create and Deploy a Stream Analytics Job 10

Exercise 3 – Visualisation in PowerBI 17

Lab 2 – Warm Path 36

Summary and Objectives 36

Exercise 1 – Provisioning IoT Hub 37

Create Device in IoT Hub 40

Connect Simulated Device 41

Exercise 2 - Create a Cosmos DB Instance 43

Exercise 3 - Create Stream Analytics Job 44

Add IoT Hub as input to Stream Analytics 45

Output Stream Analytics into Cosmos DB 46

Updating the streaming jobs query 47

Confirm Data in Cosmos DB 48

Exercise 4 – Visualisation in PowerBI 49

Create Custom Columns 53

Build your report 55

Publish and share your report 57

Lab 3 – Cold Path 59

Summary and Objectives 59

Task 1 – Route messages from IoT Hub to Blob Storage 60

Task 2 – Create Azure Databricks workspace and cluster 64

Task 3 – Create Databricks cluster with PySpark and SparkSql 64
Task 4 – Perform ELT(ETL) Using Spark 67
Task 5 – Connect PowerBI to Azure Databricks to perform BI reporting from transformed data. 70

Post-Lab Follow-up 75

After the hands-on lab 75
Task 1: Delete the resource group 75

Internet of Things Enablement Workshop

Abstract and learning objectives

This Lab is designed to guide you through an implementation of an end-to-end IoT solution using Microsoft services. In this session, you will design a lambda architecture that demonstrates warm path visualization, hot-path analysis, and cold storage. After completing the Lab, you will have a better understanding of the latest Microsoft Azure products, and a practical experience in deploying these services.

Learning objectives:

- Best practices for use of the latest PaaS offerings from Microsoft with an IoT Architecture.
- Visualisation of data through warm and hot paths, leveraging Cosmos DB and DataBricks.
- Detailed understanding of the IoT solutions and services offered through Azure.

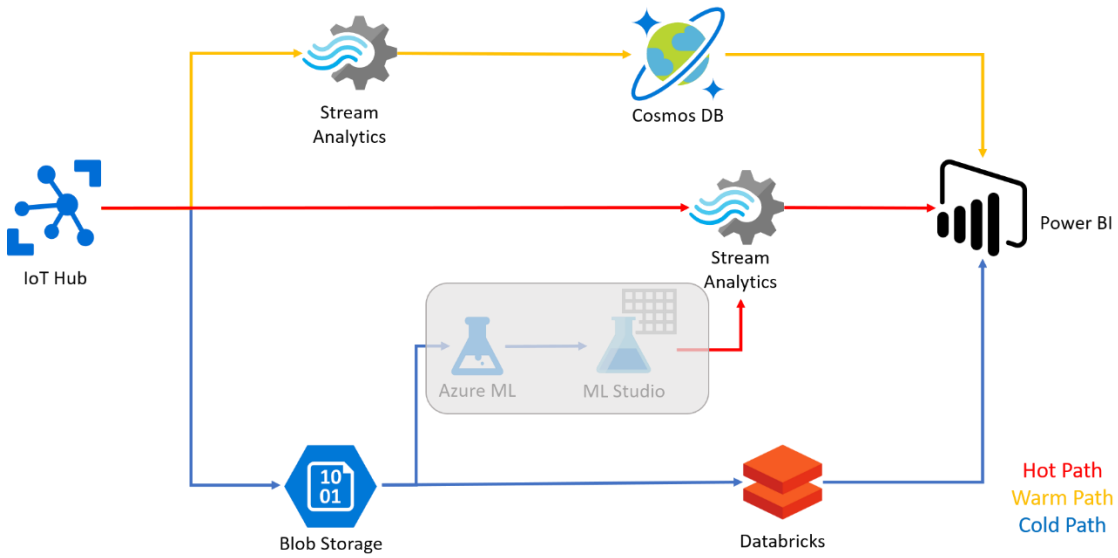


Figure 1 - Full Lab Architecture

Requirements

1. A computer with web browser capabilities and access to the internet
2. Microsoft Azure subscription must be pay-as-you-go or MSDN.
    a. Trial subscriptions will *not* work.
3. Access to PowerBI through an Enterprise or Trial License.
4. PowerBI Desktop downloaded and installed. (https://powerbi.microsoft.com/en-us/desktop/)

 ©2018 Microsoft Corporation

Before the hands-on lab

Duration: 15 minutes

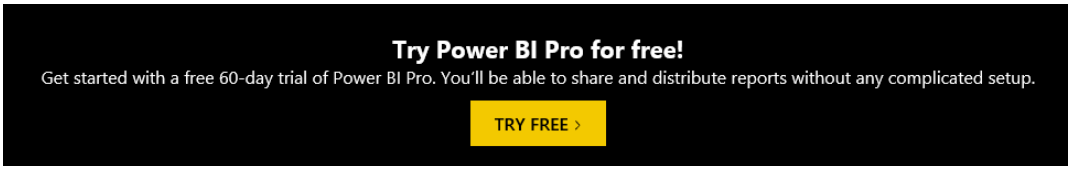
In this exercise, you will set up your environment for use in the rest of the hands-on lab. You should follow all the steps provided in the this section to prepare your environment *before* attending the hands-on lab.

Task 1: Provision Power BI

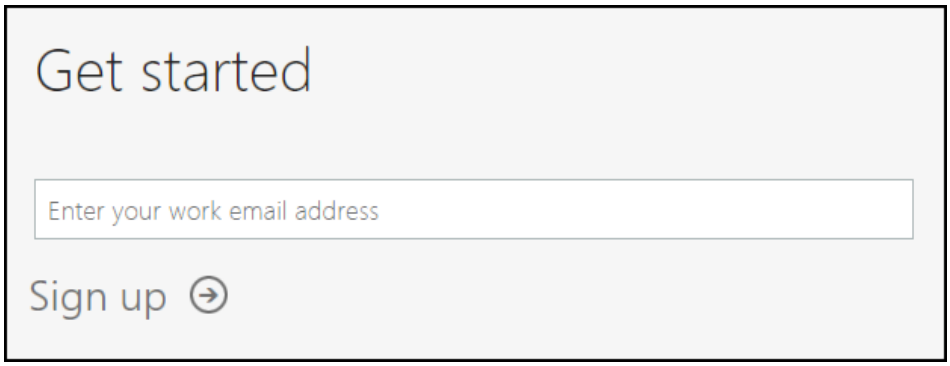
If you do not already have a Power BI account:

1. Go to https://powerbi.microsoft.com/features/.

2. Scroll down until you see the **Try Power BI for free!** section of the page, and click the **Try Free>** button.



3. On the page, enter your work email address (which should be the same account as the one you use for your Azure subscription), and select **Sign up**.



4. Follow the on-screen prompts, and your Power BI environment should be ready within minutes. You can always return to it via https://app.powerbi.com/.

Lab 1 – Hot Path

Summary and Objectives

In this section of the Lab, you will deploy an IoT Hub and analyse incoming data by adding Stream Analytics to perform hot-path analysis on the data stream and pass this to Power BI service for visualization.

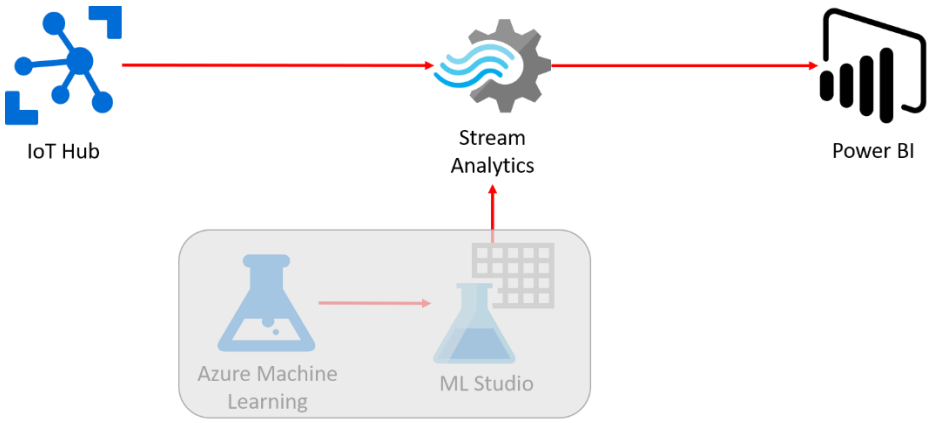


Figure 2 - Hot Path with Machine Learning

- In this Lab we will create a Stream Analytics job to select the sensor data that we want to visualize from the data stream being passed through IoT Hub, and then pass this data to power bi for live reporting.
- Stream Analytics uses a concise but powerful SQL like language to manipulate data in the stream in real-time. It has many functions including those like sliding windows to work with time series data, and allows joins including self-joins that can be used for fraud detection in real-time. The SQL we will use in this task is very simple and will just select data elements in the stream, adding a local timestamp and converting pressure data unit of measure from kilopascals to hectopascals.
- Stream Analytics is capable of processing multiple statements in a single job and sending the results to multiple outputs such as Blob Storage, CosmosDB, SQL Database, Power BI and many other endpoints. You can also have many Jobs reading the same stream of data which provides flexibility and control over processing. In this Lab Stream Analytics passes the real-time data it has selected and manipulated to a Power BI endpoint. The data is stored in a Power BI table which is stored in a dataset.
- We will then use the Power BI service to visualize real-time and aggregate data from the data stream. Power BI service is a SaaS product designed for collaboration and sharing of visual reports. Power BI reports can also be embedded in your own web applications or in Dynamix or Sharepoint online. There is also a free Power BI desktop that can be downloaded which is primarily used for authoring reports. Power BI can also be run on-premise through the Power BI Report Server.
- In the Power BI service we will create a Dashboard with Live Tiles to show real-time data as it is added to the dataset. We will use a variety of visualizations to represent the real-time data. We will also create Power BI reports to show average, minimum and maximum aggregated data from the dataset.
- While we only have data from 1 simulated device we will frequently use Deviceid as a legend in visualizations. In this way you can add devices to the solution and show many device data points in the same visualization.

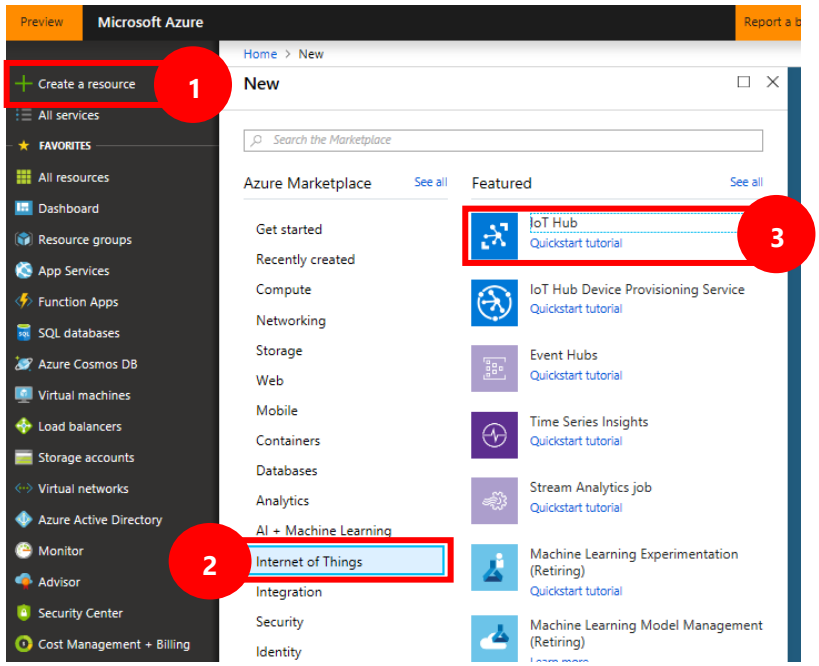
©2018 Microsoft Corporation

Exercise 1 – Provisioning IoT Hub

Duration: 20 minutes

In these steps, you will provision an instance of IoT Hub.

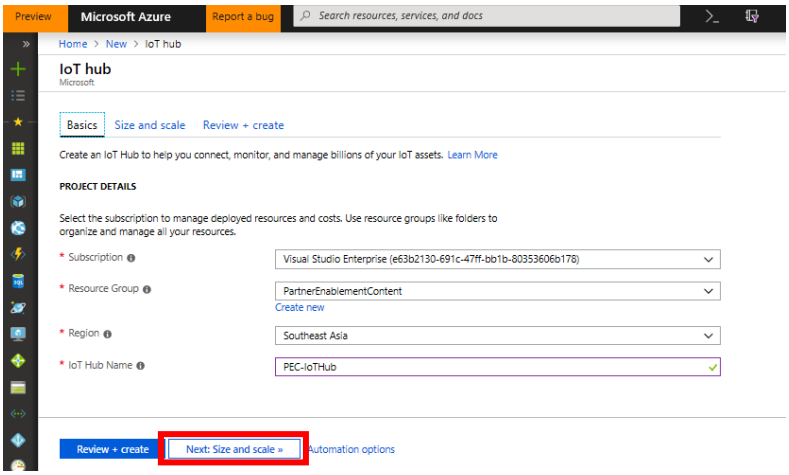
1. In a browser, navigate to the Azure Portal (https://portal.azure.com).
2. Select **+Create a resource**, then select **Internet of Things**, and select **IoT Hub**.



3. In the **IoT Hub** blade, in the **Basics** tab enter the following:
    a. *Subscription*: Select the subscription you'll be using for this lab.
    b. *Resource group*: Select, or create, the resource group you'll use for this lab
    c. *Region*: Select the region to deploy your IoT Hub. Usually this will be the same location as your resource group
    d. *IoT Hub Name*: Provide a name for your new IoT Hub

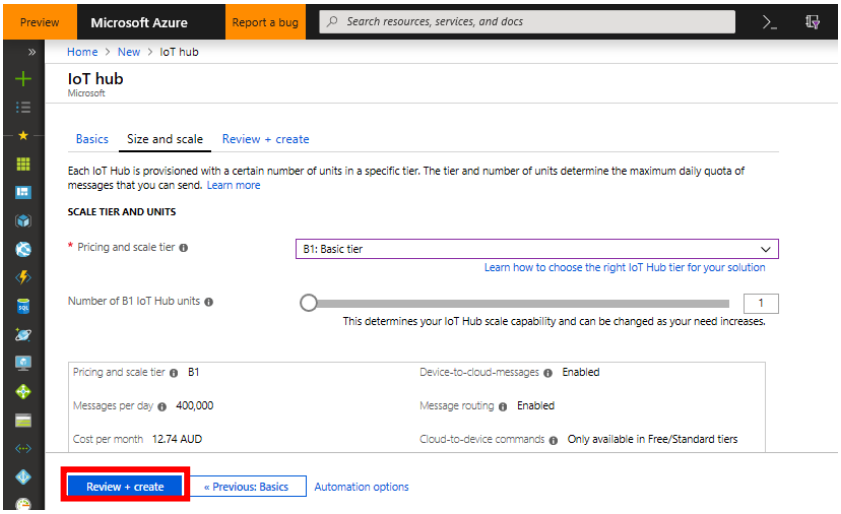
    Once complete, select the "Next: Size and scale" button at the base of the screen, or the "Size and scale" tab at the top

©2018 Microsoft Corporation



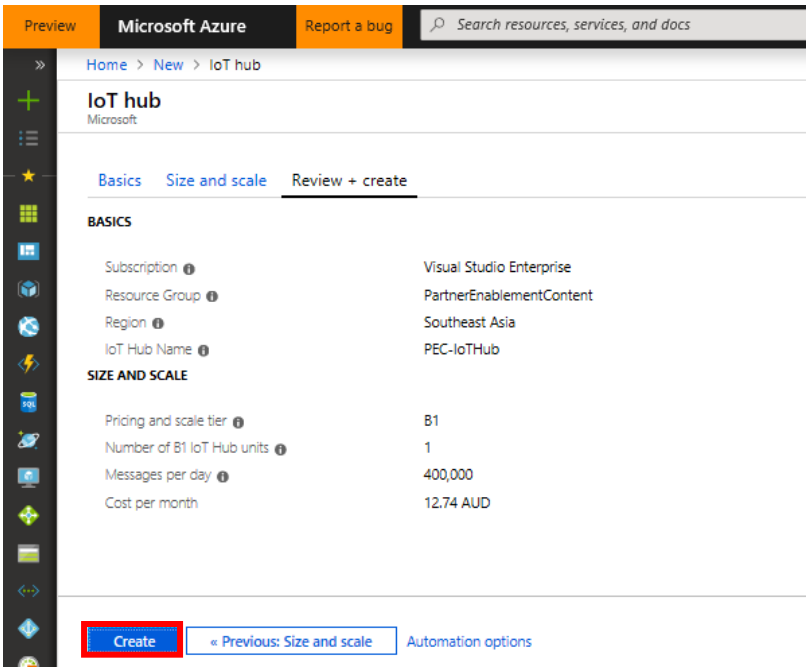
4. In this **Size and scale** screen, select the "Pricing and scale tier" in the dropdown menu
    a. Any tier will work, but for optimising cost choose in the following order:
        i. Free -> Basic -> Standard

After you've selected the desired tier, click the **Review + create** button at the base of the screen.

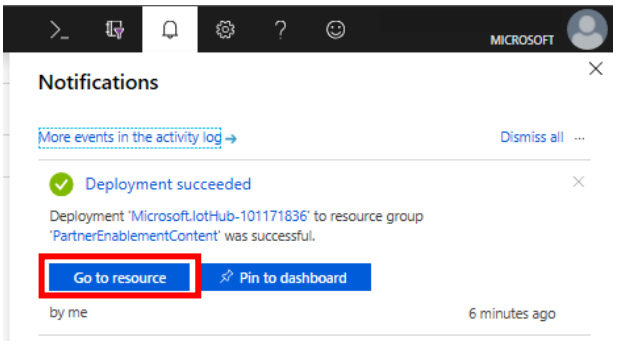


5. A summary screen will now show all the details you selected. If these are correct, select **Create** at the bottom of the screen. This will begin the deployment of your desired IoT Hub configuration.

©2018 Microsoft Corporation



6. When the IoT Hub deployment is completed, you will receive a notification in the Azure portal at the top-right of your screen. Navigate to your new IoT Hub by selecting Go to resource in the notification.

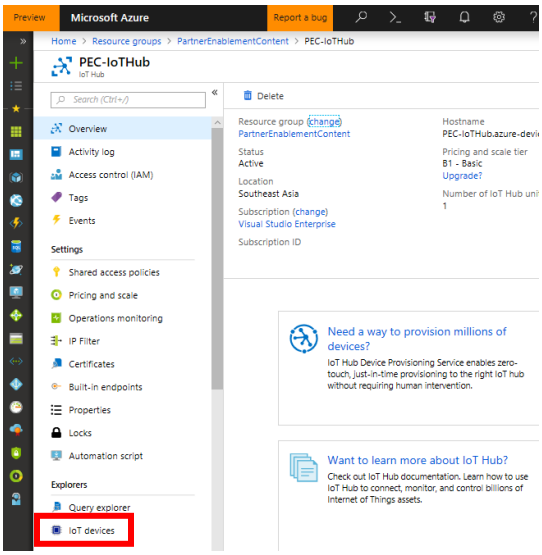


7. Now you have access to all the features and settings for your newly deployed IoT Hub. Let's connect a simulated Raspberry Pi to make sure devices can connect.

©2018 Microsoft Corporation

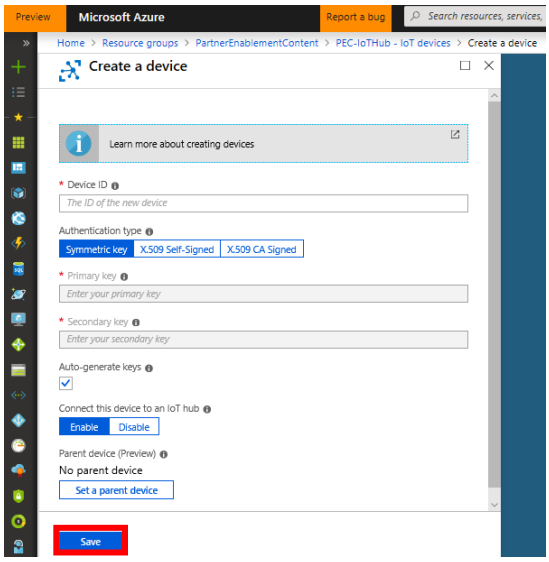
Create Device in IoT Hub

8. From the IoT Hub's Overview blade, select **IoT devices** under Explorers on the left-hand menu.



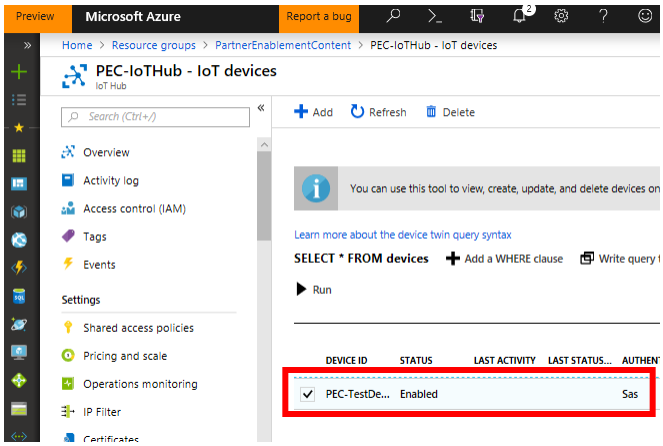
9. Select **+ Add** at the top of the screen and enter the following information on the "Create a device" blade
    a. *Device ID*: A unique ID for the new device
    b. *Authentication Type*: Leave this as Symmetric key
    c. *Auto-generate keys*: Make sure this is selected
    d. *Connect this device to an IoT Hub*: Mak sure this is enabled

    Click **Save** at the base of the screen to confirm the device creation.

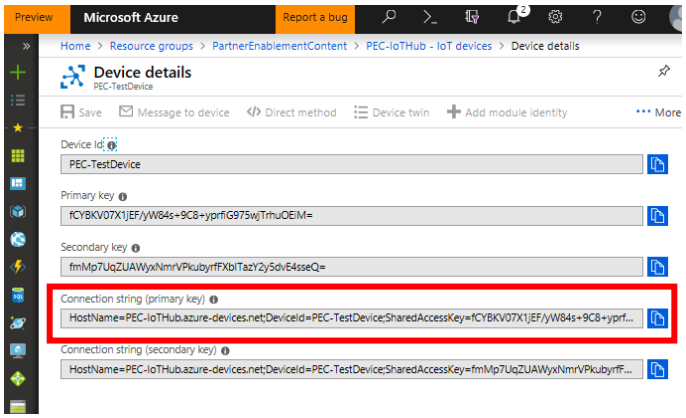


10. Once the device has been created (this may take a few minutes), select it from the list of devices

©2018 Microsoft Corporation



11. Copy the **Connection string (primary key)** and save it in a file. You'll need it for the upcoming section to register your device.



Connect Simulated Device

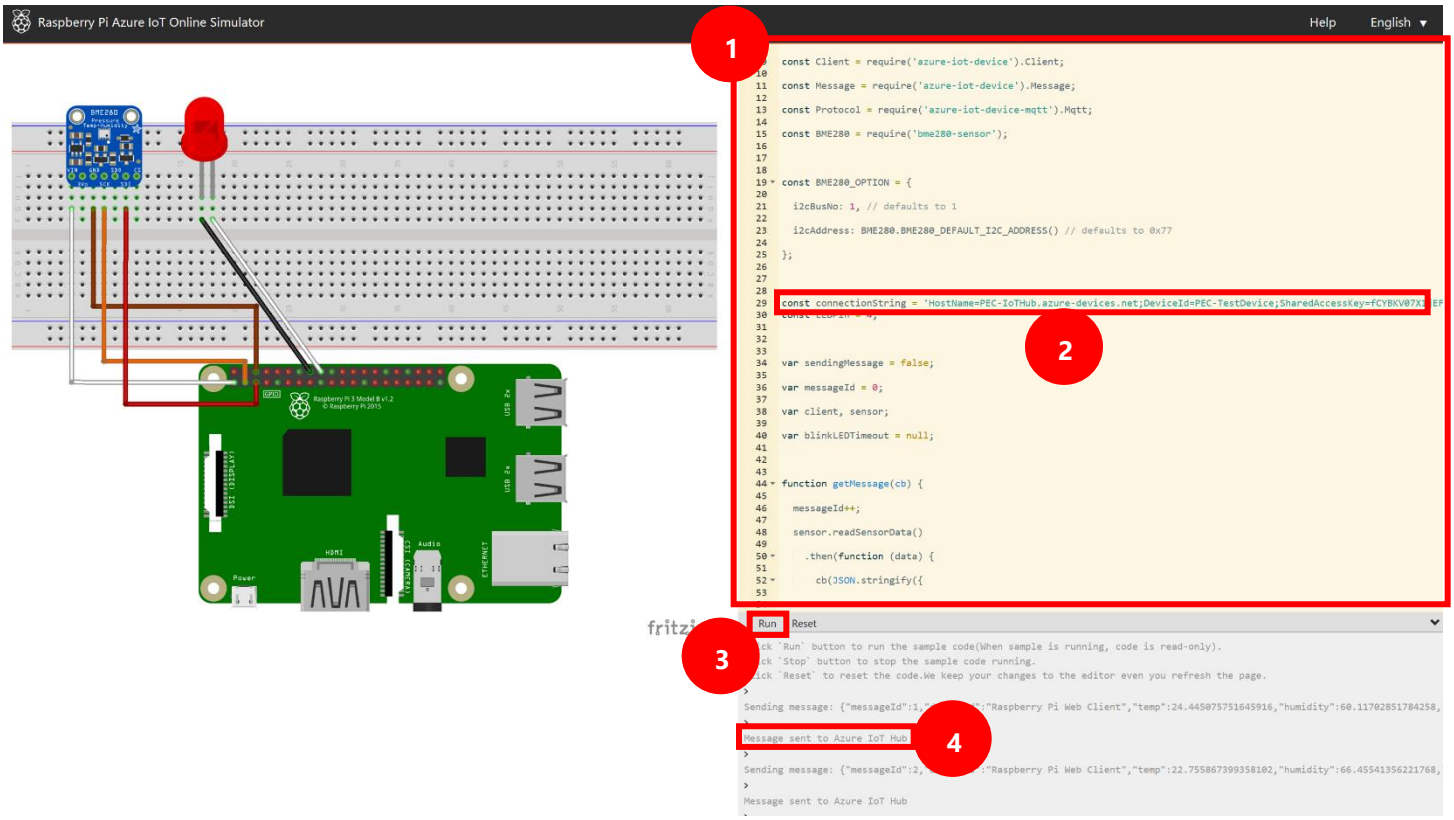
12. You'll be using a simulated Raspberry Pi as the device connecting into IoT Hub. You can access the simulator here: https://azure-samples.github.io/raspberry-pi-web-simulator/#Getstarted. This Raspberry Pi will send several measurements (Temperature, Humidity and Pressure) to the IoT Hub.

1. Copy the code from the file linked here and replace the code in the Raspberry Pi Web Simulator.

13. You'll now need to add in the Device Connection String copied earlier. Paste that string into the code by replacing the *[Your IoT hub device connection string]* text.

14. Click **Run** on your Raspberry Pi Simulator. You should see the messages printed on the log screen and confirmation that the message has been sent to IoT Hub.

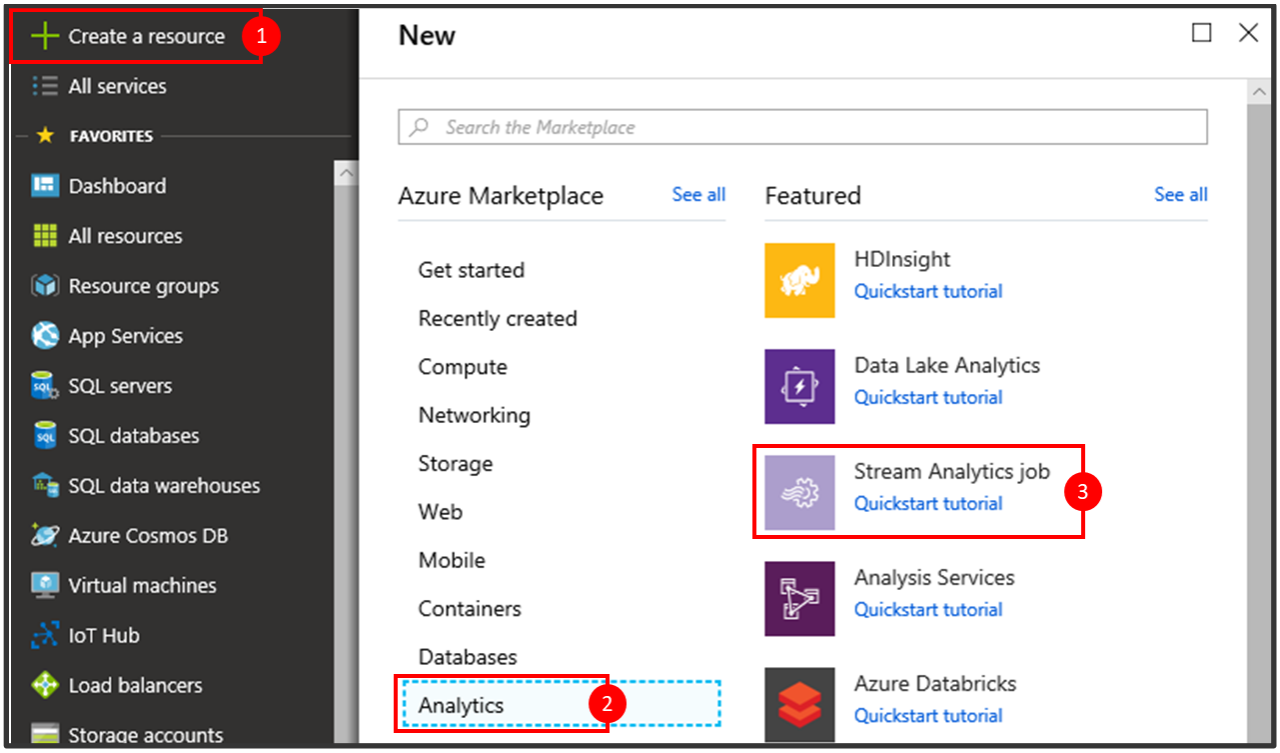
©2018 Microsoft Corporation



©2018 Microsoft Corporation

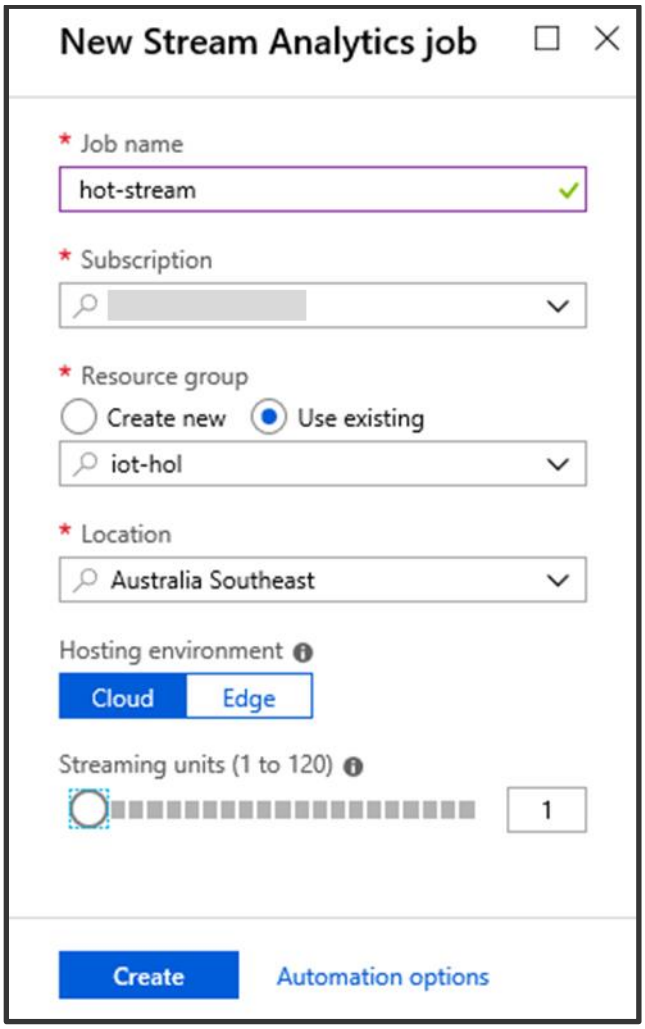
Exercise 2 – Create and Deploy a Stream Analytics Job

1. In your browser, navigate to the **Azure Portal** (https://portal.azure.com).
2. Select **+Create a Resource**, **Data + Analytics**, then select **Stream Analytics job**.



