

Enabling a Real-Time Microsoft Power BI Dashboard with Azure Stream Analytics

Contents

Overview	3
Create an Azure Event Hub	7
Start event generator application	10
Create Stream Analytics job.....	13
Create PowerBI Dashboard	18
Terms of Use.....	20

Overview

Summary

In this lab we are going to learn how to build real-time dashboards in Power BI using Azure Stream Analytics (ASA).

Scenario

Contoso is a manufacturing company in the industrial automation space and they have completely automated their manufacturing process. The machinery in this plant has sensors emitting streams of

- Sensor name
- Temperature
- Humidity
- Timestamp

data in real time. In this scenario, a production floor manager wants to have a visual dashboard showing alerts whenever a sensor report values exceeding a given threshold in a temporal window. To address this need, ASA can be configured to analyze the sensor data and send alerts to a Power BI dashboard.

Azure Stream Analytics Summary

Stream Analytics is a fully managed service providing low-latency, highly available, scalable complex event processing over streaming data in the cloud. Stream Analytics makes it easy to set up real-time analytic computations on data streaming from devices, sensors, web sites, social media, applications, infrastructure systems, and more.

With a few clicks in the Azure portal, you can author a Stream Analytics job specifying the input source of the streaming data, the output sink for the results of your job, and a data transformation expressed in a SQL-like language. You can monitor and adjust the scale/speed of your job in the Azure portal to scale from a few kilobytes to a gigabyte or more of events processed per second.

Stream Analytics leverages years of Microsoft Research work in developing highly tuned streaming engines for time-sensitive processing, as well as language integrations for intuitive specifications of such.

What can I use Stream Analytics for?

Vast amounts of data are flowing at high velocity over the wire today. Organizations that can process and act on this streaming data in real time can dramatically improve efficiencies and differentiate themselves in the market. Scenarios of real-time streaming analytics can be found across all industries: personalized, real-time stock-trading analysis and alerts offered by financial services companies; real-time fraud detection; data and identity protection services; reliable ingestion and analysis of data generated by sensors and actuators embedded in physical objects (Internet of Things, or IoT); web clickstream analytics; and customer relationship management (CRM) applications issuing

alerts when customer experience within a time frame is degraded. Businesses are looking for the most flexible, reliable and cost-effective way to do such real-time event-stream data analysis to succeed in the highly competitive modern business world.

Key capabilities and Benefits?

- **Ease of use:** Stream Analytics supports a simple, declarative query model for describing transformations. In order to optimize for ease of use, Stream Analytics uses a SQL variant, and removes the need for customers to deal with the technical complexities of stream processing systems. Using the Stream Analytics query language in the browser query editor, you get intellisense auto-complete to help you can quickly and easily implement time series queries, including temporal-based joins, windowed aggregates, temporal filters, and other common operations such as joins, aggregates, projections, and filters. In addition, in-browser query testing against a sample data file enables quick, iterative development.
- **Scalability:** Stream Analytics is capable of handling high event throughput of up to 1GB/second. Integration with Azure Event Hubs allows the solution to ingest millions of events per second coming from connected devices, clickstreams, and log files, to name a few. In order to achieve this, Stream Analytics leverages the partitioning capability of Event Hubs, which can yield 1MB/s per partition. Users are able to partition the computation into a number of logical steps within the query definition, each with the ability to be further partitioned to increase scalability.
- **Reliability, repeatability and quick recovery:** A managed service in the cloud, Stream Analytics helps prevent data loss and provides business continuity in the presence of failures through built-in recovery capabilities. With the ability to internally maintain state, the service provides repeatable results ensuring it is possible to archive events and reapply processing in the future, always getting the same results. This enables customers to go back in time and investigate computations when doing root-cause analysis, what-if analysis, etc.
- **Low cost:** As a cloud service, Stream Analytics is optimized to provide users a very low cost to get going and maintain real-time analytics solutions. The service is built for you to pay as you go based on Streaming Unit usage and the amount of data processed by the system. Usage is derived based on the volume of events processed and the amount of compute power provisioned within the cluster to handle the respective Stream Analytics jobs.
- **Reference data:** Stream Analytics provides users the ability to specify and use reference data. This could be

