ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

BIG DATA MANAGEMENT SYSTEMS: PROJECT #3 AZURE STREAM ANALYTICS

Big Data Management Systems

Supervisor: Prof. Damianos Chatziantoniou

Dimitrios Bouris (8190119)
Filippos Priovolos (8190147)

Contents

GitHub Repository

To avoid turning out repository Public we used a tool called Gitfront. GitFront is used to share private Git repositories without making them public to people who do not necessarily have GitHub accounts.
—---
The Gitfront link to view our repository can be found here.
—---

Note: *In the event of a non-responding link please contact us.*

Project Description

This assignment involves using Azure Stream Analytics to process a continuous data stream of ATM transactions and generate responses to stream queries. The data stream is emulated using a data generator which sends data within a fixed time interval. The stream schema consists of four fields: ATMCode, CardNumber, Type, and Amount. The objective is to leverage Azure Stream Analytics capabilities to analyze the data stream and provide insights and answers to specific queries. The assignment requires setting up an Event Hub for data ingestion, configuring a Storage account for reference data files, establishing a Stream Analytics Job for processing, and executing predefined queries on the data stream. The focus is on effectively utilizing Azure Stream Analytics to handle the real-time data stream and extract meaningful information. In the Azure Stream Analytics job, a plethora of queries needs to be modeled and run on the stream. The data will be saved on a Storage Blob.

Assignment

You are going to use Azure Stream Analytics to process a data stream of ATM transactions and answer stream queries.

Data

The Data Generator creates ATM data in JSON format:

- ATMCode
- CardNumber
- Type
- Amount

Example ATM event:

```
{ "ATMCode": 10, "CardNumber": 4026567514157759, "Type": 1, "Amount": 42 }
```

A set of reference files are provided for the Stream Analytics Job. These files contain the following information:

1. **Customer.json:** Personal data about customers who have made transactions. Each customer example is described by the following attributes:
    - card_number (integer)
    - rst_name (string)
    - last_name (string)
    - age (integer)
    - gender (string)
    - area_code (integer)
2. **Atm.json:** Describes ATMs as:
    - atm_code (integer)
    - area_code (integer)
3. **Area.json:** Describes areas as:
    - area_code (integer)
    - area_country (string)
    - area_city (string)

Queries

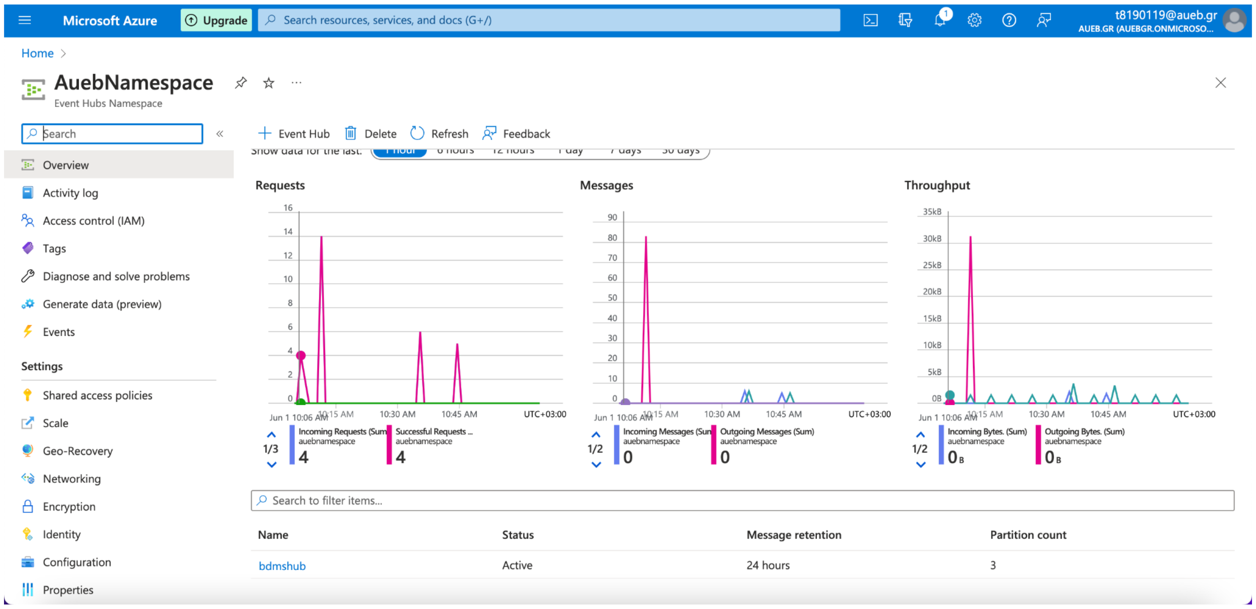
A set of queries should be modeled to process the incoming data. The queries asked are the following:

1. Show the total `Amount` of `Type = 0` transactions at `ATM Code = 21` of the last 10 minutes. Repeat as new events keep flowing in (use a sliding window).
2. Show the total `Amount` of `Type = 1` transactions at `ATM Code = 21` of the last hour. Repeat once every hour (use a tumbling window).
3. Show the total `Amount` of `Type = 1` transactions at `ATM Code = 21` of the last hour. Repeat once every 30 minutes (use a hopping window).
4. Show the total `Amount` of `Type = 1` transactions per `ATM Code` of the last one hour (use a sliding window).
5. Show the total `Amount` of `Type = 1` transactions per `Area Code` of the last hour. Repeat once every hour (use a tumbling window).
6. Show the total `Amount` per ATM's `City` and Customer's `Gender` of the last hour. Repeat once every hour (use a tumbling window).
7. Alert (SELECT "1") if a Customer has performed two transactions of `Type = 1` in a window of an hour (use a sliding window).
8. Alert (SELECT "1") if the `Area Code` of the ATM of the transaction is not the same as the "Area Code" of the `Card Number` (Customer's Area Code) - (use a sliding window)

Azure Stream Analytics Configuration

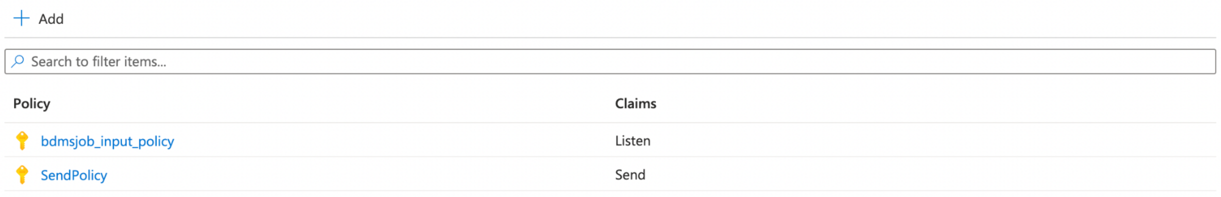
Event Hub Configuration

1. An Event Hub Namespace was created (`AuebNamespace`)
2. Then, the Event Hub was setup (`bdmshub`)
3. Two shared access policies need to be setup
    - SendPolicy
    - ReceivePolicy