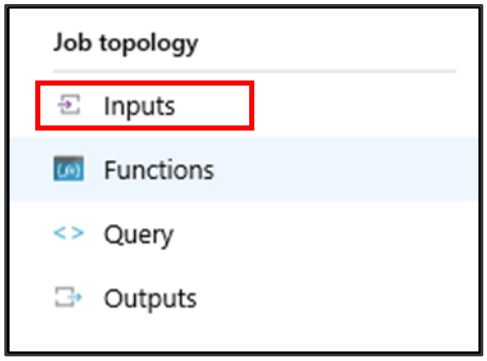
3. On the New Stream Analytics Job blade, enter the following:
   a. Job Name: Enter a name that uniquely identifies the job, in this case **hot-stream**.
   b. Subscription: Choose the same subscription you have been using thus far.
   c. Resource Group: Choose the **iot-hol** Resource Group.
   d. Location: Choose the same Location as you have for your other resources.
   e. Hosting environment: Select **Cloud**.
   f. Streaming Units: Select 1

©2018 Microsoft Corporation



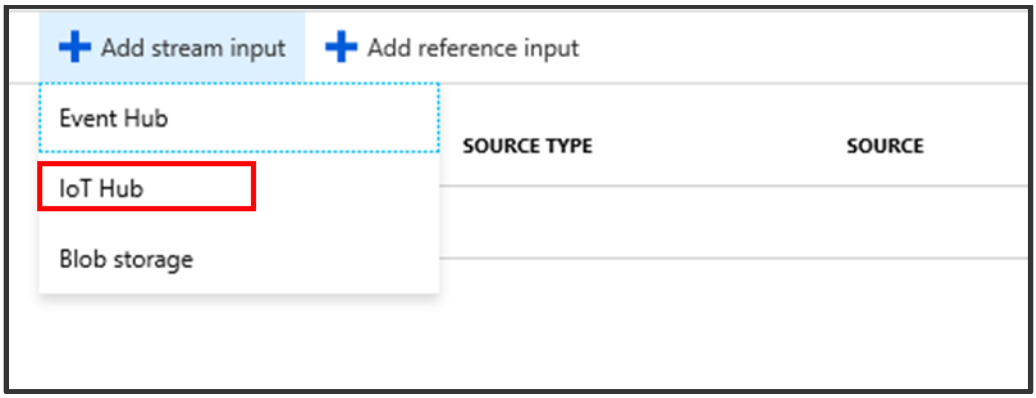
g. Select **Create** to provision the new Stream Analytics job.

4. Once provisioned, navigate to your new Stream Analytics job in the portal.
5. Select **Inputs** on the left-hand menu, under Job Topology.



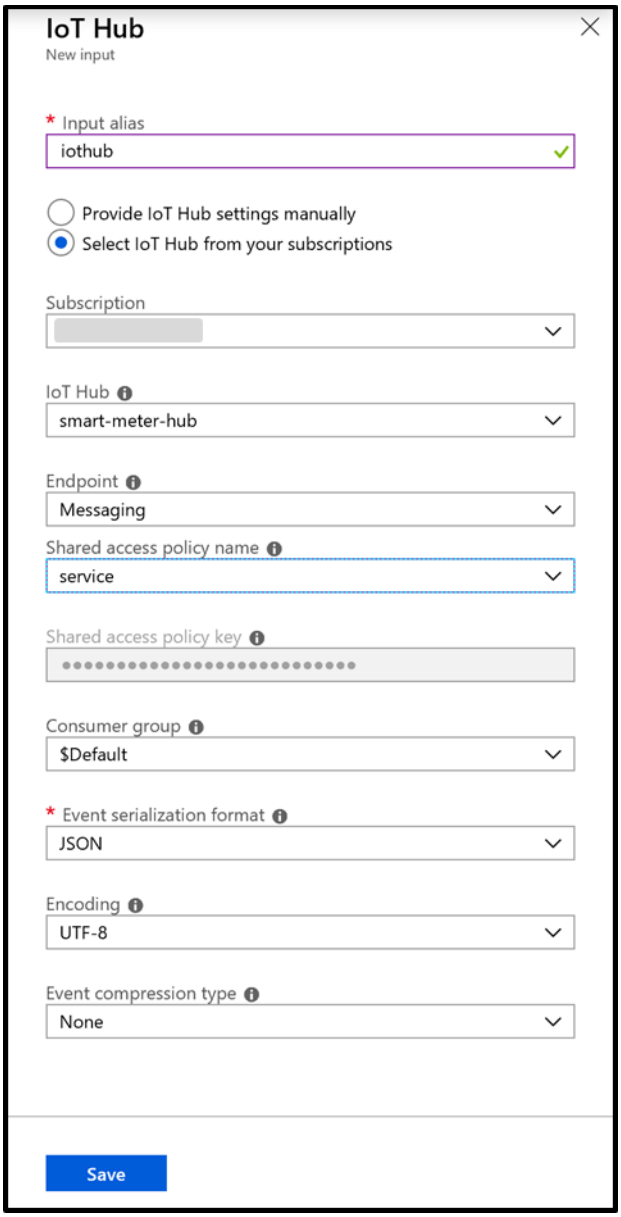
6. On the Inputs blade, select **+Add Stream Input** and **IoT Hub** from the drop-down list to add an input connected to your IoT Hub.

 ©2018 Microsoft Corporation



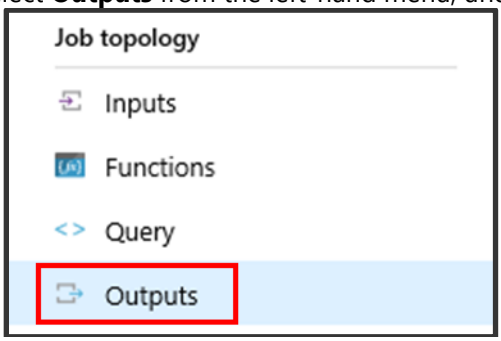
7. On the New Input blade, enter the following:
    a. Input Alias: Set the value to something that uniquely represents the source, in this case **iothub**.
    b. Choose **Select IoT hub from your subscriptions**.
    c. IoT Hub: Select your existing IoT Hub, **smart-meter-hub**.
    d. Endpoint: Choose **Messaging**.
    e. Shared Access Policy Name: Set to **Service**.
    f. Consumer Group: Leave as **$Default**.
    g. Event serialization format: Choose **JSON**.
    h. Encoding: Choose **UTF-8**.
    i. Event compression type: Leave set to **None**.

©2018 Microsoft Corporation



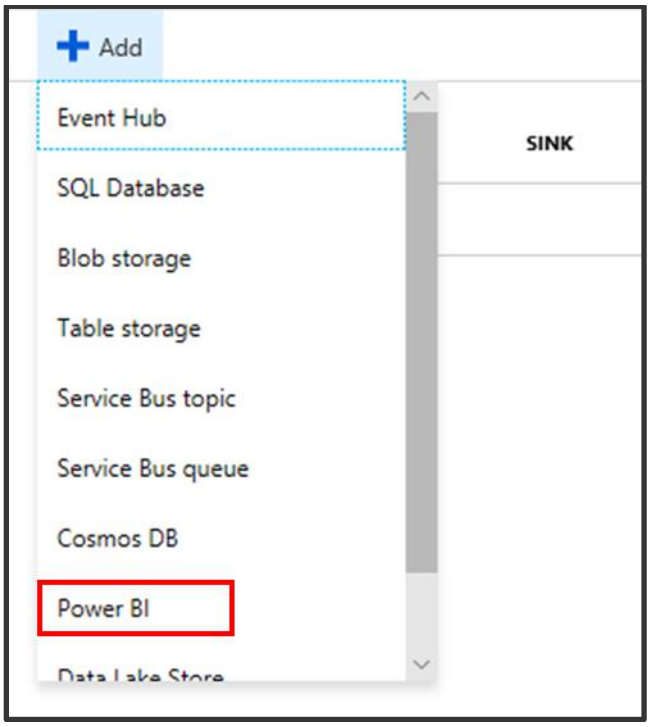
j. Select **Save**.

8. Now, select **Outputs** from the left-hand menu, under Job Topology.



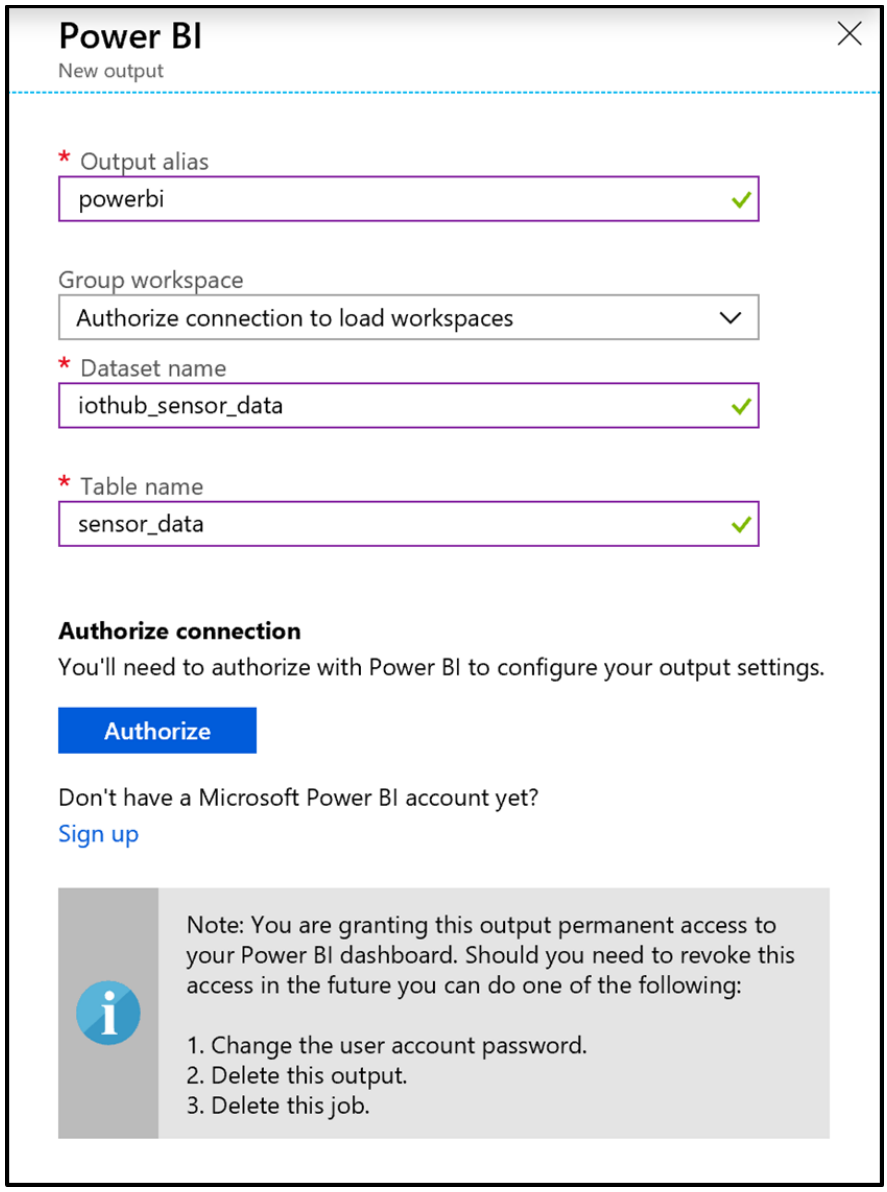
a. In the Outputs blade, select **+Add** and **Power BI** from the drop-down list to add the output destination for the query.

 ©2018 Microsoft Corporation



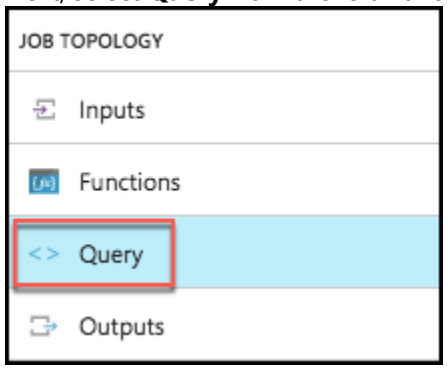
9. On the New output blade, enter the following:
    a. Output alias: Set to **powerbi**.
    b. Group workspace: Select **Authorize connection to load workspaces**
    c. Dataset Name: Set to **iothub_sensor_data**
    d. Table Name: Set to **sensor_data**
    e. Select **Authorize** to authorize the connection to your Power BI account. When prompted in the popup window, enter the account credentials you used to create your Power BI account in Before the Hands-on Lab, Task 1.

©2018 Microsoft Corporation



f. Select **Save**.

g. Next, select **Query** from the left-hand menu, under Job Topology.

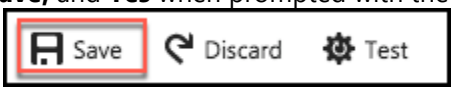


 ©2018 Microsoft Corporation

10. In the query text box, paste the following query.

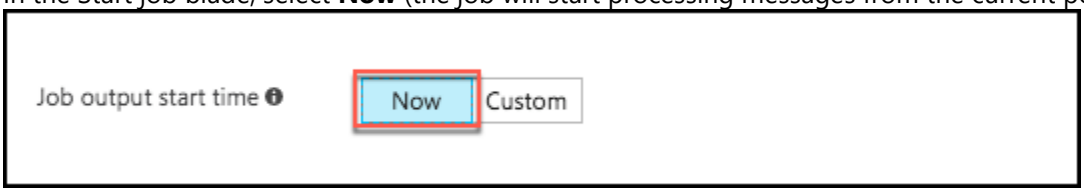
```
SELECT
    deviceId,
    temperature,
    tempmin,
    tempmax,
    temptarget,
    humidity,
    humiditymin,
    humiditymax,
    humiditytarget,
    (pressure * 100) AS pressurehpa,
    System.TimeStamp AS AEST
INTO
    powerbi
FROM
    iothub
```

11. Select **Save,** and **Yes** when prompted with the confirmation.



12. Return to the Overview blade on your Stream Analytics job, and select **Start**.

13. In the Start job blade, select **Now** (the job will start processing messages from the current point in time onward).



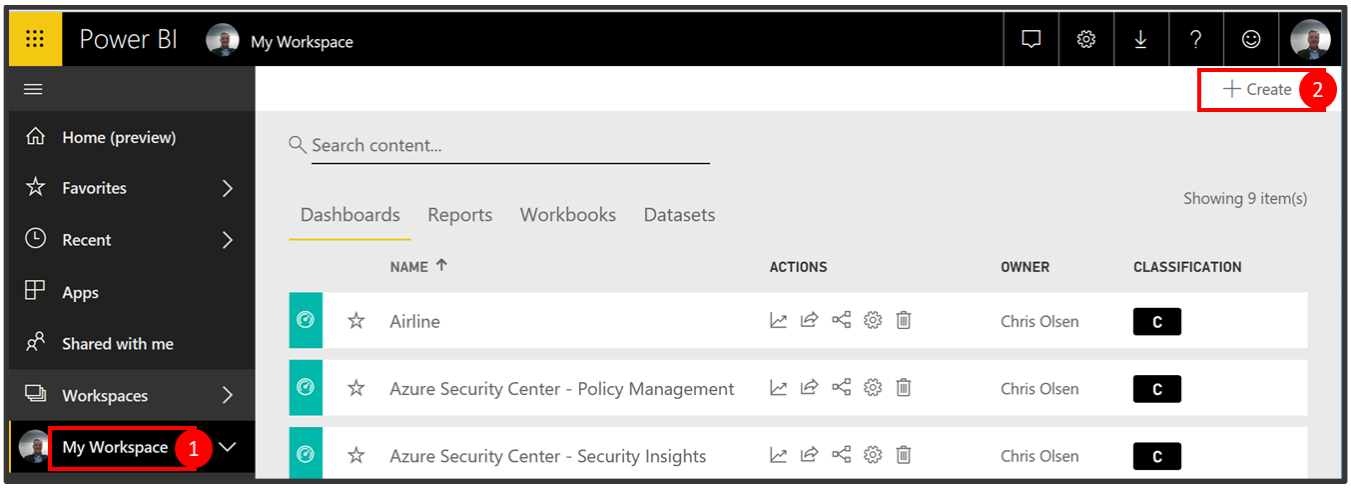
14. Select Start.
15. Allow your Stream Analytics Job a few minutes to start.

Once the Stream Analytics Job has successfully started, verify that you are showing a non-zero amount of **Input Events** on the **Monitoring** chart on the overview blade. You may need to reconnect your devices on the Smart Meter Simulator and let it run for a while to see the events.

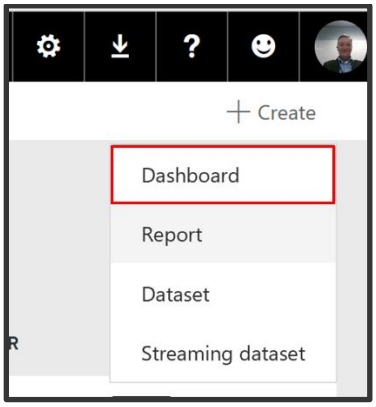
©2018 Microsoft Corporation

Exercise 3 – Visualisation in PowerBI

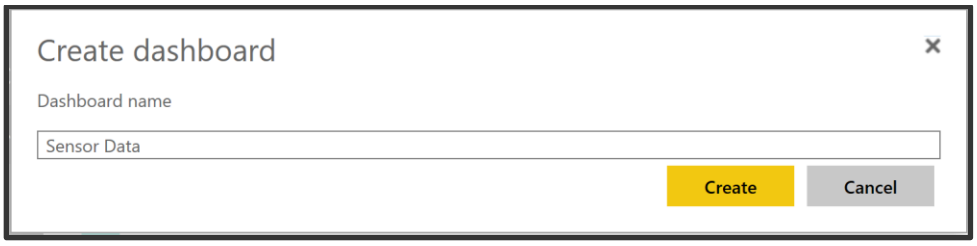
1. Login to the Power BI account that you created earlier (https://app.powerbi.com/).
2. On the Power BI Home page, click on **My Workspace** and then click on **+ Create** in the upper right corner



3. In the drop down list that is presented choose **Dashboard** to create a new dashboard.

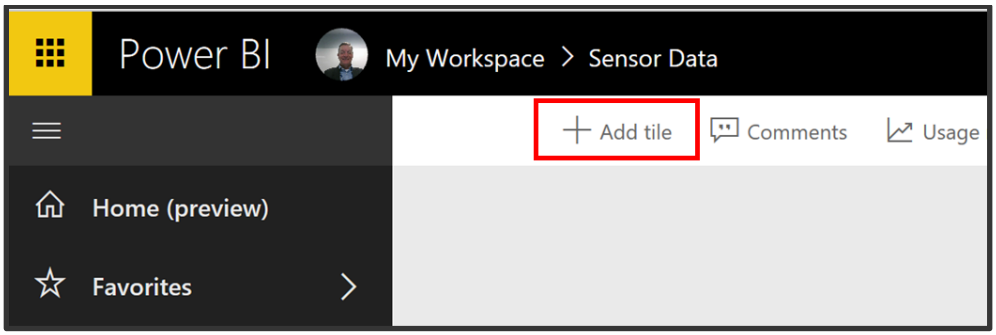


4. Enter **Sensor Data** as the Dashboard name and click Create. You will be taken to the empty Sensor Data dashboard.



5. Click **+ Add Tile** to create the first visualization on the dashboard.

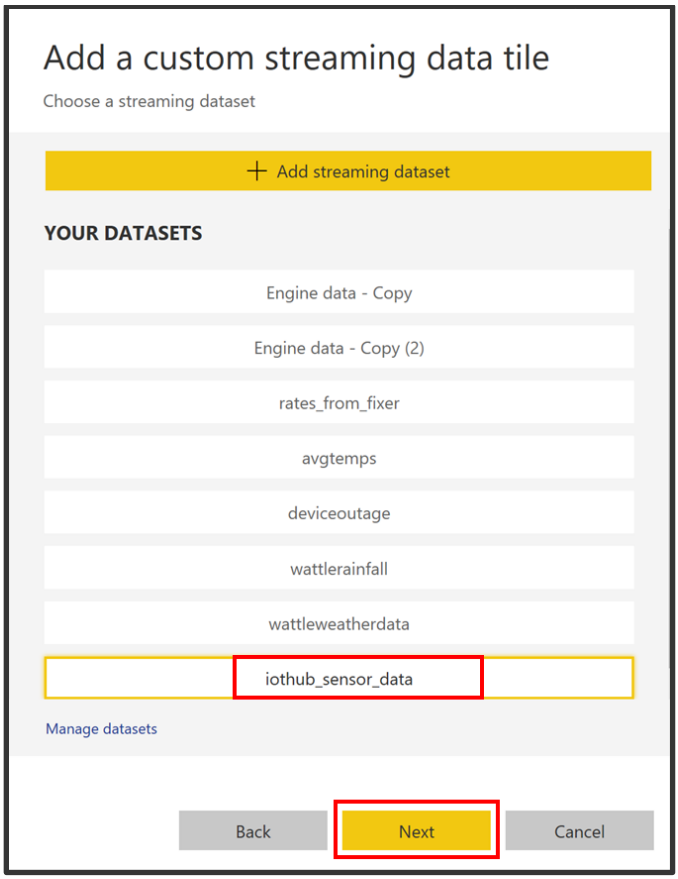
©2018 Microsoft Corporation



6. In the tile source menu, choose **Custom Streaming Data** to create a tile based on data being streamed to Power BI from Azure Stream Analytics. Then click **Next**.

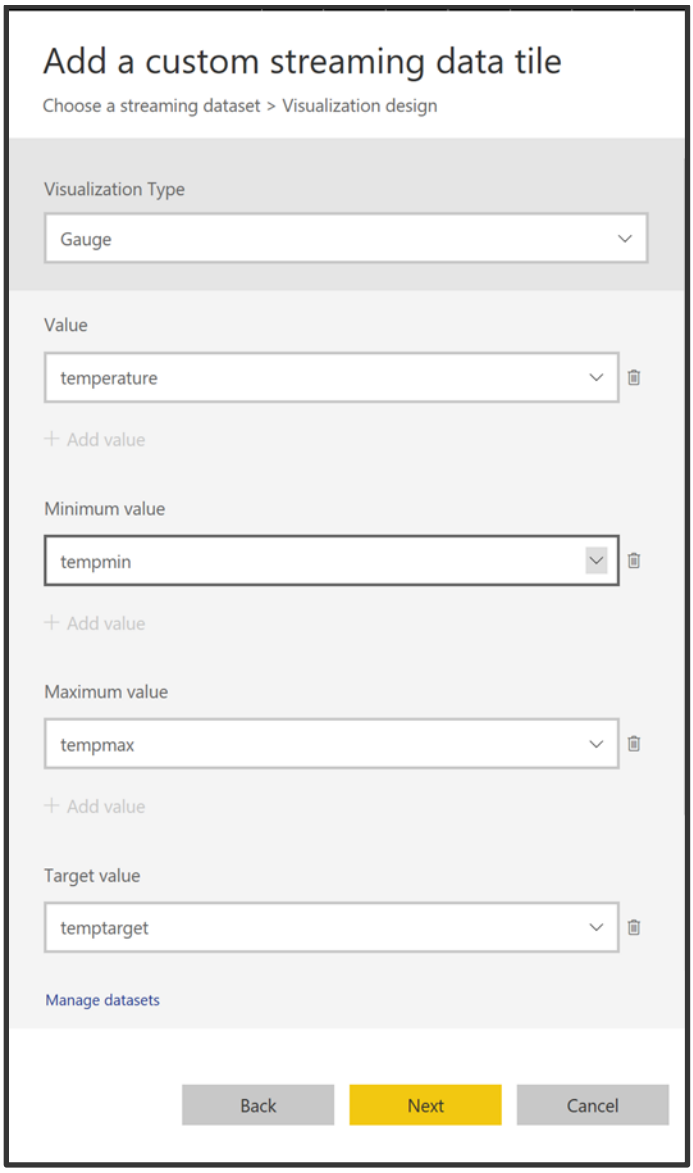
7. In the streaming dataset menu, choose the **iothub_sensor_data** dataset. This is the Powerbi dataset name you gave the Azure Stream Analytics job output.

©2018 Microsoft Corporation



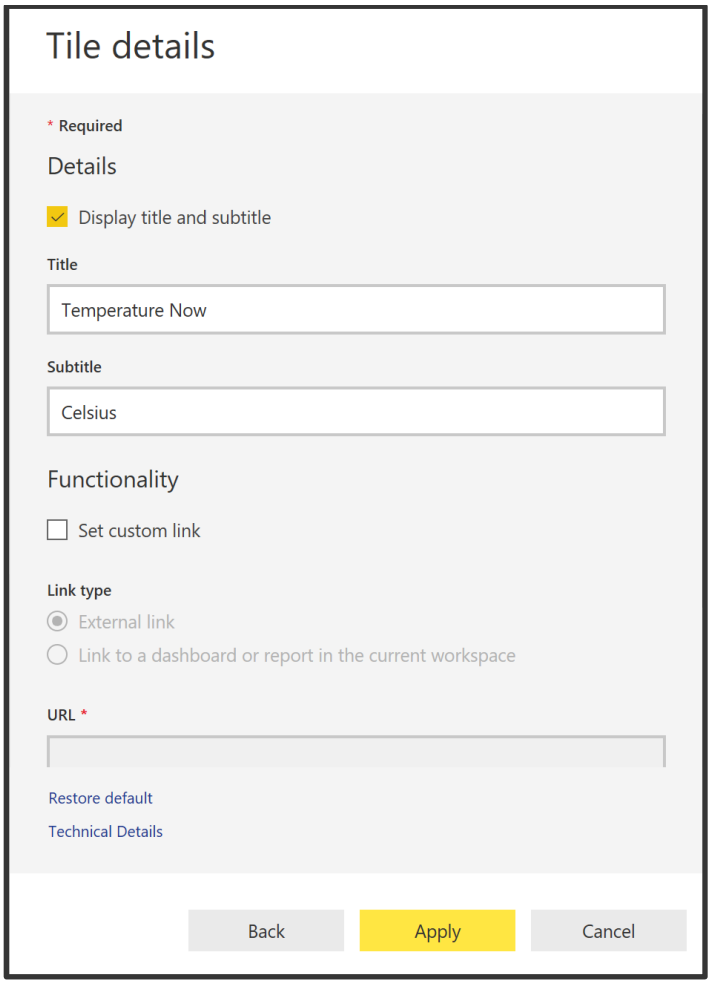
8. Click **Next**.
9. In the Visualisation Design menu, choose
    a. **Gauge** as the visualization type
    b. **Temperature** as the data stream value
    c. **Tempmin** as the data stream minimum value
    d. **Tempmax** as the data stream maximum value
    e. **Temptarget** as the data stream target value

©2018 Microsoft Corporation



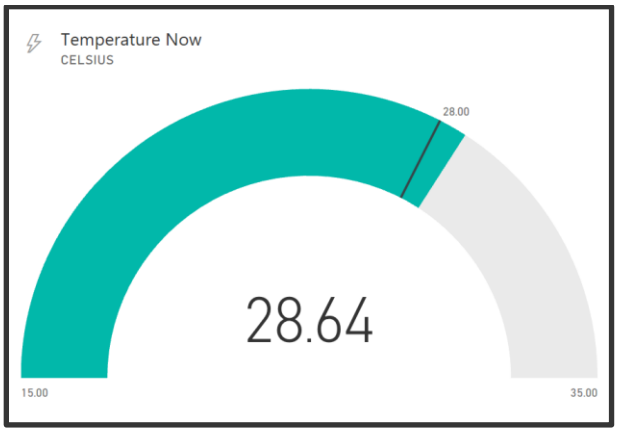
10. Click **Next**.

11. In the Tile details menu give the streaming data tile a meaningful title such as **Temperature Now**, and a subtitle that reflects the unit of measure such as **Celsius**.



12. Click **Apply**.

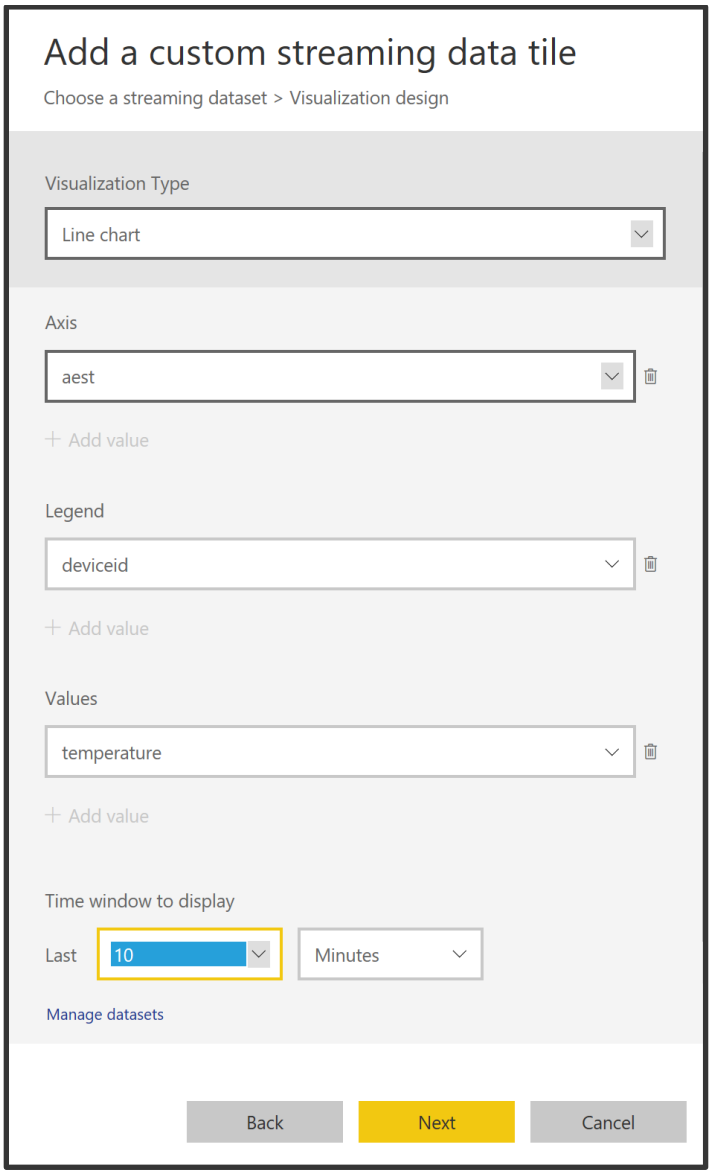
    The streaming data tile will be created on the dash board.