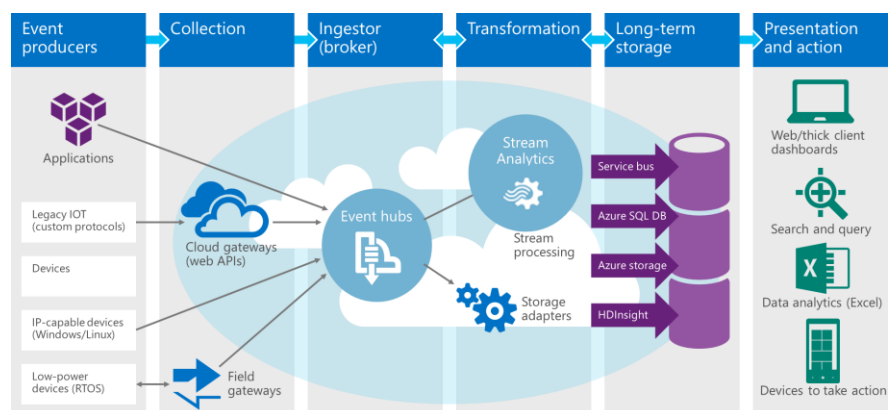
historical data or simply non-streaming data that changes less frequently over time. The system simplifies the use of reference data to be treated like any other incoming event stream to join with other event streams ingested in real time to perform transformations.

- **Connectivity:** Stream Analytics connects directly to Azure Event Hubs for stream ingestion, and the Azure Blob service to ingest historical data. Results can be written from Stream Analytics to Azure Storage Blobs or Tables, Azure SQL DB, Event Hubs, Azure Service Bus Topics or Queues, and Power BI, where it can then be visualized, further processed by workflows, used in batch analytics via Azure HDInsight or processed again as a series of events. When using Event Hubs it is possible to compose multiple Stream Analytics together with other data sources and processing engines without losing the streaming nature of the computations.

End-to-end stream processing on Microsoft Azure

Applications or devices produce messages which are sent to an *Azure Event Hub*. Event Hubs acts as the "front door" for an event pipeline, and once data is collected into an event hub, it can be transformed and stored using any real-time analytics provider or batching/storage adapters. Event Hubs decouples the production of a stream of events from the consumption of those events, so that event consumers can access the events on their own schedule. Unlike Service Bus queues and topics, Event Hubs is focused on delivering messaging stream handling at scale. Event Hubs capabilities differ from topics in that they are strongly biased towards high throughput and event processing scenarios. As a result, Event Hubs do not implement some of the messaging capabilities that are available for topics. If you need those capabilities, topics remain the optimal choice.

Stream Analytics is the *consumer* of the event hub, which queries the data in the Event Hub (in real-time) using a SQL-like language and produces an output sink to a service bus, blob storage, Azure SQL DB or HDInsight



This allows the output of the Stream Analytics query to be presented in PowerBI, Web dashboards or in devices to take action.

Stream Analytics Query Language Overview

DML Statements

- SELECT
- FROM
- WHERE
- GROUP BY
- HAVING
- CASE
- JOIN
- UNION

Statistical Functions

- VAR
- VARP
- STDEV
- STDEVP

Date and Time Functions

- DATENAME
- DATEPART
- DAY
- MONTH
- YEAR
- DATETIMEFROMPARTS
- DATEDIFF
- DATEADD

Windowing Extensions

- Tumbling Window
- Hopping Window
- Sliding Window
- Duration

Aggregate Functions

- SUM
- COUNT
- AVG
- MIN
- MAX

Scaling Functions

- WITH
- PARTITION BY

String Functions

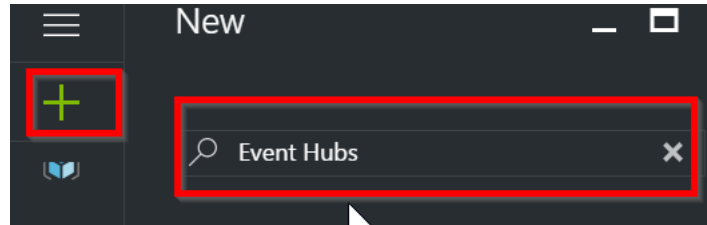
- LEN
- CONCAT
- CHARINDEX
- SUBSTRING
- PATINDEX

Create an Azure Event Hub


The sample application will generate events and push them to an Event Hub instance for real-time processing. Service Bus Event Hubs are the preferred method of event ingestion for Stream Analytics and you can learn more about Event Hubs in [Azure Service Bus documentation](#).

To create an Event Hub:

1. Log in to the Azure portal (<http://portal.azure.com>) click on **New** > search for **event hub**



2. From the results page, you should see Event Hub > Click on **Event Hub** > **Create**

NAME	PUBLISHER	CATEGORY
 Event Hubs	Microsoft	Internet of Things

3. Here we will be creating a **namespace for the service bus**, this will contain the event hub when we create a few steps time.

Enter a valid **name** (it needs to be unique to the whole of Azure), **pricing tier** (for this lab, Basic will be adequate), create a new or select a **Resource Group**, and finally choose a **region**. When happy with the settings, click the create button.