Name	Status	Message retention	Partition count
bdmshub	Active	24 hours	3

Image 1: Event Hub Namespace Main page



Policy	Claims
bdmsjob_input_policy	Listen
SendPolicy	Send

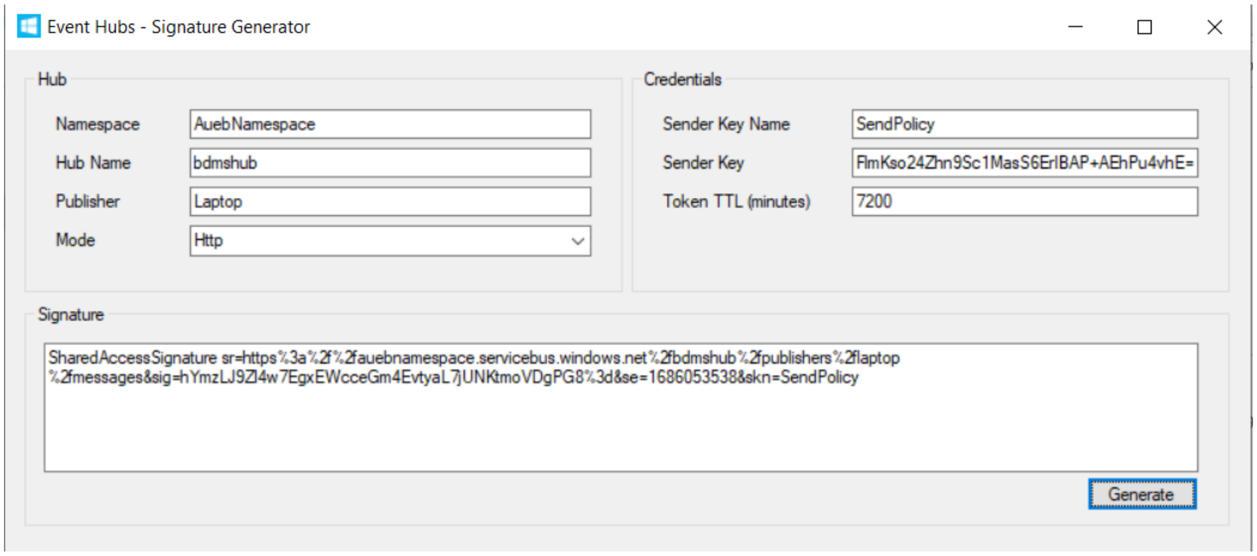
Image 2: Event Hub shared access policies

Data Generator Configuration

1. The Security Access Signature (SAS) Generator was used.
2. The variables below were specified to generate the key:
    - Namespace: `AuebNamespace`
    - Event Hub: `bdmshub`
    - Publisher: `Laptop`
    - SenderKeyName: `SendPolicy`
    - Sender Key: The `primary key` of the SendPolicy created
    - Token TTL (minutes): `7200`
3. To emulate the stream, the Data Generator was used.
4. The config variables were replaced with the variables above and the key generated from the SAS generator.

5. Then, after the generator is set up, to start the data stream, the Generator.html was opened in a browser and the button “SEND DATA” was pressed.

To verify the Data Ingestion process, the Event Hub's `Metrics` tab was monitored. The "Incoming Requests" and "Incoming Messages" metrics should increase as data comes in. Any errors are captured in the "User Errors" metric which means that the request was not successful.



Event Hubs - Signature Generator

Hub

Namespace	AuebNamespace
Hub Name	bdmshub
Publisher	Laptop
Mode	Http

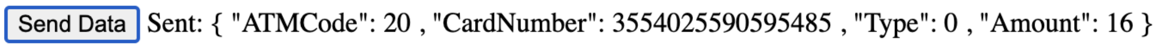
Credentials

Sender Key Name	SendPolicy
Sender Key	FlmKso24Zhn9Sc1MasS6ErIBAP+AEhPu4vhE=
Token TTL (minutes)	7200

Signature

```
SharedAccessSignature sr=https%3a%2f%2fauebnamespace.servicebus.windows.net%2fbdmshub%2fpublishers%2flaptop
%2fmessages&sig=hYmzLJ9Zl4w7EgxEWcceGm4EvtyaL7jUNKtmoVDgPG8%3d&se=1686053538&skn=SendPolicy
```

Image 3: SAS key generation



Sent: { "ATMCode": 20 , "CardNumber": 3554025590595485 , "Type": 0 , "Amount": 16 }