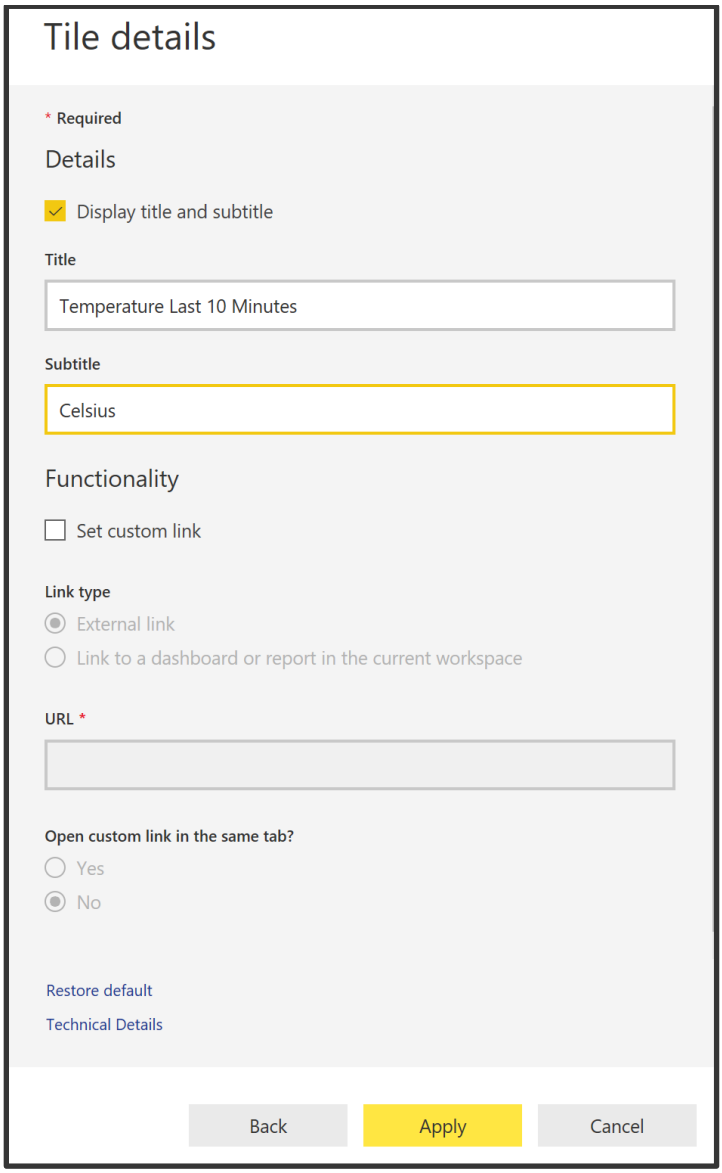
13. We will now create a second streaming dataset visual on the dashboard. To do this follow steps 5-8 as before.
14. In the Visualisation Design menu, choose:
    a. **Line Chart** as the visualisation type
    b. **AEST** as the Axis
    c. **Deviceid** as the Legend
    d. **Temperature** as the Values

©2018 Microsoft Corporation

e. Set the **Time Window to Display** to the Last **10** Minutes

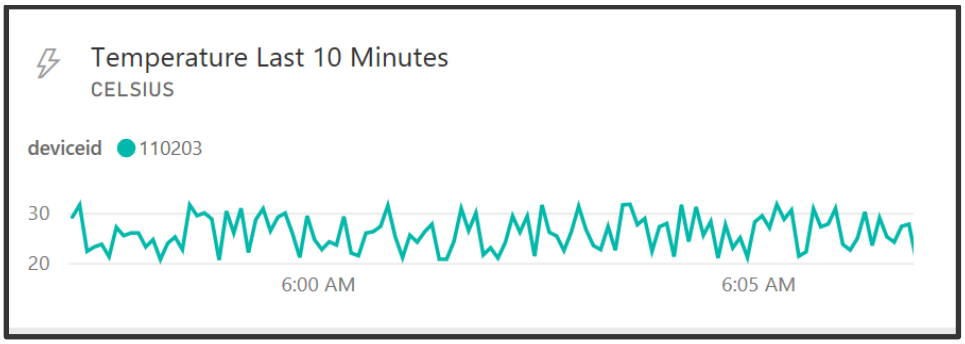


15. Click **Next**.
16. In the Tile details menu give the streaming data tile a meaningful title such as **Temperature Last 10 Minutes**, and a subtitle that reflects the unit of measure such as **Celsius**.



17. Click **Apply**.

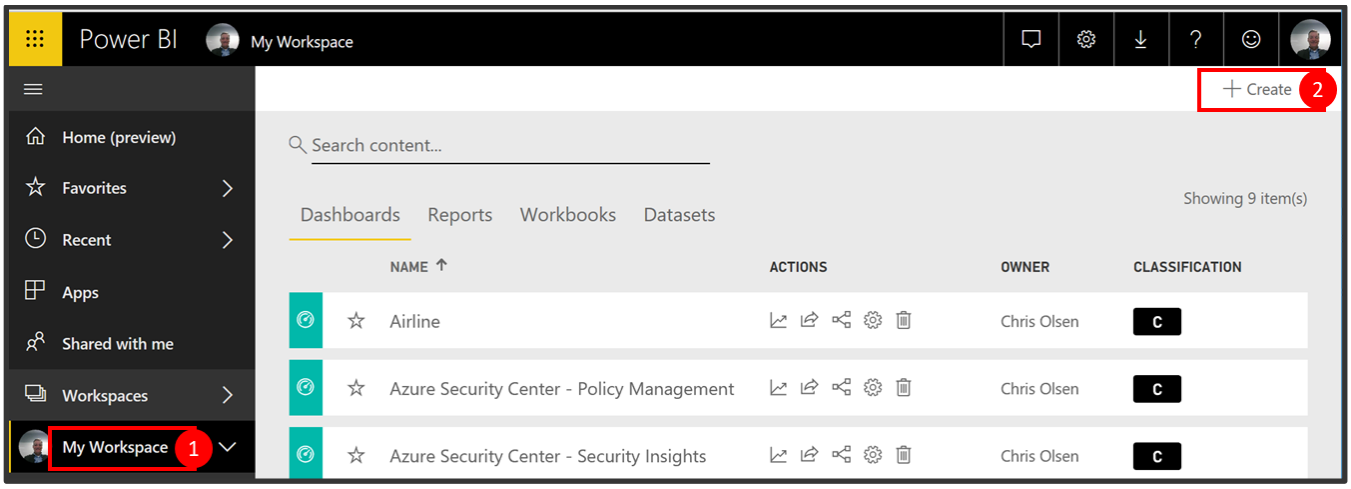
    The second streaming data tile will be created on the dash board.



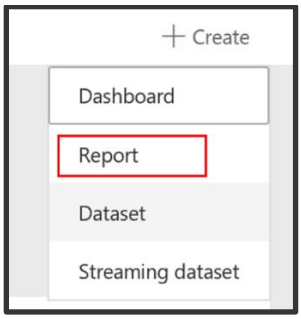
We will now add some reports to the dashboard to show average and maximum values over the full dataset.

 ©2018 Microsoft Corporation

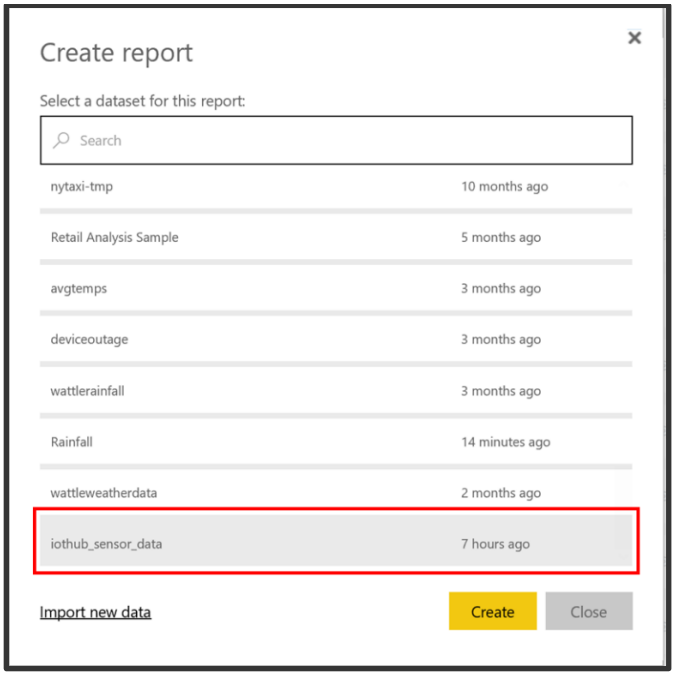
18. Go back to **My Workspace** and click **+ Create** again.



19. This time choose **Report** from the drop down list.

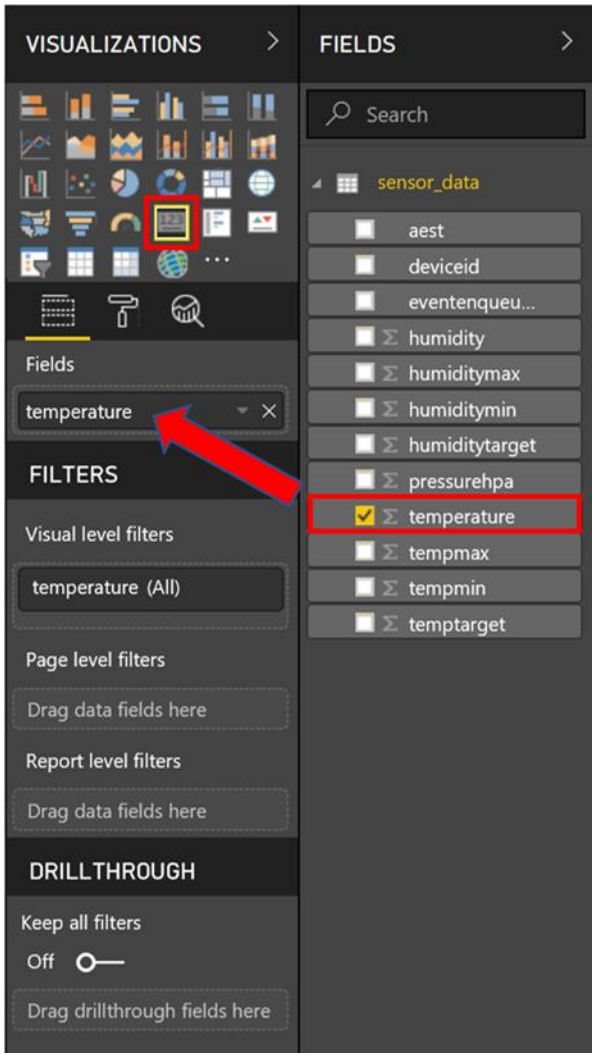


20. Select the **iothub_sensor_data** dataset and click **Create**. You will be taken to the Report creation page.



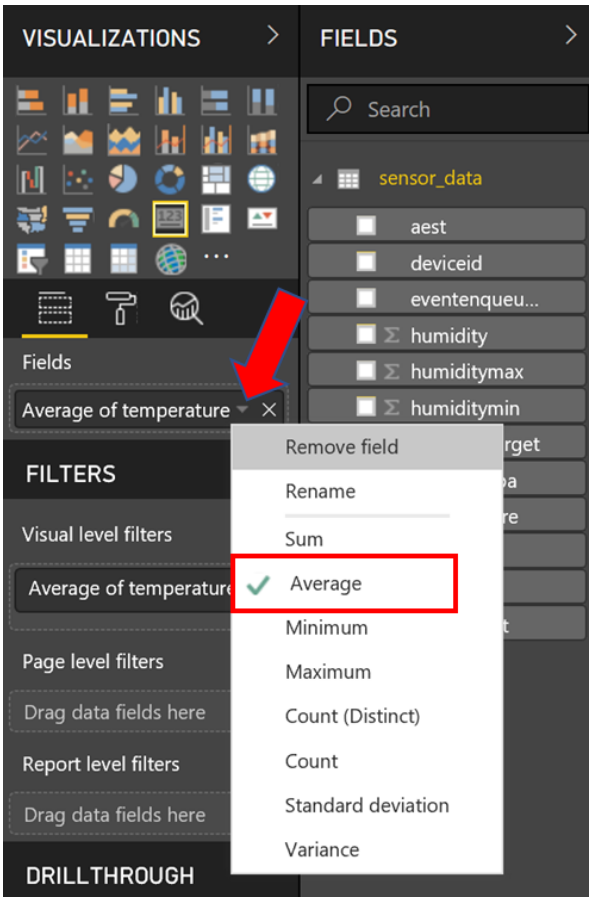
©2018 Microsoft Corporation

21. We will create a average temperature report so click the **Card** icon in the Visualizations pane and then drag the **temperature** item from the Fields pane to the Fields area in the Visualization pane (where is says 'Add data fields here').

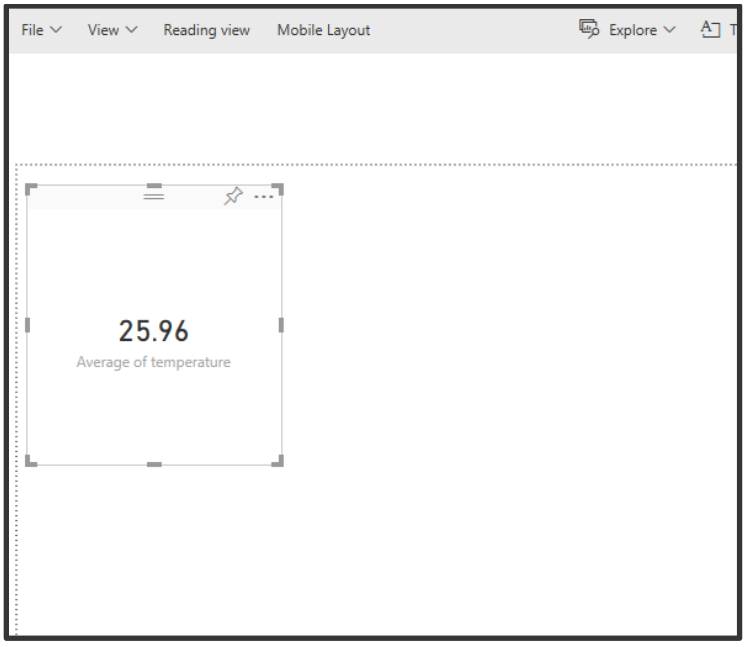


22. Now click the arrow  icon to the right of temperature you just dragged into the Fields area. A menu will pop up. Choose **Average** from the menu.

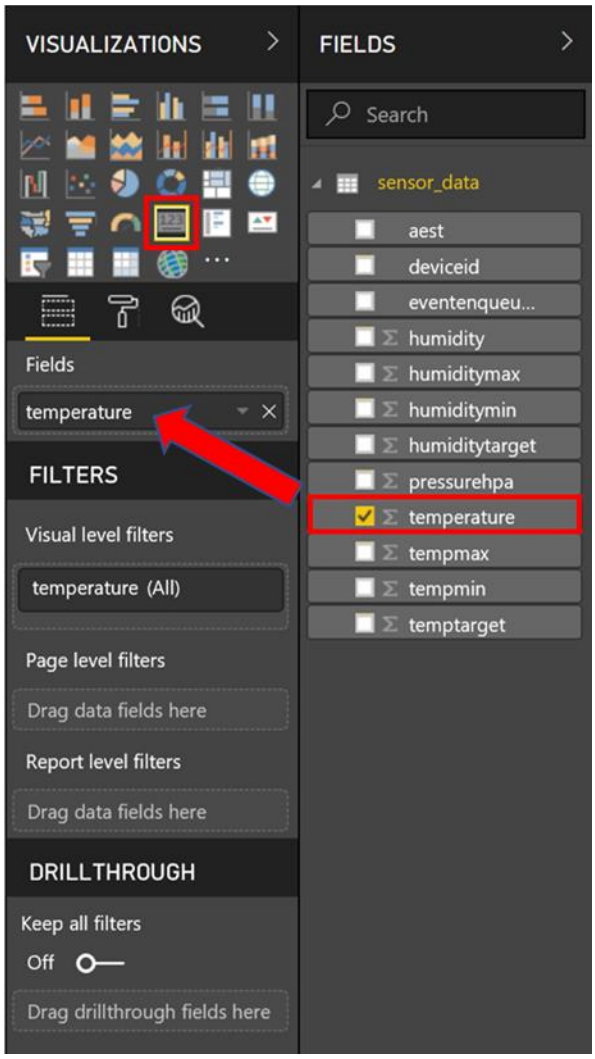
©2018 Microsoft Corporation



23. Click anywhere on the report canvas outside of your report tile to finish the report

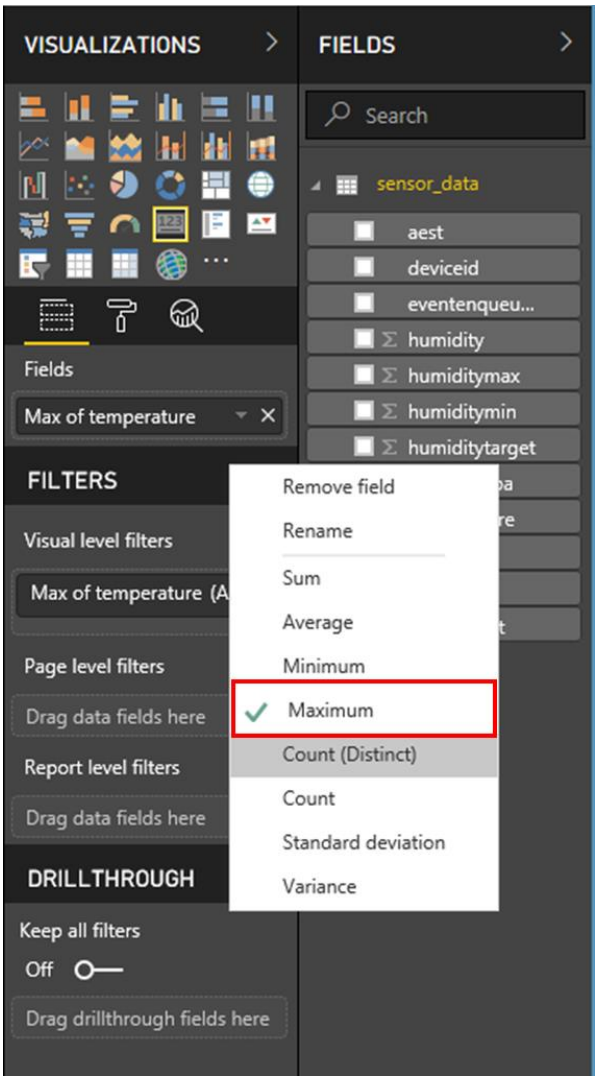


24. Now we will repeat the process to create a maximum temperature report. As before click the Card icon in the Visualizations pane and then drag the **temperature** item from the Fields pane to the Fields area in the Visualization pane (where is says 'Add data fields here').



25. Again click the arrow  icon to the right of temperature you just dragged into the Fields area. This time choose **Maximum** from the menu.

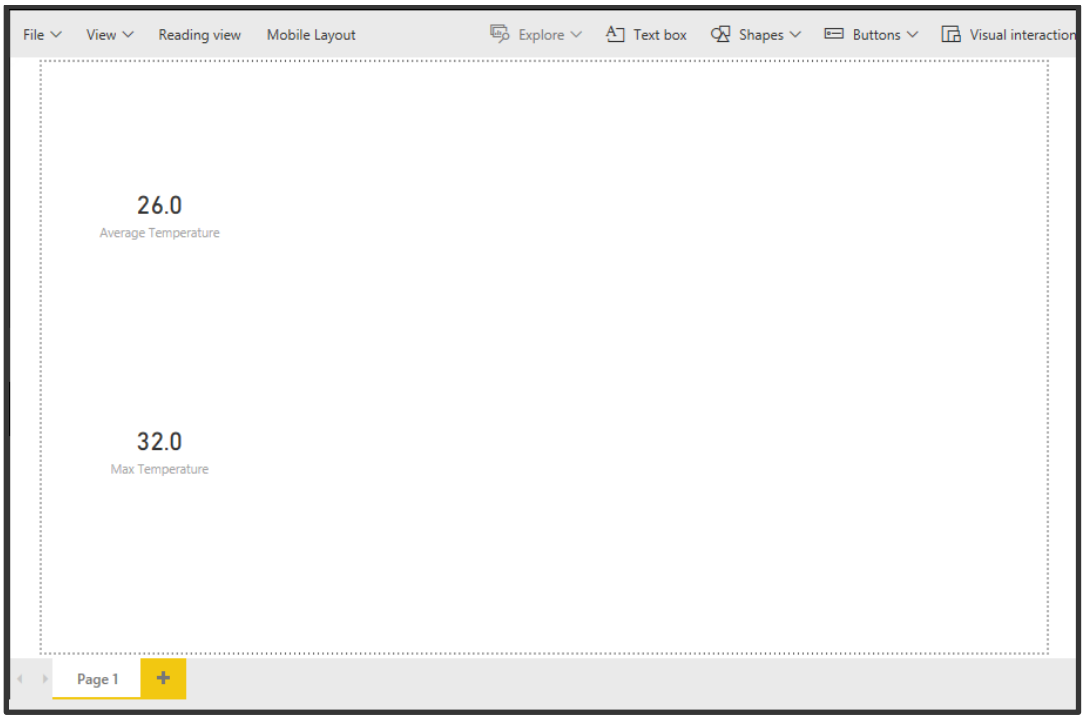
©2018 Microsoft Corporation



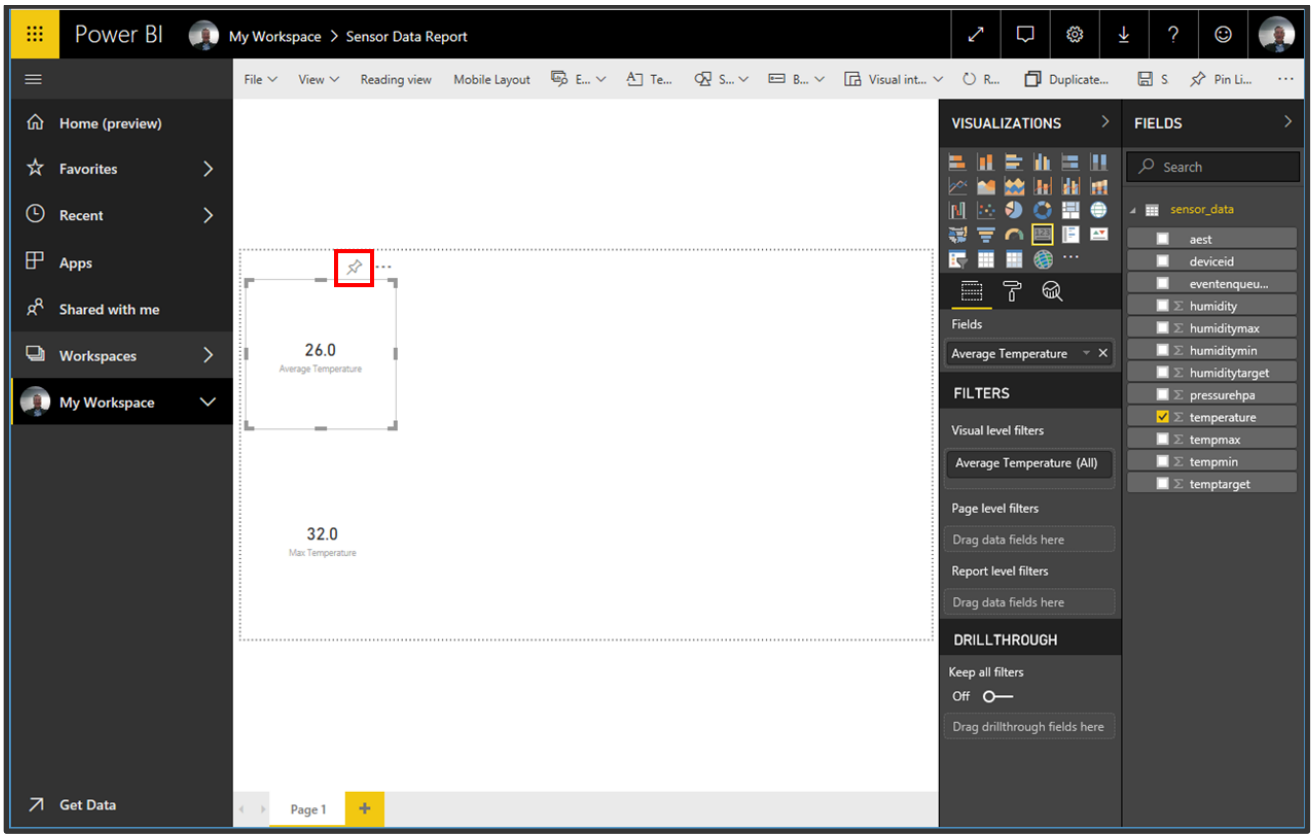
26. You can change your various parts of the report format if you wish. For example, if you would like to rename the report heading you can click the arrow  icon again in the Fields area and choose **Rename** from the menu. The heading will be highlighted, and you can change it how you want.

    You can also change the look of the report by clicking on the paint roller icon  in the Visualizations pane. For example, you can change the number of decimal places displayed by opening the **Data Label** menu and adjusting the **Value Decimal Places** field.

27. Click anywhere on the report canvas outside of your report tiles to finish the report. Your report canvas should look like this:

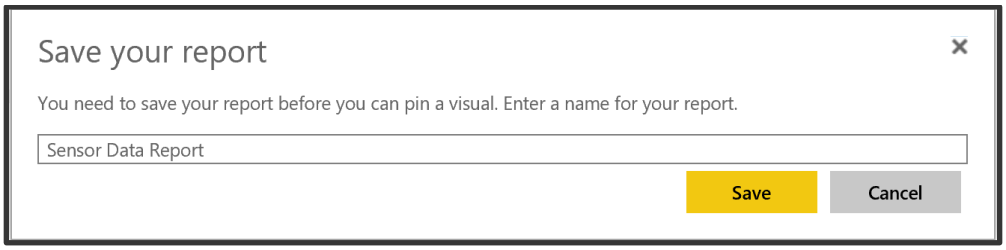


28. To pin your new reports to your Sensor Data Dashboard, click on a report and then click on the Pin  icon to pin the report to the dashboard

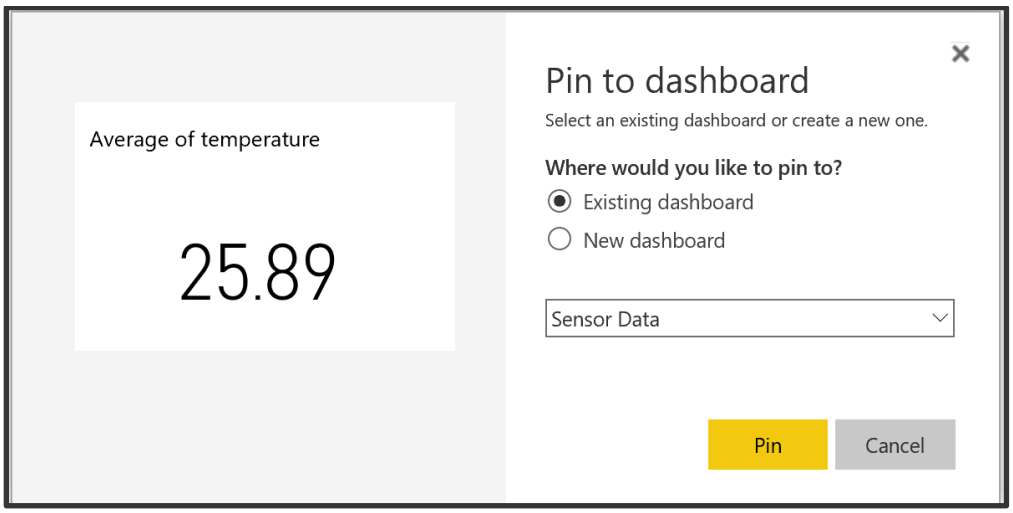


 ©2018 Microsoft Corporation

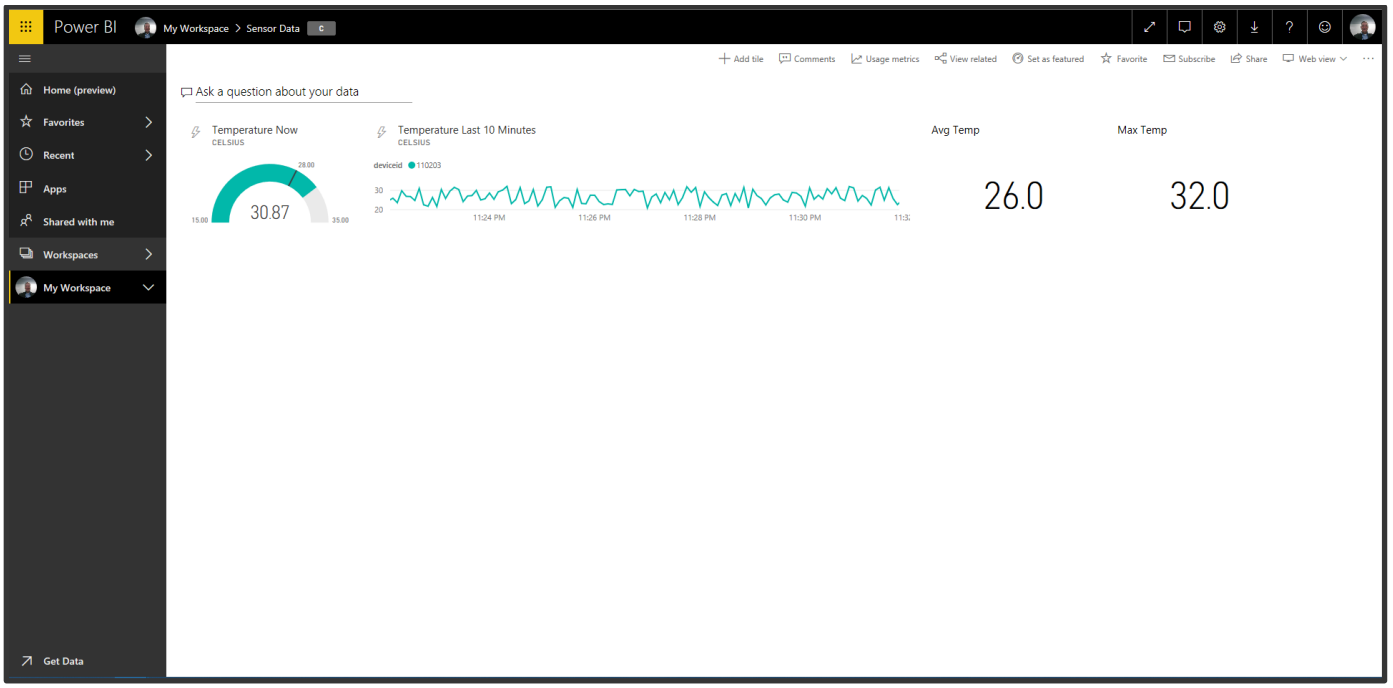
29. You will first be asked to save the report. Enter **Sensor Data Report** as the name and click **Save**.



