

Construct a data pipeline to capture and process data in real-time using Azure Event Hub, Stream Analytics, and Storage Accounts. The pipeline should ingest data from an Event Hub, store it in a raw container, process the data, and then store the processed data in a separate container

## 1. Set Up Azure Resources

- Event Hub: Create an Event Hub namespace and an Event Hub instance within it.
- Storage Account: Create a Storage Account with two containers:
  - Raw Container: For storing unprocessed data directly from the Event Hub.
  - Processed Container: For storing cleaned and transformed data.
- Stream Analytics Jobs: Configure two Stream Analytics jobs:
  - Job 1: Ingests data from the Event Hub and stores it in the raw container.
  - Job 2: Reads data from the raw container, processes it (e.g., filtering, aggregation), and stores the results in the processed container.

## 2. Configure Event Hub and Generate Data

- Event Hub Configuration: Set up necessary authentication policies and connection strings.
- Data Generation: Use a sample script or Azure Event Hub data generator to simulate data with fields like `timestamp`, `sensorId`, and `temperature`.

## 3. Stream Analytics Job 1: Event Hub to Raw Container

- Input: Connect the job to your Event Hub.
- Sample Query:

```
SELECT *
INTO [raw-output]
FROM [event-hub-input]
```

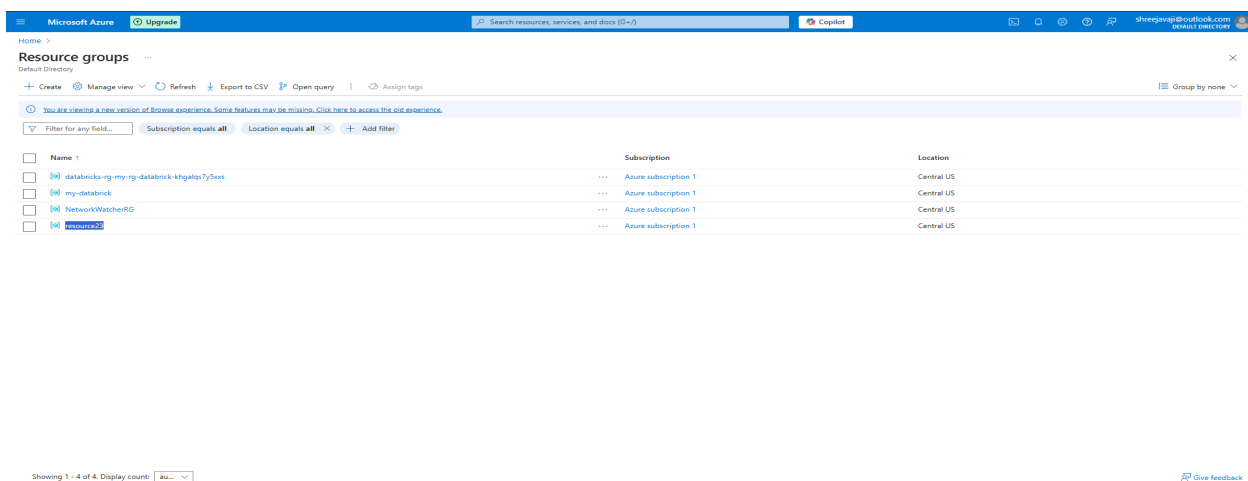
- Output: Configure the output to the raw container with a path pattern like:

```
raw/{date}/{hour}/events.json.
```

