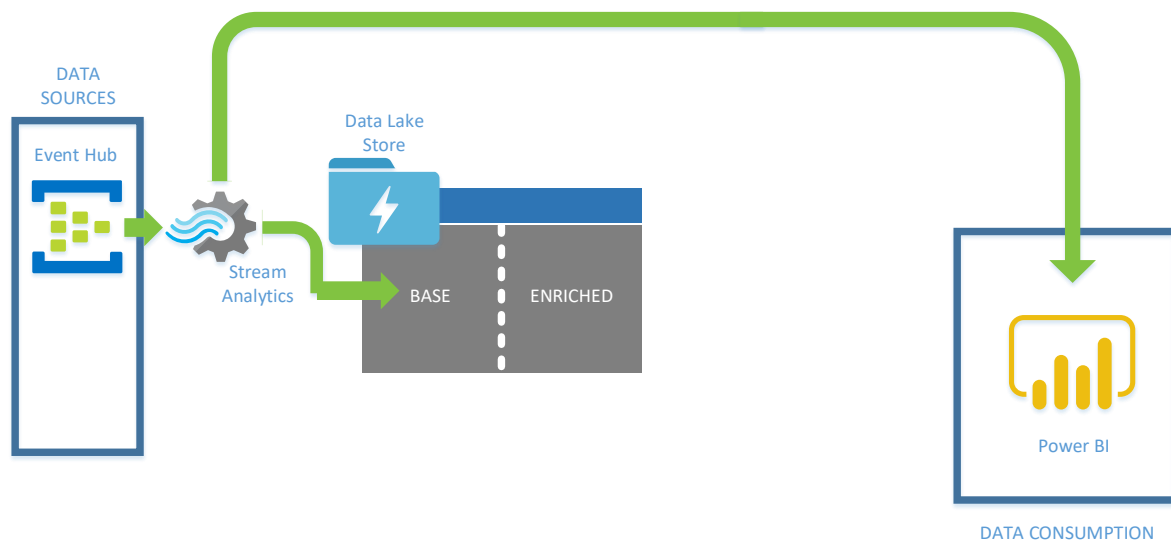


### Lab Overview and Objective

Ingest real-time data from a pre-created message blaster app running in an Azure Virtual Machine into your own Azure Event Hub, through to Azure Stream Analytics and visualised using Power BI streaming dashboard tiles. As well as writing the data into Azure Data Lake.

This lab covers the below section of our overall architecture.



### Steps

**Narrative:** first, we need some more PaaS tech deploying so we can start facilitating real-time data in our solution.

1. Deploy the required Azure services via the Portal.

- a. Create an Azure Event Hub
  - i. **Tier:** Basic (1 Consumer Group)
  - ii. **Enable Kafka:** No
  - iii. **Make Zone Redundant:** No
  - iv. **Region:** West Europe
  - v. **Throughput Units:** 1
  - vi. **Enable Auto-Inflate:** No
- b. Create an Azure Stream Analytics Service
  - i. **Region:** West Europe
  - ii. **Hosting Environment:** Cloud
  - iii. **Streaming Units:** 1



c. Create a PowerBI.com Workspace

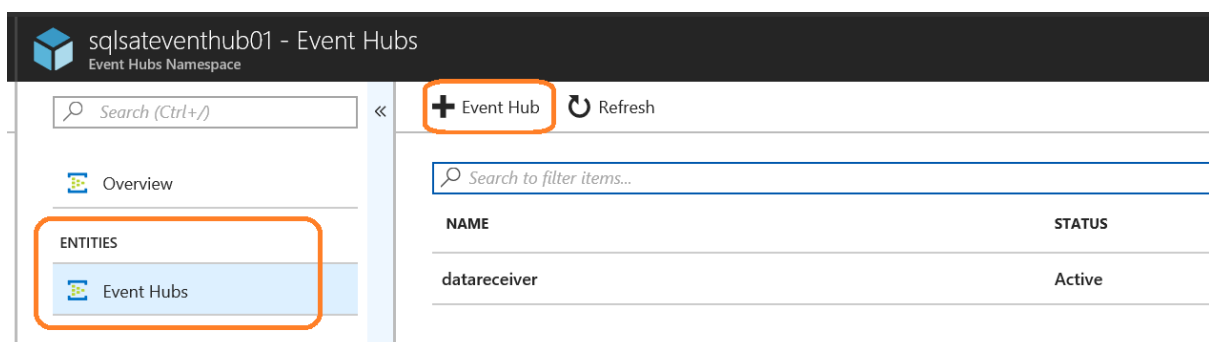
- i. Using the same 'Work/School' account used for your Azure subscription navigate to [PowerBI.com](https://PowerBI.com) and login.
- ii. If you already have a workspace, just confirm it's still accessible and move onto the next step.
- iii. If you don't have a workspace and PowerBI.com account follow the prompts to create one for free.



**Narrative:** now we have services available to handle our real-time data feed we need to configure them ingest a data stream.