Create namespace
Event Hubs - PREVIEW

* Name
skaalab ✓
.servicebus.windows.net

* Pricing tier
Basic >

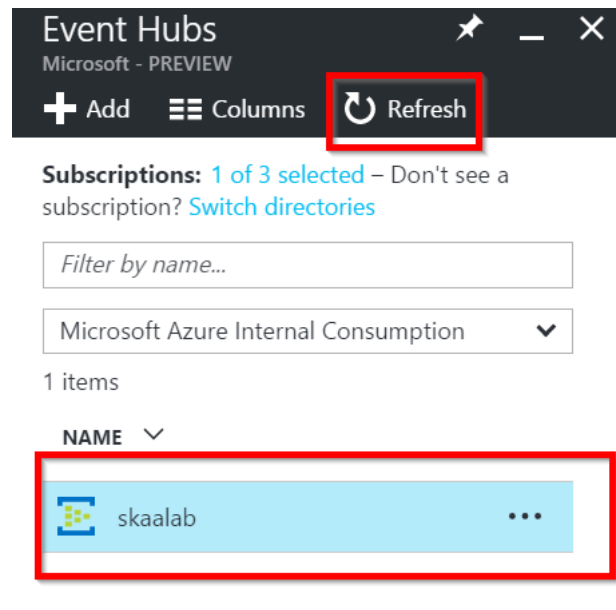
* Subscription
Microsoft Azure Internal Consumption ▼

* Resource group ⓘ
☐ Create new ☒ Use existing
ASA06 ▼

* Location
North Europe ▼

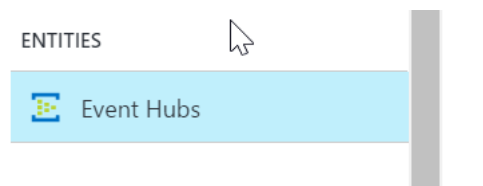
4. Once the **Namespace** has been deployed, you should see it listed in your event hubs (n.b. you may need to hit the refresh

button on the event hub blade to see it). Click on the **namespace** you created.

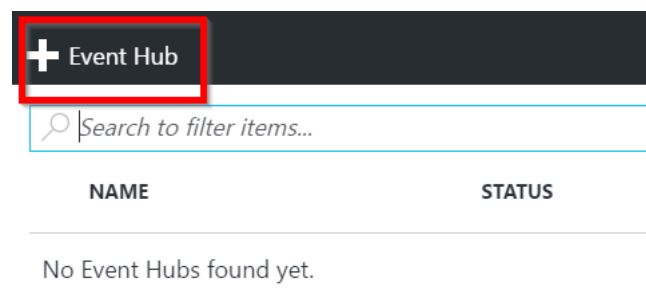


5. We will now create the actual **event hub** that will be used in the namespace we just created.

After clicking on the namespace (above) you will see a menu of options. Under the entities heading, click on **event hubs**



6. Click on **+Event Hub** in the next blade



7. Give the event hub a name and click create (leave the default values as they are) – see below:

Create Event Hub

skaalab - PREVIEW

*

Name

myhub

✓

Partition Count ⓘ

2

Message Retention ⓘ

1

Archive ⓘ

On

Off

Time window (minutes)

5

Size window (MB)

300

*

Container

Nothing selected

>

Storage Account ⓘ

Nothing selected

Create

Start event generator application

We have provided a client application that will generate sample sensor reading and push those readings into an Azure Event Hub. Follow the steps below to set up this application.

1. Unzip the **SensorEventGenerator.zip** file that is contained in the same folder as this document. Ignore any warnings about long paths.

Name	Date modified	Type	Size
AALAB06 - Real-time dashboards with Strea...	8/14/2016 11:21 AM	PDF File	1,487 KB
SensorEventGenerator	8/14/2016 11:21 AM	Compressed (zippe...	33,589 KB

