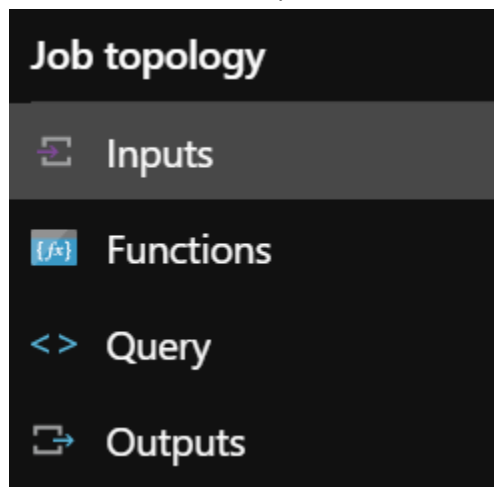


Ensure all your resources are in the same region or you will get charged extra.

Ensure you use the cheapest option for all the resources you created (Use Basic instead of Standard whenever applicable)

1. Create Blob
 - a. Create a storage account
 - b. Create a blob container
2. Create Event Hub
 - a. Create **Namespace**
 - b. Create event hub **instance** within namespace
3. Create Stream Analytics Job
 - a. Set the event hub instance you have created as the **input**
 - b. Set the Blob container you have created as the **output**(use Parquet as format)



4. Producer setup
 - a. Open producer.py
 - b. Install event hubs library for Python using pip (might also be available through other tools like Anaconda but no guarantees)
 - c. Fill **conn_str** and **eventhub_name** with the connection string and event hub name of your event hub **instance**(not the namespace)

```
#create a producer client
producer = EventHubProducerClient.from_connection_string(
    conn_str="YOUR HUB CONNECTION STRING",
    eventhub_name="YOUR HUB NAME"
)
```

5. Test connection
 - a. Run producer.py

- b. If you set up everything correctly, you will see “Send messages in xx seconds.” In your terminal output. There will be roughly one message sent per second for a total of 250

```
(start) PS C:\Users\Mao\source\repos\PythonSEP\SepDemo>
rs/Mao/source/repos/PythonSEP/SepDemo/streaming/produce
Send messages in 1.8554797172546387 seconds.
Send messages in 2.910527229309082 seconds.
Send messages in 3.9642348289489746 seconds.
Send messages in 5.023609399795532 seconds.
Send messages in 6.073307275772095 seconds.
Send messages in 7.128117322921753 seconds.
Send messages in 8.18008017539978 seconds.
Send messages in 9.311246633529663 seconds.
Send messages in 10.349803924560547 seconds.
Send messages in 11.400628566741943 seconds.
```

messages

- c. You should be able to see data coming in the overview dashboard for both your event hub instance and you stream analytics job.
6. Create query
 - a. In the “Query” tab in your stream analytics job, check the input preview, it should look like this

Input preview Test results

Showing events from 'eventhub0518'. This list of events might not be complete. Select a specific time range to show all events during that period.

View in JSON | Table | Raw | Refresh | Select time range | Upload sample input | Download sample data

id	timestamp	uv	temperature	humidity	EventProcessedUtcTime	PartitionId	EventEnqueuedUtcTime
"53b62047-5c27-4337-8a42-8a55b2c5...	"2021-05-18 15:40:45.600303"	0.8173306901933518	82	74	"2021-05-18T16:29:06.573590Z"	0	"2021-05-18T15:40:45.601000Z"
"323437d9-6082-4b4f-ba31-586eca0...	"2021-05-18 15:40:44.544469"	0.9627495672069886	86	80	"2021-05-18T16:29:06.573590Z"	1	"2021-05-18T15:40:44.544000Z"
"25340577-1b1d-4329-8a02-82e0c3c3...	"2021-05-18 15:40:43.491538"	0.6929518962862925	71	94	"2021-05-18T16:29:06.573590Z"	0	"2021-05-18T15:40:43.506000Z"
"b7d3c69b-3926-472a-9945-13283695...	"2021-05-18 15:40:42.452359"	0.9029463394874346	95	99	"2021-05-18T16:29:06.573590Z"	1	"2021-05-18T15:40:42.470000Z"
"d68756b2-43b7-47e4-836d-8612a448...	"2021-05-18 15:40:41.381740"	0.3114809440104569	87	93	"2021-05-18T16:29:06.573590Z"	0	"2021-05-18T15:40:41.396000Z"
"53b62047-5c27-4337-8a42-8a55b2c5...	"2021-05-18 15:40:40.341436"	0.8371657175502658	92	90	"2021-05-18T16:29:06.573590Z"	1	"2021-05-18T15:40:40.345000Z"
"53b62047-5c27-4337-8a42-8a55b2c5...	"2021-05-18 15:40:39.298206"	0.432489329073822	78	84	"2021-05-18T16:29:06.573590Z"	0	"2021-05-18T15:40:39.303000Z"
"ac58165-ec99-4d4e-a432-d04d03ee...	"2021-05-18 15:40:38.250759"	0.773385728728592	74	97	"2021-05-18T16:29:06.573590Z"	1	"2021-05-18T15:40:38.267000Z"
"b7d3c69b-3926-472a-9945-13283695...	"2021-05-18 15:40:37.214654"	0.6747739082824629	89	70	"2021-05-18T16:29:06.573590Z"	0	"2021-05-18T15:40:37.224000Z"
"323437d9-6082-4b4f-ba31-586eca0...	"2021-05-18 15:40:36.160310"	0.3436670470964972	82	91	"2021-05-18T16:29:06.573590Z"	1	"2021-05-18T15:40:36.173000Z"
"323437d9-6082-4b4f-ba31-586eca0...	"2021-05-18 15:40:35.118077"	0.20460173731552023	76	80	"2021-05-18T16:29:06.573590Z"	0	"2021-05-18T15:40:35.131000Z"
"ac58165-ec99-4d4e-a432-d04d03ee...	"2021-05-18 15:40:34.075010"	0.7596266179394422	82	90	"2021-05-18T16:29:06.573590Z"	1	"2021-05-18T15:40:34.079000Z"
"25340577-1b1d-4329-8a02-82e0c3c3...	"2021-05-18 15:40:33.020108"	0.5842782316279658	91	78	"2021-05-18T16:29:06.573590Z"	0	"2021-05-18T15:40:33.037000Z"