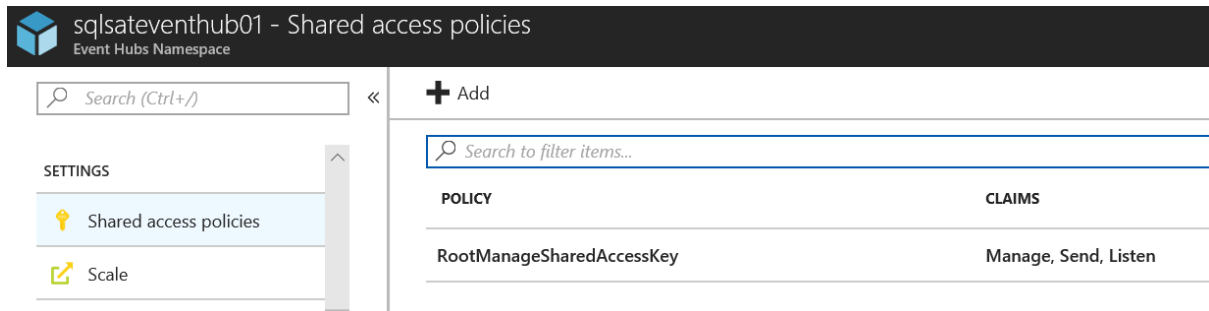
For our data stream we will use a pre-built message blaster app running on our server. This will randomly select and send a record from an Adventure Works sales order details table once you've provided your event hub connection details.

2. Create an Event Hub namespace and subscribe your Event Hub to the blaster apps messages.
  - a. Navigate to your newly deployed Azure Event Hubs service via the portal.
  - b. In the portal blade under **Entities** choose **Event Hubs**.
  - c. In the next blade that's presented click **+ Event Hub** on the tool bar.



- d. Give the Event Hub a suitable name and click **Create**.
  - i. **Partition Count:** 2
  - ii. **Message Retention:** 1
  - iii. **Capture:** Off

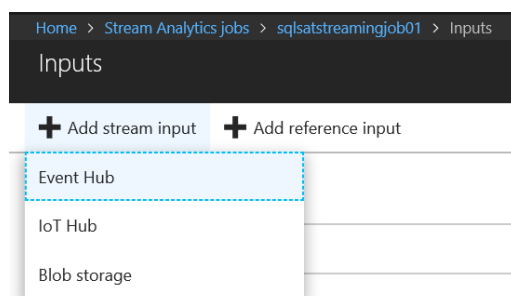
- e. Back on the main Event Hub blade now choose **Shared Access Policies** under **Settings** and select the **RootManageSharedAccessKey**.



- f. In the blade the presents on the right of your browser screen copy the **Connect string – primary key** value.
- g. Email the connection string to [paul@mrpaulandrew.com](mailto:paul@mrpaulandrew.com).
- h. Get Paul to add your Event Hub connection string to the message blaster app configuration. Coffee helps with this process.

**Narrative:** your event hub will now be receiving messages from the source data. We now need to work on stream analytics to handle the downstream data flows.

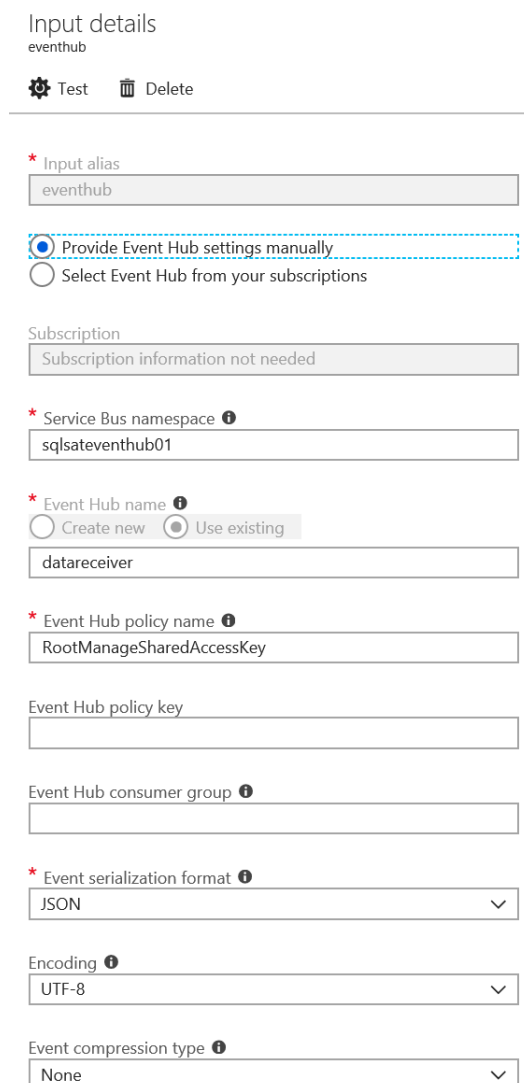
3. Configure the two inputs and two outputs for your Azure Stream Analytics service – we'll tell it what to do with these connections in a later step.
  - a. Open your Azure Stream Analytics service in the portal.
  - b. Choose **Inputs** under the **Job Topology** section.
  - c. Input 1 – from your Event Hub. Choose **+ Add Stream Input** and then select **Event Hub** from the drop-down menu.



