Real-Time data streaming

with

Azure EventHub, Azure Stream Analytics job &

Azure Data Explorer

Lab Overview and Objectives

**Scenario:**  Imagine a meteorological organization that aims to enhance weather forecasting and improve disaster response strategies through real-time data analysis. The organization collects vast amounts of weather data from various sources, including weather stations, satellites, IoT sensors, and social media feedback. To remain competitive and responsive to changing weather conditions, they require a robust solution that allows for the continuous ingestion and processing of this data.

To achieve this, the organization implements a comprehensive solution using Azure services, including:

* **Azure Event Hubs** for real-time data ingestion of weather data streams.
* **Azure Stream Analytics** for processing and analysing the streaming data.
* **Azure Blob Storage** for durable data storage.
* **Azure Data Explorer** for advanced data analysis and visualization.

**Azure Event Hubs for Real-Time Data Ingestion:**

Azure Event Hubs is a fully managed, real-time data ingestion service designed to handle millions of events per second. It acts as a scalable platform for streaming data from multiple sources.

In this scenario, Event Hubs allows the retail company to ingest diverse data streams from various channels—such as sales transactions, website interactions, and customer feedback—into their data processing pipelines in real time.

Key Features:

- High Throughput: Supports millions of events per second, ensuring timely data ingestion.

- Partitioning: Enables distributed processing, enhancing data processing efficiency.

- Integration: Easily integrates with other Azure services, such as Azure Stream Analytics and Azure Databricks.

**Azure Stream Analytics for Real-Time Processing:**

Azure Stream Analytics is a real-time analytics service designed to process and analyze streaming data from various sources.

It enables the retail company to apply real-time analytics to the incoming data, performing operations such as filtering, aggregating, and enriching the data.

Key Features:

- Real-Time Insights: Allows for immediate analysis of incoming data streams.

- SQL-Like Queries: Users can write familiar SQL queries to process and analyze data.

**Azure Blob Storage for Data Storage:**

Azure Blob Storage is a scalable cloud storage solution for unstructured data.

It serves as a central repository for storing processed data in various formats (e.g., CSV, JSON), making it accessible for further analysis.

Key Features:

- Cost-Effective: Offers a low-cost option for storing large volumes of data.

- Flexible Structure: Data can be organized in a hierarchy for easy retrieval.

What You'll Learn in This Lab

1. Azure Event Hubs:

- Set up for real-time data ingestion.

- Handle millions of events per second from diverse sources.

2. Azure Stream Analytics Job:

- Process and analyze streaming data in real time.

- Write SQL-like queries to filter, aggregate, and enrich data.

3. Azure Blob Storage:

- Store processed data in various formats for accessibility.

- Organize data efficiently for easy retrieval.

4. Azure Data Explorer:

* Azure Data Explorer (ADX) is a fast and highly scalable data analytics service for real-time analysis of large volumes of data.
* It allows users to ingest, explore, and visualize data quickly using a powerful query language called Kusto Query Language (KQL).
* ADX is ideal for analysing time-series data, logs, and telemetry, making it suitable for applications such as monitoring and diagnostics

After Completing This Lab, You Will Be Able To

1. Configure Azure Event Hubs for real-time ingestion
2. Ingest Data to Azure Event Hub Using Cloud Shell
3. Create Azure Storage and containers for durable storage of data
4. Configure Azure Stream Analytics job to read streaming JSON data from Azure Event Hub and store it in your Blob Storage container as CSV file
5. Create Azure Data Explorer clusters, databases and tables
6. Ingest data from Azure Blob Storage into Azure Data Explorer (ADX) to create tables and perform one-time data loading.
7. Continuously load data from Azure Event Hubs into Azure Data Explorer (ADX) tables for real-time analytics and build dashboards

**Prerequisites**

**Azure Account:** This lab requires you to sign up for Azure. Please sign up for an [Azure Student account](https://azure.microsoft.com/en-us/free/students/) to perform this lab. It is free and includes various Azure services.

**TASK 1: Creating Azure Event Hub**

Create an Event Hubs Namespace

Step 1. Navigate to Event Hubs:

* In the Azure portal, use the search bar at the top and type "Event Hubs."
* Select Event Hubs from the search results.