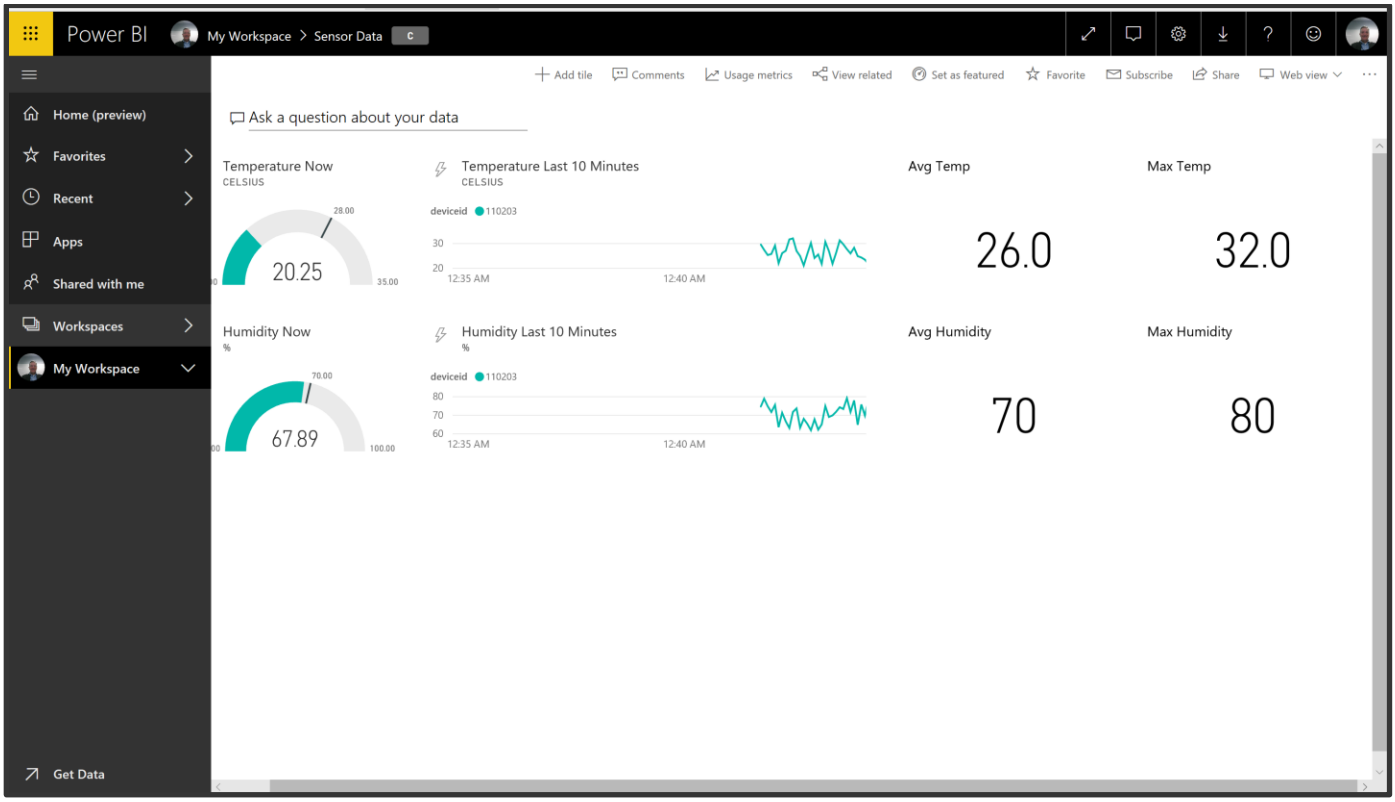
30. Ensure that the **Sensor Data** dashboard is chosen and click **Pin**.



31. Repeat the Pin process for the other report.
32. Navigate back to your **Sensor Data** Dashboard by clicking on **My Workspace**. Your Dashboard should look something like this:



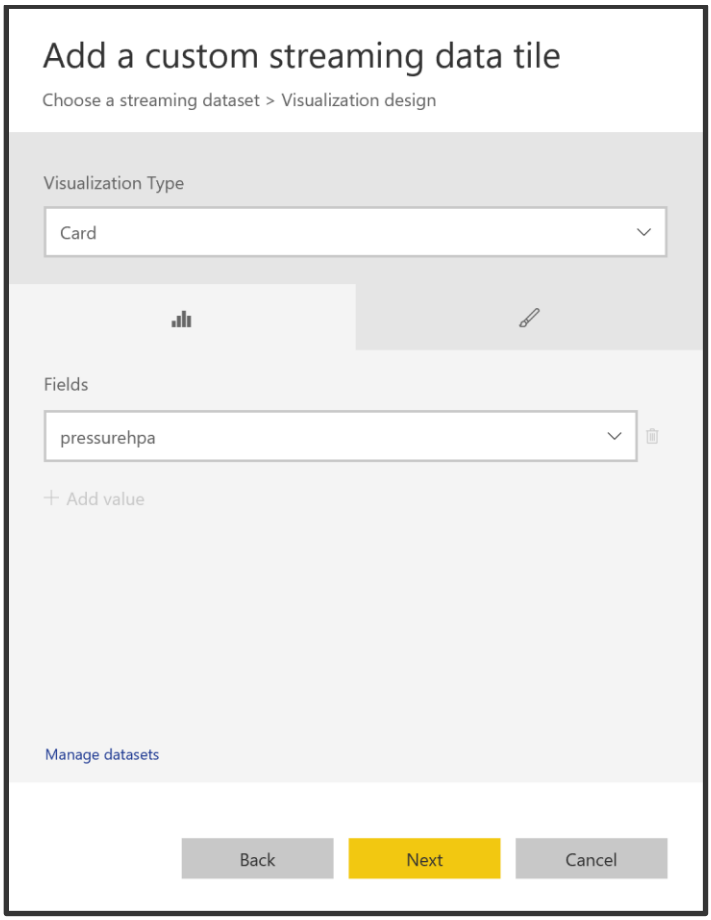
33. You have now created a dashboard for live and aggregate temperature data streaming to Power BI from the simulated device via IoT Hub and Stream Analytics. Now repeat **steps 5-30** using **Humidity** data to add live and aggregate humidity data to the dashboard. Your dashboard will now look something like this.



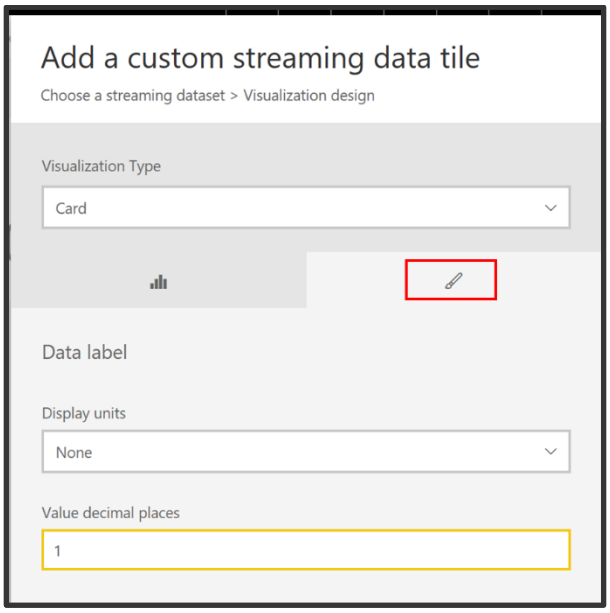
34. Now repeat **steps 5-30** using **Pressure** data to add live and aggregate air pressure data to the dashboard.

    Instead of using a gauge for the visualization in **step 9** try using a **Card** instead with a Field of **pressurehpa.**

©2018 Microsoft Corporation



35. Click the Paintbrush icon tab  and change the **Display Units** to **None** and the **Value Decimal** Places to **1**.

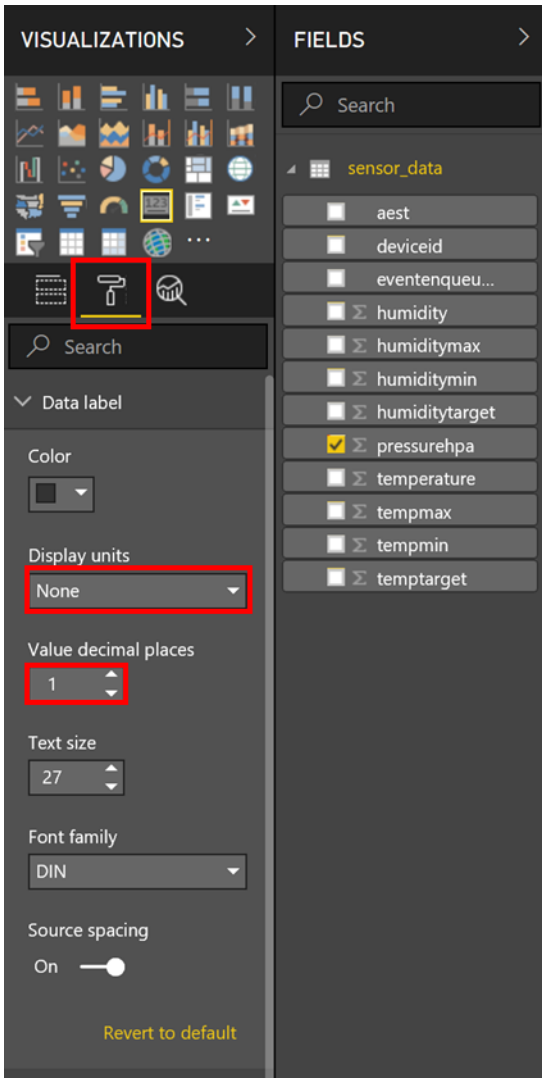


36. Click **Next**.
37. Give the Card a suitable title and subtitle and click **Apply**.
38. Complete the rest of the **steps 10-30** for Pressure. Instead of using Average function try using Minimum.

©2018 Microsoft Corporation

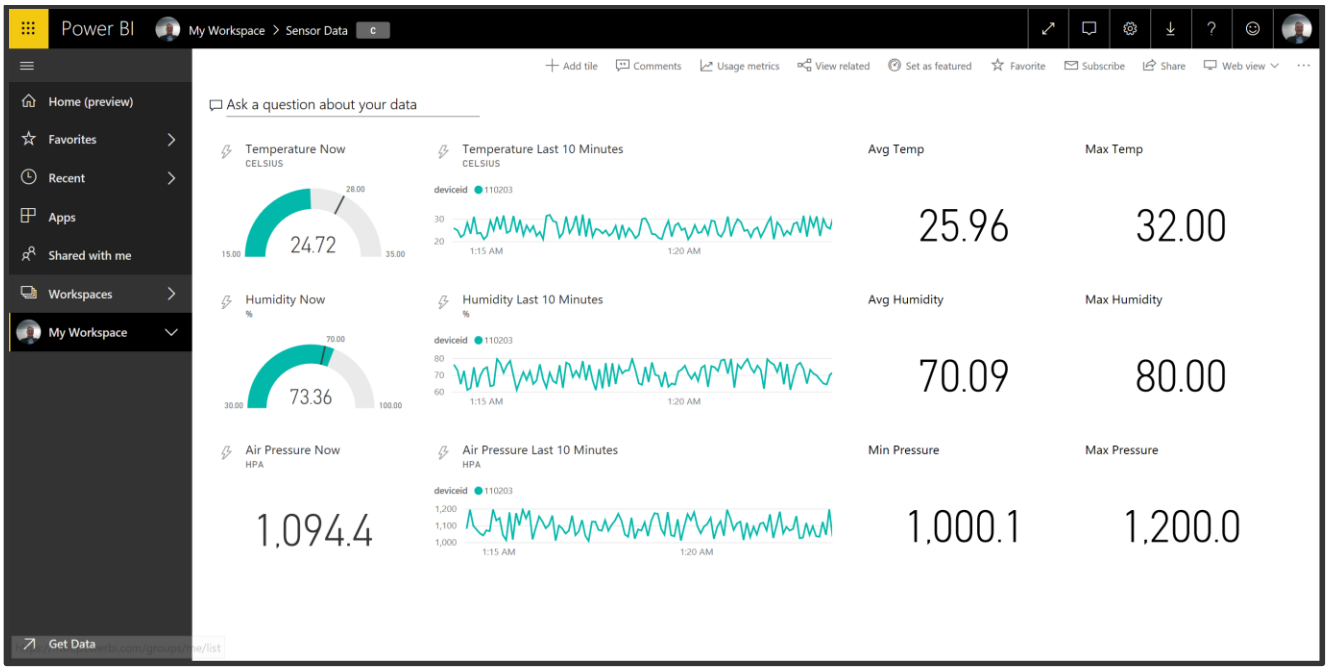
For the two pressure reports (min and max) you will need to format the **Display Units** to **None** and adjust the **Value Decimal Places** to **1**.

You can reach these by clicking on the paint roller icon  in the Visualizations pane and opening the **Data Label** menu.

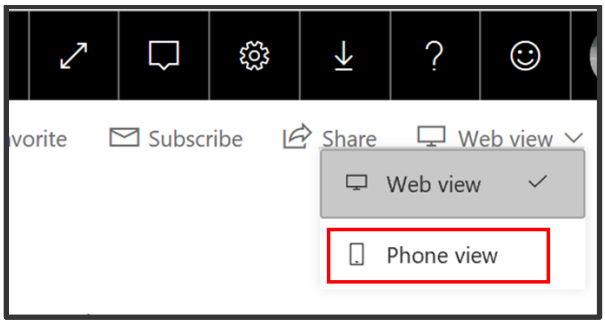


You have now successfully built a complete Power BI real-time dashboard for the simulated Temperature, Humidity and Air Pressure data. Your dashboard should look something like this:

©2018 Microsoft Corporation

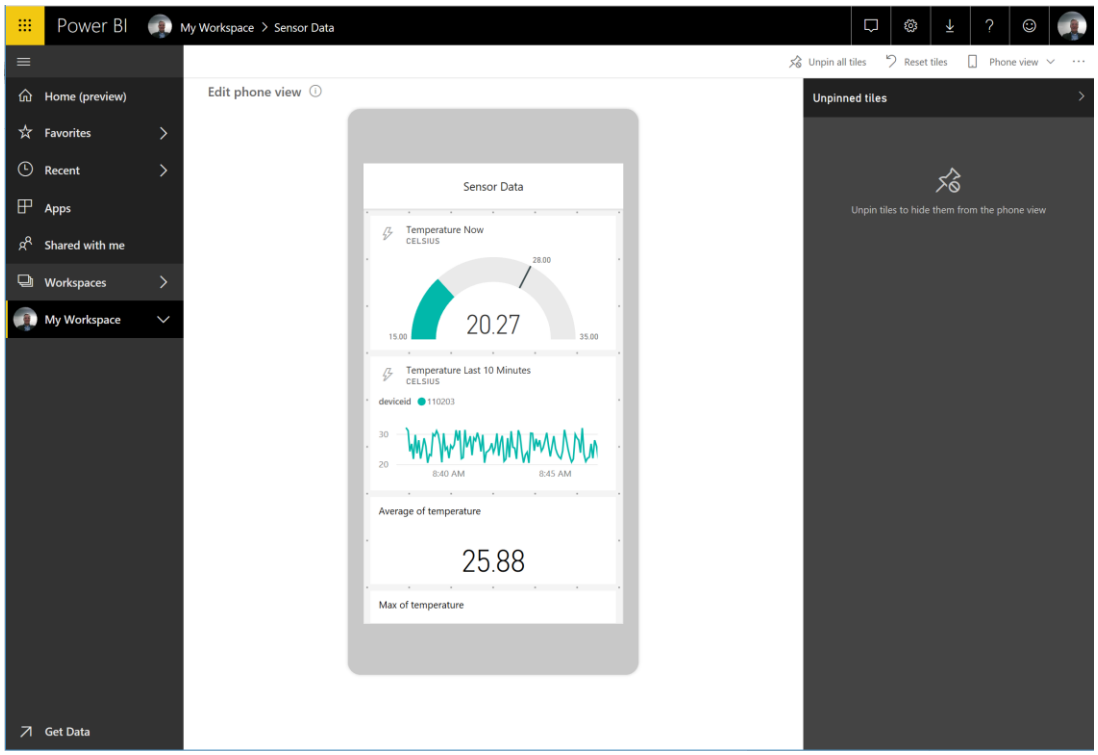


One of Power BI's advantages is that it automatically creates a mobile view of your dashboards and reports. To view your dashboards mobile view click on the **Web View** tab in the top right corner of the dashboard and select **Phone View**.



From the Phone View page you can edit the mobile view by rearranging, deleting or adding tiles to get the mobile dashboard view you want.

©2018 Microsoft Corporation



You can access the mobile view of your dashboard by installing Power BI on your iOS, Android or Windows 10 mobile devices.

This completes Lab 1 – Hot Path.

©2018 Microsoft Corporation

Lab 2 – Warm Path

Summary and Objectives

In this section of the Lab, you'll instantiate Microsoft's keystone IoT service, IoT Hub, and leverage the power of CosmosDB to store and manipulate your data, and visualise in PowerBI. This architecture is usually used for aggregation of data with some manipulation prior to visualisation

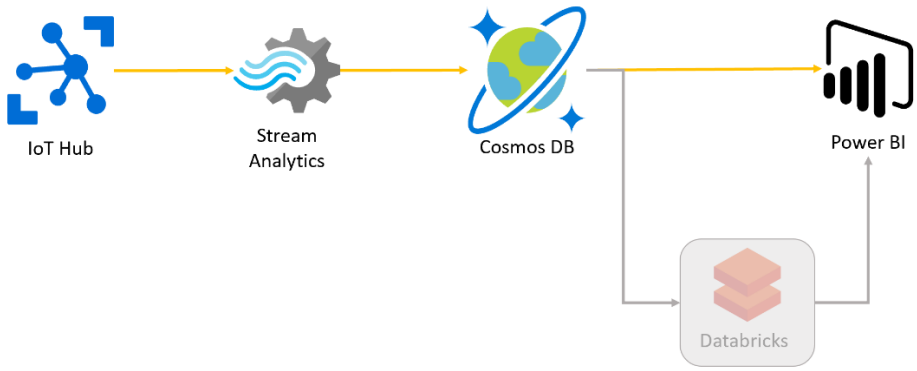


Figure 3 - Warm Path Architecture with Cosmos DB

To learn more about the services used in this Lab, see the following getting started guides:

- IoT Hub
- Azure Stream Analytics
- Cosmos DB
- Power BI

©2018 Microsoft Corporation

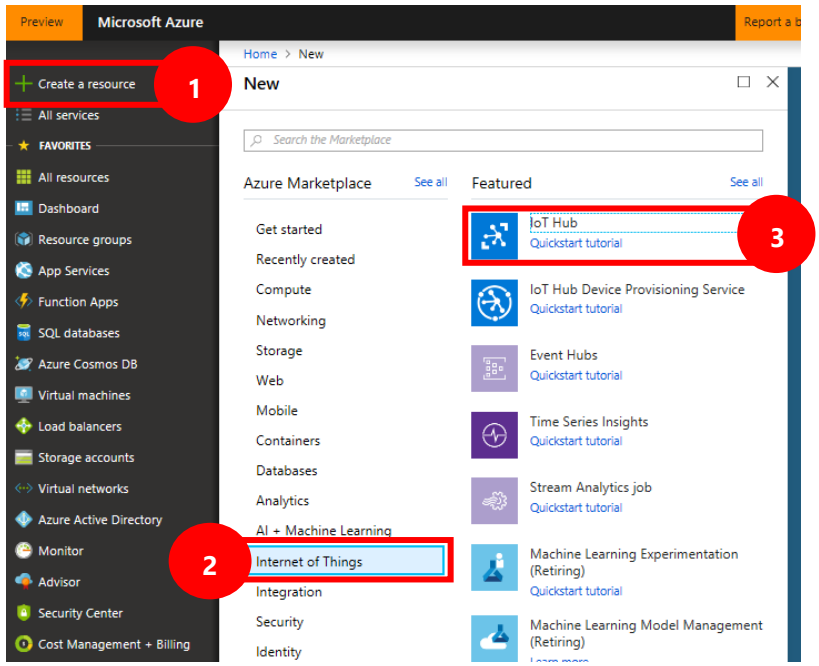
Exercise 1 – Provisioning IoT Hub

Duration: 20 minutes

Note: If you completed Lab 1, skip to Exercise 2.

In these steps, you will provision an instance of IoT Hub.

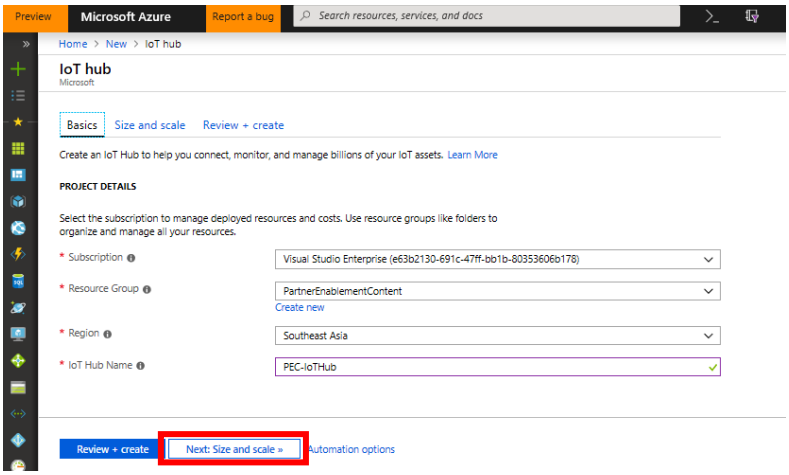
2. In a browser, navigate to the Azure Portal (https://portal.azure.com).
3. Select **+Create a resource**, then select **Internet of Things**, and select **IoT Hub**.



4. In the **IoT Hub** blade, in the **Basics** tab enter the following:
    a. *Subscription*: Select the subscription you'll be using for this lab.
    b. *Resource group*: Select, or create, the resource group you'll use for this lab
    c. *Region*: Select the region to deploy your IoT Hub. Usually this will be the same location as your resource group
    d. *IoT Hub Name*: Provide a name for your new IoT Hub

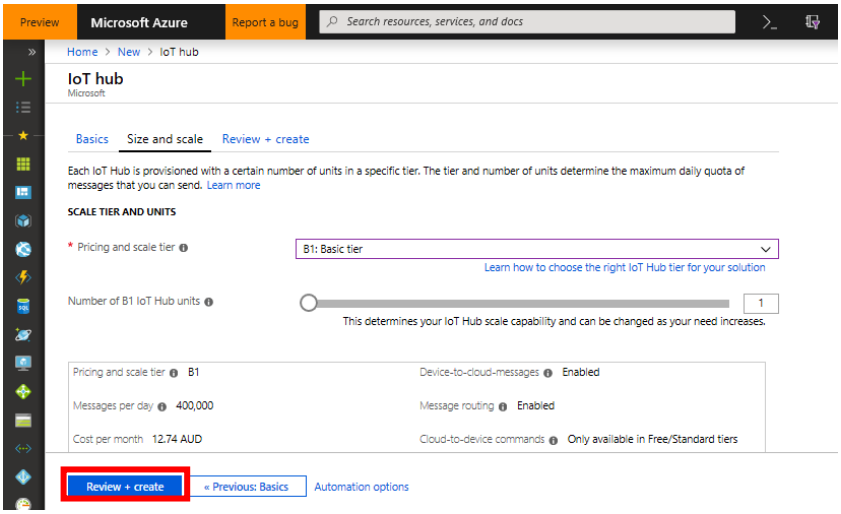
    Once complete, select the "Next: Size and scale" button at the base of the screen, or the "Size and scale" tab at the top

©2018 Microsoft Corporation



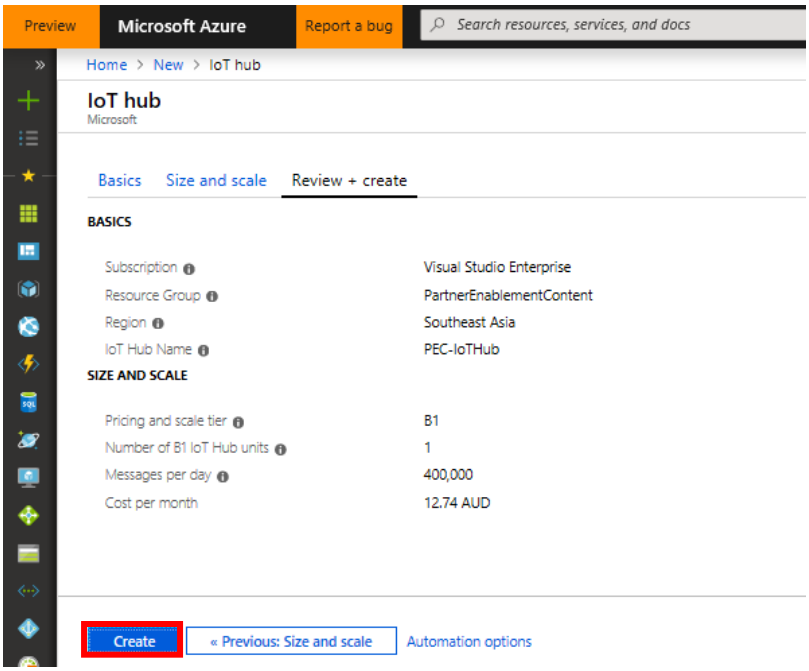
5. In this **Size and scale** screen, select the "Pricing and scale tier" in the dropdown menu
    a. Any tier will work, but for optimising cost choose in the following order:
        i. Free -> Basic -> Standard

    After you've selected the desired tier, click the **Review + create** button at the base of the screen.

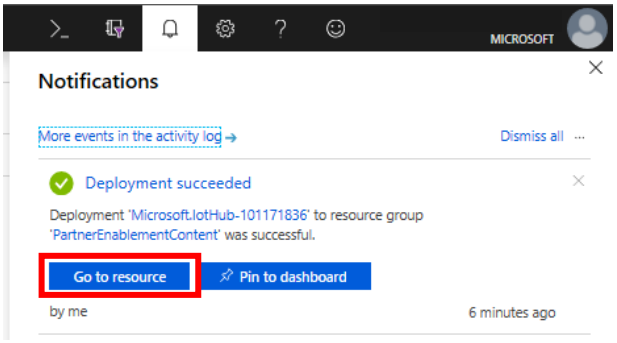


6. A summary screen will now show all the details you selected. If these are correct, select **Create** at the bottom of the screen. This will begin the deployment of your desired IoT Hub configuration.

©2018 Microsoft Corporation



7. When the IoT Hub deployment is completed, you will receive a notification in the Azure portal at the top-right of your screen. Navigate to your new IoT Hub by selecting Go to resource in the notification.

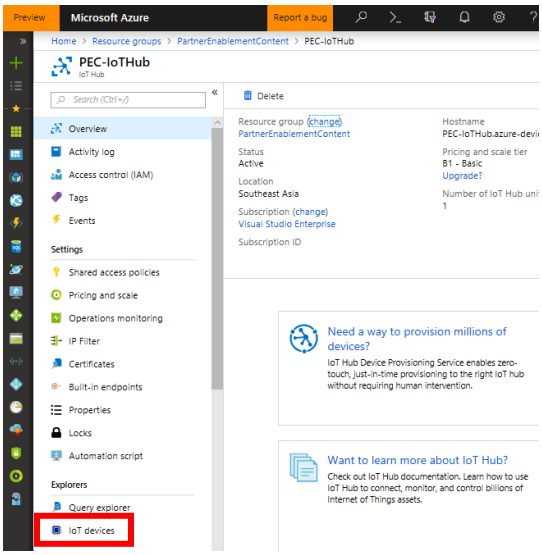


8. Now you have access to all the features and settings for your newly deployed IoT Hub. Let's connect a simulated Raspberry Pi to make sure devices can connect.

©2018 Microsoft Corporation

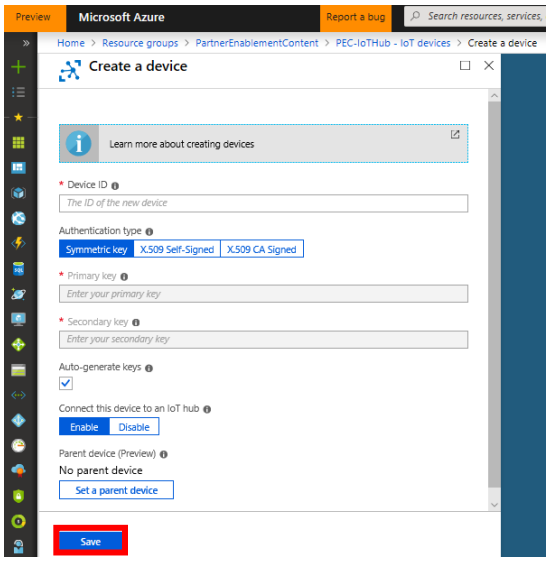
Create Device in IoT Hub

9. From the IoT Hub's Overview blade, select **IoT devices** under Explorers on the left-hand menu.



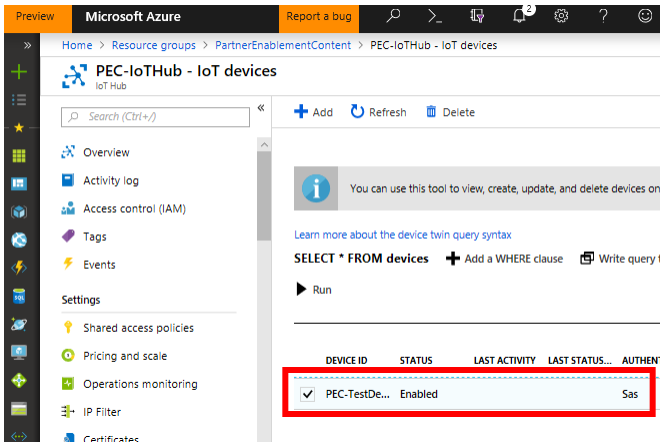
10. Select **+ Add** at the top of the screen and enter the following information on the "Create a device" blade
    a. *Device ID*: A unique ID for the new device
    b. *Authentication Type*: Leave this as Symmetric key
    c. *Auto-generate keys*: Make sure this is selected
    d. *Connect this device to an IoT Hub*: Make sure this is enabled

    Click **Save** at the base of the screen to confirm the device creation.

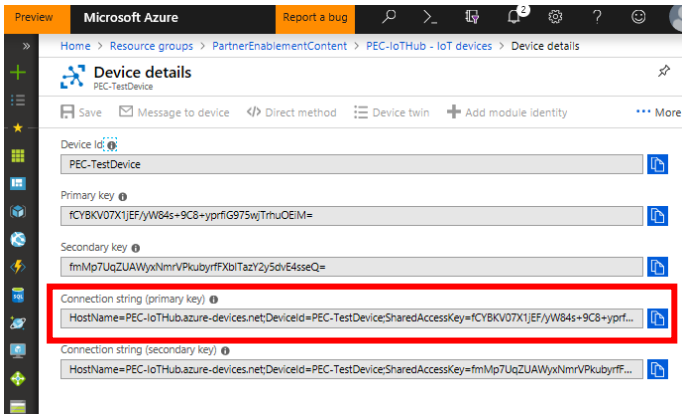


11. Once the device has been created (this may take a few minutes), select it from the list of devices

©2018 Microsoft Corporation



12. Copy the **Connection string (primary key)** and save it in a file. You'll need it for the upcoming section to register your device.