Image4: Successful “send data” operation

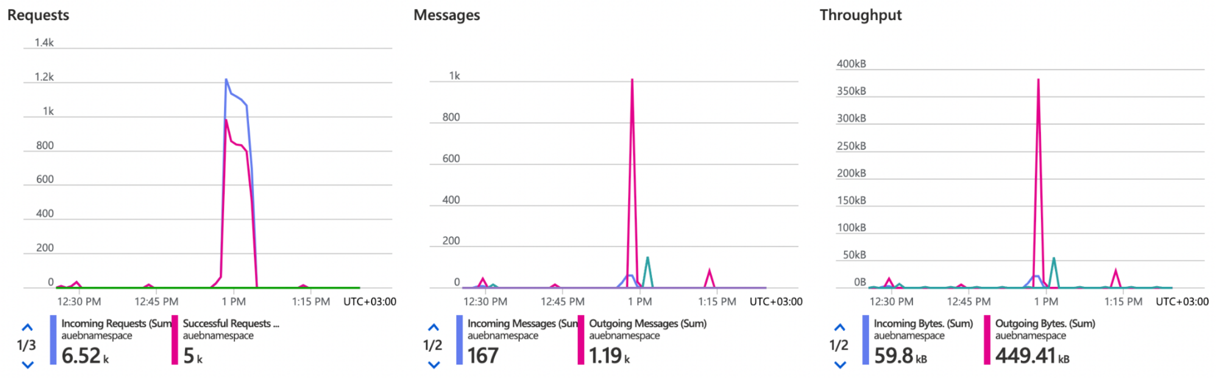
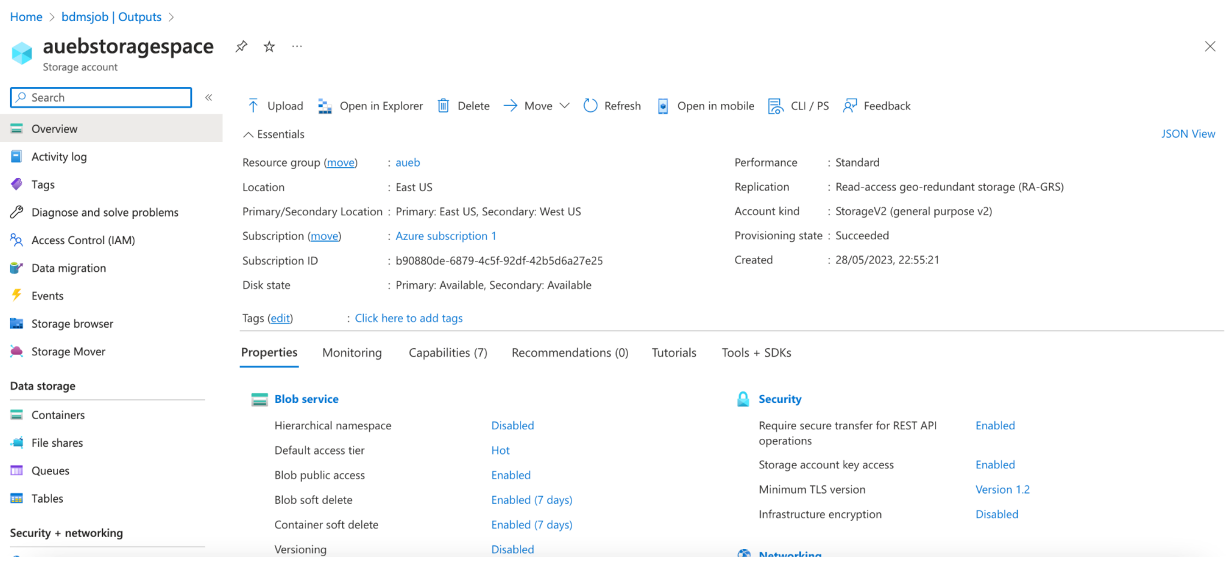


Image 5: Event Hub metrics during Data Ingestion

Blob storage configuration

1. A Blob Storage Namespace was created(`auebstorage`)
2. A new Container was created to store the results of the analytics job (`atmresultscontainer`)
3. A new Container was created to store the reference files (`atmrefcontainer`)
    - The reference files were uploaded



Resource group (move)	: aueb
Location	: East US
Primary/Secondary Location	: Primary: East US, Secondary: West US
Subscription (move)	: Azure subscription 1
Subscription ID	: b90880de-6879-4c5f-92df-42b5d6a27e25
Disk state	: Primary: Available, Secondary: Available
Tags (edit)	: Click here to add tags
Performance	: Standard
Replication	: Read-access geo-redundant storage (RA-GRS)
Account kind	: StorageV2 (general purpose v2)
Provisioning state	: Succeeded
Created	: 28/05/2023, 22:55:21

Blob service

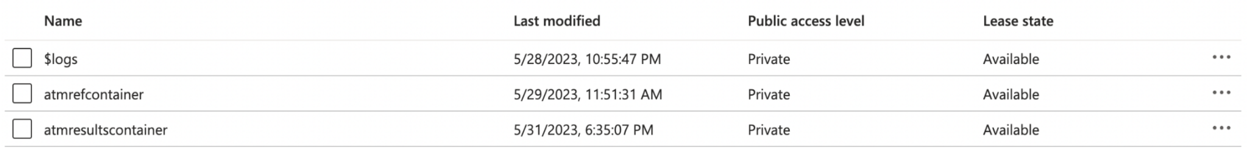
Hierarchical namespace	Disabled
Default access tier	Hot
Blob public access	Enabled
Blob soft delete	Enabled (7 days)
Container soft delete	Enabled (7 days)
Versioning	Disabled