Step 2. Add a New Namespace:

* Click on the create Event Hubs namespace button.
* Fill in the required fields:
* Name: Enter a unique name for your Event Hub namespace (e.g., ` Weatherdataspace `).
* Subscription: Choose your Azure subscription.
* Resource Group: Select an existing resource group or create a new one (e.g., `WeatherDataGroup`).
* Location: Choose a region close to your data sources (e.g., `East US`).
* Pricing Tier: Select the pricing tier Standard
* Keep the Throughput Units as 1 which is a default value

Step 3. Review + Create:

* Click Review + create.
* After validating your settings, click Create to create the Event Hubs namespace. You will see Deployment is in progress and this may take a few moments.
* Once the deployment is complete, proceed with the below steps

Create an Event Hub

Step 1. Navigate to the Created Namespace:

After the namespace is created, go to the resource by clicking on its name.

Step 2. Add an Event Hub:

* In the namespace overview, find and select Event Hubs under the Entities section.
* Click on + Event Hub to create a new Event Hub.
* Configure Event Hub Settings:
* Fill in the required fields:
* Name: Enter a name for your Event Hub (e.g., `WeatherDataHub`).
* Partition Count: Choose the number of partitions (default is usually fine but consider your expected load). Keep it as 1
* Message Retention: Set the retention period for messages (default is 1 day; adjust based on your requirements).

Step 3. Create the Event Hub:

Click on Review + Create to validate and once the validation is successful then click on Create to finalize the creation of your Event Hub.

Configure Shared Access Policies

Step 1. Access Policies:

- In the Event Hub namespace, navigate to the Shared access policies under the Settings section.

- Click on the + Add button to create a new access policy.

Step 2. Set Access Permissions:

- Enter a name for the policy (e.g., `WeatherDataPolicy`).

- Select permissions:

- Send: Check this to allow sending messages to the Event Hub.

- Listen: Check this to read from the Event Hub.

- Click Create to add the policy.

3. Copy Connection String:

- After creating the policy, click on it to view its details.

- Copy the Connection String–Primary Key for later use in your data ingestion code.

**Task 1 Summary:**

You have now successfully created an Azure Event Hub for ingesting weather data. You can use the connection string to send data from your Python script or any other data source

Endpoint=sb://weatherdataspace.servicebus.windows.net/;SharedAccessKeyName=WeatherDataPolicy;SharedAccessKey=QYWsEv3RLdGXptSvt5vyr/i1LAiNdWZa1+AEhBX5DLg=

**Task 2: Ingest Data to Azure Event Hub Using Cloud Shell**

Step 1: Open Azure Cloud Shell

Access Cloud Shell: Click on the Cloud Shell icon (a terminal icon) at the top of the portal.

Step 2: Set Up Your Python Environment

* Select Bash: If prompted, choose the Bash environment in Cloud Shell.
* Select Subscription if prompted and keep the default settings for other options
* Install Required Libraries: Install the Azure Event Hubs SDK if it’s not already available in Cloud Shell:

pip install azure-eventhub

Step 3: Create Your Python Ingestion Script

Create a New Python File: Use a text editor like `nano` to create a new Python script:

nano ingest\_weather\_data.py

Add the Ingestion Code: Copy and paste the following code into the file, replacing `Your\_Event\_Hub\_Connection\_String` and `Your\_Event\_Hub\_Name` with your actual Event Hub connection string and name:

In the place of Your\_Event\_Hub\_Connection\_String, you need to add the connection string -primary key of Weather data policy which we created in task1

**Note**: You can either create a file and copy- paste the below code or upload the file ingest\_weather\_data.py which is there in MyCourses to the cloud shell and proceed with the next steps

The below code will generate the streaming data and ingest it to Event Hubs with the attached Event hub name and connection string

import time

import random

import json

from azure.eventhub import EventHubProducerClient, EventData

from datetime import datetime

# Connection string and Event Hub name

connection\_string = "”

eventhub\_name = "weatherdatahub"

def generate\_weather\_data():

"""Generate random weather data."""

return {

"temperature": random.randint(-10, 40), # Random temperature between -10 and 40

"humidity": random.randint(0, 100), # Random humidity between 0 and 100

"windSpeed": random.randint(0, 20), # Random wind speed between 0 and 20

"windDirection": random.choice(["N", "NE", "E", "SE", "S", "SW", "W", "NW"]),

"precipitation": random.choice([0, 1]), # 0 for no precipitation, 1 for precipitation

"conditions": random.choice(["Sunny", "Cloudy", "Partly Cloudy", "Rain", "Snow"]),

"eventtime": datetime.utcnow().isoformat() + "Z" # UTC timestamp in ISO format

}

def main():