Connect Simulated Device

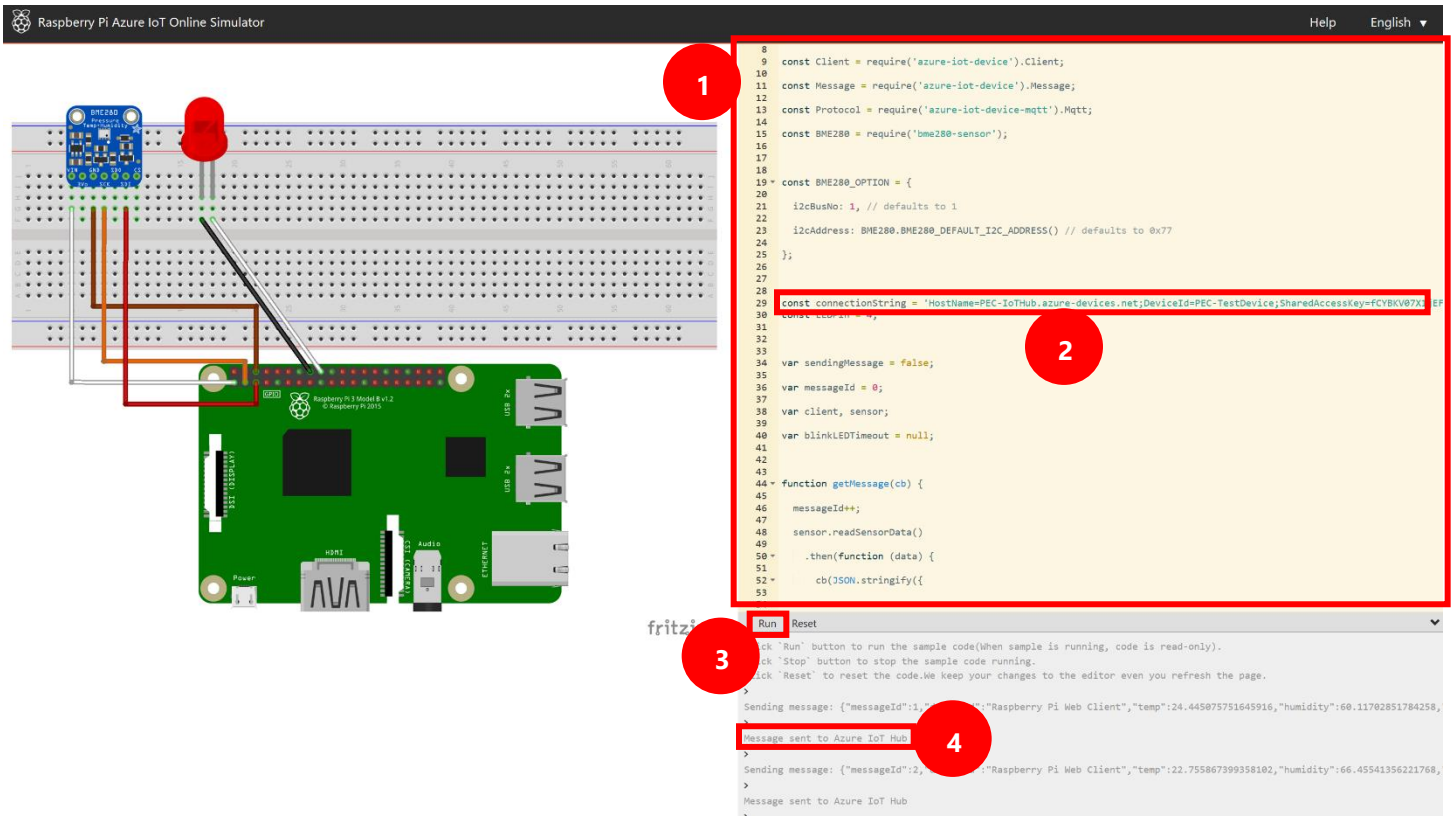
13. You'll be using a simulated Raspberry Pi as the device connecting into IoT Hub. You can access the simulator here: https://azure-samples.github.io/raspberry-pi-web-simulator/#Getstarted. This Raspberry Pi will send several measurements (Temperature, Humidity and Pressure) to the IoT Hub.

14. Copy the code from the file linked here and replace the code in the Raspberry Pi Web Simulator.

15. You'll now need to add in the Device Connection String copied earlier. Paste that string into the code by replacing the *[Your IoT hub device connection string]* text.

16. Click **Run** on your Raspberry Pi Simulator. You should see the messages printed on the log screen and confirmation that the message has been sent to IoT Hub.

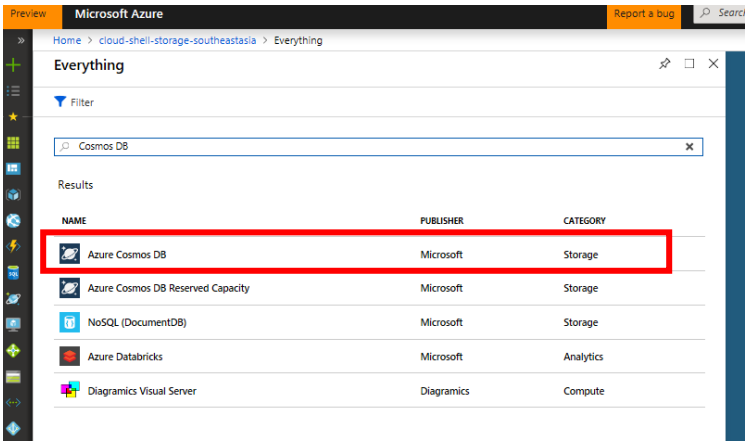
 ©2018 Microsoft Corporation


Raspberry Pi Azure IoT Online Simulator
Help
English
1
2
3
4
Run
Reset
Message sent to Azure IoT Hub
Message sent to Azure IoT Hub

Exercise 2 - Create a Cosmos DB Instance

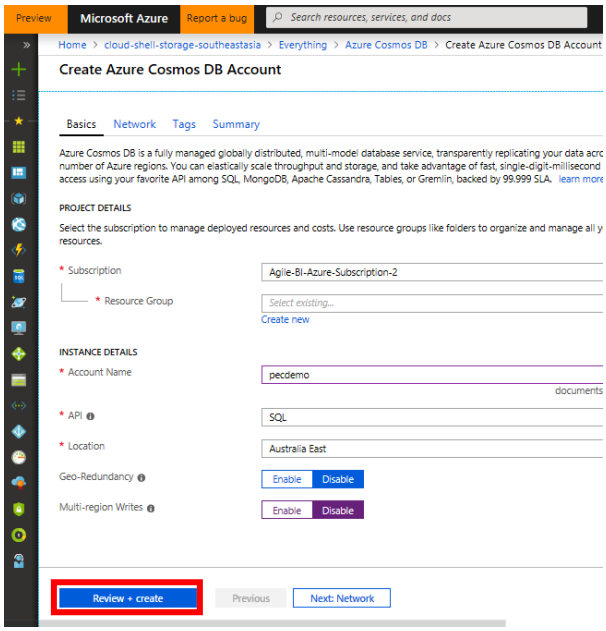
1. Select your resource group and then click on **Add** to create a new resource.

2. Search for **Cosmos DB** and select the resource from the results.



3. In the **Basics** tab, fill in the following information
   a. *Subscription*: Select the Subscription you've been using throughout this exercise
   b. *Resource Group:* Select the Resource group you used to deploy your other resources
   c. *Account Name*: This will be used to reference you Cosmos DB account
   d. *API*: This determines the data model used in your Cosmos DB instance. For this exercise, select **SQL**
   e. *Location:* Select the region where your service will be deployed
   f. *Geo-Redundancy*: Set to **Disable** for this exercise.
   g. *Multi-region Writes:* Set to **Disable** for this exercise.

   We won't be changing any values in subsequent tabs, so select **Review + create** when complete.



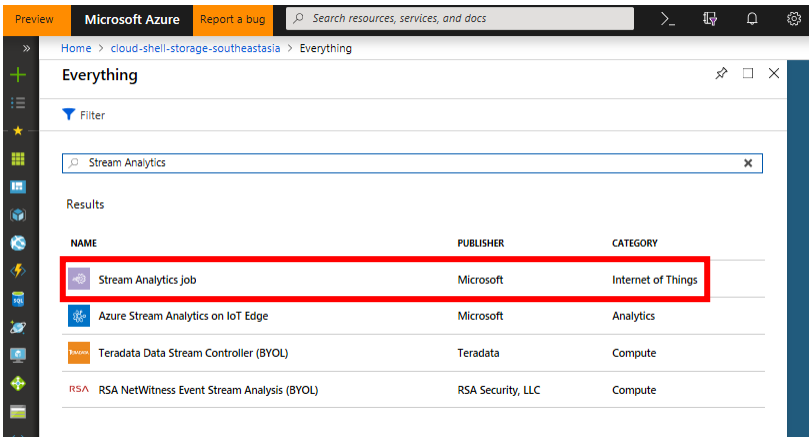
4. You will be notified once the service has been provisioned. That's it!

©2018 Microsoft Corporation

Exercise 3 - Create Stream Analytics Job

1. Select your resource group and then click on **Add** to create a new resource.

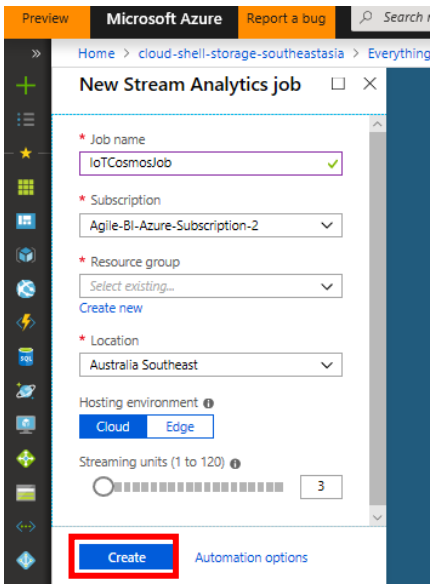
2. In the search box, type **Stream Analytics** and then select it from the list of results.



3. In the blade that opens on the right, click **Create**.

4. Fill out the form with the following values:
   a. *Job name:* Give your job a unique and descriptive name
   b. *Subscription:* Select the Subscription you've been using throughout this exercise
   c. *Resource Group:* Select the Resource group you used to deploy your other resources
   d. *Location:* Select the same location as your other resources
   e. *Hosting environment:* Select **Cloud**

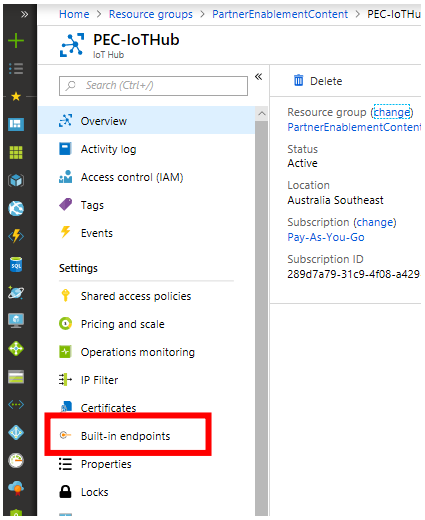
   Click **Create** when you've input all the required information.



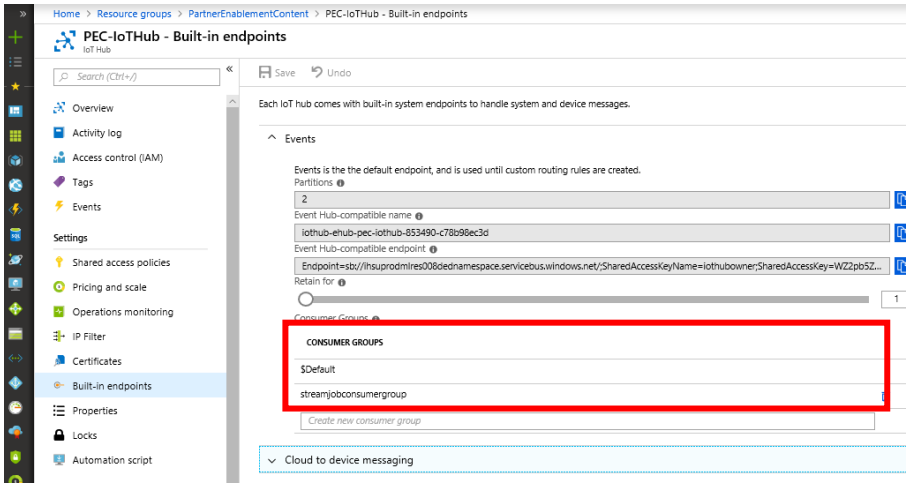
Add IoT Hub as input to Stream Analytics

1. Return to your resource group and select the **IoT Hub** resource you created previously.

2. In the left navigation area of the IoT Hub blade under **Settings**, select **Built-in Endpoints**.



3. Under **Events**, enter ***streamjobconsumergroup*** in the text input under **Consumer Groups**. This will create a new consumer group with that name. Select anywhere outside the text box to save.



4. Return to your Resource Group and select the **Stream Analytics job** you created in the previous section.

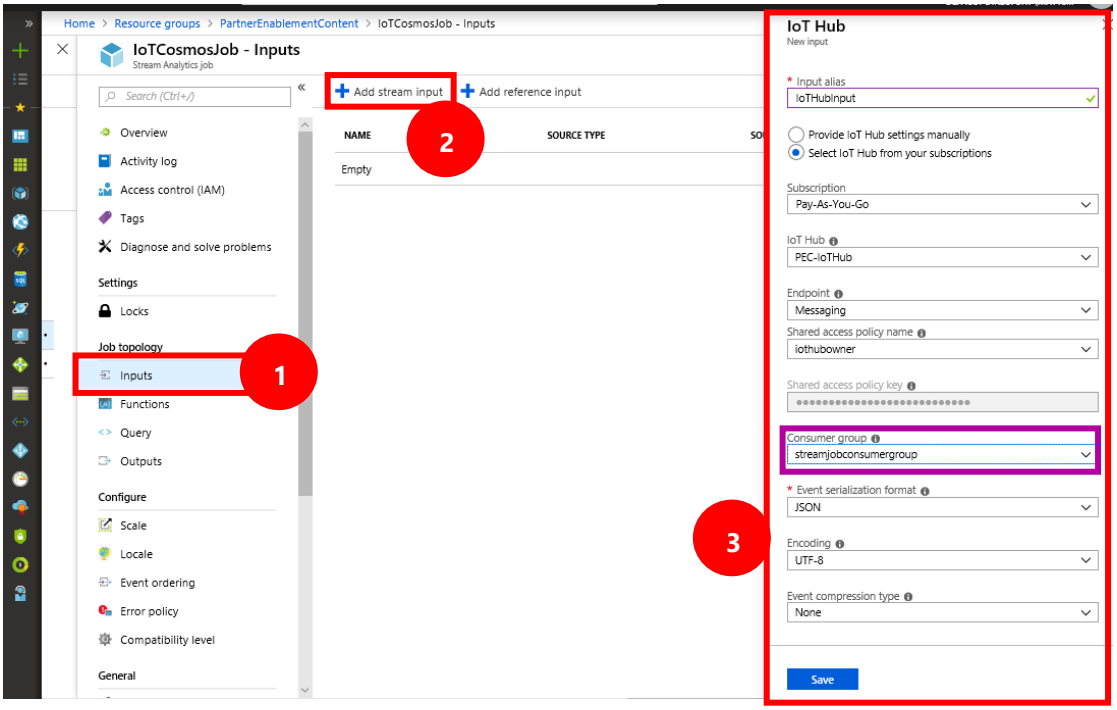
5. In the left sidebar under **Job Topology**, select **Inputs**

6. Click on **Add Stream Input** and select **IoT Hub** from the dropdown.

7. Enter *iothubinput* for the **Input alias**, then **click** "Select IoT Hub from your subscriptions"

©2018 Microsoft Corporation

8. Select your active subscription and the IoT Hub you provisioned previously. All other details should populate automatically. Make sure to select the newly create consumer group (streamjobconsumergroup) from the dropdown.

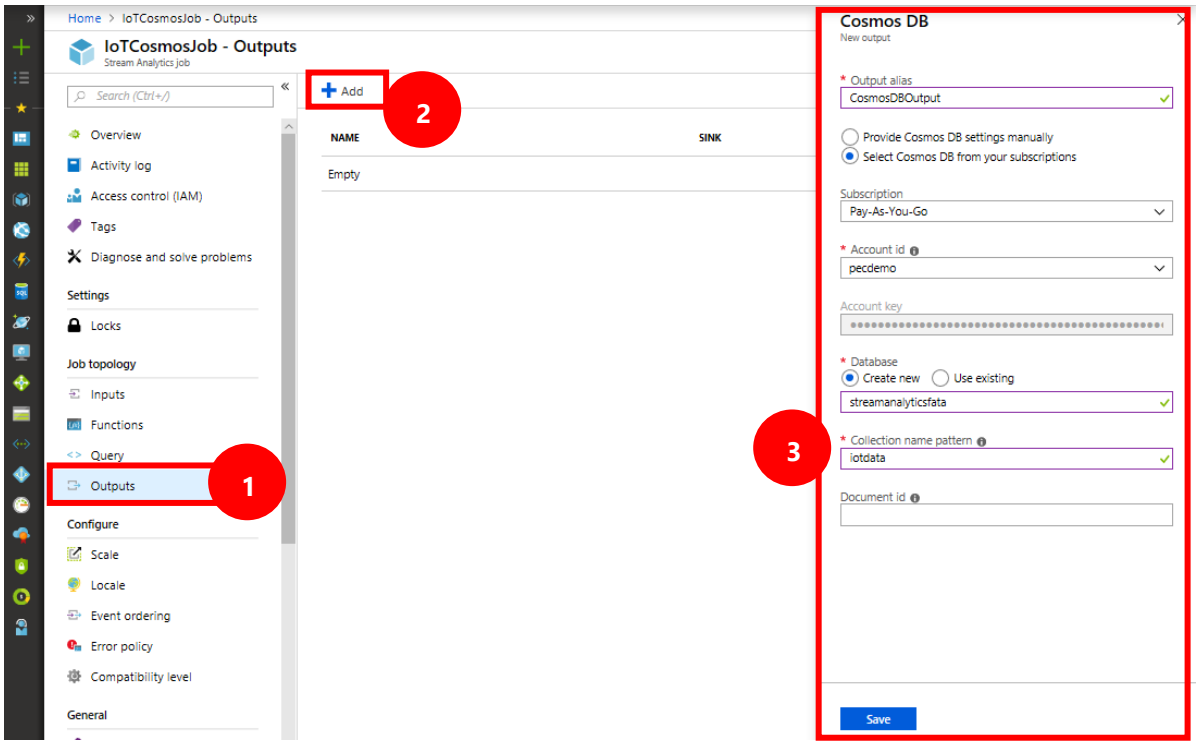


9. **Save** the input.

Output Stream Analytics into Cosmos DB

1. Under the same **Stream Analytics job**, select **Outputs** in the sidebar.
2. Click on **Add** in the top bar and select **Cosmos DB** from the dropdown.
3. Choose the "Select Cosmos DB from your subscriptions options, then enter the following information:
    a. *Output alias:* CosmosDBOutput
    b. *Account id:* the existing Cosmos DB account provisioned previously
    c. *Database:* Create New
    d. *Database Name:* streamanalyticsdata
    e. *Collection Name Pattern:* iotdata
4. **Save** the output.

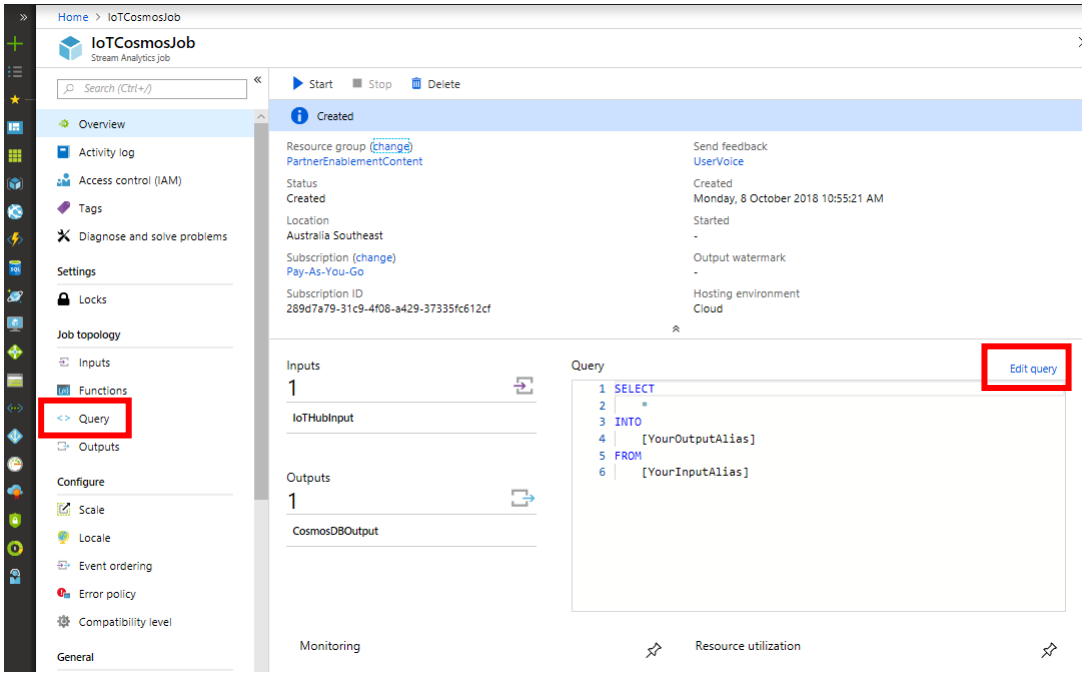
©2018 Microsoft Corporation



5. In the left navigation of your streaming job, click on **Overview**.

Updating the streaming jobs query

1. On the overview page of your streaming analytics job, click on **Edit query**, or select **Query** from the left navigation.



 ©2018 Microsoft Corporation

2. **Update** the query

```
SELECT
    deviceId,
    AVG(temperature) AS avgTemperature,
    AVG(pressure) AS avgPressure,
    AVG(humidity) AS avgHumidity
INTO
    CosmosDBOutput
FROM
    IoTHubInput TIMESTAMP BY EventEnqueuedUtcTime
GROUP BY
    deviceId,
    TUMBLINGWINDOW(second, 60)
```

3. At the top, press **Start** to start the streaming job, then **Start** again in the pop-up blade on the right. This may take a few minutes to start.

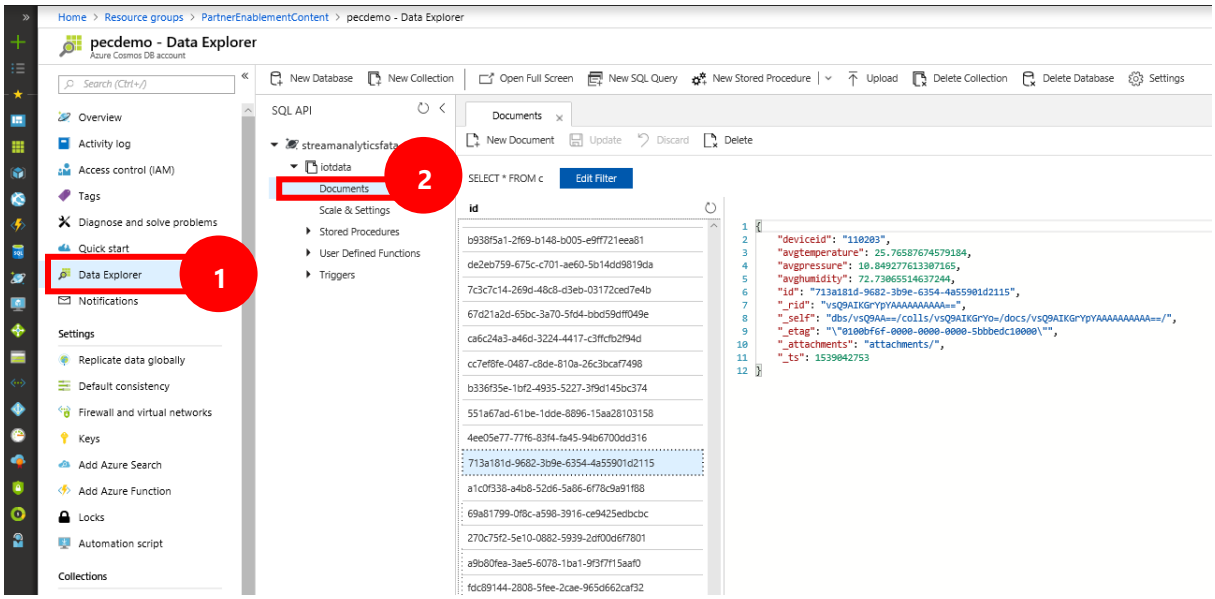
Confirm Data in Cosmos DB

1. Go back to your resource group and click on your **Cosmos DB** instance.

2. In the left navigation, click on **Data Explorer**.

3. Under the database created in the previous section **streamanalyticsdata,** click the collection previous created (**iotdata**), then select **Documents**

4. You should see a list of Documents appear, each with a unique ID. Select any Document to see the contents.

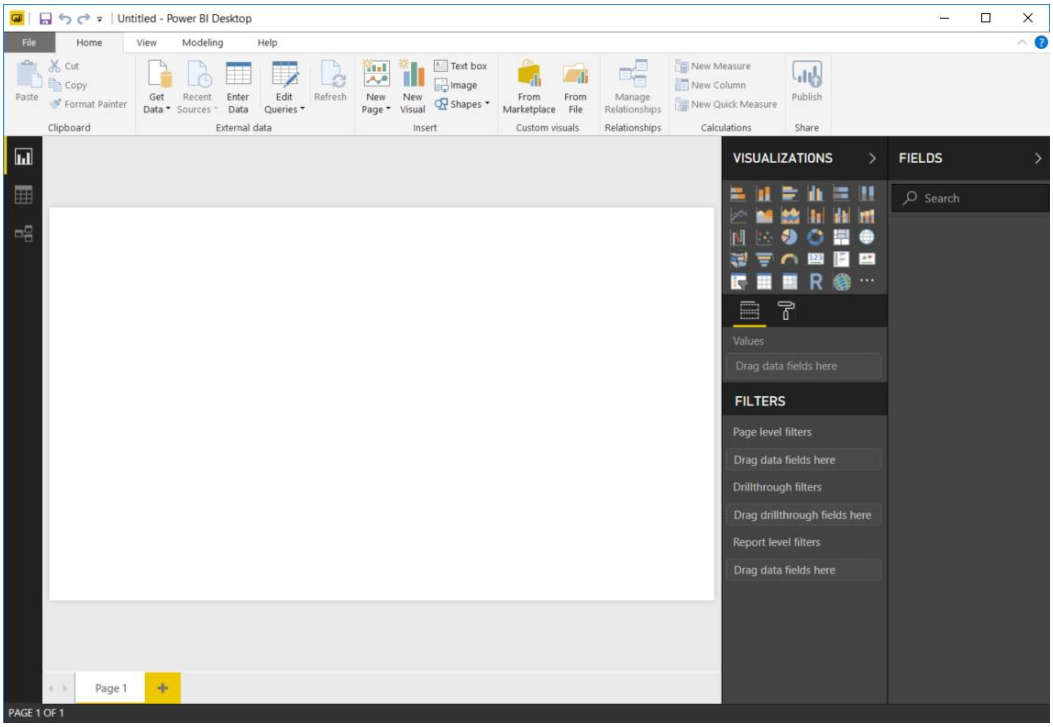


Exercise 4 – Visualisation in PowerBI

(https://docs.microsoft.com/en-us/azure/cosmos-db/powerbi-visualize)

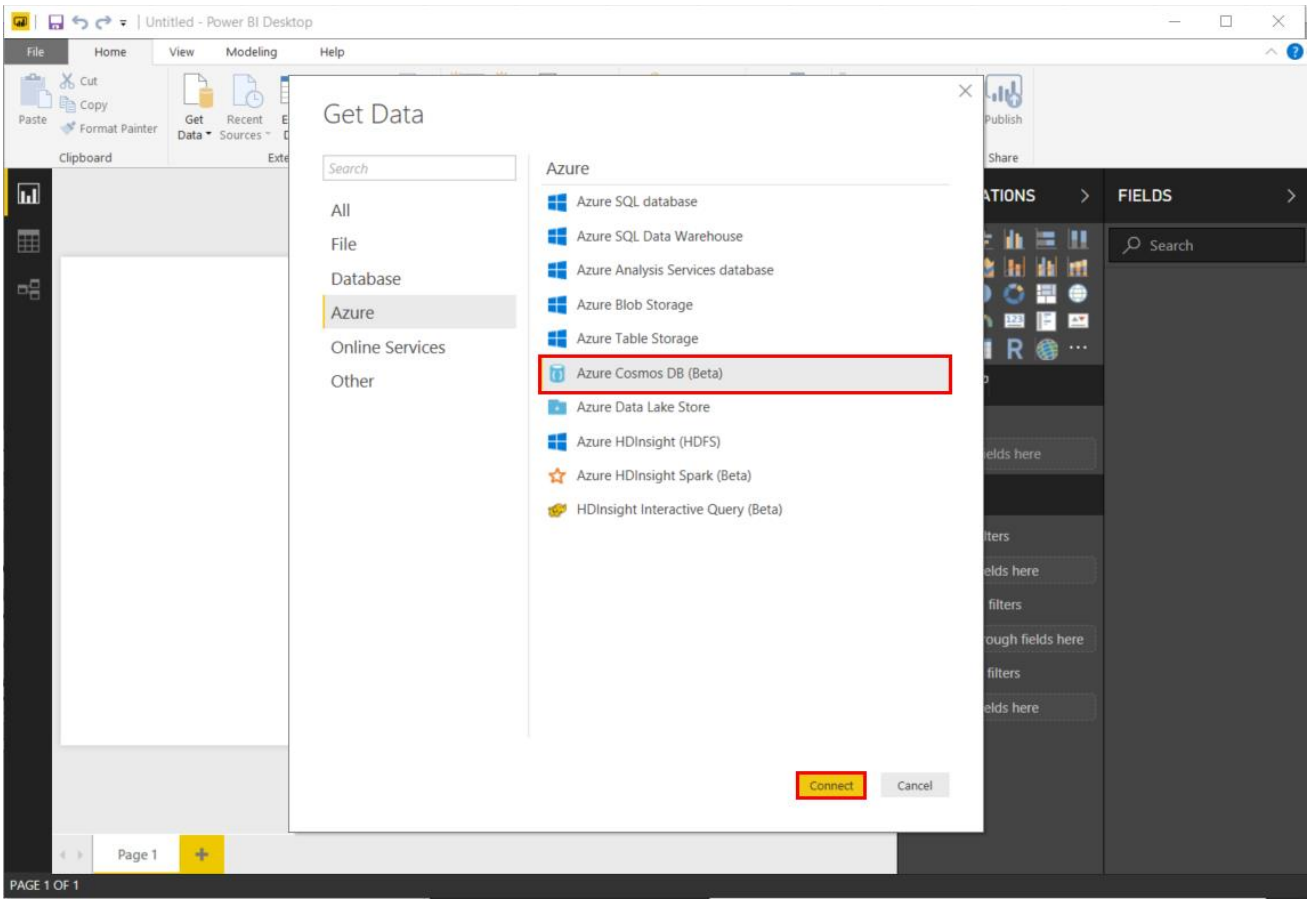
In this section, we'll be demonstrating PowerBI Desktop to a visualisation your CosmosDB account.

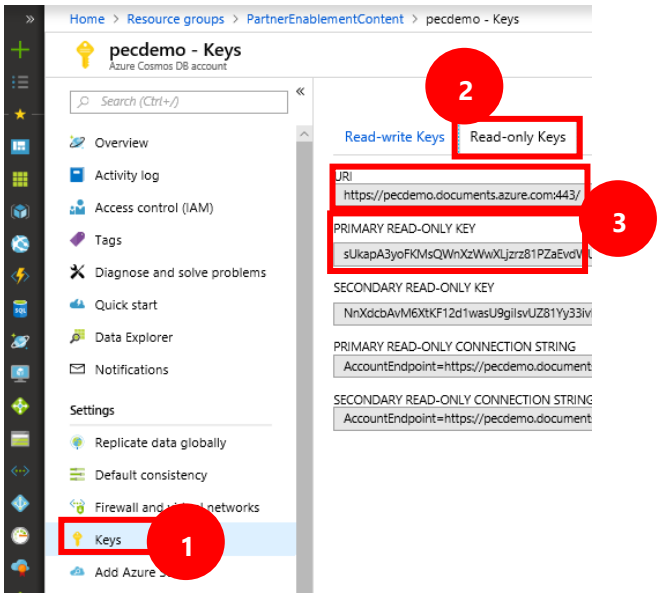
1. Download the latest version of Power BI Desktop
2. Open the application and dismiss the pop-up screen. You should now see the **Report** view of Power BI.