Security

Require secure transfer for REST API operations	Enabled
Storage account key access	Enabled
Minimum TLS version	Version 1.2
Infrastructure encryption	Disabled

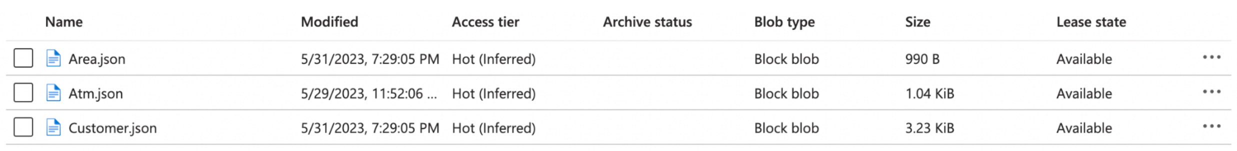
Networking

Image 6: Storage Blob namespace



Name	Last modified	Public access level	Lease state
$logs	5/28/2023, 10:55:47 PM	Private	Available
atmrefcontainer	5/29/2023, 11:51:31 AM	Private	Available
atmresultscontainer	5/31/2023, 6:35:07 PM	Private	Available

Image 7: Containers created



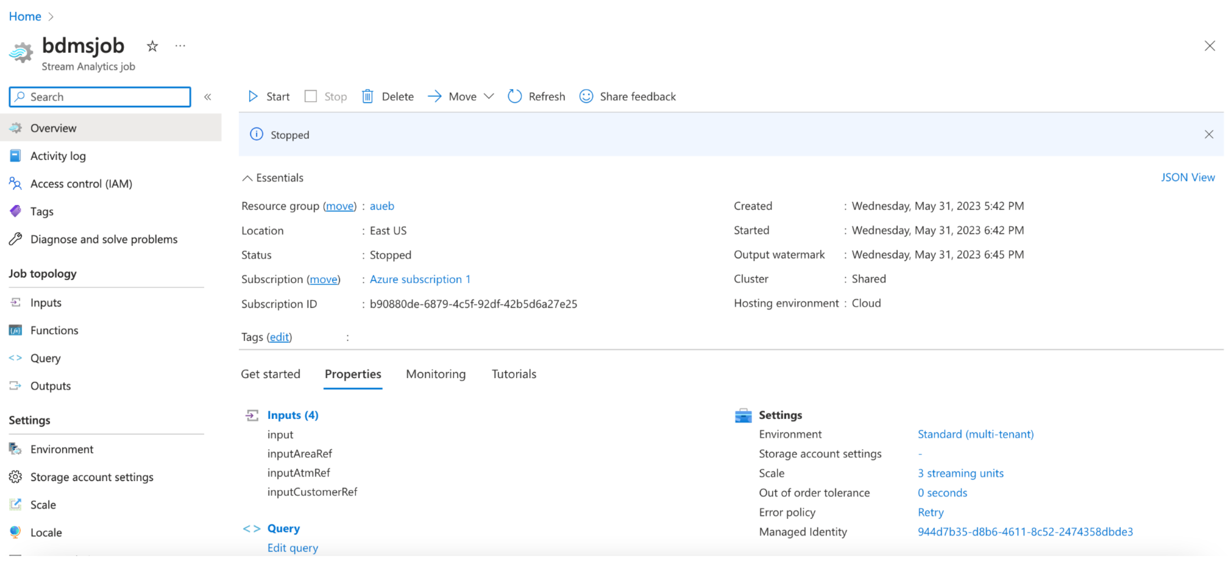
Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
Area.json	5/31/2023, 7:29:05 PM	Hot (Inferred)		Block blob	990 B	Available
Atm.json	5/29/2023, 11:52:06 ...	Hot (Inferred)		Block blob	1.04 KiB	Available
Customer.json	5/31/2023, 7:29:05 PM	Hot (Inferred)		Block blob	3.23 KiB	Available