- d. In the resulting blade on the right of your browser window populate the values similar to the below example.

Within the context of you Azure tenant the stream analytics job and find and authenticate against other services without providing connection details manually.

Choose **Select Event Hub from your Subscriptions**.



Input details  
eventhub

⚙️ Test 🗑️ Delete

\* Input alias  
eventhub

☒ Provide Event Hub settings manually  
☐ Select Event Hub from your subscriptions

Subscription  
Subscription information not needed

\* Service Bus namespace ⓘ  
sqlsateventhub01

\* Event Hub name ⓘ  
☐ Create new ☒ Use existing  
datareceiver

\* Event Hub policy name ⓘ  
RootManageSharedAccessKey

Event Hub policy key

Event Hub consumer group ⓘ

\* Event serialization format ⓘ  
JSON

Encoding ⓘ  
UTF-8

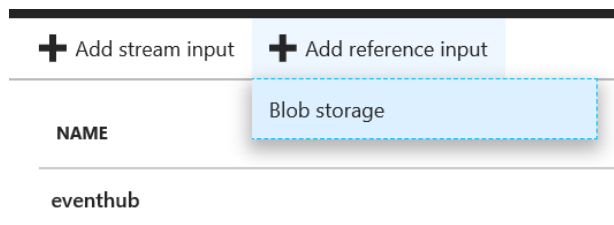
Event compression type ⓘ  
None

- e. Input 2 – a reference dataset from our pre-created blob storage account.



- f. Back to your Event Hub. Choose **Inputs** under the **Job Topology** section.

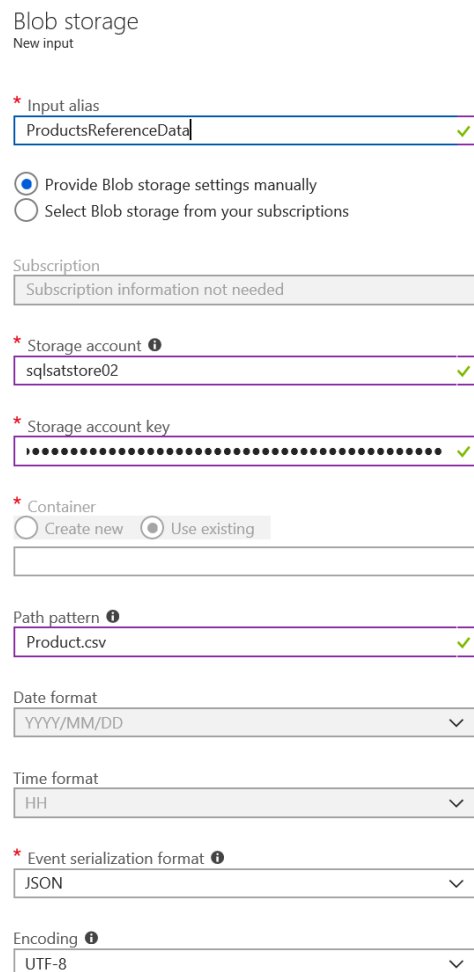
- g. Choose **+ Add Reference Input** and then select **Event Hub** from the drop-down menu.



- h. In the resulting blade on the right of your browser window populate the values similar to the below example.

Complete the connection details manually providing the storage account name and key that can be found in the [GitHub](#) repository under lab 1.

Choose **Provide Blob Storage Settings Manually**.



- i. Output 1 – to Azure Data Lake Storage.
- j. Back to your Event Hub. Choose **Outputs** under the **Job Topology** section.
- k. In the resulting blade on the right of your browser window populate the values similar to the below example.



Within the context of you Azure tenant the stream analytics job and find and authenticate against other services without providing connection details manually. However, Azure Data Lake will need an additional session token for the connection to be established.



Choose **Select Event Hub from your Subscriptions**.

Note, the **Path Prefix Pattern** is what you created in Data Lake in Lab 1.

- l. Use the **Authorization** button at the bottom on the blade to get a token.

### Output details

datalake

 Test  Delete

\* Output alias

datalake

☒ Provide Data Lake Store settings manually

☐ Select Data Lake Store from your subscriptions

Subscription

Subscription information not needed

\* Account name

sqlsatstore01

\* Path prefix pattern ⓘ

RAW/External/AdventureWorks/Streamed/{date}/ ✓

Example: cluster1/logs/{date}/{time}

Date format

YYYY/MM/DD

Time format

HH

\* Event serialization format ⓘ

CSV

Delimiter ⓘ

comma (,)