## Document for 1st Job:

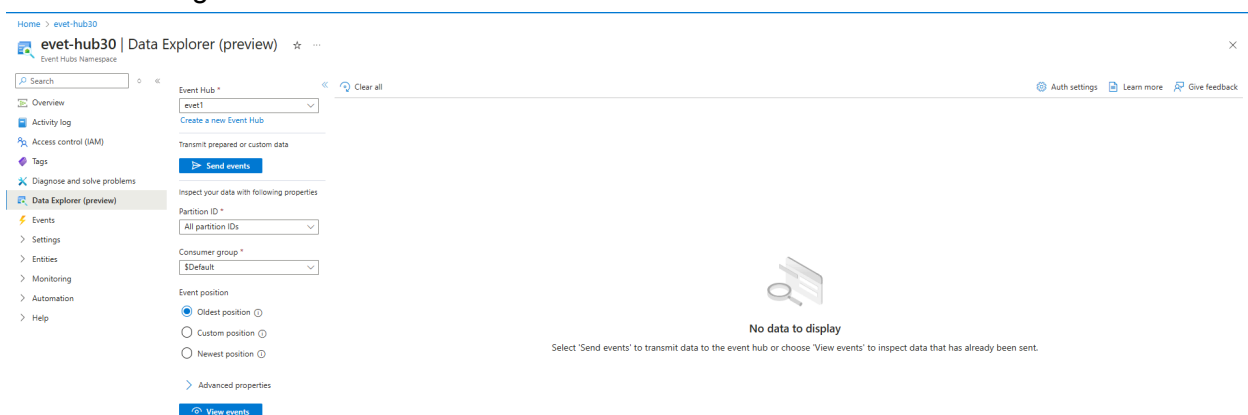
### Step 1: Create resource group

- Resource group name: “resource23”



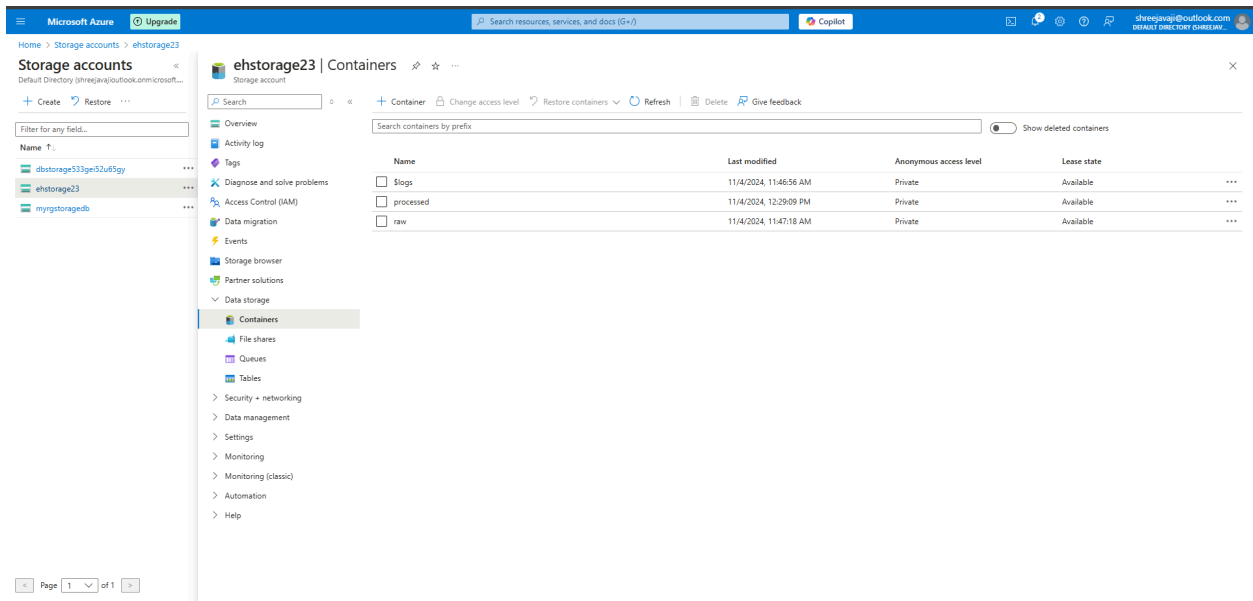
### Step 2: Create Event hub namespace and event hub instance.

- Creating a event hub namespace: “evet-hub30”
- Creating a event hub instance: “evet1”