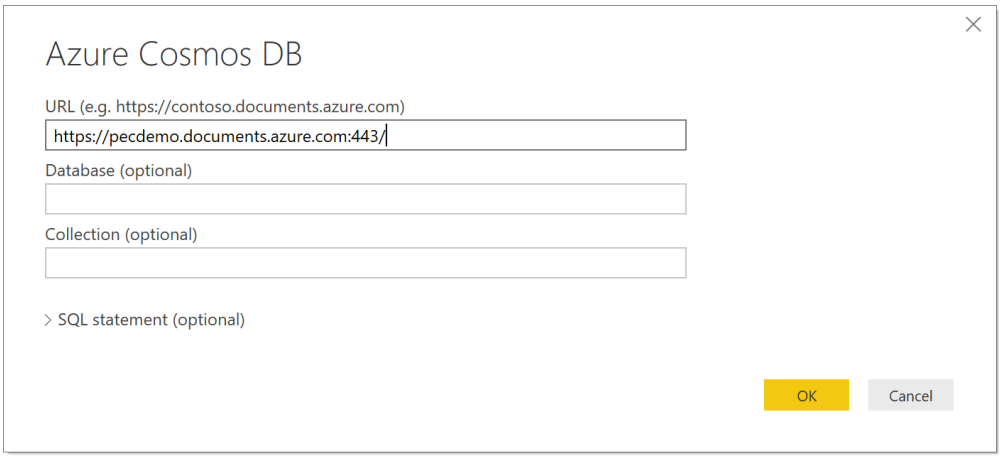


3. Select the Home ribbon, then click on Get Data. The Get Data window should appear.
4. Click on Azure, select Azure Cosmos DB (Beta), and then click Connect.

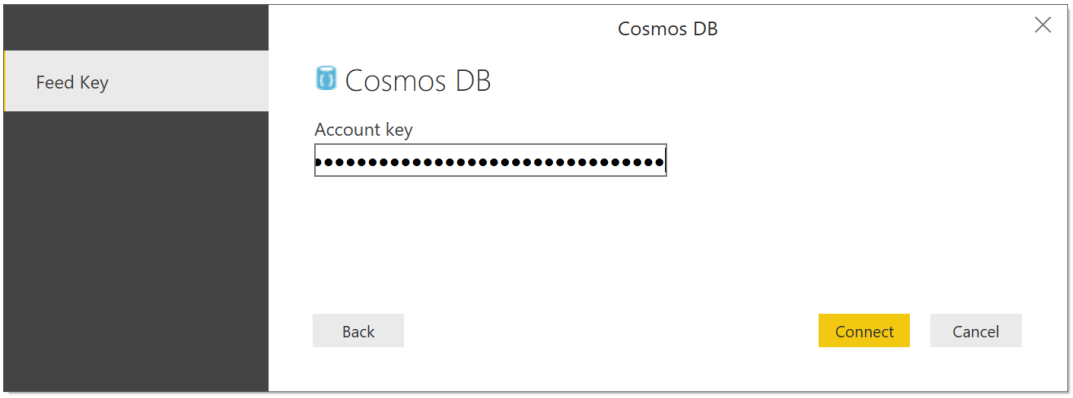
©2018 Microsoft Corporation



5. On the **Preview Connector** page, click **Continue**. The Azure Cosmos DB window appears.
6. Specify the Azure Cosmos DB account endpoint URL you would like to retrieve the data from as shown below, and then click **OK**. You can find the **URI** and **Keys** under the Cosmos DB you deployed in the Azure Portal [Make sure to select the *Read-only Keys* tab].
7. 

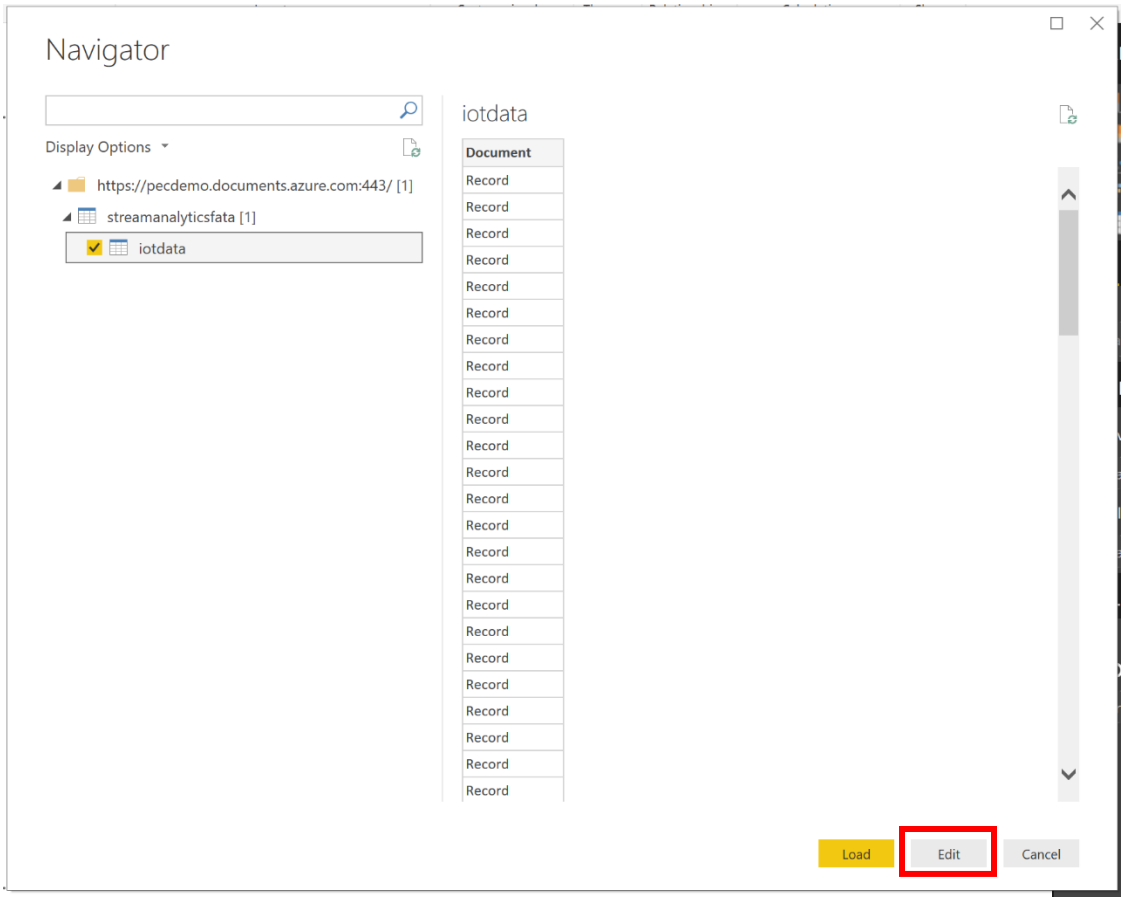


8. You'll then be prompted for the Account Key. Copy the key you found in the Azure portal in Step 6. Again make sure you're using the *Read-only Keys*. Past the **Primary Key** into the Account Key box.



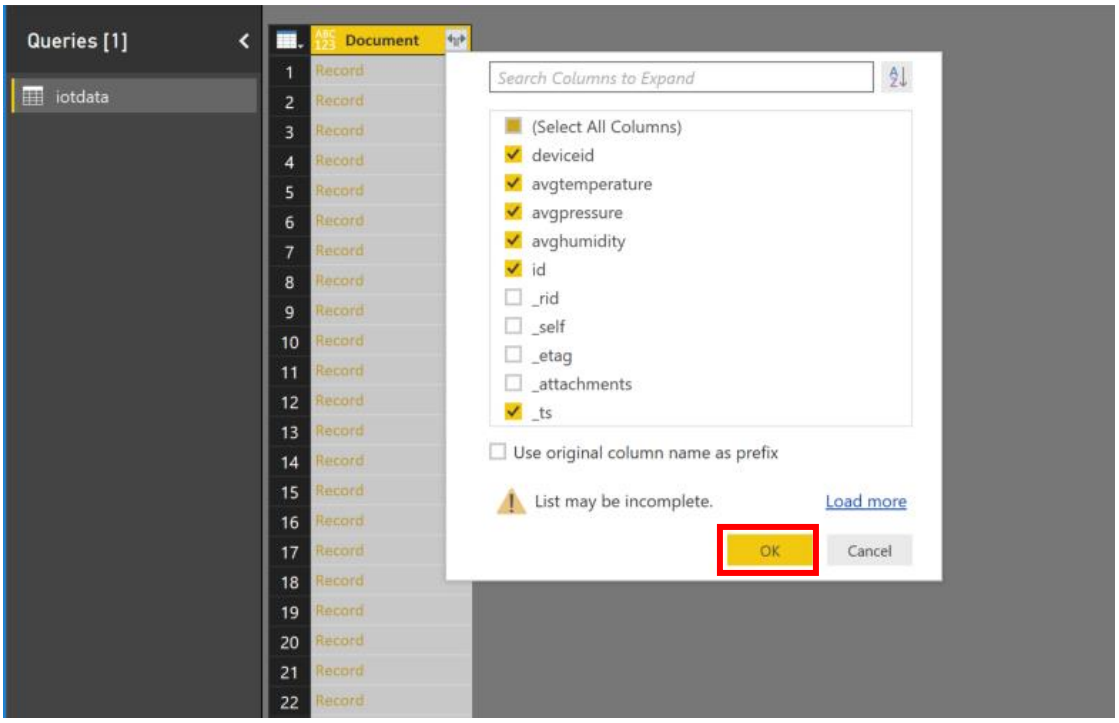
9. When the account is successfully connected, the **Navigator** pane appears. The **Navigator** shows a list of databases under the account.
10. Click and expand on the database where the data for the report comes from.
11. Now, select a collection that contains the data to retrieve.

©2018 Microsoft Corporation



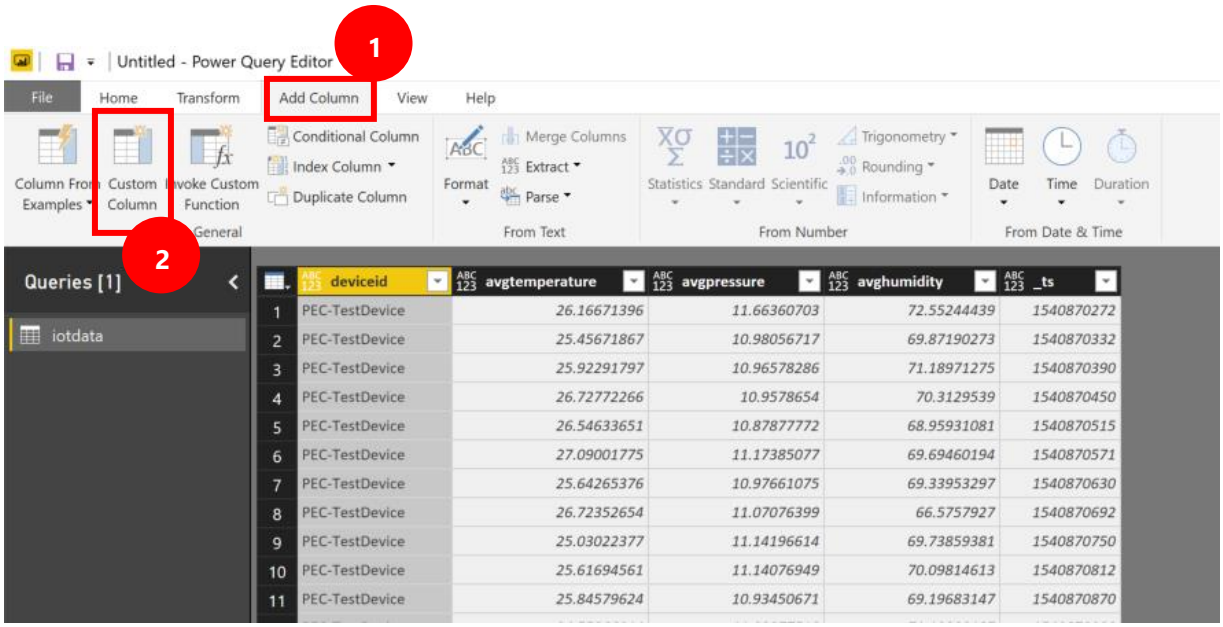
12. Click **Edit** to launch the Query Editor in a new window to transform the data.
13. Click on the expander at the right side of the Document column header. The context menu with a list of fields will appear. Select the fields you need for your report:
    1. *deviceID*
    2. avgtemperature
    3. avgpressure
    4. avghumidity
    5. *_ts*
14. Uncheck the Use original column name as prefix box, and then click OK.

©2018 Microsoft Corporation



Create Custom Columns

15. You'll need to convert *_ts* from Unix time to UTC. To do this, we'll create a **custom column**. Select the **add column** tab at the top of the screen.

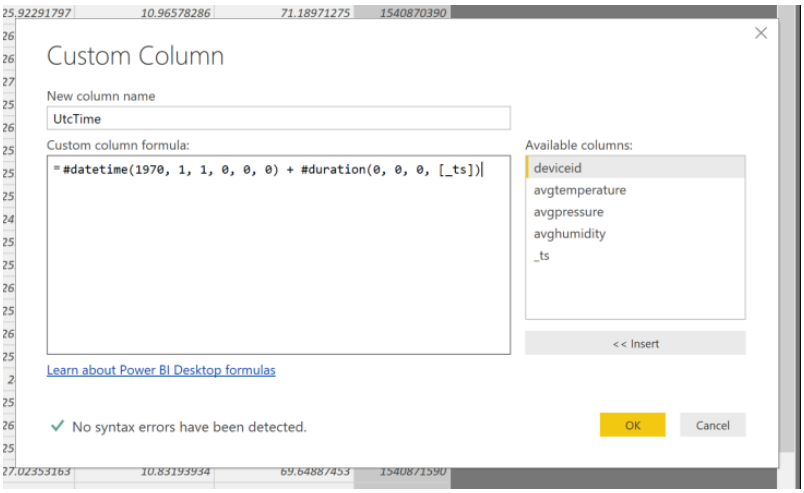


16. In the pop-up window, name your new column *UtcTime* and enter the following formula:

```
#datetime(1970, 1, 1, 0, 0, 0) + #duration(0, 0, 0, [_ts])
```


©2018 Microsoft Corporation

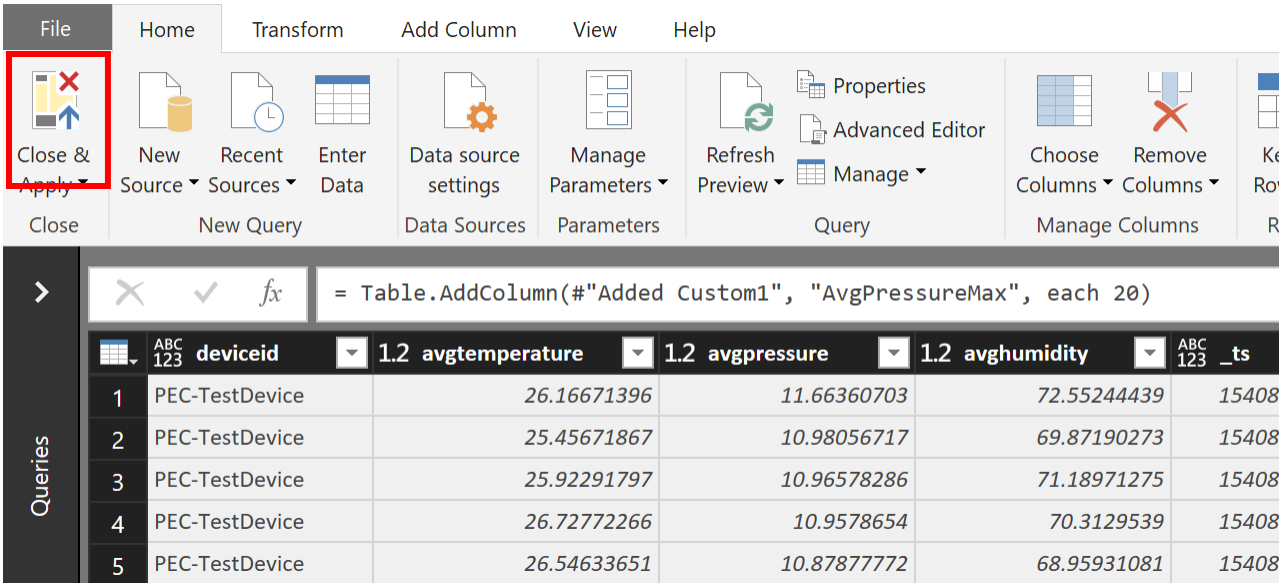
17. Select ok to finalise.



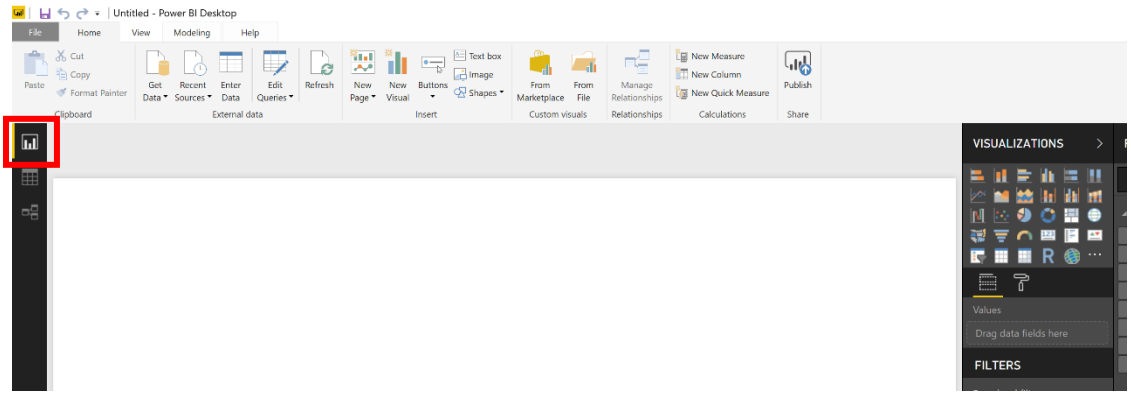
18. With the new column added, we'll need to state the data types for each column. Select the small data icon next to each of the column labels and select the correct data type as follows:
    1. *Deviceid (ignore)*
    2. Avgtemperature -> Decimal Number
    3. Avgpressure -> Decimal Number
    4. Avghumidity -> Decimal Number
    5. *_ts (ignore)*
    6. UtcTime -> Date/Time
19. Finally, we'll add two new columns for our dashboard. Create two more **custom columns** with the following details:
    1. Name: *AvgPressureMin*; Value: *0*
    2. Name: *AvgPressureMax*; Value: *20*
    3. Remember to change the data types to "decimal"
20. With all the columns added, select the **Home** tab, then **Close and Apply**

	deviceid	avgtemperature	avgpressure	avghumidity	_ts	UtcTime	avgPressureMin	AvgPressureMax
1	PEC-TestDevice	26.16671396	11.66360703	72.55244439	1540870272	30/10/2018 3:31:12 AM	0	20
2	PEC-TestDevice	25.45671867	10.98056717	69.87190273	1540870332	30/10/2018 3:32:12 AM	0	20
3	PEC-TestDevice	25.92291797	10.96578286	71.18971275	1540870390	30/10/2018 3:33:10 AM	0	20
4	PEC-TestDevice	26.72772266	10.9578654	70.3129539	1540870450	30/10/2018 3:34:10 AM	0	20
5	PEC-TestDevice	26.54633651	10.87877772	68.95931081	1540870515	30/10/2018 3:35:15 AM	0	20

©2018 Microsoft Corporation



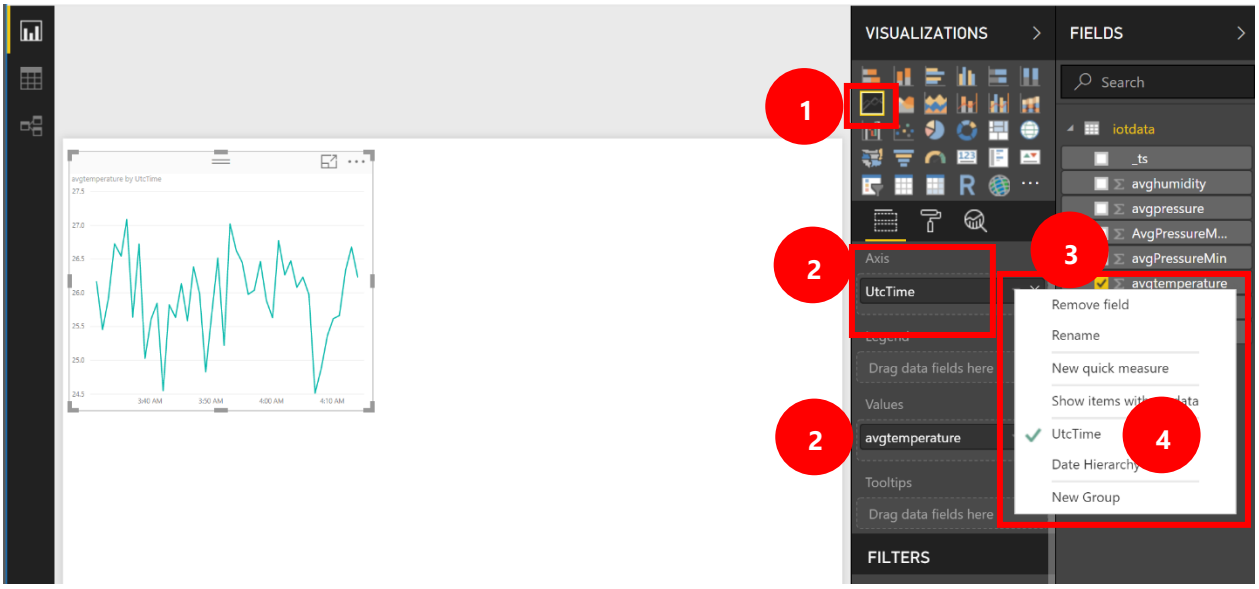
21. You should now see a blank workspace. If not, select the “Report” tab on the left of the screen.



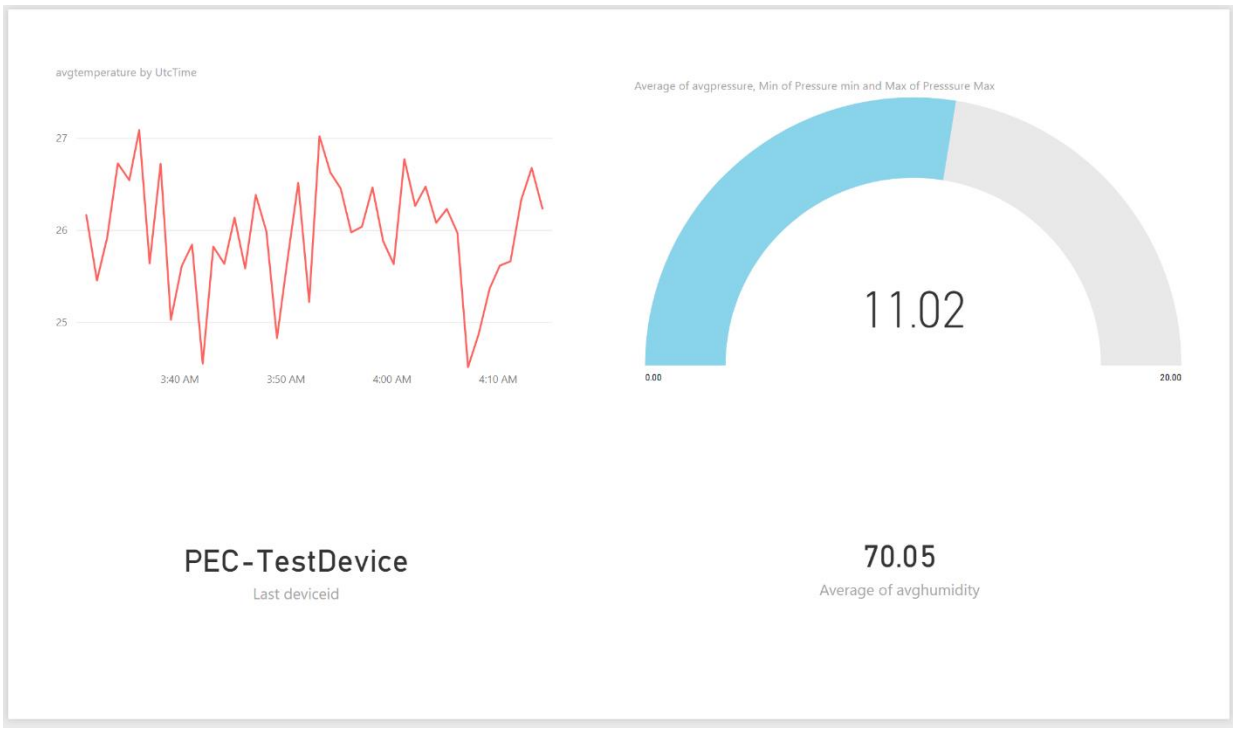
Build your report

1. Select the line graph icon and click anywhere on the white page to create a tile.
2. Click and drag **UtcTime** and **AvgTemperature** into the **Axis** and **Values** field respectively
3. Select the down arrow next to **UtcTime** and select *UtcTime* from the dropdown menu. You should see the line graph populate with varying values.

©2018 Microsoft Corporation



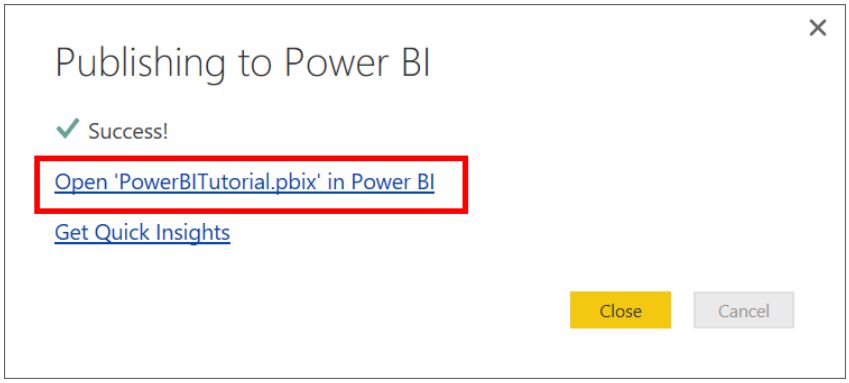
4. Experiment with the different visualisation options to build your graph. Try using the AvgPressureMin and AvgPressureMax values to build a gauge tile.



Publish and share your report

Note: To share your report, you must have an account in PowerBI.com.

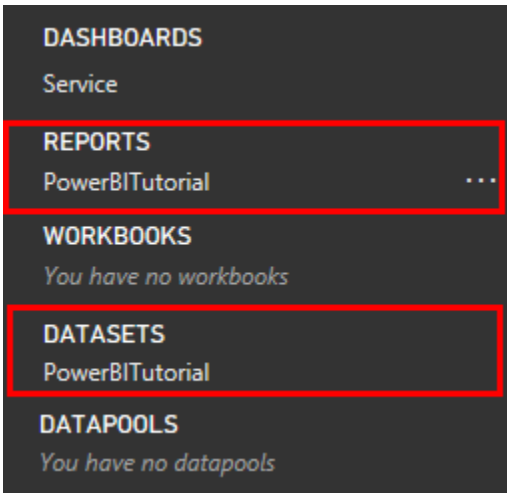
1. In the Power BI Desktop, click on the Home ribbon.
2. Click Publish. You are be prompted to enter the user name and password for your PowerBI.com account.
3. Once the credential has been authenticated, the report is published to your destination you selected.
4. Click **Open 'PowerBITutorial.pbix' in Power BI** to see and share your report on PowerBI.com.



CREATE A DASHBOARD IN POWERBI.COM

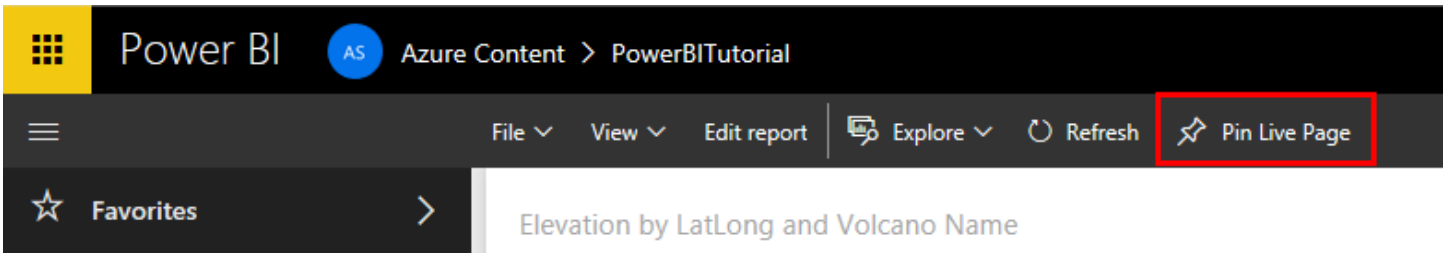
Now that you have a report, lets share it on PowerBI.com

When you publish your report from Power BI Desktop to PowerBI.com, it generates a **Report** and a **Dataset** in your PowerBI.com tenant. For example, after you published a report called **PowerBITutorial** to PowerBI.com, you will see PowerBITutorial in both the **Reports** and **Datasets** sections on PowerBI.com.



©2018 Microsoft Corporation

To create a sharable dashboard, click the **Pin Live Page** button on your PowerBI.com report.



Then follow the instructions in [Pin a tile from a report] to create a new dashboard.

You can also do ad hoc modifications to report before creating a dashboard. However, it's recommended that you use Power BI Desktop to perform the modifications and republish the report to PowerBI.com.

This completes Lab 2 – Warm Path.

Lab 3 – Cold Path

Summary and Objectives

In this lab we will be storing the raw device telemetry messages from IoT Hub in Blob storage using the "routing" feature of IoT Hub. Then we use Azure Databricks to perform ETL activities and ultimately make the data available to PowerBI for reporting and visualisation.

The following Azure tools and technologies will be covered in this lab:

1. IoT Hub routing and custom endpoints to store raw device telemetry data
2. Azure Databricks (PySpark and SparkSql) to preform ETL on the raw data
3. Azure Databricks tables to store transformed data
4. PowerBI Spark connector to query the transformed data using Azure Databricks.