Encoding ⓘ

UTF-8

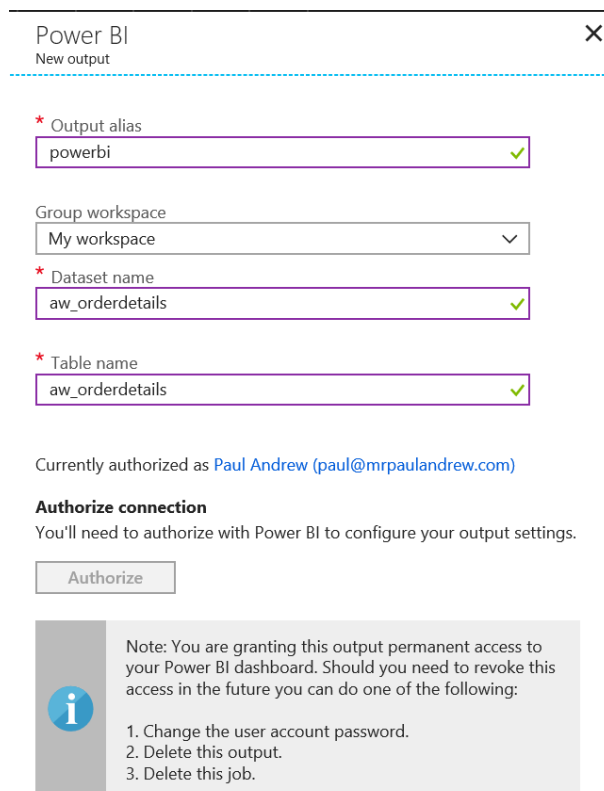
- a. Output 2 – to Power BI.
- b. Back to your Event Hub. Choose **Outputs** under the **Job Topology** section.



- c. In the resulting blade on the right of your browser window populate the values similar to the below example.

PowerBI is an external service to Azure so we simply specify a workspace and dataset name for the output.

- d. Similar to data lake use the **Authorization** button at the bottom on the blade to get a token.



The screenshot shows the 'Power BI' configuration blade for a new output. It includes the following fields and values:

- Output alias:** powerbi (with a green checkmark)
- Group workspace:** My workspace (dropdown menu)
- Dataset name:** aw\_orderdetails (with a green checkmark)
- Table name:** aw\_orderdetails (with a green checkmark)

Below the fields, it states: 'Currently authorized as Paul Andrew (paul@mrpaulandrew.com)'.

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

There is an 'Authorize' button.

**Note:** You are granting this output permanent access to your Power BI dashboard. Should you need to revoke this access in the future you can do one of the following:

1. Change the user account password.
2. Delete this output.
3. Delete this job.

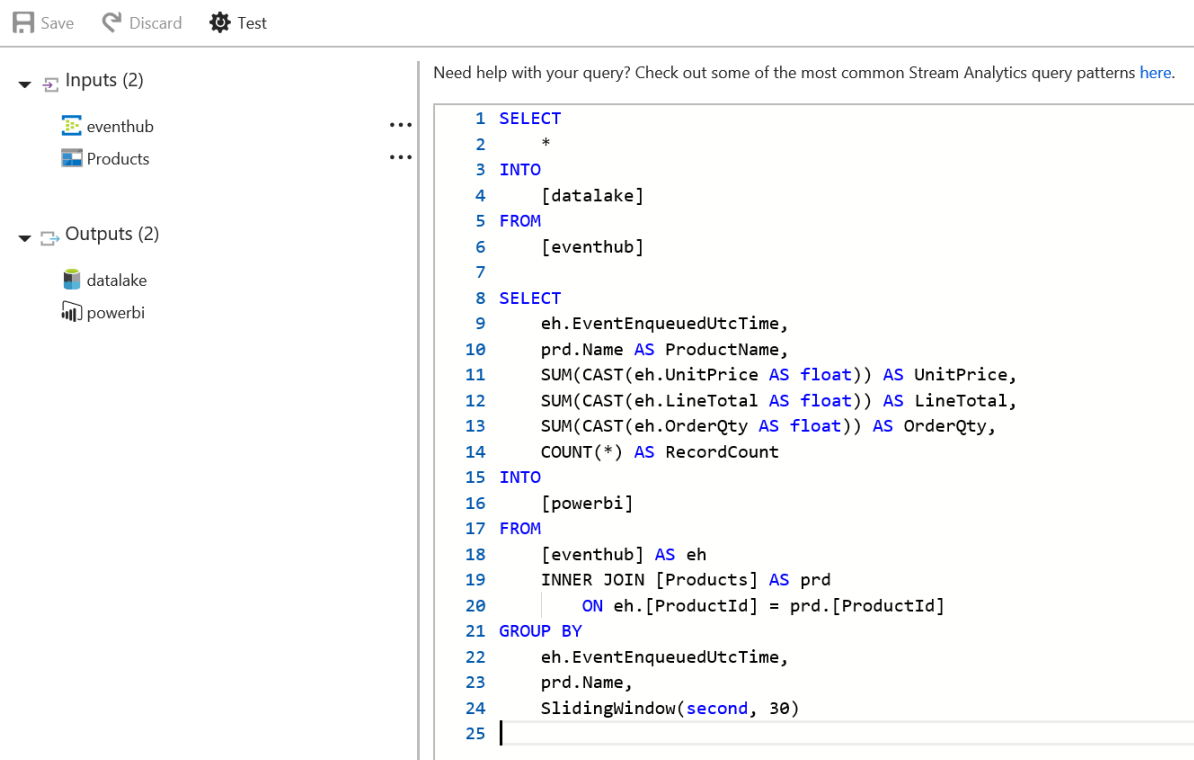
**Narrative:** we now have the sources and destinations configured within our Stream Analytics job. Finally, we need a query to join them together and shape out data as it flows through the service.