# Create a producer client

producer = EventHubProducerClient.from\_connection\_string(connection\_string, eventhub\_name=eventhub\_name)

try:

while True:

weather\_data = generate\_weather\_data()

event\_data = EventData(json.dumps(weather\_data)) # Convert dict to JSON string

producer.send\_event(event\_data)

print(f"Sent event: {weather\_data}")

time.sleep(5) # Send every 5 seconds (adjust as needed)

except Exception as e:

print(f"Error sending events: {e}")

finally:

producer.close()

if \_\_name\_\_ == "\_\_main\_\_":

main()

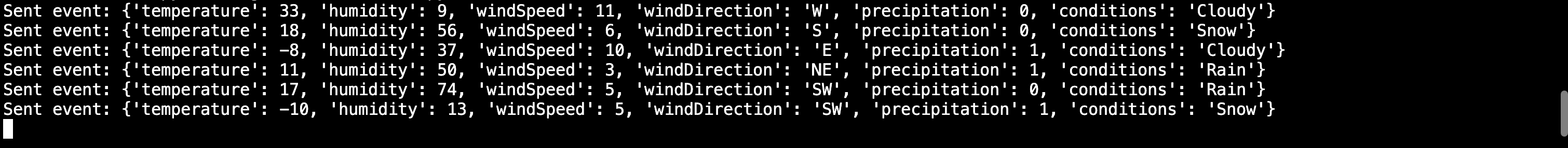
Save and Exit: Press `CTRL + X`, then `Y`, and hit `Enter` to save and exit the editor.

Step 4: Run the Ingestion Script

Execute the Script: Run the Python script to start sending weather data to your Event Hub:

python ingest\_weather\_data.py

Monitor Output: The script will print the generated weather data being sent to the Event Hub every second. You can stop the script by pressing `CTRL + C`.



Step 5: Verify Data Ingestion

Check Azure Portal: You can verify that the data is being ingested by checking the metrics of your Event Hub in the Azure portal. Look for incoming messages in the Metrics section.

Note: Stop the script execution by pressing `CTRL + C`.

**Task 2 Summary:**

You have successfully ingested weather data into your Azure Event Hub using a Python script in Azure Cloud Shell.

**TASK 3 – Create an Azure Storage Account:**

Create an Azure Storage Account

Step 1. Create a Storage Account:

* + Click on Create a resource in the upper-left corner.
  + Search for Storage Account and select it from the dropdown.
  + Click Create.

Step 2. Configure the Storage Account:

* Basics Tab
* Subscription: Select your Azure subscription.
* Resource Group: Choose an existing resource group which we created for this lab (WeatherDataGroup)
* Storage Account Name: Enter a unique name (e.g., `weatherdatastorage123`).
* Region: Choose the Azure region closest to you.
* Performance: Choose Standard.
* Replication: Choose a replication option (e.g., LRS for locally redundant storage).
* Advanced Tab (optional): You can leave defaults
* Review + Create Tab: Review your settings and click Create. Wait for the deployment to complete.

Once the deployment is complete, proceed with the below steps

Create a Blob Container

Step 1. Go to Your Storage Account:

* Once the storage account is created, navigate to it from the Azure Portal dashboard or through All resources.

Step 2. Create a Blob Container:

* + In the Data storage account menu, click on Containers.
  + Click on + Container at the top of the Containers page.
  + Name Your Container: Enter a name for your container (e.g., `weatherdata`).
  + Public Access Level: Set this to Private (no anonymous access) for security.
  + Click Create to create the container

Now that you have set up your Azure storage account and container for weather data, you can store streaming data from Azure Event Hub into this storage account using Azure Stream Analytics

**TASK 3 Summary:**

You now have a storage account, and a container set up in Azure to store weather data.

**TASK 4: Create an Azure Stream Analytics Job**

Step 1. Create a Stream Analytics Job:

* + Click on Create a resource in the upper-left corner.
  + Search for Stream Analytics Job and select it.
  + Click Create.
  + Fill in the required fields:
  + Job name: Enter a unique name for your job (e.g., `WeatherDataJob`).
  + Subscription: Select your Azure subscription.
  + Resource Group: Choose an existing resource group WeatherDataGroup
  + Location: Select the Azure region.
  + Hosting Environment as cloud
  + Click Review + Create, then Create.

Step 2: Configure Input from Event Hub

* Open the Stream Analytics Job: Once created, navigate to the job.