2. Open the **SensorEventGenerator** Folder and then the **App** folder:

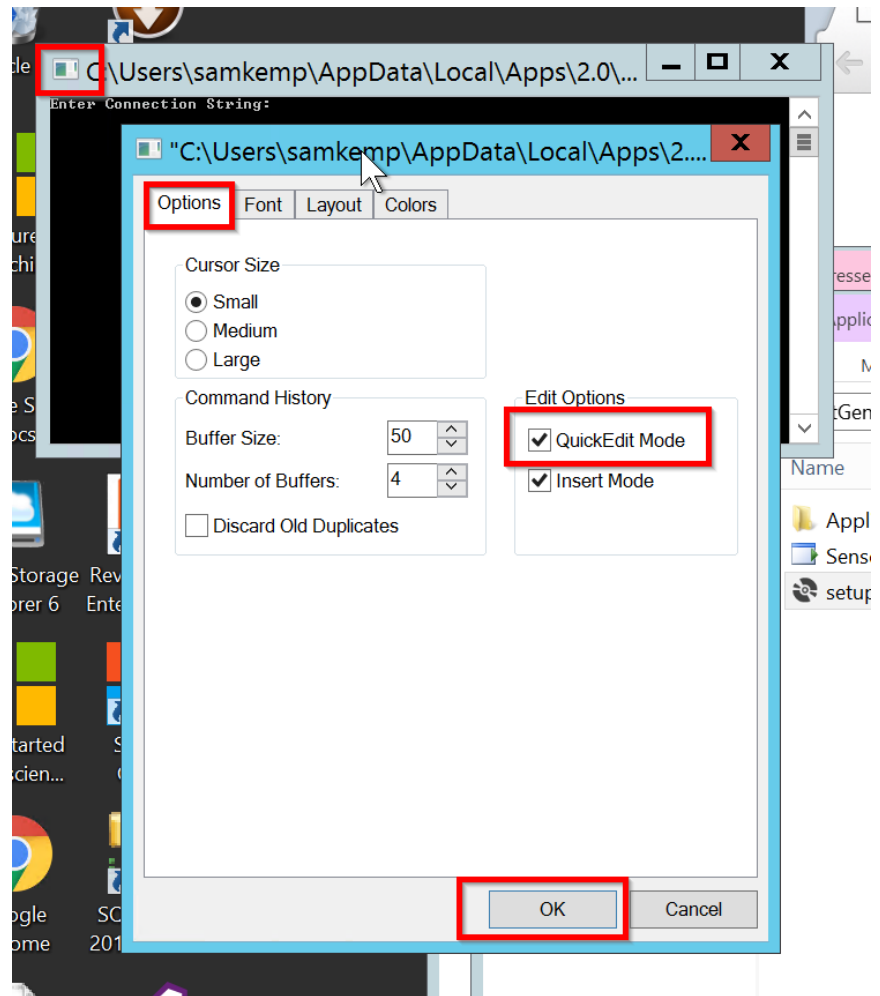
« SensorEventGenerator ▶ SensorEventGenerator ▶				Search Senso
Name	Date modified	Type	Size	
App	8/14/2016 11:24 AM	File folder		
SensorEventGenerator	8/14/2016 11:24 AM	File folder		

3. Run the **setup** program:

« SensorEventGenerator ▶ SensorEventGenerator ▶ App ▶				Search App
Name	Date modified	Type	Size	
Application Files	8/14/2016 11:24 AM	File folder		
SensorEventGenerator	8/14/2016 11:24 AM	Application Manifest	6 KB	
setup	8/14/2016 11:24 AM	Application	525 KB	

n.b. if you want to remove this console application after the lab you can do so through the usual Add/Remove programs in Control Panel (the application is called SensorEventGenerator).

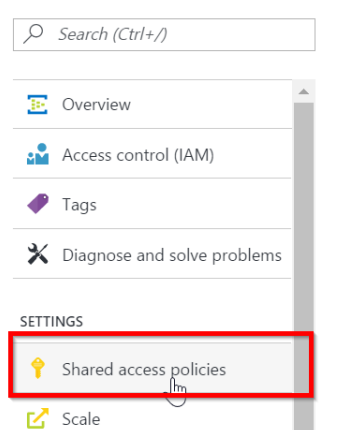
4. The console app will start running. In order to copy-and-paste your event hub connection details easily, right-click in the top right corner > **Properties** > on the **options** tab > check the **QuickEdit mode** > **OK**. See the next page for a diagram.



5. In the **Azure portal** – <http://portal.azure.com> – click on **Event Hubs** then select the namespace you created at the beginning of this lab.