4. Add a stream analytics query to our job.



- Back to your Event Hub. Choose **Query** under **Job Topology**.
- In the blade presented you should have your two inputs and two outputs defined on the left of the canvas.
- Add two queries as follows, renaming to your choice of input and output names as required.

Add the queries using copies that can be found in the [GitHub](#) repository under lab 2.



Save Discard Test

▼ Inputs (2)

- eventhub
- Products

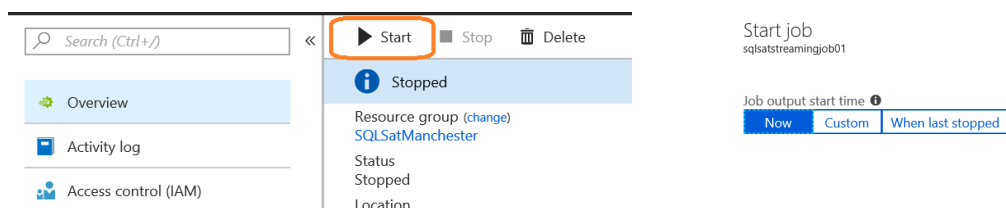
▼ Outputs (2)

- datalake
- powerbi

Need help with your query? Check out some of the most common Stream Analytics query patterns [here](#).

```
1 SELECT
2   *
3 INTO
4   [datalake]
5 FROM
6   [eventhub]
7
8 SELECT
9   eh.EventEnqueuedUtcTime,
10  prd.Name AS ProductName,
11  SUM(CAST(eh.UnitPrice AS float)) AS UnitPrice,
12  SUM(CAST(eh.LineTotal AS float)) AS LineTotal,
13  SUM(CAST(eh.OrderQty AS float)) AS OrderQty,
14  COUNT(*) AS RecordCount
15 INTO
16  [powerbi]
17 FROM
18  [eventhub] AS eh
19  INNER JOIN [Products] AS prd
20    ON eh.[ProductId] = prd.[ProductId]
21 GROUP BY
22   eh.EventEnqueuedUtcTime,
23   prd.Name,
24   SlidingWindow(second, 30)
25
```

- Once added. Click **Save**.
- Back to your Event Hub. Click **Start** and if prompted. **Start out start time as Now**.