Image 8: Reference files uploaded

Stream Analytics Job Configuration

1. *A new Stream Analytics Job was created (`bdmsjob`)*
2. *The Event Hub created before is added as Input to the Job*
3. *The blob container created is added (`atmrefcontainer`) as Input. Each reference file is added as a separate Input with the following names:*
    - `InputAreaRef`*: The reference Blob storage with the* `Area.json` *file specified in the path.*
    - `InputAtmRef`*: The reference Blob storage with the* `Atm.json` *file specified in the path.*
    - `InputCustomerRef`*: The reference Blob storage with the* `Customer.json` *file specified in the path.*
4. *The blob storage (`atmresultscontainer`) is added as an output to the job*
    - `OutputBlob`
5. *Finally, the sql queries were added to the Analytics Job.*
    - *Each query saves the input in a different file*

The "Sample" option on the input can be utilized to test the incoming data. A JSON file will be returned which is expected to contain the data produced from the Generator and some attributes added by the Event Hub: `EventProcessedUtcTime`, `PartitionId` *and* `EventEnqueuedUtcTime`.



Stopped

Resource group (move) : aueb
Location : East US
Status : Stopped
Subscription (move) : Azure subscription 1
Subscription ID : b90880de-6879-4c5f-92df-42b5d6a27e25
Tags (edit) :

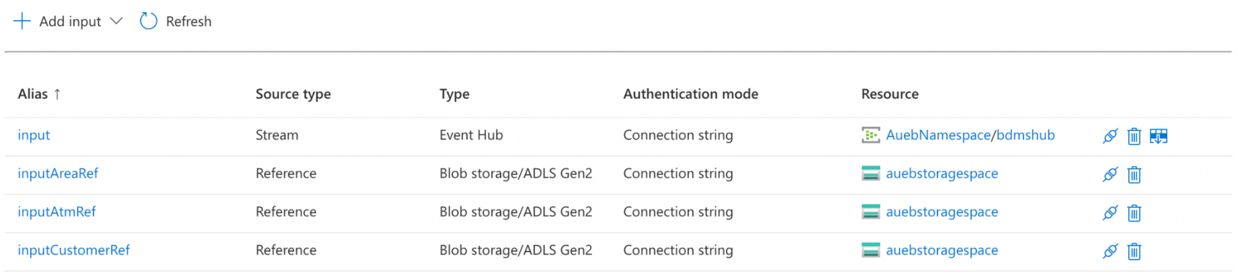
Created : Wednesday, May 31, 2023 5:42 PM
Started : Wednesday, May 31, 2023 6:42 PM
Output watermark : Wednesday, May 31, 2023 6:45 PM
Cluster : Shared
Hosting environment : Cloud

Inputs (4)
input
inputAreaRef
inputAtmRef
inputCustomerRef

Query

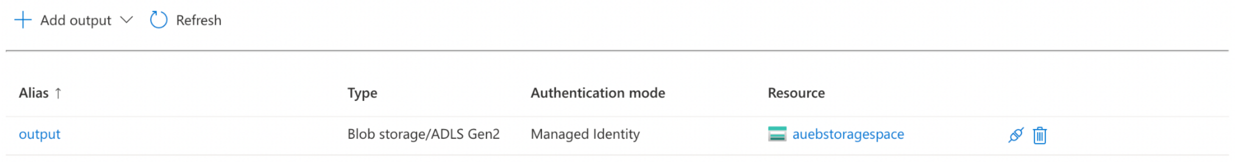
Settings	
Environment	Standard (multi-tenant)
Storage account settings	-
Scale	3 streaming units
Out of order tolerance	0 seconds
Error policy	Retry
Managed Identity	944d7b35-d8b6-4611-8c52-2474358dbde3