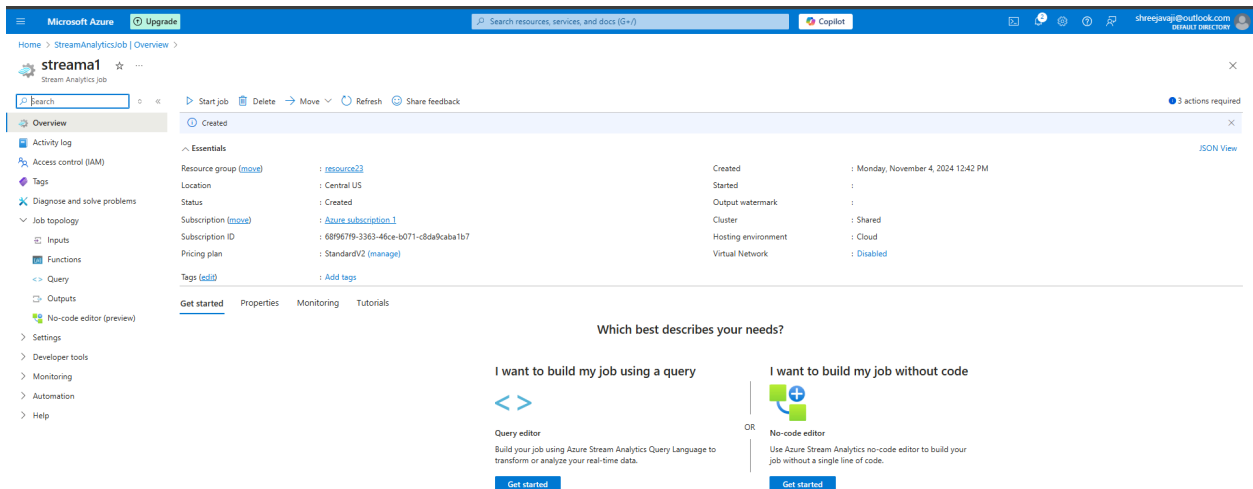
### Step 3: Create storage account

- Storage account name: “ehstorage23”
- Create 2 containers:
- Container 1 name: “raw”
- Container 2 name: “processed”

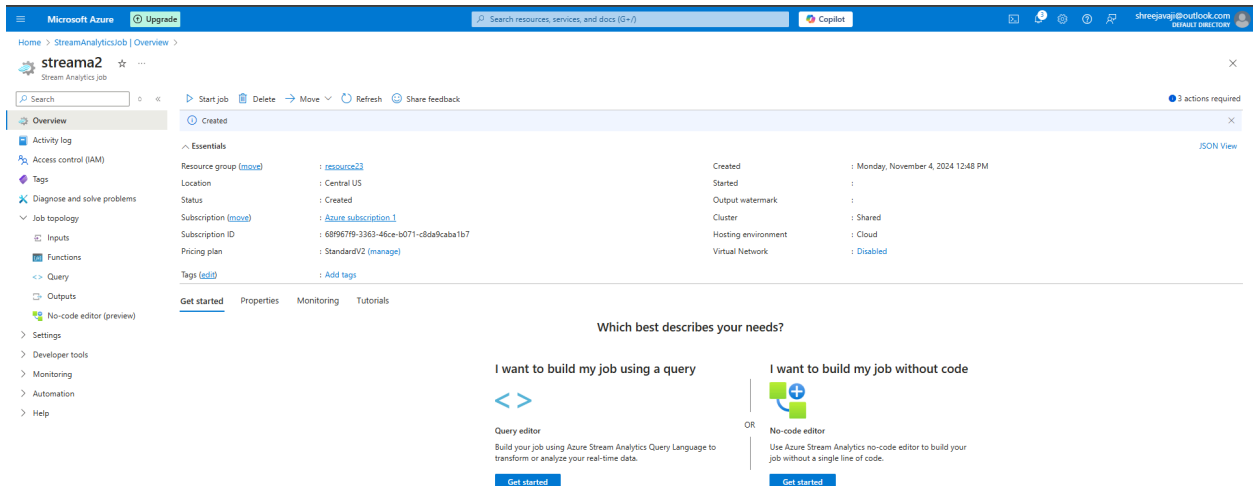


#### Step 4: Create Stream Analytics Job

- 1st Stream analytics name: “streama1”