Search (Ctrl+/) « Start Stop Delete

Stopped

Resource group (change)  
SQLSatManchester

Status  
Stopped

Location

Start job  
sqlsatstreamingjob01

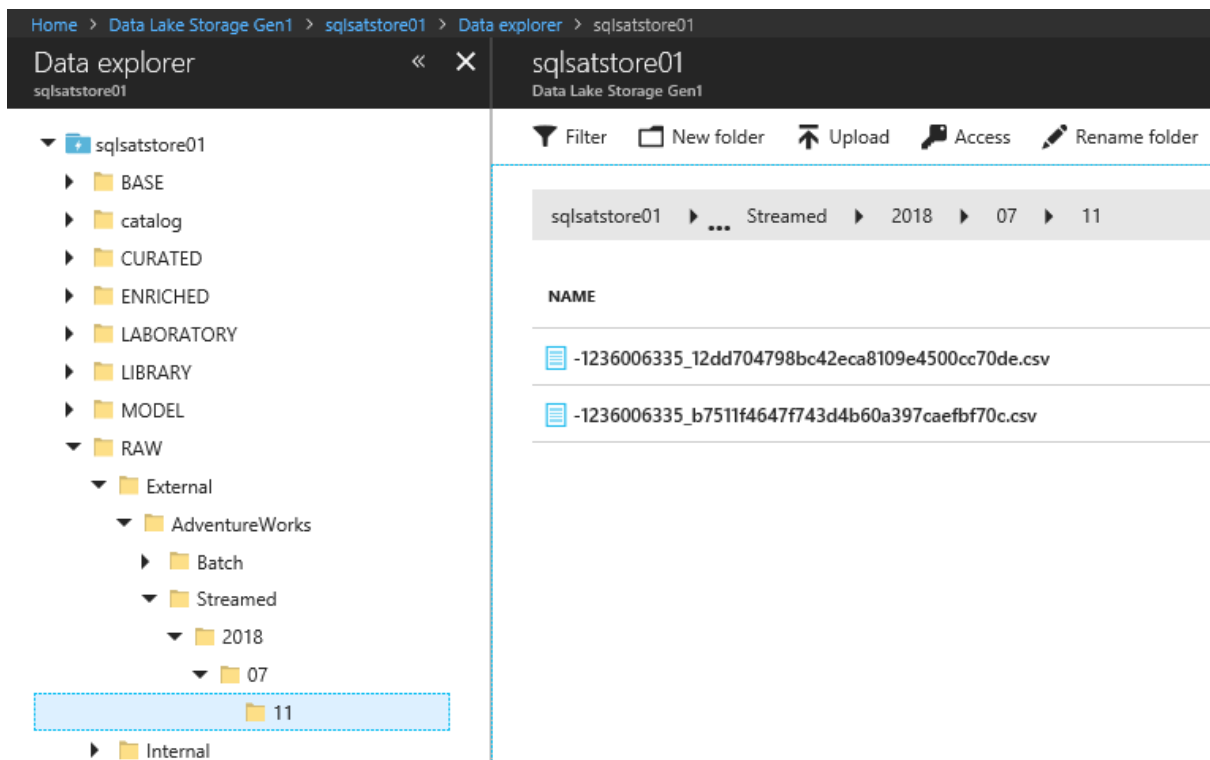
Job output start time ⓘ

Now Custom When last stopped



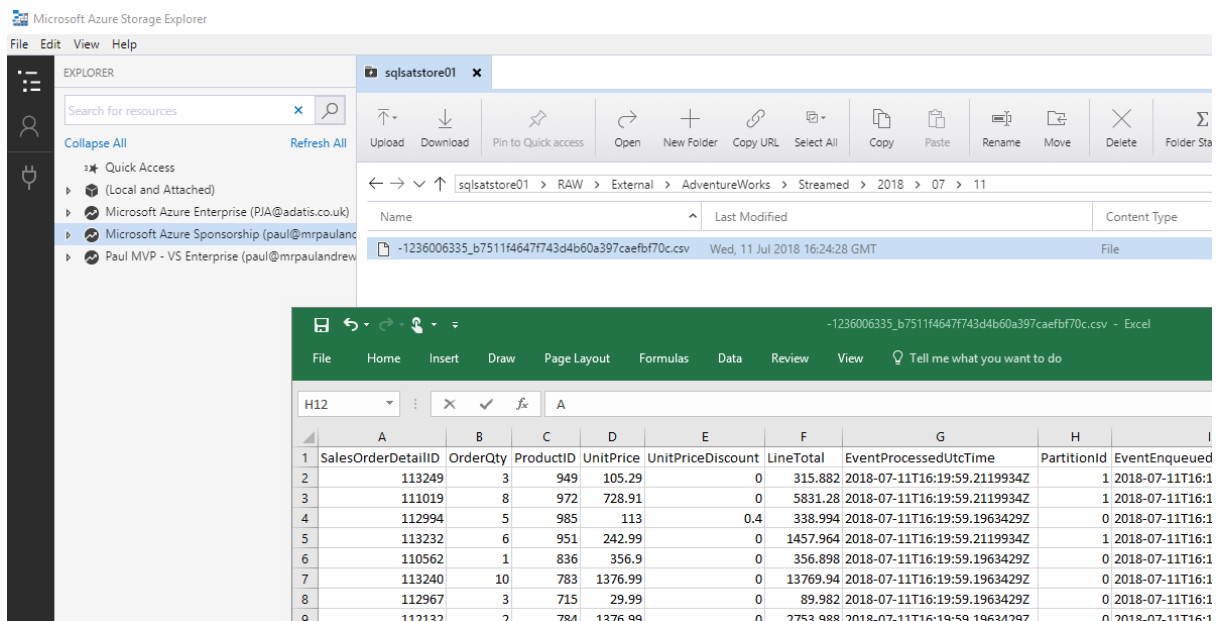
**Narrative:** data is flowing! You should now have real-time data ingested by your Event Hub and flowing through Stream Analytics to your Data Lake and Power BI outputs. Let's confirm this and create some streaming Power BI visuals.

5. Check Data Lake Store for data. This data should be arriving in Data Lake if you have correctly configured output 1 of your Stream Analytics job. To check the storage, we can use various tools. Follow options A, B or C below.
  - a. Option 1 – use the Azure Portal.
    - i. Navigate to your Data Lake Storage deployed and setup in Lab 1.
    - ii. Go to **Data Explorer** under the **Data Lake Storage** section.
    - iii. Browse to your defined output directory:  
(RAW/External/AdventureWorks/Streamed).



- iv. Review a couple of the files received in the browser reading blade. They should be in a CSV format.

- b. Option 2 – use the [Microsoft Storage Explorer](#) app.
  - i. Open the storage explorer app.
  - ii. Navigate to the Data Lake Storage deployed and setup in Lab 1.
  - iii. Browse to your defined output directory:  
(RAW/External/AdventureWorks/Streamed).
  - iv. Review a couple of the files received in a text editor. They should be in a CSV format.