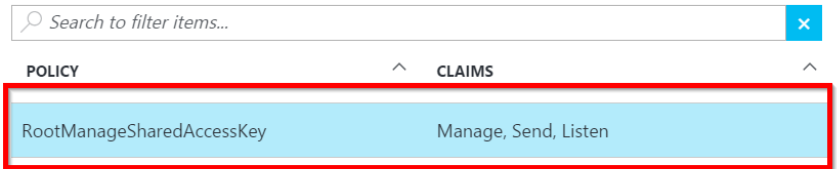
NAME	TYPE
skaalab	Event Hubs

6. Under the Settings heading, click on **Shared access policies**



7. Then Click on **RootManageSharedAccessKey**

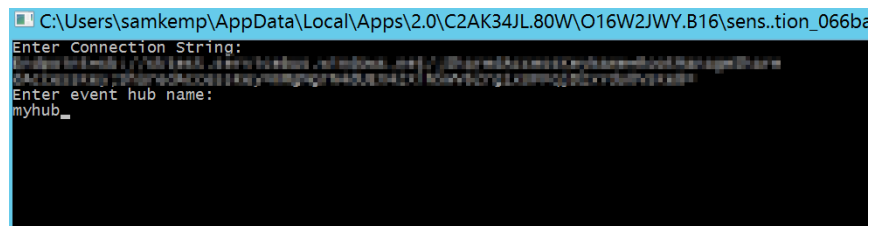
Shared access policies



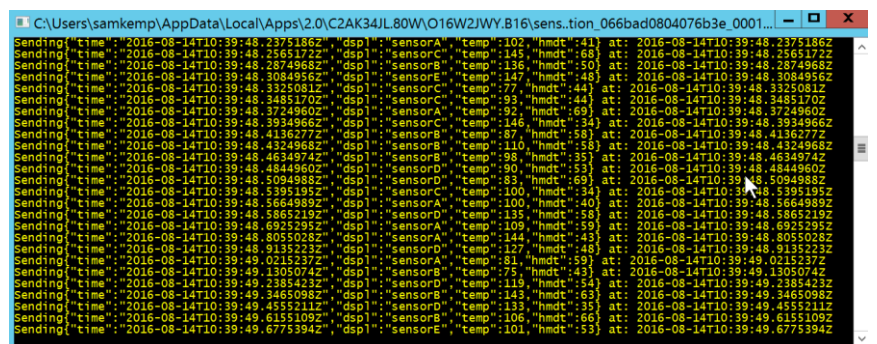
- Click on the little clipboard on the third item down (CONNECTION STRING PRIMARY KEY).

PRIMARY KEY	5Wykml0SKY0vztBMtnnDCd0enqls7xlZS1HV32JfvSc=	
SECONDARY KEY	DKyAHZjGvu0nRz1Ep2ex1yyLNy9+qMoRk1Ch9b9RnE=	
CONNECTION STRING-PRIMARY KEY	Endpoint=sb://skaalab.servicebus.windows.net/SharedA	
CONNECTION STRING-SECONDARY KEY	Endpoint=sb://skaalab.servicebus.windows.net/SharedA	

- Paste into the Console App by right clicking anywhere in the console > hit return
- Enter the **event hub name** defined earlier in the lab and hit return



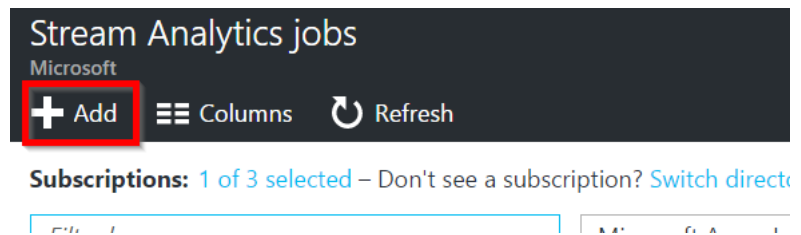
- After a few seconds, you should see sensor readings being sent to Azure Event Hubs:



- Leave the Console Application running for the duration of the lab – this will allow you to see real-time information updating in PowerBI

Create Stream Analytics job

1. In Azure portal (<http://portal.azure.com>) open **Stream Analytics** and click **Add**



2. Enter a valid **job name** and select an existing **Resource Group** (or create a new one). When you have a green tick by Job Name, click **create**.

A screenshot of the 'New Stream Analytics Job' form in the Azure portal. The form has a dark header with the title 'New Stream Analytics J...' and window control buttons. Below the header, there are four sections, each with a red asterisk and a label: 'Job name' (with a value of 'aalabtest' and a green checkmark), 'Subscription' (with a value of 'Microsoft Azure Internal Consumption'), 'Resource group' (with a value of 'ASA06' and radio buttons for 'Create new' and 'Use existing', where 'Use existing' is selected), and 'Location' (with a value of 'North Europe').

3. Once the job has been provisioned, you should see it listed in your Stream analytics blade:

A screenshot of the 'Stream Analytics jobs' blade in the Azure portal, showing the newly created job 'aalabtest'. The blade has a dark header with the title 'Stream Analytics jobs' and the Microsoft logo. Below the header, there are three buttons: '+ Add', 'Columns', and 'Refresh'. Below the buttons, it says 'Subscriptions: 1 of 3 selected' and 'Don't see a subscription? Switch direct'. At the bottom, there is a table with columns for 'Job name', 'Subscription', 'Resource group', and 'Location'. The table contains one row with the job name 'aalabtest', subscription 'Microsoft Azure Internal Consumption', resource group 'ASA06', and location 'North Europe'. The job name 'aalabtest' is highlighted with a green box.

4. Clicking on the job name above, you should see the following overview in a Blade:

Settings
Start
Stop
Delete

Created

Essentials

Resource group
[ASA06](#)

Status
Created

Location
North Europe

Subscription name
[Microsoft Azure Internal Consumption](#)

Subscription ID
5b0bd56f-84b3-4ec2-b7a1-41fd06c7edd3

Send feedback
[UserVoice](#)

Created
Monday, December 5, 2016, 9:16:43 PM

Started
-

Last output
-

Inputs
0
No results.