Image 9: Azure Stream Analytics Job



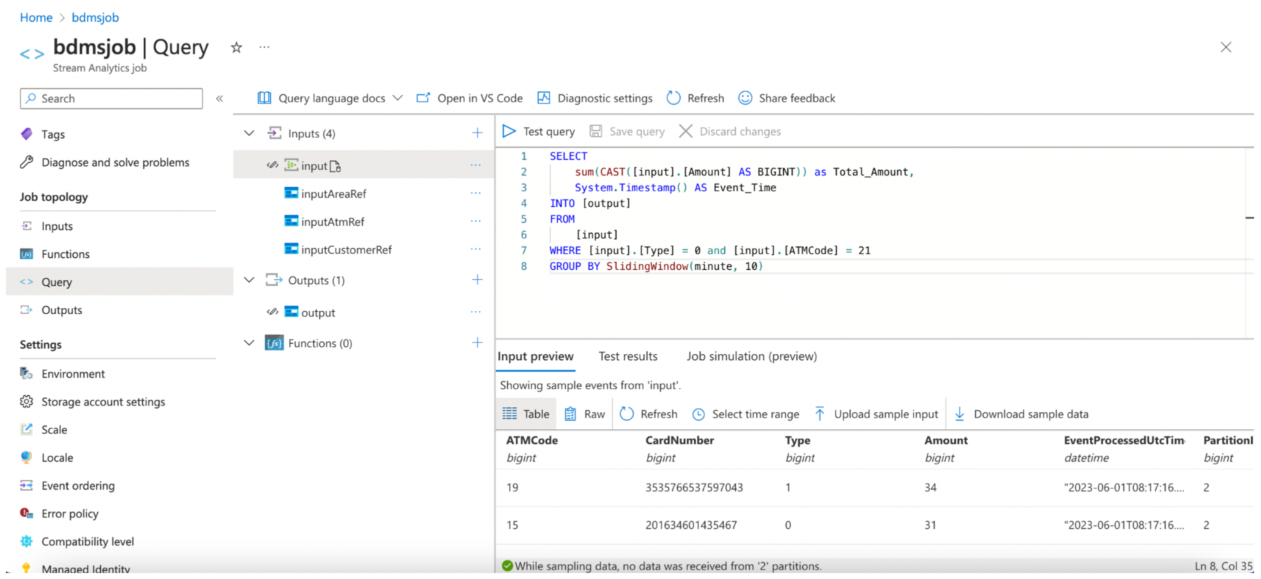
Alias ↑	Source type	Type	Authentication mode	Resource
input	Stream	Event Hub	Connection string	AuebNamespace/bdmshub
inputAreaRef	Reference	Blob storage/ADLS Gen2	Connection string	auebstoragespace
inputAtmRef	Reference	Blob storage/ADLS Gen2	Connection string	auebstoragespace
inputCustomerRef	Reference	Blob storage/ADLS Gen2	Connection string	auebstoragespace

Image 10: Job Inputs



Alias ↑	Type	Authentication mode	Resource
output	Blob storage/ADLS Gen2	Managed Identity	auebstoragespace

Image 11: Job Output



```
SELECT
    sum(CAST([input].[Amount] AS BIGINT)) as Total_Amount,
    System.Timestamp() AS Event_Time
INTO [output]
FROM
    [input]
WHERE [input].[Type] = 0 and [input].[ATMCode] = 21
GROUP BY SlidingWindow(minute, 10)
```

Showing sample events from 'input'.

ATMCode bigint	CardNumber bigint	Type bigint	Amount bigint	EventProcessedUtcTim datetime	PartitionI bigint
19	3535766537597043	1	34	"2023-06-01T08:17:16....	2
15	201634601435467	0	31	"2023-06-01T08:17:16....	2

While sampling data, no data was received from '2' partitions.

Image 12: Job Query Sample

Testing the Job

Starting the process

1. Start the Data Generator and the Stream Analytics Job

2. The data ingestion can be monitored in the Stream Analytics Job's `Monitoring` tab.
3. After some minutes stop the Generator and the Stream Analytics Job.
4. The output of the Stream Analytics Job should be found in the Blob Storage container specified above (`atmresultscontainer`).