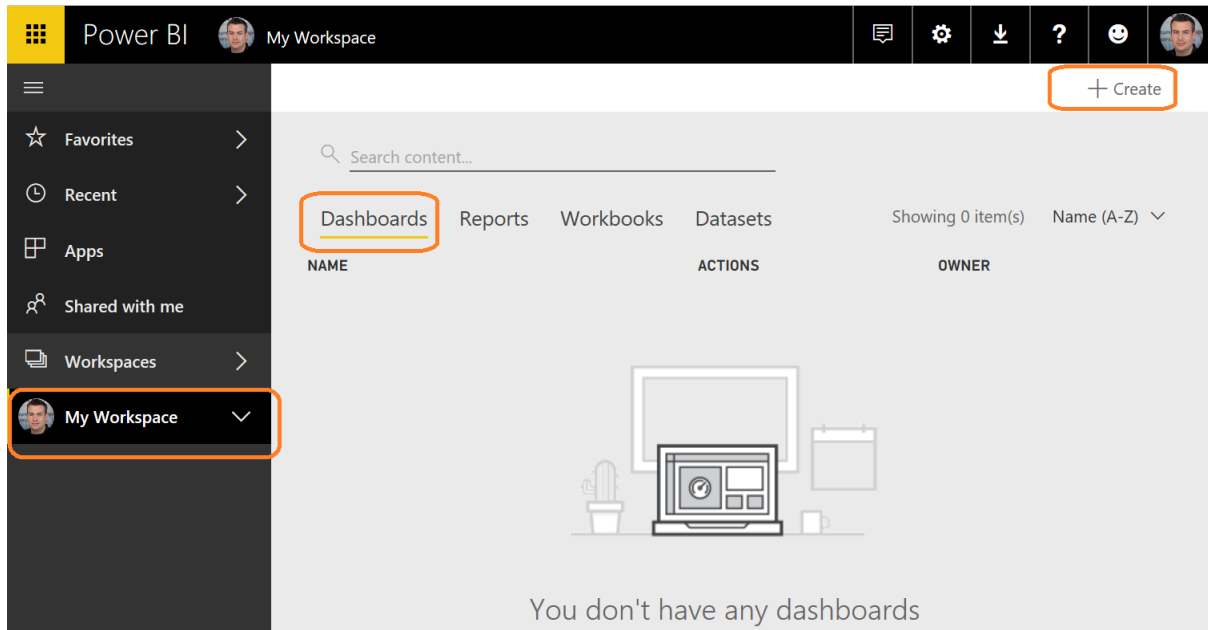


	A	B	C	D	E	F	G	H	I
1	SalesOrderDetailID	OrderQty	ProductID	UnitPrice	UnitPriceDiscount	LineTotal	EventProcessedUtcTime	PartitionId	EventEnqueued
2	113249	3	949	105.29	0	315.882	2018-07-11T16:19:59.2119934Z	1	2018-07-11T16:1
3	111019	8	972	728.91	0	5831.28	2018-07-11T16:19:59.2119934Z	1	2018-07-11T16:1
4	112994	5	985	113	0.4	338.994	2018-07-11T16:19:59.1963429Z	0	2018-07-11T16:1
5	113232	6	951	242.99	0	1457.964	2018-07-11T16:19:59.2119934Z	1	2018-07-11T16:1
6	110562	1	836	356.9	0	356.898	2018-07-11T16:19:59.1963429Z	0	2018-07-11T16:1
7	113240	10	783	1376.99	0	13769.94	2018-07-11T16:19:59.1963429Z	0	2018-07-11T16:1
8	112967	3	715	29.99	0	89.982	2018-07-11T16:19:59.1963429Z	0	2018-07-11T16:1
9	112132	2	784	1376.99	0	2753.988	2018-07-11T16:19:59.1963429Z	0	2018-07-11T16:1

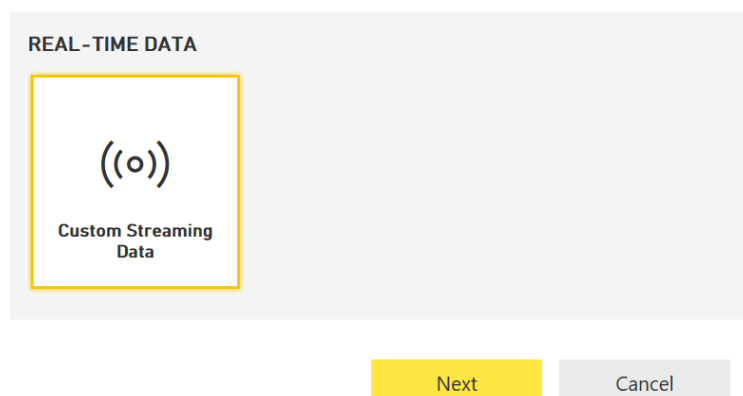
- c. Option 3 – use the Cloud Explorer extension for Visual Studio.
  - i. You can figure it out ☺

**Narrative:** having confirmed data is flowing in data lake let's check our other output to Power BI.

6. If you don't still have it open in a tab from step 1. Using the same 'Work/School' account used for your Azure subscription navigate to [PowerBI.com](https://PowerBI.com) and login.
  - a. Create a new dashboard. Click on **My Workspace**, then select **Dashboards** and click **Create**.