**Add Input:**

* In the job topology in the side menu bar, click on Inputs in the left menu.
* Click on + Add input and select Event Hub.
* Fill in the details:
* Input alias: Name your input (e.g., `WeatherInputJson`).
* Use the existing Event Hub name – Weatherdatahub and use existing event hub consumer group $Default
* Select the Event hub policy we created under Event Hub Policy name - WeatherDataPolicy
* Keep the authentication mode as connecting string
* Event serialization format - Choose JSON
* Click Save.

Step 3: Configure Output to Blob Storage

**Add Output:**

* In the job topology in the side menu bar, click on Outputs.
* Click on + Add output and select Blob storage /ADLS Gen2.
* Fill in the details:
* Output alias: Name your output (e.g., `WeatherOutputCsv`).
* Storage account: Select the storage account you created earlier.
* Container: Select the container where you want to store the data (e.g., `weatherdata`). Use the existing container which we created
* Keep the authentication mode as connecting string
* Select the write mode to append
* Event serialization format - Choose CSV – Comma(,)
* Click Save.

Step 4: Define a Query

Set Up the Query:

* In the left menu under Job topology, click on Query.
* Write a SQL-like query to process your incoming data.
* This query selects all columns from the Event Hub input and sends it to the Blob Storage output.
* Save the query

SELECT

\*

INTO

[WeatherOutputCsv]

FROM

[WeatherInputJson]

Note : Query can't be edited while a job is running. You can

stop the job to edit the query.

Grant permission for the Stream Analytics Job to access Azure Container.

* Navigate to Storage account and click on your storage account and go to the container you created.
* Click on Access Control IAM and click on Add role assignment
* Under Roles, select Storage Blob Data Contributor role and then click on Next
* Under Members, select Managed Identity and click on + Select members
* Choose the subscription and select the Stream Analytics job under Managed identity and select the WeatherDatajob member we created for this lab
* Click on Review + assign.

A screenshot of a computer

Description automatically generated

Step 5: Start the Stream Analytics Job

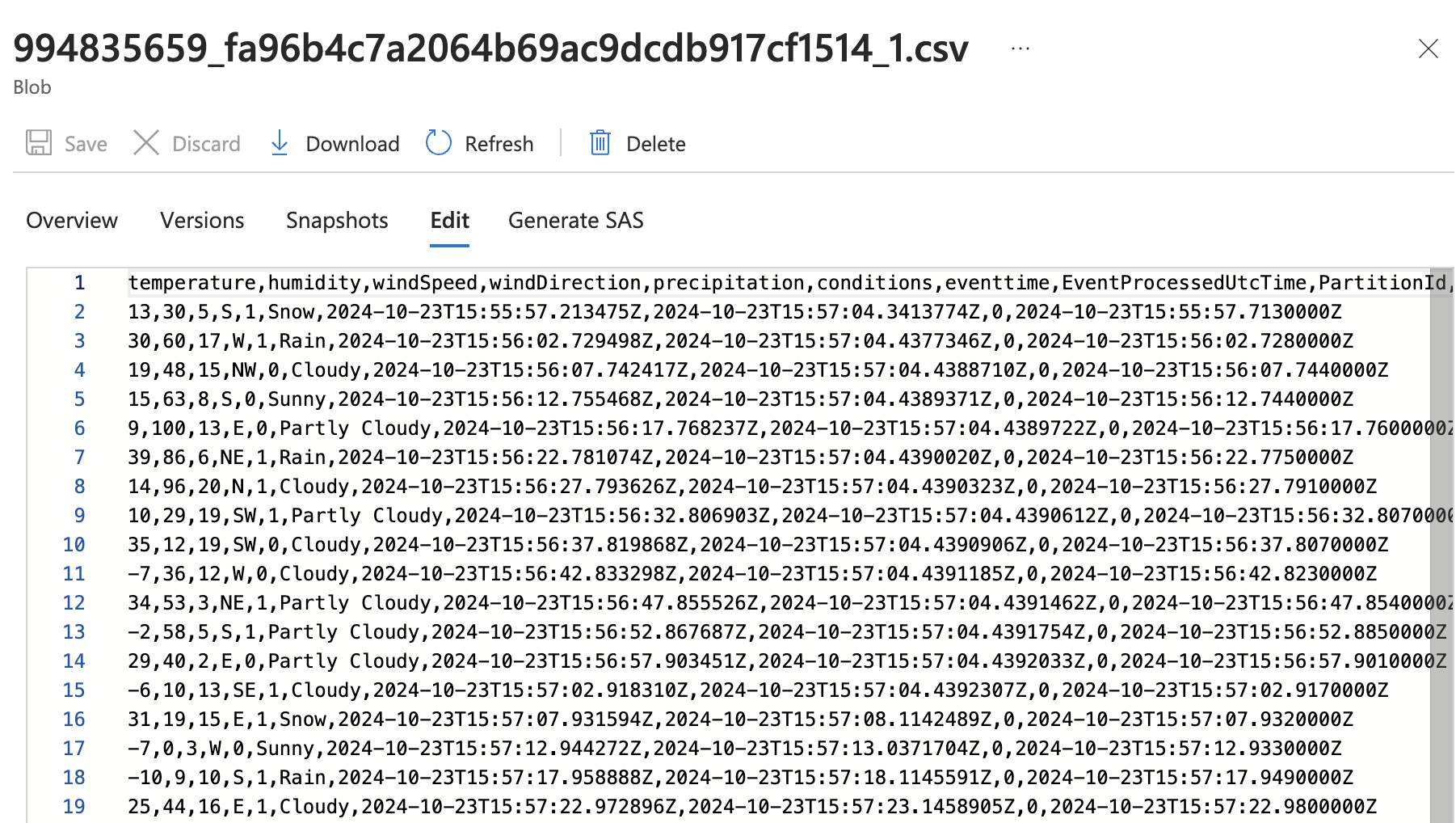
Start the Job:

* + After you have configured the input, output, and query, click on Start job in the job overview.
* Choose the Start option and confirm to start processing the data. Keep the default settings for Stream units, Environment and Job output start time as Now.
  + Navigate to Cloud Shell and exec the below command to stream the data to Event hubs

python ingest\_weather\_data.py

Step 6: Monitor the Job

Once you trigger the command, data will be generated and sent to the stream every 5 seconds, as configured, and stored in the container. Navigate to the storage account named `weatherdatastorage1234`, click on the container, and you will see the streamed data there. Click on the container and select Edit to view the CSV data streamed from Azure Event Hubs.



- Use the Monitoring section in the Stream Analytics job to view metrics and check for errors.

**Note: Stop the script execution by pressing `CTRL + C` in the cloud shell**

**Task 4 Summary**

You have now successfully set up Azure Stream Analytics to read streaming JSON data from Azure Event Hub and store it in your Blob Storage container as CSV file

**TASK 5 – CREATING AZURE DATA EXPLORER CLUSTERS & DATABASE**

Step 1: Create a New Resource

* + In the left-hand menu, click on Create a resource.
  + In the "Search the Marketplace" box, type Azure Data Explorer Cluster and select it from the dropdown list.
  + On the Azure Data Explorer page, click the Create button.

Step 2: Configure the Data Explorer Cluster

* Select the Azure subscription you want to use for the cluster.
* Select an existing resource group or create a new one by clicking on Create new.
* Cluster Name: Provide a unique name for your cluster . E.g. weathercluster1
* Region: Choose the Azure region where you want to deploy the cluster (e.g., East US).
* In the Compute Specification, select the workload as Dev/test
* And toggle the availability zones to No zones
* Review and Create:
* Click on Review + create and check all the settings. Then click on Create to deploy your cluster.

This may take some time.

Once the deployment is successful Go to the resource and Click on Data🡪 Databases

Creating Databases:

* Click on + Add Database and enter the Database name -weatherdb
* Keep the Retention period and Cache period as 1 and Create the database

Task 5 Summary:

In this task, we created Azure Data Explorer (ADX) clusters and databases, which are essential for storing and analysing large volumes of data in real time.

**Task 6 – Loading data from Azure Storage to Azure Data Explorer(One-time data loading)**

Ingest Data

* Navigate to the Azure Data Explorer Cluster weathercluster1 and select Data Ingestion 🡪Ingest

Using the Ingest option we can set up data ingestion from various sources, such as Azure Blob Storage, Event Hubs, or IoT Hub.

In this lab, we will be ingesting data from Azure storage which is one-time data capture

A screenshot of a computer

Description automatically generated

* This will open a new window to Azure Data Explorer.
* Navigate to Home🡪Get Data 🡪Select a data Source.
* Here the data source is Azure storage. So click on Azure Storage .
* In the configure section, you will find the Cluster and database you created earlier. Clicking on Database, You can enter the table name for the new table
* And configure the data source as shown below.