- b. Within your stream analytics job, write a SAQL window to do a Tumbling Window of 5 seconds
- c. Get **average uv, temperature, humidity and time** for each device within each window
 - i. Hint: estimated time is calculated by first converting the datetime to seconds since the Epoch(unix time stamp), get the average for each device within the current window, and then convert the unix time stamp back to datetime type.
- d. If average humidity and temperature for a device in a window are **both** higher than 85, mark that window as “Sweaty”
- e. Test your query with the data received from event hub

Query language docs | Open in Visual Studio | UserVoice

Inputs (1)

Test query | Save query | Discard changes

</> eventhub0518

- f. Your test result should look like this

Input preview		Test results				
Showing 39 rows from 'blob0422'.						
id	AverageUV	AverageTemp	AverageHumidity	IsSweaty	EstimatedTime	
"ff9f0714-c235-4a11-b818-4cec992eb896"	0.6809428358306182	90	95	1	"2021-05-18T15:40:03.0000000Z"	
"b7d3c69b-3926-472a-9945-13283695cf44"	0.5961249318818505	93.5	82.5	0	"2021-05-18T15:40:01.0000000Z"	
"4eb40547-80e5-4e6d-9e9f-e20f0f8b9df6"	0.9808423920243619	77	76	0	"2021-05-18T15:40:00.0000000Z"	
"44cee7c0-ae8d-47ac-8b39-69ef2e50cf0c"	0.5382794654357046	83	77	0	"2021-05-18T15:40:04.0000000Z"	
"253d0577-1bfd-4329-8a02-82e0c3c3410f"	0.3380572953527309	70	70	0	"2021-05-18T15:40:06.0000000Z"	
"4eb40547-80e5-4e6d-9e9f-e20f0f8b9df6"	0.054076536384488616	82	97	0	"2021-05-18T15:40:08.0000000Z"	
"44cee7c0-ae8d-47ac-8b39-69ef2e50cf0c"	0.7202291863079375	77.5	94	0	"2021-05-18T15:40:08.0000000Z"	
"4715262f-be04-4617-8783-b2bb6a04feb8"	0.44842202537348685	76	91	0	"2021-05-18T15:40:05.0000000Z"	
"253d0577-1bfd-4329-8a02-82e0c3c3410f"	0.602272615075533	95	87	1	"2021-05-18T15:40:13.0000000Z"	
"dd8756b2-43b7-47e4-83fd-f612a4482d83"	0.1316612492716308	98	95	1	"2021-05-18T15:40:14.0000000Z"	
"323437d9-6082-4b4f-ba31-586eeca08d54"	0.7915171146301364	82	79	0	"2021-05-18T15:40:12.0000000Z"	
"53b62047-5c27-4337-8442-8a55b2c54f6a"	0.13744209565639298	80	76	0	"2021-05-18T15:40:11.0000000Z"	
"ff9f0714-c235-4a11-b818-4cec992eb896"	0.4293633824361228	80	96	0	"2021-05-18T15:40:18.0000000Z"	
"b7d3c69b-3926-472a-9945-13283695cf44"	0.6216676054430621	92	82	0	"2021-05-18T15:40:19.0000000Z"	
✔ Success						

- g. After you are happy with the test result, go back to your stream analytics dashboard, click on "Start" and wait for the job to start running
- h. Now, rerun producer.py and check your blob container for the resulted Parquet files.
- i. You now have a pipeline that looks like this:

Producer.py -> Event Hubs -> Stream Analytics -> Blob Storage