- b. Choose **Dashboard** from the dropdown menu and give it an inappropriate name in the presented pop up. Then click **Create**.
  - c. On the dashboard blank canvas now choose **+ Add Title** from the top right-hand toolbar.
  - d. Choose **Custom Streaming Data** from the panel presented.



- e. Select your dataset as defined in output 2 of your Stream Analytics job.

### Add a custom streaming data tile

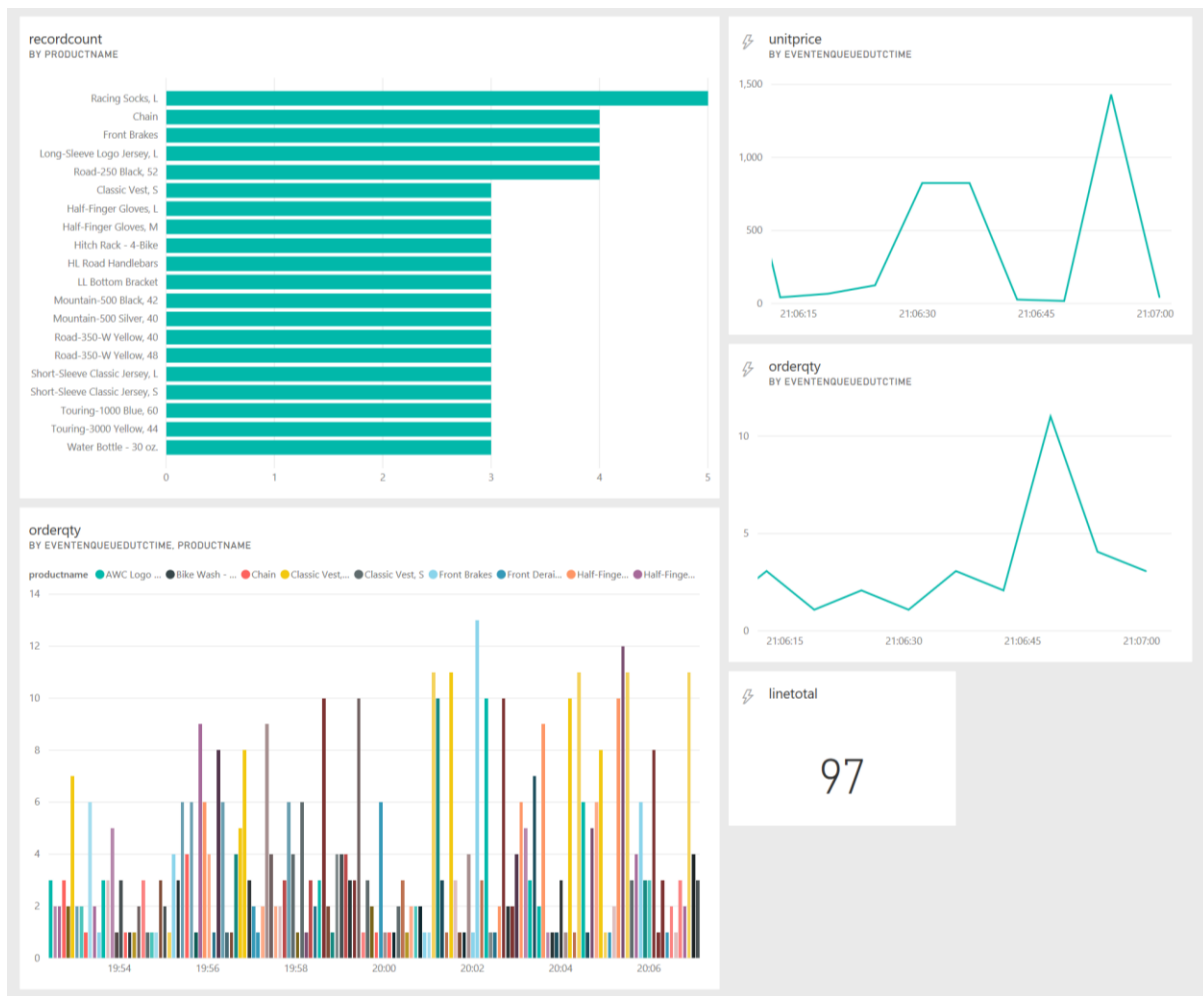
Choose a streaming dataset

+ Add streaming dataset

**YOUR DATASETS**

aw\_orderdetails

- f. Follow the wizard prompts to add a visual to your dashboard. Add a few visuals of varying types using the sales order data being sent from Stream Analytics. Example below. This dashboard includes both streaming visuals as well as visuals created from the 'push' dataset.



That concludes lab 2.

To recap, we've created.

- An Azure Event Hub.
  - Setup a data stream from a hypothetical 3<sup>rd</sup> party system.
- An Azure Stream Analytics job.
  - Configured;
    - An input from Event Hub
    - An input from a static reference dataset in Azure Blob Storage.
    - An output to Data Lake
    - An output to Power BI.com
- A PowerBI.com account.
  - Added streaming data titles to our dashboard.