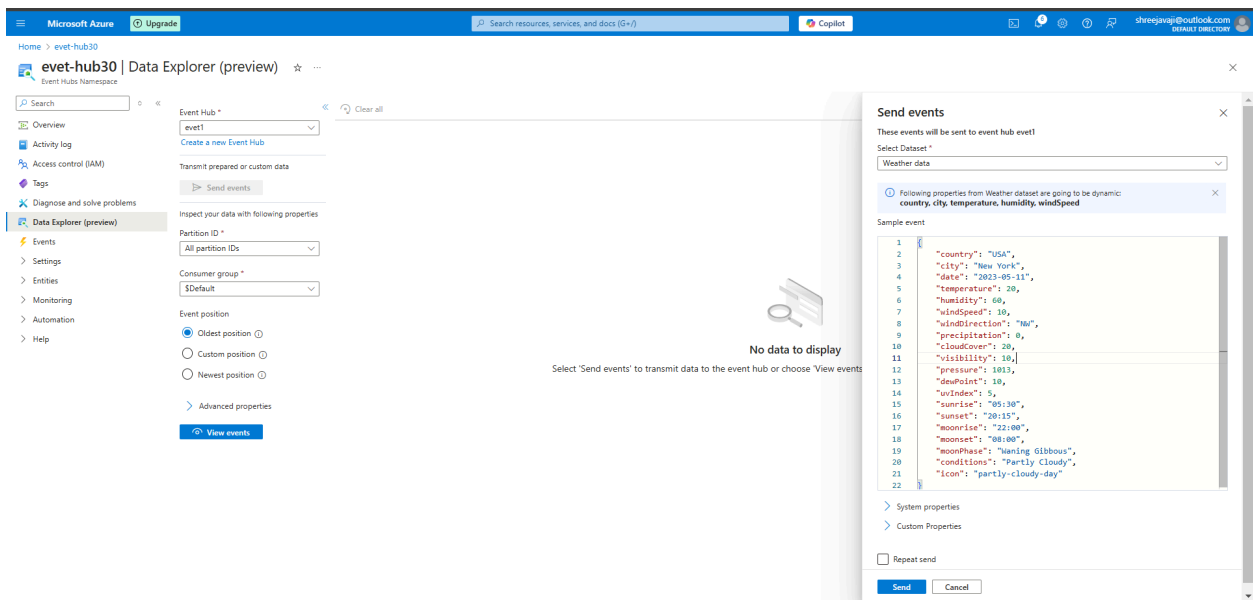


- 2nd Stream Analytics name: “streama2”



## Step 5: Go to Event hub - “evet-hub30”

- Select - Data Explorer
- Select - Send Events
- In “Select datasets” : select “Weather Data”

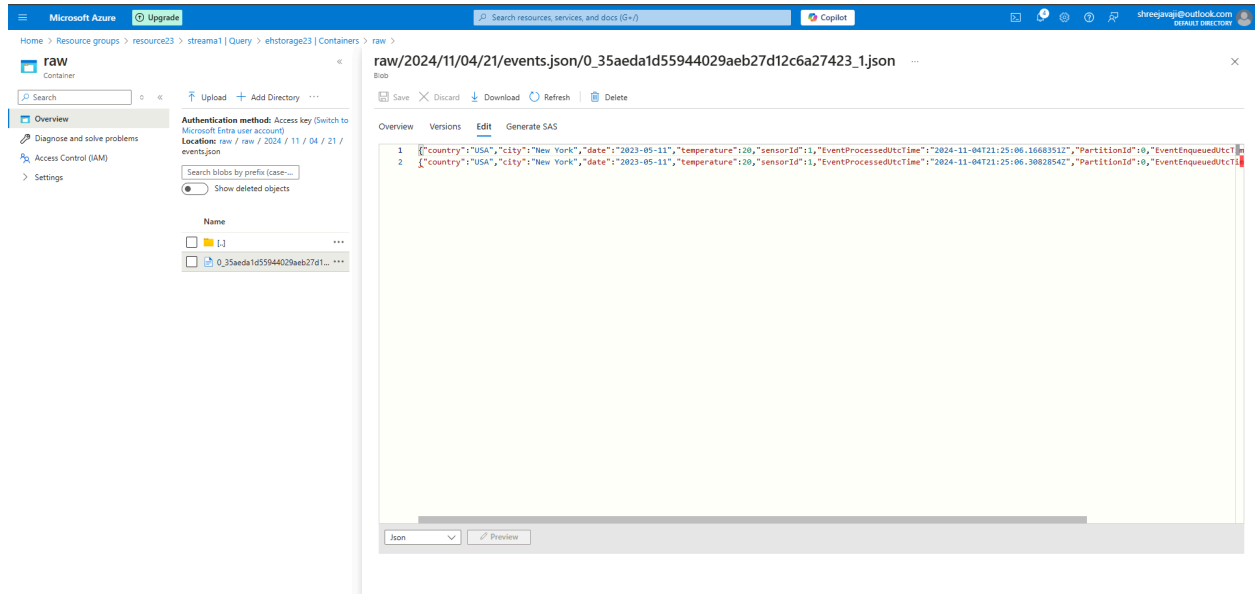


## Step 6: Go to Stream Analytics - streama1

- Select - Job Topology
- Select - Inputs
- Select - Add Inputs
- Select - “Event hub” and cross check the data.
- Select - Save
- Click on - Test Inputs