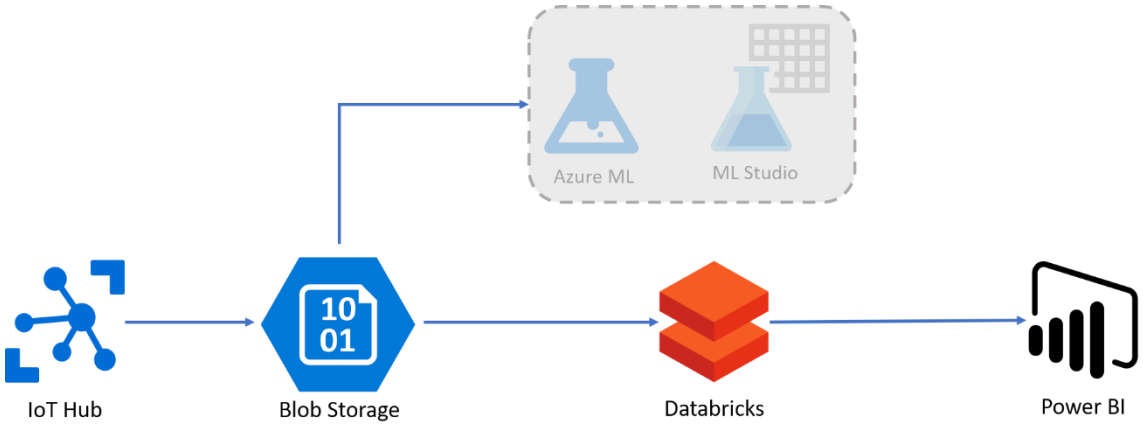


Figure 4 - Cold Storage and Machine Learning

©2018 Microsoft Corporation

Task 1 – Route messages from IoT Hub to Blob Storage

Duration: 15 Minutes

In this task we introduce the concept of routing in IoT Hub. Also, we set up a custom endpoint for Azure Blob storage to store raw message for further processing in cold path.

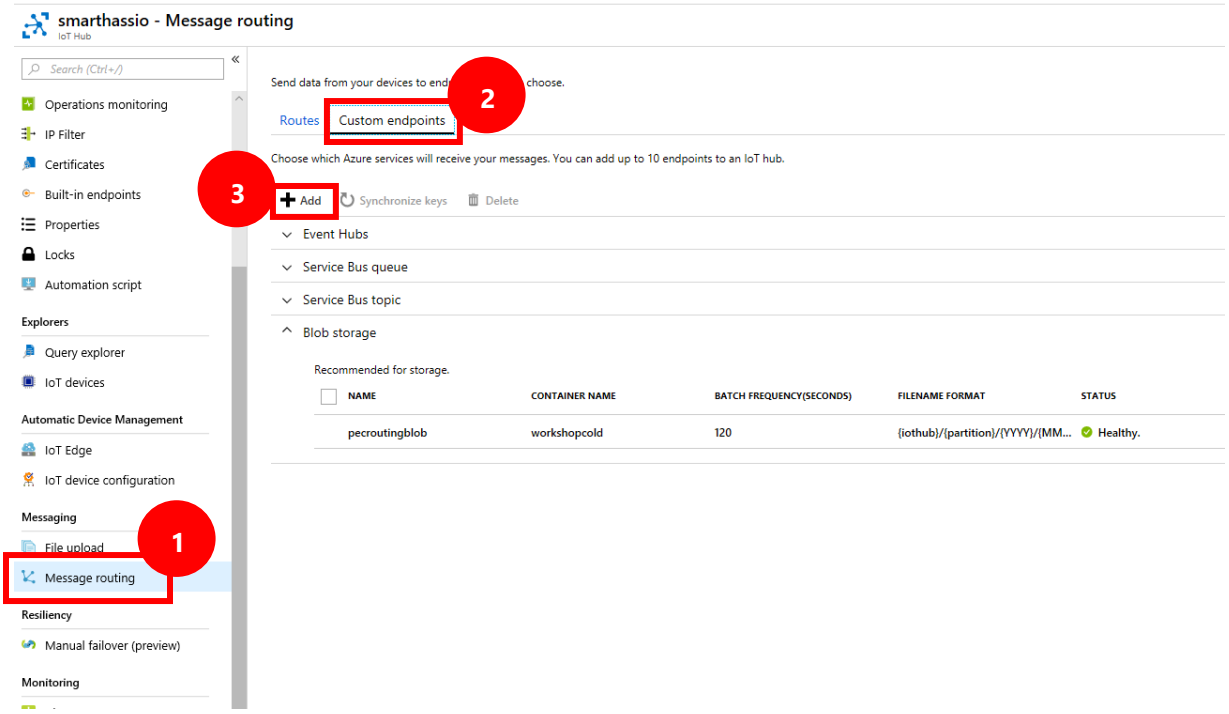
Message routing enables sending telemetry data from your IoT devices to built-in Event Hub-compatible endpoints or custom endpoints such as blob storage, Service Bus Queue, Service Bus Topic, and Event Hubs. While configuring message routing, you can create routing queries to customize the route that matches a certain condition. Once set up, the incoming data is automatically routed to the endpoints by the IoT Hub.

To learn more about IoT hub message routing refer to: https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-devguide-messages-d2c

An IoT hub has a default built-in-endpoint (**messages/events**) that is compatible with Event Hubs. You can create custom endpoints to route messages to by linking other services in your subscription to the IoT Hub. In this task we setup a custom endpoint for Azure Blob Storage.

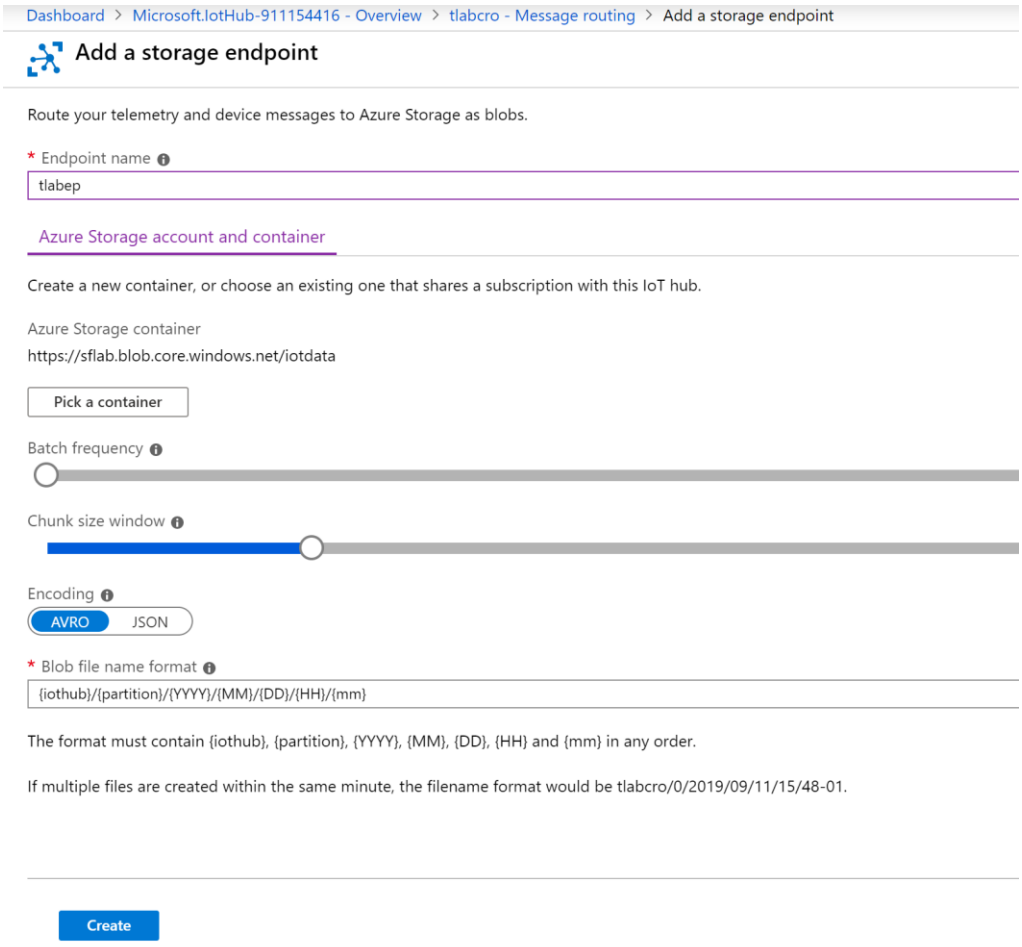
©2018 Microsoft Corporation

1. Create a blob storage custom endpoint:
    a. In your IoT Hub go to “Message routing”
    b. On the message routing tab click on “Custom endpoints”
    c. Click “Add”

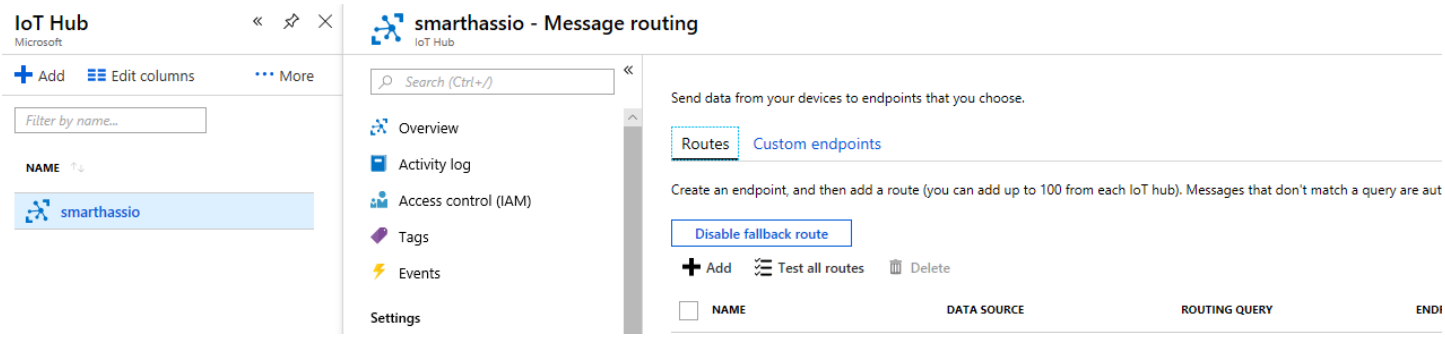


    d. In the new open window type in a name for the endpoint
    e. Select a Blob storage account and container using “Pick a Container” button. (Note: If you have a big number of storage accounts and containers make sure you make note of the location as you will require it next steps)
    f. Adjust the batch frequency to 60 seconds
    g. Leave the rest of the setting as is (including AVRO encoding).
    h. Click create to save and close

©2018 Microsoft Corporation

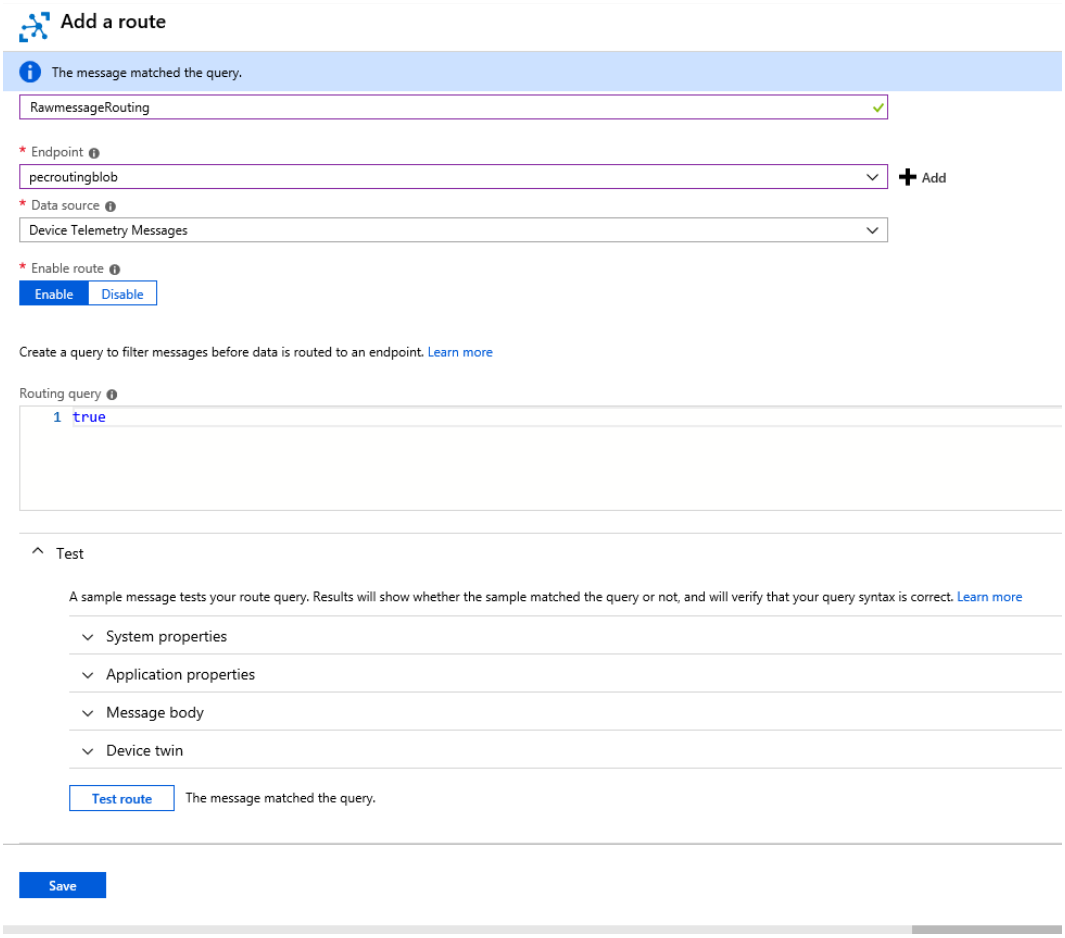


2. Setup message routing to the newly created Blob endpoint
    a. On the same screen go back to “Routes” tab
    b. Click add



    c. On the “Add Route” window type in a name for the new route
    d. Select the Blob endpoint you created in the previous step
    e. Select “Device Telemetry Messages” as data source
    f. Leave the rest of the field to default and click test at the bottom of the page
    g. Save and close

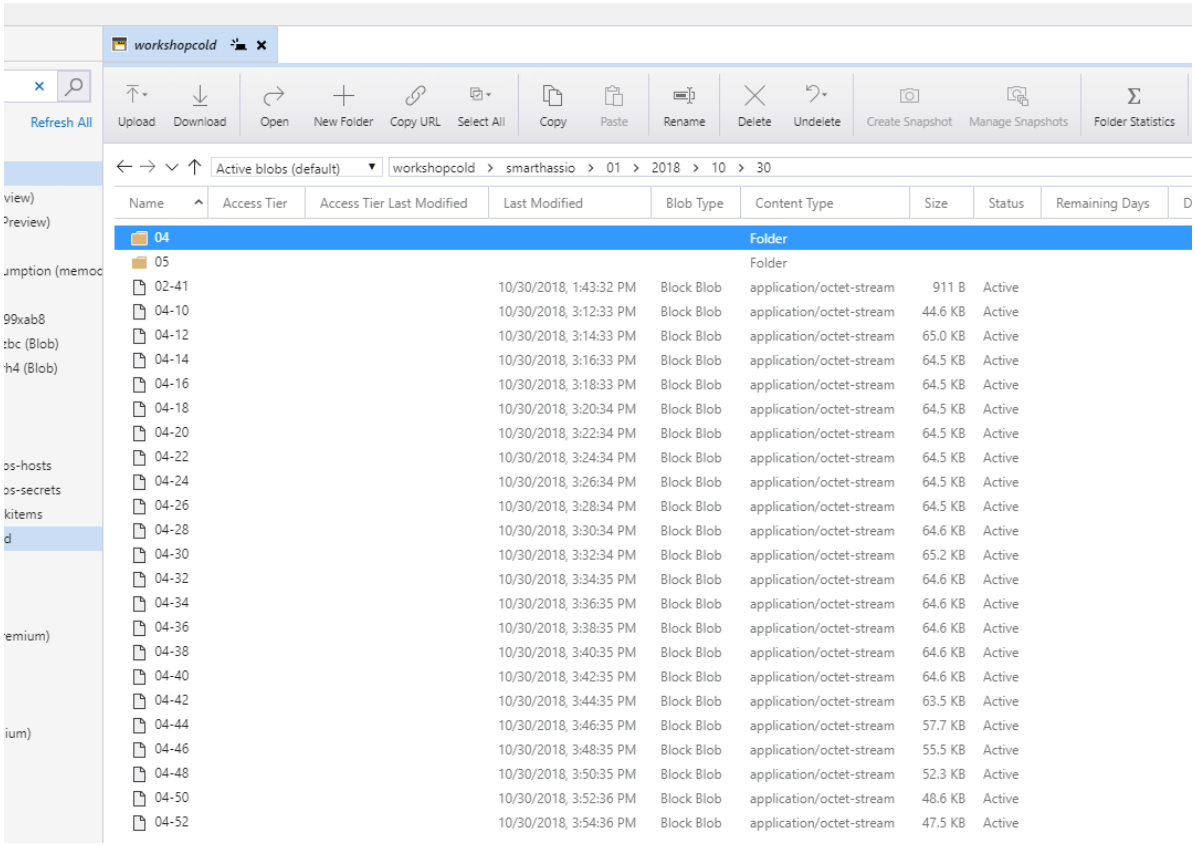
©2018 Microsoft Corporation



Note: We have not changed the routing query for this task as we are intending to route all telemetry messages to our endpoint as cold raw message storage. Message routing enables users to route different data types namely, device telemetry messages, device lifecycle events, and device twin change events to various endpoints. You can also apply rich queries to this data before routing it to receive the data that matters to you. This article describes the IoT Hub message routing query language, and provides some common query patterns. To learn more about this feature of IoT hub read more at: https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-devguide-routing-query-syntax

3. Verify the new IoT Hub Routes are created
    a. On the "Message routing" setting tab in IoT Hub verify that the new route is created correctly as per the instructions.
    b. Make sure your Raspberry Pi emulator is running
    c. Allow sometime for messages to accumulate in the Hub
    d. Go to your selected Blob storage and container using Azure Storage Explorer and confirm there are files being created as per the naming format in the custom endpoint

 ©2018 Microsoft Corporation



Task 2 – Create Azure Databricks workspace and cluster

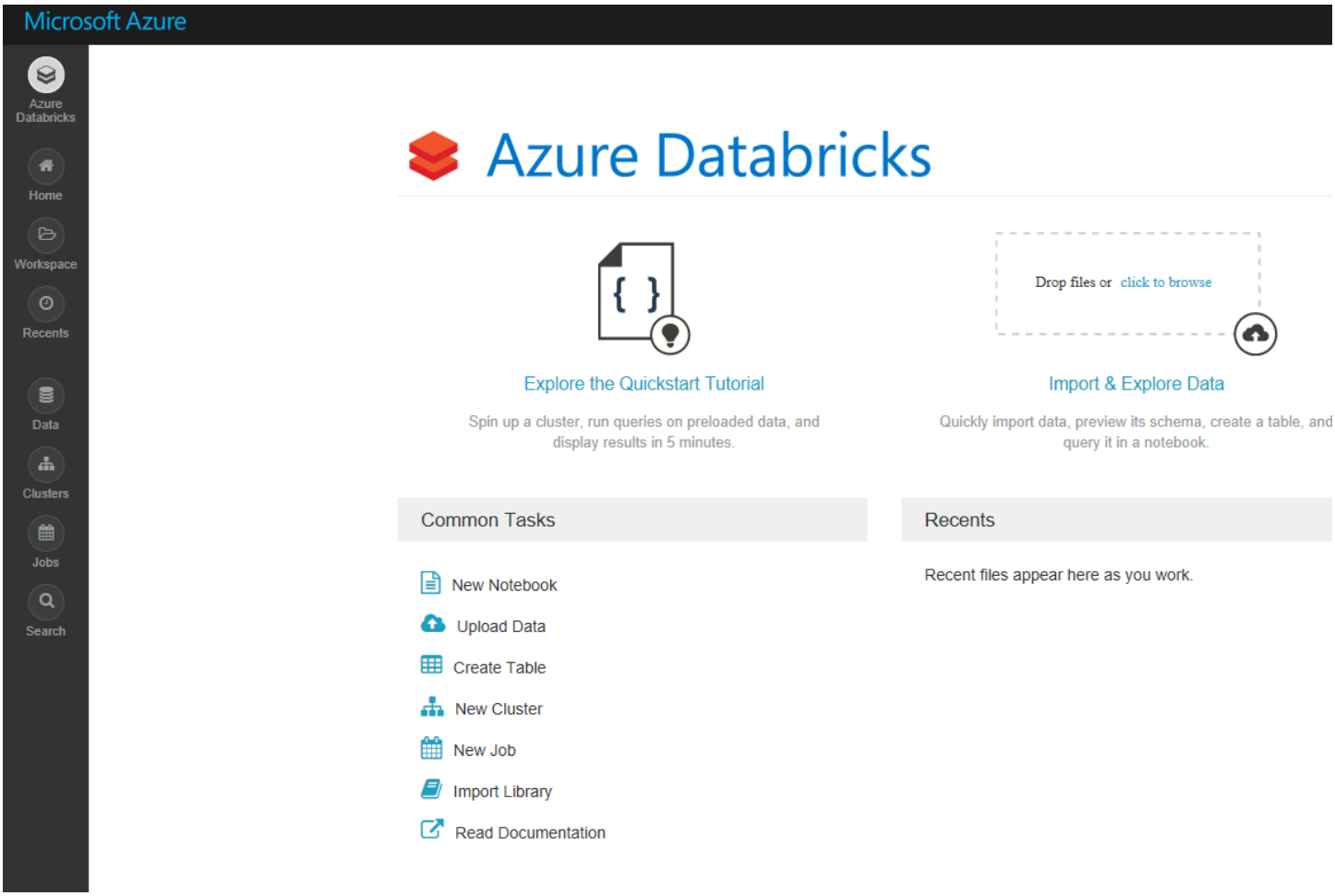
Using the portal, provision an Azure Databricks workspace or service.

The provisioning should be very straight forward, but remember to choose either “Premium or Trial (Premium)” in Pricing tier. The “standard tier” works fine for most the activities below except it lacks the feature for connecting to PowerBI.

Task 3 – Create Databricks cluster with PySpark and SparkSql

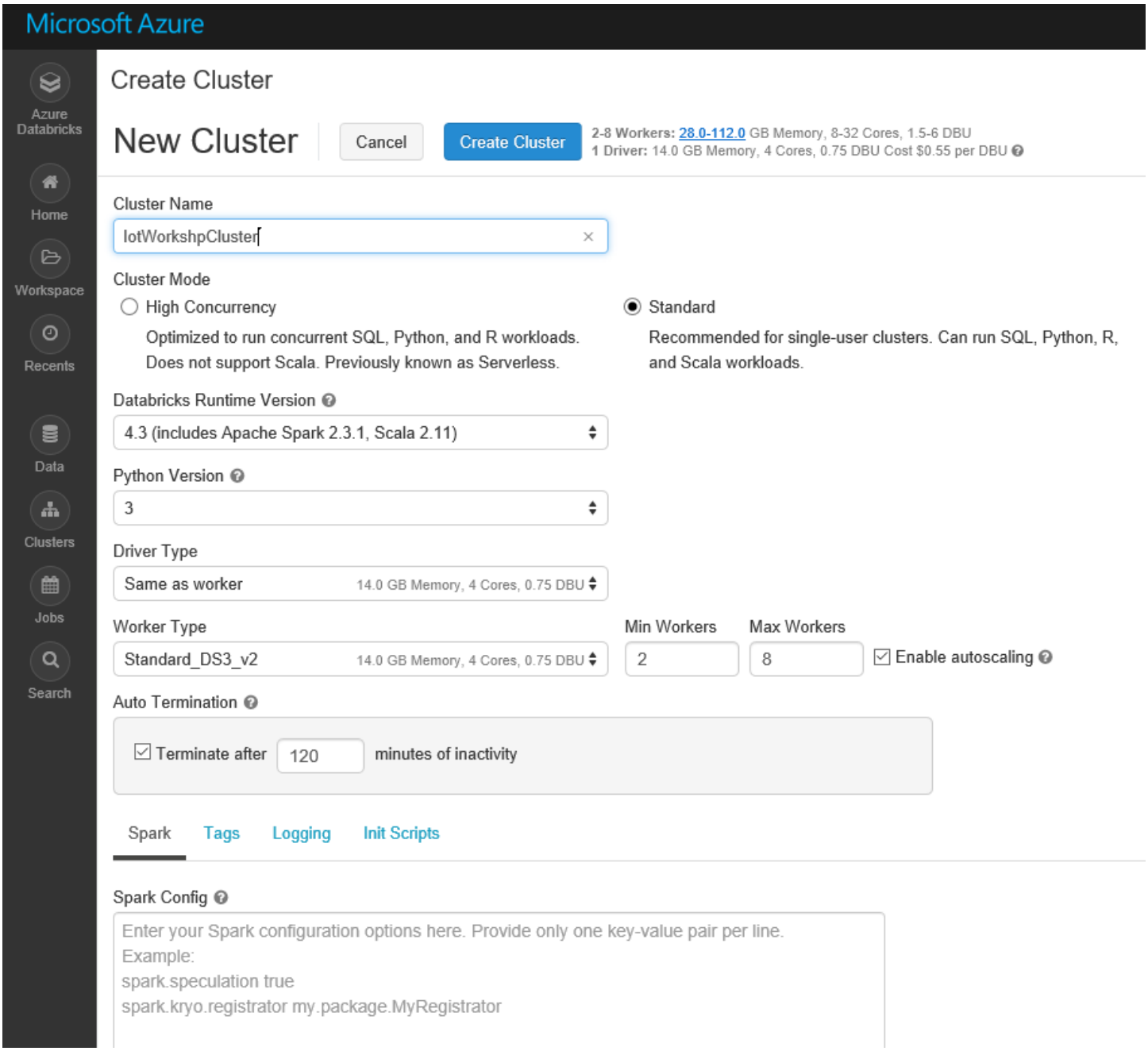
1. In your newly provisioned Azure Databricks service click “Launch Workspace” to open the control plane of Azure Databricks.

©2018 Microsoft Corporation



2. From the left-hand side pane click clusters to open the cluster page and then click "+create cluster" button to open the new cluster page.
3. In this step you can customise the Databricks (Spark) cluster that you'd need to provision.
    a) Enter a cluster name
    b) For cluster mode select "standard"
    c) Change the Python version to "3"
    d) [optional] change the virtual machine size for worker nodes and driver node OR leave as is.
    e) Click create to start the cluster provisioning process. (Note: It may take several minutes for the cluster to get provisioned)

 ©2018 Microsoft Corporation

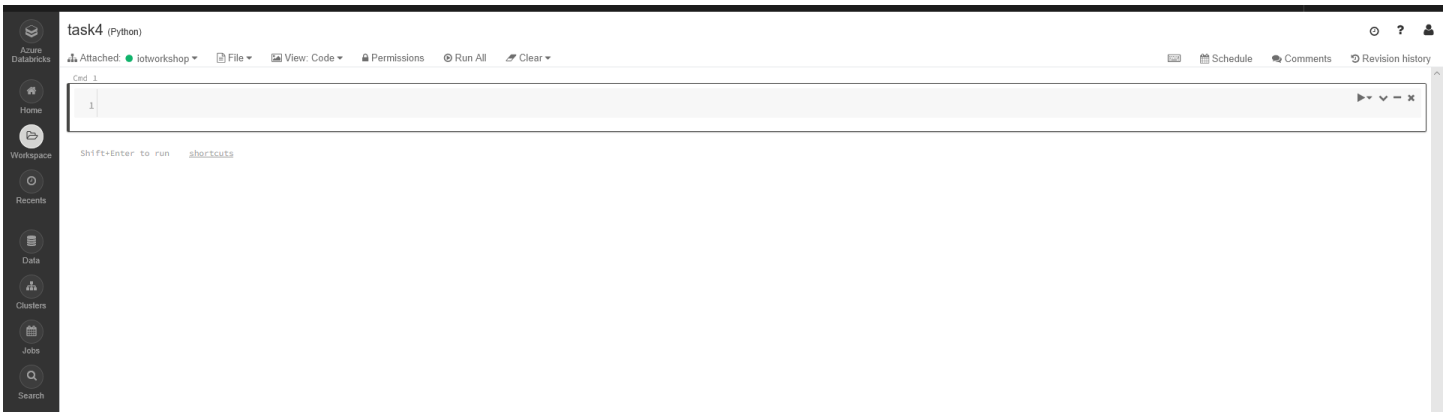


4. Once the cluster is provisioned from the "workspace" tab on left hand side of the screen create a new "Python" notebook and choose the newly created cluster. This will create a Databricks notebook which in many ways resembles Jupyter Notebooks. (If you are not familiar with Jupyter Notebooks, don't worry you will not need extensive knowledge of that to complete this task but for taking full advantage of Azure Databricks it is highly recommended to familiarise yourself with it.
5. In this step we will be composing a series of Pyspark code blocks to Extract, Transform and load the raw device data from Blob storage to SparkDB tables for PowerBI to visualise it.

©2018 Microsoft Corporation

Task 4 – Perform ELT(ETL) Using Spark

1. Open the newly created notebook and it should look like the below screen



Note: Each of the following code fragments gets placed in a separate cell in the notebook. You create new cells by clicking on the "+" sign (at the bottom) when in an existing cell.

1. Setup Access to blob storage:
   a) Copy the below code block in the first code cell and replace the values with your storage account name and key. (It is also possible to connect to storage account using SAS tokens in a very similar fashion) to read more about this refer to : https://docs.databricks.com/spark/latest/data-sources/azure/azure-storage.html

```
spark.conf.set(
  "fs.azure.account.key.<Storage account name>.blob.core.windows.net",
  "<Storage Account Key")
```

2. Set the Spark configuration to load Avro file regardless of the file extension.

```
spark.conf.set("avro.mapred.ignore.inputs.without.extension", "false")
```

3. Load all the files created by IoT Hub routing to a Spark dataframe and show a limited number of lines from the DF to verify the load was successful. Copy the below code to the third code cell and replace the variables with your blob container name, storage account name and IoT hub name

```
df = spark.read.format("com.databricks.spark.avro").load("wasbs://<container
name>@<storage account name>.blob.core.windows.net/<Iot Hub
name>/01/{*}/{*}/{*}/{*}/{*}")
df.show()
```

One you run all three code cells consecutively you should see something like the below screen.

©2018 Microsoft Corporation

Cmd 4

```
df = spark.read.format("com.databricks.spark.avro").load("wasbs://workshopcold@hdiausnetstorage.blob.core.windows.net/smarthassio/01/{*}/{*}/{*}/{*}/{*}")
df.show()
```

(1) Spark Jobs

df: pyspark.sql.dataframe.DataFrame = [EnqueuedTimeUtc: string, Properties: map ... 2 more fields]

```
+--------------------+--------------------+--------------------+--------------------+
|     EnqueuedTimeUtc|          Properties|    SystemProperties|                Body|
+--------------------+--------------------+--------------------+--------------------+
|2018-10-30T04:53:...|[temperatureAlert...|[connectionDevice...|[7B 22 6D 65 73 7...|
|2018-10-30T04:53:...|[temperatureAlert...|[connectionDevice...|[7B 22 6D 65 73 7...|
|2018-10-30T04:53:...|[temperatureAlert...|[connectionDevice...|[7B 22 6D 65 73 7...|
|2018-10-30T04:53:...|[temperatureAlert...|[connectionDevice...|[7B 22 6D 65 73 7...|
|2018-10-30T04:53:...|[temperatureAlert...|[connectionDevice...|[7B 22 6D 65 73 7...|
|2018-10-30T04:53:...|[temperatureAlert...|[connectionDevice...|[7B 22 6D 65 73 7...|
|2018-10-30T04:53:...|[temperatureAlert...|[connectionDevice...|[7B 22 6D 65 73 7...|
|2018-10-30T04:53:...|[temperatureAlert...|[connectionDevice...|[7B 22 6D 65 73 7...|
|2018-10-30T04:53:...|[temperatureAlert...|[connectionDevice...|[7B 22 6D 65 73 7...|
|2018-10-30T04:53:...|[temperatureAlert...|[connectionDevice...|[7B 22 6D 65 73 7...|
|2018-10-30T04:53:...|[temperatureAlert...|[connectionDevice...|[7B 22 6D 65 73 7...|
|2018-10-30T04:53:...|[temperatureAlert...|[connectionDevice...|[7B 22 6D 65 73 7...|
```

Note: If you look closely at the printed dataframe you will see the loaded Avro files consists of 4 fields and the actual part of the device data is storage in "body" field. Which is binary encoded values.