A screenshot of a computer

Description automatically generated

* Select the source as Select container
* Choose the subscription
* Choose the storage account and Container that you created for the weather data from the drop down.
* Clicking on next, you will be able to preview the data from the container and then click on Finish
* Once the table is created, you can view the Summary of getting data from Azure storage to Azure Data Explorer cluster database
* Toggle First row is column header to exclude the header columns and click on Finish

A screenshot of a computer

Description automatically generated

* Navigate to Azure Data Explorer🡪Query.

Here you will be able to Query the data from Azure storage

SELECT \* FROM weather where temperature is not null

The above query will display the results like below

A screenshot of a computer

Description automatically generated

TASK 6 Summary:

In this task we established a connection between Azure Data Explorer and the Azure Storage account and then we transferred the data from the storage account into the desired table within Azure Data Explore

**Note**: You can now stop the Azure Stream Analytics Job.

Navigate to Azure Stream Analytics job 🡪 Stop job

**TASK 7 – Continuous data loading from Event Hubs**

* Navigate to the Azure Data Explorer Cluster weathercluster1 and select Data Ingestion 🡪Ingest
* This will open a new window to Azure Data Explorer.
* Navigate to Home🡪 Get Data 🡪Select a data Source.
* Here the data source is Event Hubs.
* Click on the Azure Data Explorer cluster connection that we created and select the existing table which we created (weather ) in task 6.

Configure the data source

* Choose the subscription .
* Select the Event hub name space and Event hub from the dropdown
* Choose the consumer group as Default and default Data connection name

A screenshot of a computer

Description automatically generated

Preview the data and click on Edit columns and delete the duplicate columns and select the corresponding source columns from the dropdown for each column names

A screenshot of a computer

Description automatically generated

Data preview should look like below

A screenshot of a computer

Description automatically generated

Summary of the Get data from Event Hubs to Azure Data Explorer will look like below

A screenshot of a computer

Description automatically generated

Execute the below query in the Azure Data Explorer🡪Query.

SELECT \* FROM weather where temperature is not null

**GRANTING ACCESS TO AZURE DATA EXPLORER:**

* Navigate to the Event Hub Namespace we created.
* Event Hubs Instance 🡪 Access Control(IAM) 🡪 Add Role Assignment
* Select the Role - Azure Event Hubs Data Receiver
* Assign Access to managed identity- Azure Data Explorer cluster and then select the Azure Data Explorer that you created and then click on Review+assign

A screenshot of a computer

Description automatically generated

* + Navigate to Cloud Shell and exec the below command to stream the data to Event hubs

python ingest\_weather\_data.py

Note: Let the above command run for 10 minutes and then stop the script execution by pressing `CTRL + C`.

This will ingest data from Event Hubs to Azure Data Explorer table weather

Execute the below query.

select \* from weather

Note: There could be some potential delays in streaming data from Azure Event Hubs to the weather table in Azure Data Explorer. Reasons for these delays may include:

1. **Data Ingestion Latency**: Natural delays in processing events.
2. **Throughput Limits**: Exceeding Event Hubs’ throughput can slow data transfer.
3. **Resource Availability**: High load on Azure Data Explorer may affect performance.

Wait for some time and query to check if the data is loaded to weather table in Azure Data Explorer.

Execute the below Query to view the average temperature recorded for every 5 minutes

SELECT

DATEADD(MINUTE, DATEDIFF(MINUTE, 0, eventtime) / 5 \* 5, 0) AS time,

AVG(temperature) AS avg\_temperature

FROM

weather

WHERE

temperature IS NOT NULL

GROUP BY

DATEADD(MINUTE, DATEDIFF(MINUTE, 0, eventtime) / 5 \* 5, 0)

ORDER BY

time;

The above query returns the average temperature of the weather data for every 5 mins

**Adding visualization to this query and Pin to Dashboard**

* Click on Add Visual and enter Title name as Average Temperature and Visual type as Line chart
* Click on Pin to Dashboard and enter the Title name as Average Temperature
* Select Create a new dashboard and give the dashboard name – Average Temperature
* Click on pin

A screenshot of a computer

Description automatically generated

Navigate to dashboards in Azure Data Explorer to view the real time visualizations.

Refresh the dashboard to view the real time visualizations on the streaming data.

A screenshot of a video

Description automatically generated

TASK 7 Summary

In this task, streaming data from Azure Event Hubs was ingested into the Azure Data Explorer (ADX) table, enabling real-time data processing. Visualizations and dashboards were created to analyse and present the streaming weather data effectively. This setup allows for immediate insights and monitoring of weather conditions as new data arrives.

**Note: Delete all resources created for this lab once the lab is complete.**

**Lab complete**

Congratulations! You have completed the lab.