- Select - Outputs
- Select - Blob Storage/ADLS Gen2
- Select storage account - "ehstorage23"
- Select Container - "raw"
- Path Pattern - `raw/{date}/{hour}/events.json`.
- Save the path.
- Go to Event Hub - "evnt-hub30"
- Put the sample code:
 

```
{
  "country": "USA",
  "city": "New York",
  "date": "2023-05-11",
  "temperature": 20,
  "sensorId": 1
}
```
- Go to Stream Analytics - "streama1"
- Select - Query
- Select - Start Job - Start
- Go to Event Hub - "evnt-hub30"
- Click on send
- Go to Stream Analytics - "streama1"
- Click on refresh, it will fetch the data.
- Click on output "raw" select open blob storage.
- Select - container
- Select - raw
- Data will display.



- Stop the job.

## Stream Analytics Job 2: Raw Container to Processed Container

- Input: Configure the job to read from the raw container.
- Sample Query:

```
SELECT
    sensorId,
    AVG(temperature) AS AvgTemperature,
    System.Timestamp AS ProcessedTime
INTO [processed-output]
FROM [raw-input]
GROUP BY
    sensorId,
    TumblingWindow(minute, 5)
```

- Output: Configure the output to the processed container with a path pattern like:

```
processed/{date}/{hour}/processed_data.json.
```

# Document for 2st Job:

## Step 1: Go to Stream Analytics - “streama2”

- Select - Job Topology
- Select - Inputs
- Select - Add Inputs
- Select - “Blob Storage/ADLS Gen2” and cross check the data.
- Select storage account - “ehstorage23”
- Select Container - “raw”.
- Select - Save
- Click on - Test Inputs

The screenshot displays the Microsoft Azure portal interface for a Stream Analytics job named 'streama2'. The left-hand navigation pane shows the 'Inputs' tab selected under the 'Job topology' section. The main area shows a table of inputs with one entry: 'raw', which is a 'Stream' source type of 'Blob storage/ADLS Gen2' using 'Connection string' authentication. The right-hand pane, titled 'Input details', provides configuration options for the 'raw' input. It includes fields for 'Input alias' (set to 'raw'), 'Subscription' (set to 'Azure subscription 1'), 'Storage account' (set to 'ehstorage23'), 'Container' (set to 'raw'), 'Authentication mode' (set to 'Connection string'), 'Storage account key' (masked), 'Path pattern' (with a link to 'Use common path pattern: Application Insights'), 'Date format' (set to 'YYYY/MM/DD'), and 'Time format' (set to 'HH'). A 'Save' button is at the bottom of the details pane. A warning message at the top of the details pane states: 'Inputs can't be added or edited while a job is running. You can stop the job to add or edit inputs.'

- Select - Outputs
- Select - Blob Storage/ADLS Gen2
- Select storage account - “ehstorage23”
- Select Container - “Processed”
- Path Pattern - `processed/{date}/{hour}/processed_data.json`.
- Save the path.
- Click on - Test outputs

The screenshot displays the Microsoft Azure portal interface for a Stream Analytics job named 'streama2'. The 'Outputs' tab is active, showing a table with one output named 'processed'. The 'Output details' pane on the right provides configuration options for this output, including event serialization format (JSON), format (Line separated), encoding (UTF-8), write mode (Append), path pattern, date format, time format, and minimum rows.

Alias	Type	Authentication mode	Resource
processed	Blob storage/ADLS Gen2	Connection string	ehstorage23

**Output details for 'processed':**

- Event serialization format: JSON
- Format: Line separated
- Encoding: UTF-8
- Write mode: ☒ Append, as results arrive
- Path pattern: processed/(date)/(hour)/processed\_data.json
- Date format: YYYY/MM/DD
- Time format: HH
- Minimum rows: 1

- Select - Query
- Mention this query:

SELECT

```
sensorId,
AVG(temperature) AS AvgTemperature,
System.Timestamp AS ProcessedTime
```

INTO [processed-output]

FROM [raw-input]

GROUP BY

```
sensorId,
TumblingWindow(minute, 5)
```

- **Select - refresh**
- **Select - Start Job - Start**
- Click on refresh, it will fetch the data.
- Click on output “processed” select open blob storage.
- Select - container
- Select - processed
- Data will display.