Query
< >

Outputs
0
No results.

Monitoring

- Click on **Inputs** > Add. This will reveal the following dialog below. For the **Input alias** enter **sensorinput** and select the **Service bus namespace** to be what you entered in the previous section (the event hub will automatically populate). Click **Create**.

New input

* Input alias

* Source Type ⓘ
Data stream

* Source ⓘ
Event hub

* Subscription
Use event hub from current subscription

* Service bus namespace
skaalab

* Event hub name
myhub

* Event hub policy name
RootManageSharedAccessKey

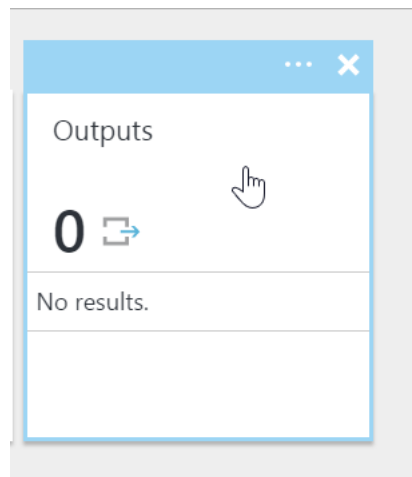
Event hub consumer group ⓘ

* Event serialization format ⓘ
JSON

Encoding ⓘ
UTF-8

Create

6. Back on the Stream Analytics Job blade click **Outputs > Add**.



7. Give your output alias the name **pbioutput** and select **Sink** to be PowerBI (it should ask you to authorize). Click on the Authorize button and follow the on-line instructions to authenticate.

* Output alias

pbioutput



* Sink ⓘ

Power BI



Authorize Connection

You'll need to authorize with Power BI to configure your output settings.

Authorize

8. Once authorized, fill in the other details for **dataset name** and **table name** (these can be given the same name as per below):

Group Workspace

My Workspace



* Dataset Name

aalabtest



If the dataset or table already exists in your Microsoft Power BI subscription, it will be overwritten.

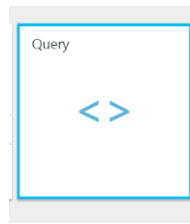
* Table Name

aalabtest

Currently authorized as [Sam Kemp](#)
(samkemp@microsoft.com)

9. We will now set up the query for the real-time job.

On your Stream Analytics Job blade, click on **Query**

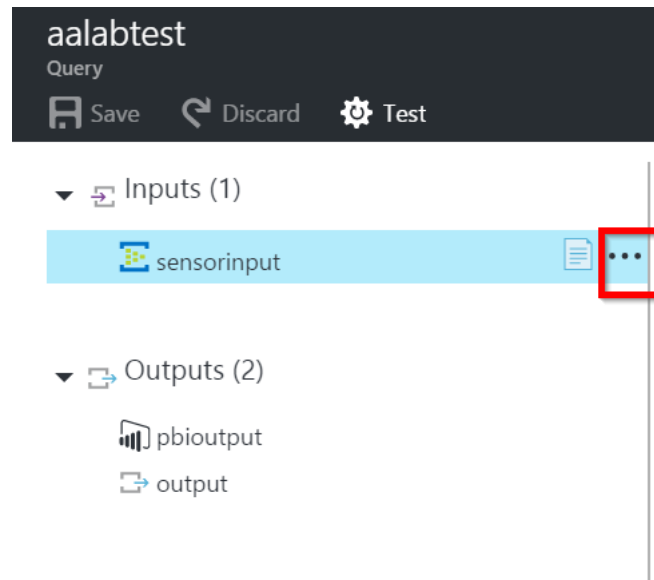


10. Delete the template SQL and Copy-and-paste the following query

```
SELECT * FROM InputStream
```

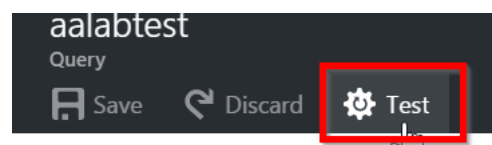
11. Click Save

12. It is often a good idea to test the query on some sample data. To do this click on the ellipsis next to the Input alias



13. Select **Sample Data from Input** > Accept the default start time and duration and click **OK**.

14. The Test button should now be enabled > click it!



15. You should see some results. Our query is a simple pass through and in the next step we will define something more useful.

[Download results](#)

TIME	DSPL	TEMP	HMDT	EVENTPROCESSEDUTCTIME	PARTITIONID	EVENTENQUEUEDUTCTIME
"2016-12-05T17:42:31..."	"sensorD"	75	34	"2016-12-05T17:43:00..."	1	"2016-12-05T17:42:31..."
"2016-12-05T17:42:31..."	"sensorE"	124	59	"2016-12-05T17:43:00..."	1	"2016-12-05T17:42:31..."