4. In order to access the information inside the body field there is some processing to be done: copy the below code into the forth code cell.

```
Rdd = df.select(df.Body.cast("string"))
Rdd.take(10)
```

Code explanation:

First line is selecting only the "body" column of the dataframe and loading it to and RDD and casting the binary value to clear text using the cast function. If you check the output of the "take" function you will notice that the binary data casted to string is actually of JSON format.

5. Transform the JSON data to tabular for further processing.

```
jsonRdd = Rdd.rdd.map(lambda x: x[0])
data = spark.read.json(jsonRdd)
data.show()
```

here we pass the lambda function to map function to place every json field into its respective tabular column and finally using the show() function to verify the outcome.

```
jsonRdd = Rdd.rdd.map(lambda x: x[0])
data = spark.read.json(jsonRdd)
data.show()
```

(2) Spark Jobs

data: pyspark.sql.dataframe.DataFrame = [deviceId: string, humidity: double ... 10 more fields]

```
+--------+-----------------+-----------+-----------+--------------+--------------------+---------+------------------+------------------+-------+-------+----------+
|deviceId|         humidity|humiditymax|humiditymin|humiditytarget|    measureTimeStamp|messageId|          pressure|       temperature|tempmax|tempmin|temptarget|
+--------+-----------------+-----------+-----------+--------------+--------------------+---------+------------------+------------------+-------+-------+----------+
|  110203| 62.57508855236319|        100|         30|            70|2018-10-30T04:53:...|     2489|11.134419692393324|20.873589736962778|     35|     15|        28|
|  110203| 65.12650076621078|        100|         30|            70|2018-10-30T04:53:...|     2490|10.225074351838192| 30.68643675103555|     35|     15|        28|
|  110203| 69.22599118423186|        100|         30|            70|2018-10-30T04:53:...|     2491|11.070396680868452| 24.03952686696753|     35|     15|        28|
|  110203| 71.42505118387052|        100|         30|            70|2018-10-30T04:53:...|     2492| 10.58221129649506|20.580277490568143|     35|     15|        28|
|  110203|60.283435817728076|        100|         30|            70|2018-10-30T04:53:...|     2493|10.394513975884344|25.805300608147363|     35|     15|        28|
|  110203| 63.40365204714364|        100|         30|            70|2018-10-30T04:53:...|     2494|10.510768315411495| 29.69869168630771|     35|     15|        28|
|  110203| 71.50582981030391|        100|         30|            70|2018-10-30T04:53:...|     2495|10.316010415835587|25.827213606139665|     35|     15|        28|
|  110203| 64.97596467792825|        100|         30|            70|2018-10-30T04:53:...|     2496|10.411995427884614| 27.76047744986031|     35|     15|        28|
|  110203|  69.3623548836468|        100|         30|            70|2018-10-30T04:53:...|     2497|10.549101129127923|28.939016459474246|     35|     15|        28|
|  110203| 76.90058718951197|        100|         30|            70|2018-10-30T04:53:...|     2498| 11.04811687455605|21.018563865338667|     35|     15|        28|
|  110203| 79.63524803324437|        100|         30|            70|2018-10-30T04:53:...|     2499|10.303492775675775| 28.79489493213333|     35|     15|        28|
|  110203| 77.68350948074794|        100|         30|            70|2018-10-30T04:53:...|     2500|10.829119377686844|31.854300197331575|     35|     15|        28|
|  110203| 77.03143917995692|        100|         30|            70|2018-10-30T04:53:...|     2501|10.351200962687567|31.808854984339337|     35|     15|        28|
|  110203| 77.32433095804026|        100|         30|            70|2018-10-30T04:53:...|     2502|10.523074849596021|27.111331449983133|     35|     15|        28|
|  110203| 70.41608061866393|        100|         30|            70|2018-10-30T04:53:...|     2503|10.833730867342119|28.888224375166338|     35|     15|        28|
|  110203| 73.79078233628903|        100|         30|            70|2018-10-30T04:53:...|     2504|11.135186318354082| 31.19728434901526|     35|     15|        28|
|  110203| 62.59890108509376|        100|         30|            70|2018-10-30T04:53:...|     2505|11.344774977481656|21.877127608195444|     35|     15|        28|
|  110203| 63.97046115348219|        100|         30|            70|2018-10-30T04:53:...|     2506|11.356256985455746|28.048854492698275|     35|     15|        28|
|  110203| 62.14337017138395|        100|         30|            70|2018-10-30T04:53:...|     2507|10.836833073431473|27.535191059262612|     35|     15|        28|
|  110203| 76.70646790986906|        100|         30|            70|2018-10-30T04:54:...|     2508|10.022304137557153| 31.14127094462838|     35|     15|        28|
+--------+-----------------+-----------+-----------+--------------+--------------------+---------+------------------+------------------+-------+-------+----------+
only showing top 20 rows
```

6. If you pay attention closely to the output you will notice the "measuretime" field is appeared as text instead of timestamp so we need to transform it to the correct type. Also the data set consists of many constant values which will have no value in a PowerBI dashboard or report. So we are using the select function to only select the fields are reports going to need.

```
from pyspark.sql.functions import to_timestamp
refineddf = data.select(to_timestamp("measureTimeStamp","yyyy-MM-
dd'T'kk:mm:ss.SSSXXX").alias("measuretime"),"humidity","pressure","temperature")
refineddf.show()
```

7. Since we are taking a full load approach for this data set and we are planning on aggregating it on hourly basis for reporting purposes we need to make sure any incomplete hour will not show in the final data set. The next code bloc achieves this using a filter function

```
from pyspark.sql.functions import col, asc
from pyspark.sql.functions import unix_timestamp, current_timestamp
refineddf = refineddf.filter("measuretime is not null and
UNIX_TIMESTAMP(current_timestamp())- UNIX_TIMESTAMP(measuretime) > 3600")
```

8. In this step we need to convert the manipulated data in the dataframe into a Databricks table so PowerBI will be able to consume it.

```
refineddf.write.format("parquet").saveAsTable("iotSensorData")
```

9. Out the full dataset table we now create another table with only aggregate data at "hourly" level using SparkSQL.

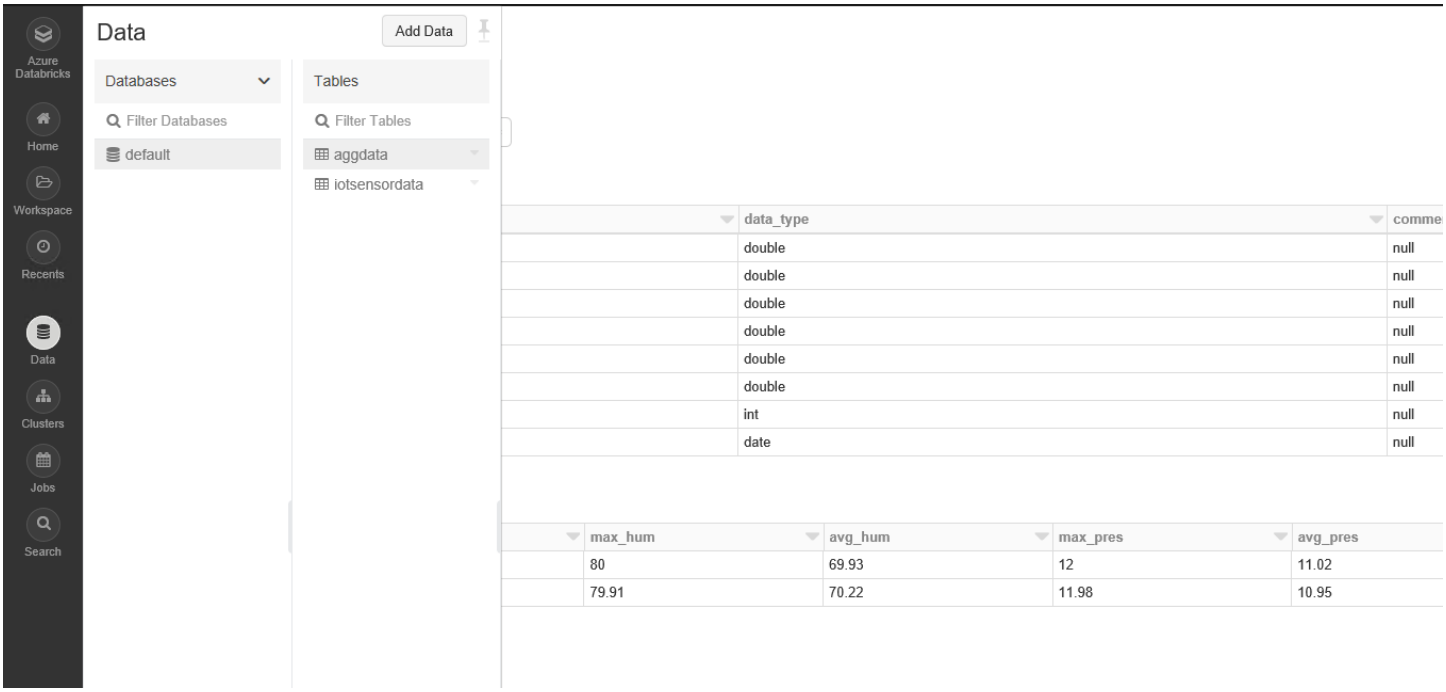
```
%sql
drop table if exists aggdata;
create table aggdata as
select
 round(max(temperature),2) as max_temp
,round(avg(temperature),2) as avg_temp
,round(max(humidity),2) as max_hum
,round(avg(humidity),2) as avg_hum
,round(max(pressure),2) as max_pres
,round(avg(pressure),2) as avg_pres
,hour(measuretime) as `hour_of_day`
,date(measuretime) as `date`
from iotsensordata
group by hour(measuretime),date(measuretime)
```

10. Finally, to verify the data in the newly created tables we run some SELECT queries

```
%sql
select * from aggdata order by date,hour_of_day;
select * from iotsensordata order by measuretime desc limit 20;
```

You can also check and verify the tables created using the "Data" tab on the left-hand side of the Azure Databricks environment

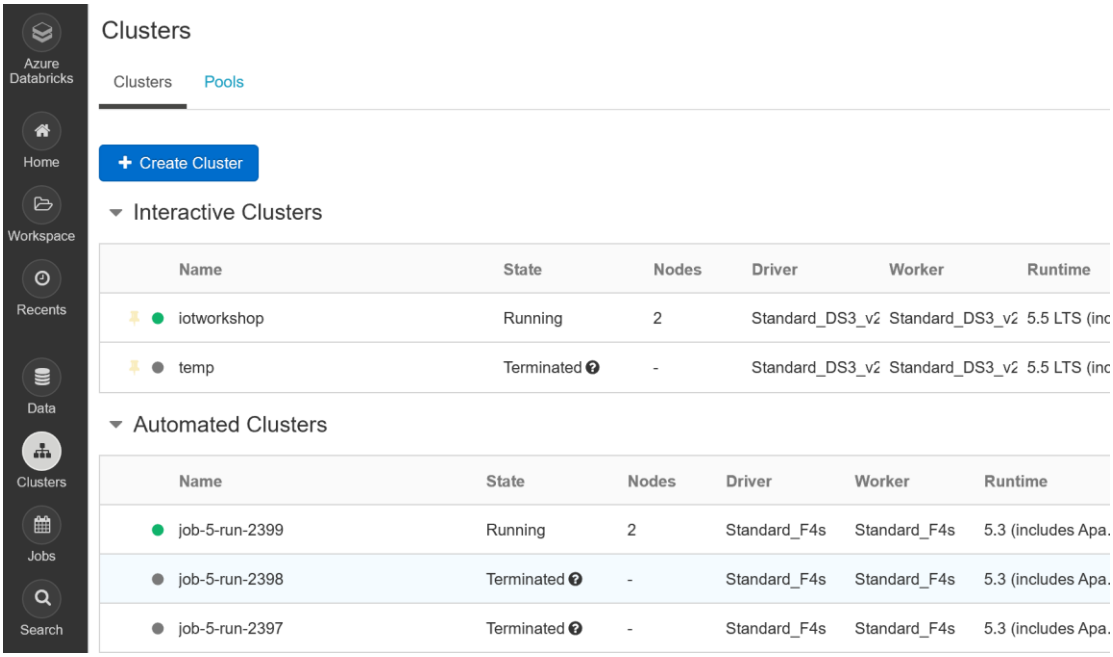
 ©2018 Microsoft Corporation



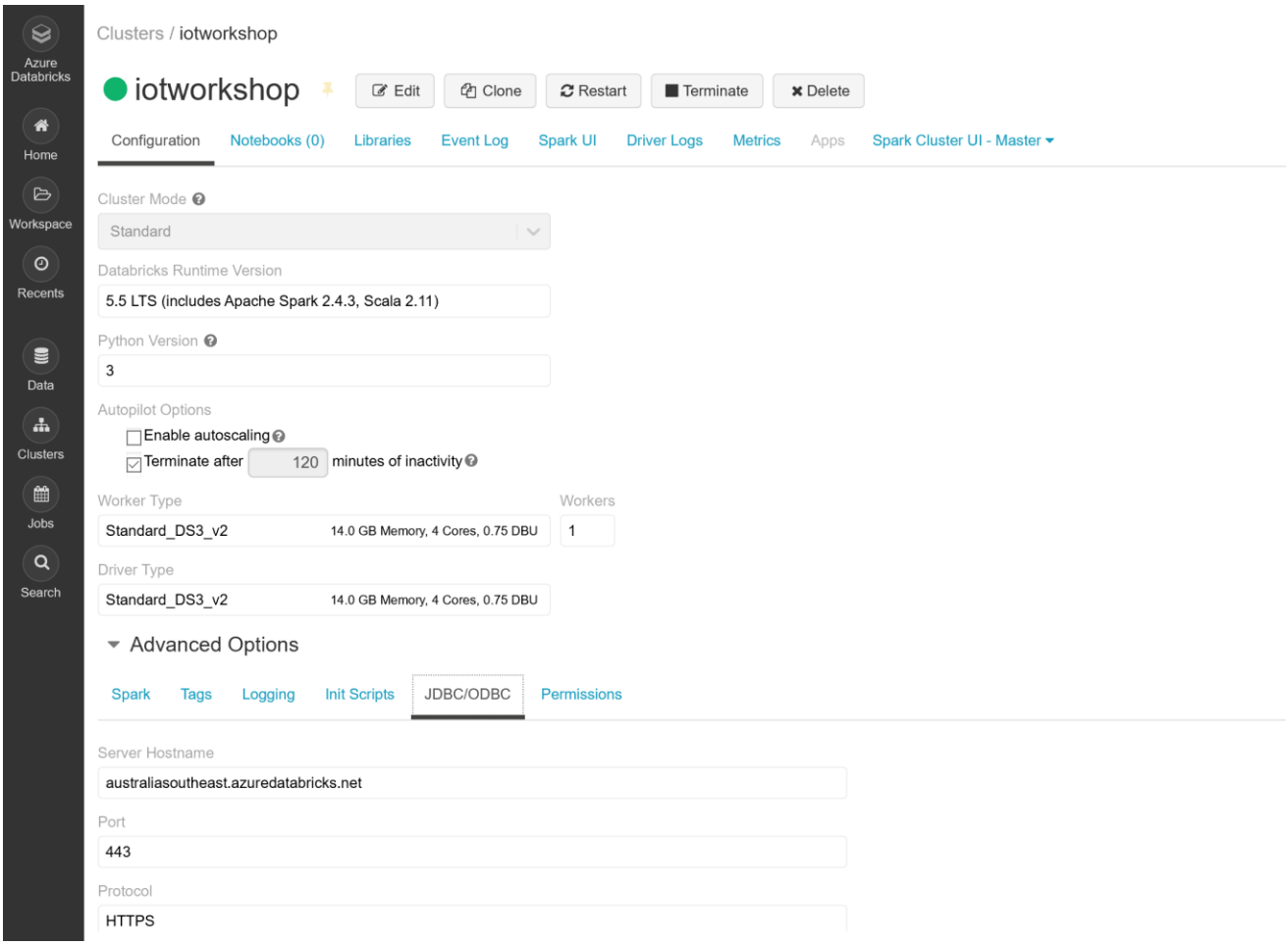
Task 5 – Connect PowerBI to Azure Databricks to perform BI reporting from transformed data.

In this step we will setup a connection between PowerBI and Azure Databricks in order to run reports and visualise the data sets created in Azure Databricks.

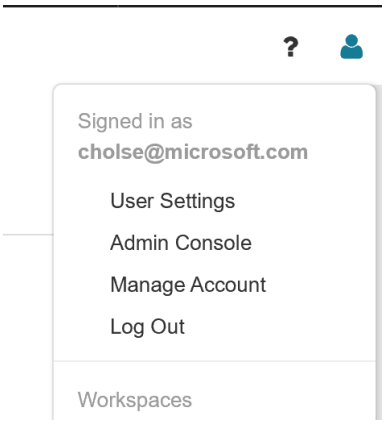
1. From the cluster table in Azure Databricks environment select your cluster, open the Advanced Options area and go to JDBC/ODBC tab
2. Copy the JDBC URL and place it in a Notepad or any text editor for later steps

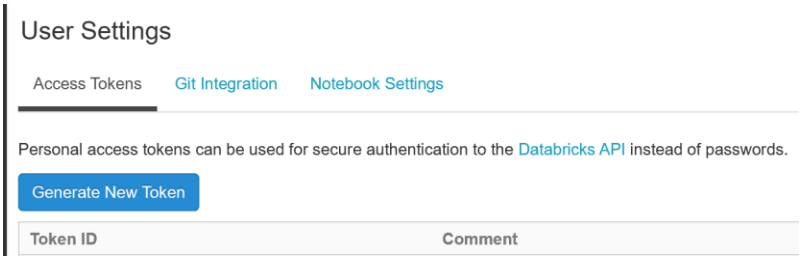


©2018 Microsoft Corporation



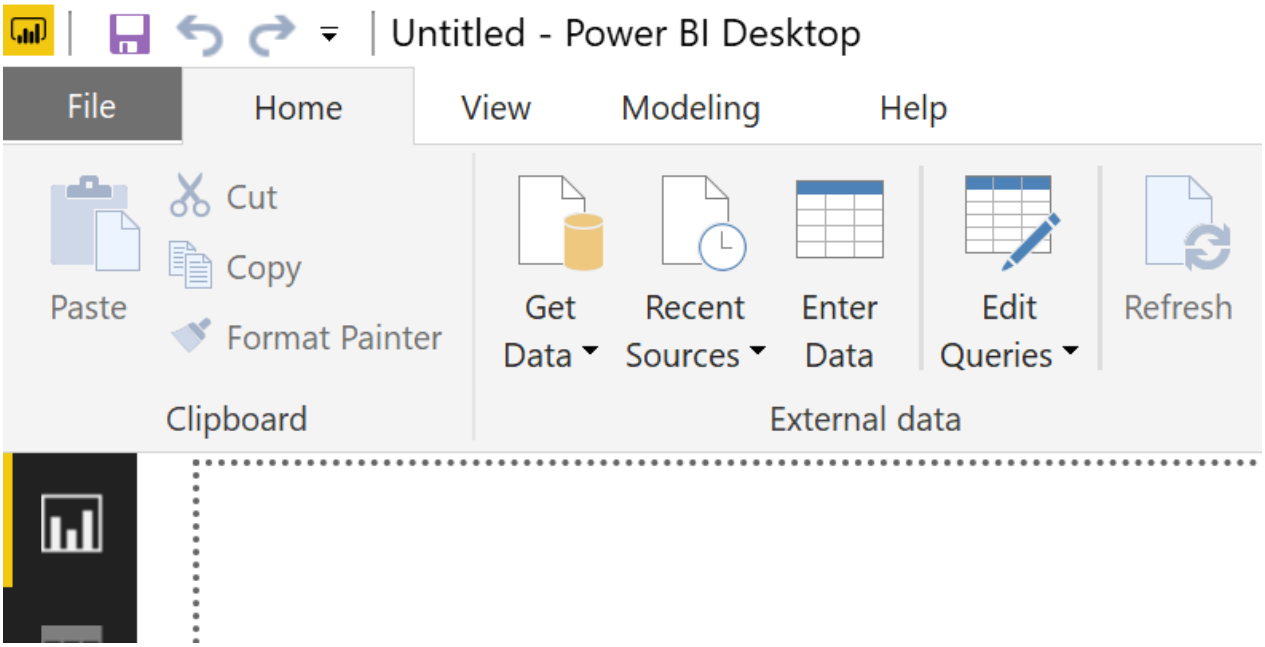
3. From Databricks we need to generate a token for PowerBI. Select the symbol then User Settings. Click Generate New Token and copy it in the same text editor window.



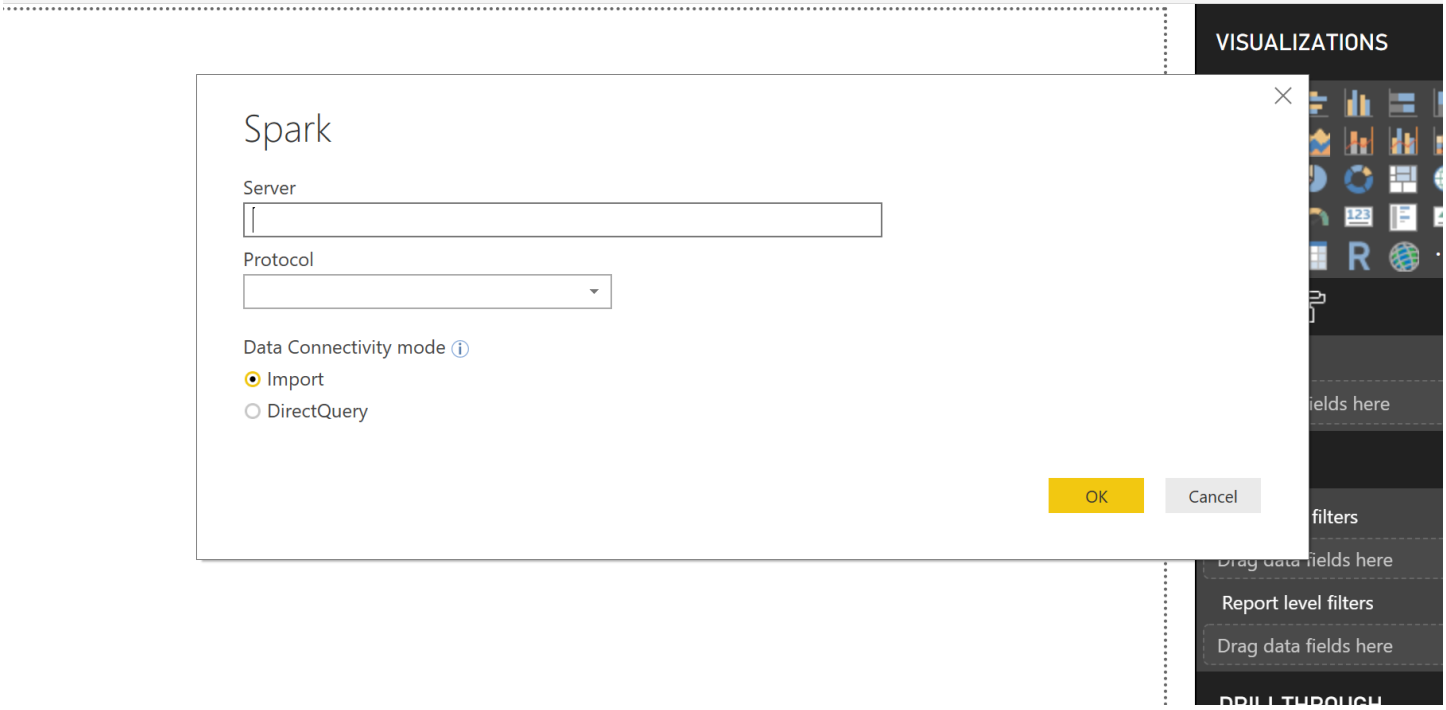


 ©2018 Microsoft Corporation

4. In PowerBI Desktop click "Get Data" and select "More"



5. From other category select "Spark" and you will be presented with the below form



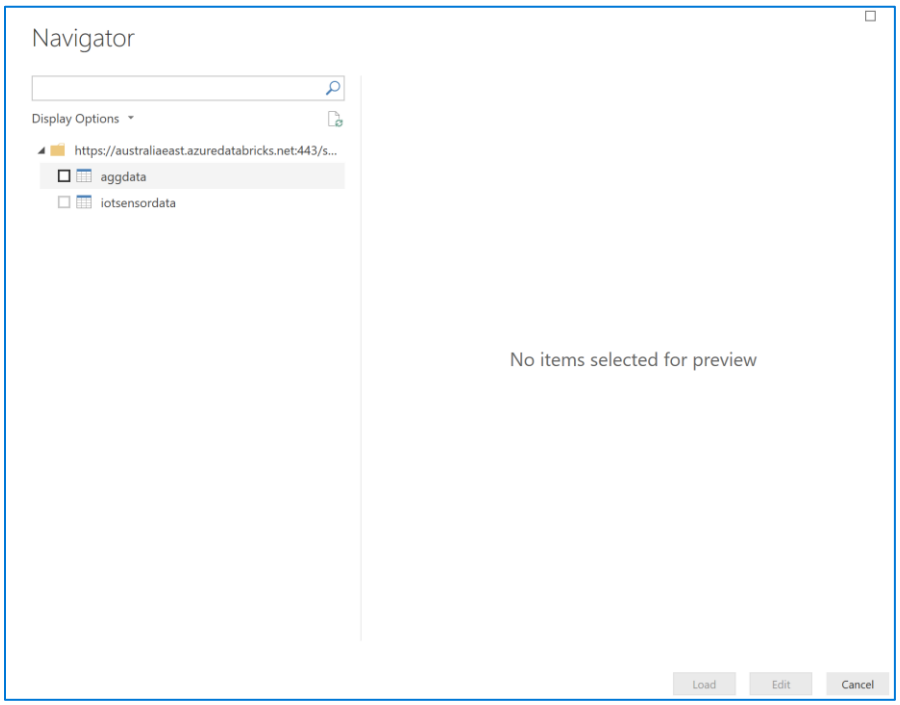
6. Remove the crossed part and replace the "jdbc:hive2" part at the beginning of the URL with "https"

```
~~jdbc:hive2~~://australiaeast.azuredatabricks.net:443/~~default;transportMode=http;ssl=true;httpPath=~~sql/protocolv1/o/3928796931465149/1030-020314-clef59
```

So that the Server address looks something like this:

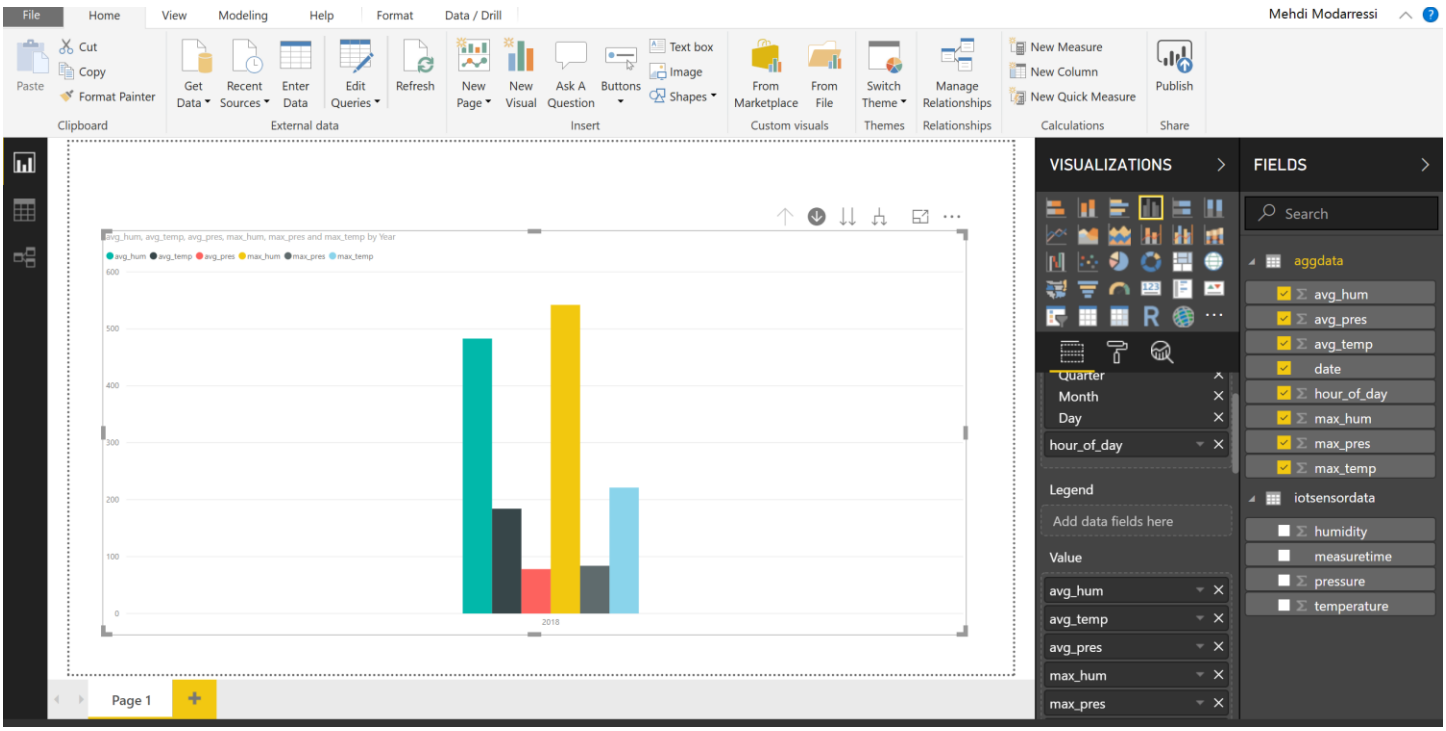
```
https://australiaeast.azuredatabricks.net:443/sql/protocolv1/o/3928796931465149/1030-020314-clef59
```

7. Paste the URL in the form and select “HTTP” as protocol. Also make sure the “Data connectivity mode” is on “Direct Query”
8. In the next form enter “token” as username and the token you got in the previous step as the password. You should be presented with the table selection window like below



9. Select both tables and click “Load”
10. Go to “Edit queries” and click “Close and Apply” to confirm lack of relationship between tables.
11. From “aggdata” table place the “date” and “hour_of_day” fields in Axis of a graph and the rest of the measures in Value part. The result should be like below.

©2018 Microsoft Corporation



12. Repeat the same steps to create a report graph out of the full table “iotsensordata”
13. If you allow sometime for new data to be written to Blob storage and re-run the steps in the Databricks notebook you can refresh the report to show newly written data. This is due to the fact that we have selected “direct query” mode rather than “Import” connectivity mode.

This is the end of Lab 3 – Cold path processing

Post-Lab Follow-up

After the hands-on lab

Duration: 10 minutes

In this exercise, attendees will deprovision any Azure resources that were created in support of the lab.

Task 1: Delete the resource group

1. Using the Azure portal, navigate to the Resource group you used throughout this hands-on lab by selecting **Resource groups** in the left menu.
2. Search for the name of your research group and select it from the list.
3. Select **Delete** in the command bar and confirm the deletion by re-typing the Resource group name and selecting **Delete**.

©2018 Microsoft Corporation