16. Replace the query with the following SQL statement and click **Save**

```
SELECT
    System.Timestamp as OutputTime,
    dspl as SensorName,
    Avg(temp) as AvgTemperature
INTO
    PBIOutput
FROM
    SensorInput TIMESTAMP BY time
GROUP BY TumblingWindow(second, 30), dspl
HAVING Avg(temp) > 100
```

17. Click **Test** and you should see that we have the following results

[Download results](#)

OUTPUTTIME	SENSORNAME	AVGTEMPERATURE
"2016-12-05T17:50:00.0000000Z"	"sensorD"	107
"2016-12-05T17:50:00.0000000Z"	"sensorA"	103
"2016-12-05T17:50:00.0000000Z"	"sensorB"	124.5
"2016-12-05T17:50:00.0000000Z"	"sensorC"	109.75
"2016-12-05T17:50:30.0000000Z"	"sensorA"	104.5

18. You are now ready to start the Stream Analytics Job > Click **Start** on the main job blade i.e.

The screenshot shows the Azure Stream Analytics job blade for a job named 'Created'. At the top, there is a toolbar with buttons for 'Settings', 'Start' (highlighted with a red box), 'Stop', and 'Delete'. Below the toolbar, the 'Essentials' section displays job details: Resource group 'ASA06', Status 'Created', Location 'North Europe', Subscription name 'Microsoft Azure Internal Consumption', and Subscription ID '5b0bd56f-84b3-4ec2-b7a1-41fd06c7edd3'. To the right of these details are links for 'Send feedback UserVoice', 'Created Monday, December 5, 2016, 9:16:43 PM', 'Started -', and 'Last output -'. The 'Job Topology' section shows a diagram with 'Inputs' (1 input: sensorinput), 'Query' (represented by a code icon), and 'Outputs' (1 output: pbioutput). The 'Monitoring' section is partially visible at the bottom.

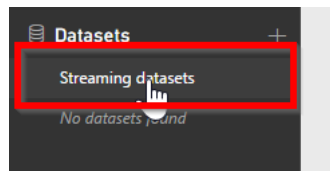
19. It will take a couple of minutes to start the Job, once it is running following the next steps on the following pages

Create PowerBI Dashboard

Now that we have defined an event stream, an Event Hub input to ingest events, and a query to perform a transformation over the stream, the last step is to create a PowerBI Dashboard.

Follow the steps below:

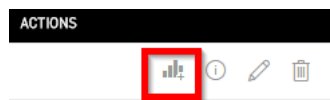
1. Log in to PowerBI - <https://powerbi.microsoft.com/en-us>
2. On the landing page you should see the **Streaming Datasets**



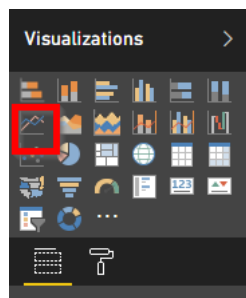
3. On the list of **Streaming datasets**, you should see the name of the event hub that you created in the earlier steps.



4. Under the Actions column, click on the little chart as highlighted below

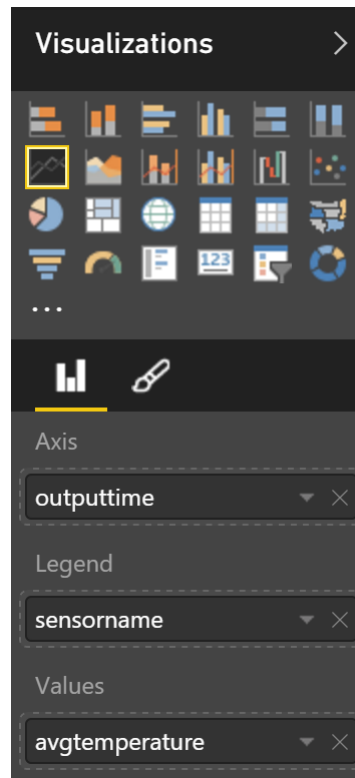


5. Click on the **Line Chart** icon in the Visualisations blade.

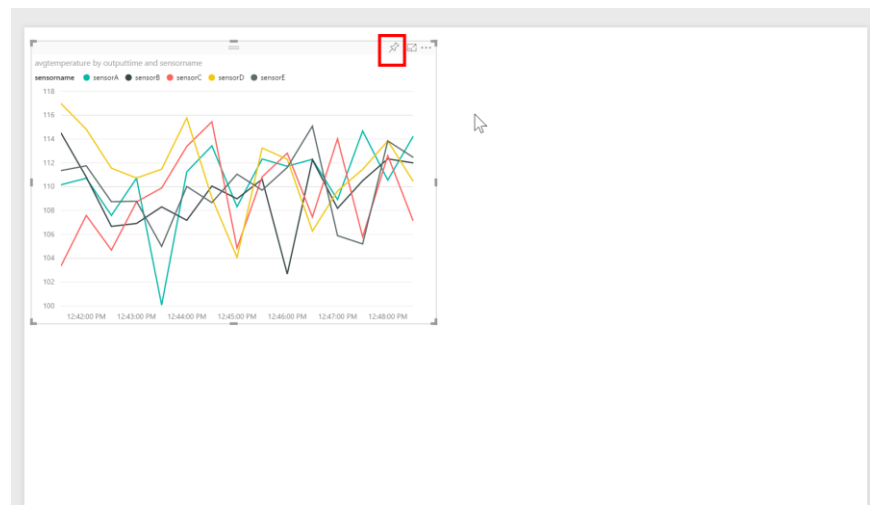


6. Drag on to Axis **outputtime**.
7. Drag on to Legend **sensorname**.
8. Drag on to Values **avgtemperature**.

The Visualisations blade should look as follows:



9. The report canvas should have a Chart:



10. Click on the **Pin** (as highlighted by the red box above) > Create a new Report > Pin to dashboard (select a name for a **New Dashboard**).

11. Click on the Dashboard you just created and you should see your chart updating every 30secs (PowerBI refreshes every 10 seconds, but we are tumbling a 30 second window).

12. Explore

- other visualisations
- tables
- natural language search feature

Terms of Use

© 2016 Microsoft Corporation. All rights reserved.

By using this Hands-on Lab, you agree to the following terms:

The technology/functionality described in this Hands-on Lab is provided by Microsoft Corporation in a “sandbox” testing environment for purposes of obtaining your feedback and to provide you with a learning experience. You may only use the Hands-on Lab to evaluate such technology features and functionality and provide feedback to Microsoft. You may not use it for any other purpose. You may not modify copy, distribute, transmit, display, perform, reproduce, publish, license, create derivative works from, transfer, or sell this Hands-on Lab or any portion thereof.

COPYING OR REPRODUCTION OF THE HANDS-ON LAB (OR ANY PORTION OF IT) TO ANY OTHER SERVER OR LOCATION FOR FURTHER REPRODUCTION OR REDISTRIBUTION IS EXPRESSLY PROHIBITED.

THIS HANDS-ON LAB PROVIDES CERTAIN SOFTWARE TECHNOLOGY/PRODUCT FEATURES AND FUNCTIONALITY, INCLUDING POTENTIAL NEW FEATURES AND CONCEPTS, IN A SIMULATED ENVIRONMENT WITHOUT COMPLEX SET-UP OR INSTALLATION FOR THE PURPOSE DESCRIBED ABOVE. THE TECHNOLOGY/CONCEPTS REPRESENTED IN THIS HANDS-ON LAB MAY NOT REPRESENT FULL FEATURE FUNCTIONALITY AND MAY NOT WORK THE WAY A FINAL VERSION MAY WORK. WE ALSO MAY NOT RELEASE A FINAL VERSION OF SUCH FEATURES OR CONCEPTS. YOUR EXPERIENCE WITH USING SUCH FEATURES AND FUNCTIONALITY IN A PHYSICAL ENVIRONMENT MAY ALSO BE DIFFERENT.

FEEDBACK. If you give feedback about the technology features, functionality and/or concepts described in this Hands-on Lab to Microsoft, you give to Microsoft, without charge, the right to use, share and commercialize your feedback in any way and for any purpose. You also give to third parties, without charge, any patent rights needed for their products, technologies and services to use or interface with any specific parts of a Microsoft software or service that includes the feedback. You will not give feedback that is subject to a license that requires Microsoft to license its software or documentation to third parties because we include your feedback in them. These rights survive this agreement.

MICROSOFT CORPORATION HEREBY DISCLAIMS ALL WARRANTIES AND CONDITIONS WITH REGARD TO THE HANDS-ON LAB, INCLUDING ALL WARRANTIES AND CONDITIONS OF MERCHANTABILITY, WHETHER EXPRESS, IMPLIED OR STATUTORY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. MICROSOFT DOES NOT MAKE ANY ASSURANCES OR REPRESENTATIONS WITH REGARD TO THE ACCURACY OF THE RESULTS, OUTPUT THAT DERIVES FROM USE OF THE VIRTUAL LAB, OR SUITABILITY OF THE INFORMATION CONTAINED IN THE VIRTUAL LAB FOR ANY PURPOSE.

DISCLAIMER

This lab contains only a portion of the features and enhancements in Microsoft Azure. Some of the features might change in future releases of the product.