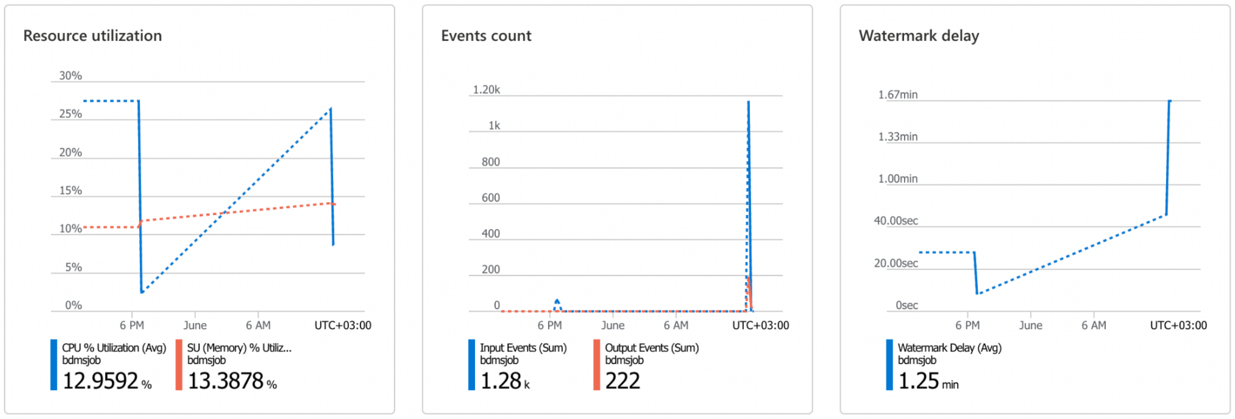


Image 13: Stream Analytics ‘Monitor” Graphs

Results

The queries created and their output when tested is reported here.

Query 1

Show the total `Amount` of `Type = 0` transactions at `ATM Code = 21` of the last 10 minutes. Repeat as new events keep flowing in (use a sliding window).

```
SELECT
    sum(CAST([input].[Amount] AS BIGINT)) as Total_Amount,
    System.Timestamp() AS Event_Time
INTO [output]
FROM
    [input]
WHERE [input].[Type] = 0 and [input].[ATMCode] = 21
GROUP BY SlidingWindow(minute, 10)
```

Output:

```
{"Total_Amount":11,"Event_Time":"2023-06-01T09:55:31.3370000Z"}
{"Total_Amount":21,"Event_Time":"2023-06-01T09:56:48.6080000Z"}
```

Query 2

Show the total `Amount` of `Type = 1` transactions at `ATM Code = 21` of the last hour. Repeat once every hour (use a tumbling window).

```
SELECT
    sum(CAST([input].[Amount] AS BIGINT)) as Total_Amount,
    System.Timestamp() AS Time
INTO [output]
FROM
    [input]
WHERE [input].[Type] = 1 and [input].[ATMCode] = 21
GROUP BY TumblingWindow(hour, 1)
```

Output:

```
{"Total_Amount":204,"Time":"2023-06-01T10:00:00.0000000Z"}
```

Query 3

Show the total `Amount` of `Type = 1` transactions at `ATM Code = 21` of the last hour. Repeat once every 30 minutes (use a hopping window).

```
SELECT
    sum(CAST([input].[Amount] AS BIGINT)) as Total_Amount,
    System.Timestamp() AS Time
INTO [output]
FROM
    [input]
WHERE [input].[Type] = 1 and [input].[ATMCode] = 21
GROUP BY HoppingWindow(minute, 60, 30)
```

Output:

```
{"Total_Amount":50,"Time":"2023-06-01T10:30:00.0000000Z"}
{"Total_Amount":50,"Time":"2023-06-01T11:00:00.0000000Z"}
```

Query 4

Show the total `Amount` of `Type = 1` transactions per `ATM Code` of the last one hour (use a sliding window).

```
SELECT
    [input].[ATMCode],
    sum(CAST([input].[Amount] AS BIGINT)) as Total_Amount,
    System.Timestamp() AS Time
INTO [output]
FROM
    [input]
WHERE Type = 1
GROUP BY [input].[ATMCode],
         SlidingWindow(hour, 1)
```

Output:

```
{"ATMCode":15,"Total_Amount":102,"Time":"2023-06-01T09:55:22.4460000Z"}
{"ATMCode":10,"Total_Amount":123,"Time":"2023-06-01T09:55:22.4770000Z"}
{"ATMCode":15,"Total_Amount":135,"Time":"2023-06-01T09:55:25.3210000Z"}
{"ATMCode":18,"Total_Amount":33,"Time":"2023-06-01T09:55:26.2580000Z"}
{"ATMCode":20,"Total_Amount":29,"Time":"2023-06-01T09:55:30.2900000Z"}
{"ATMCode":12,"Total_Amount":102,"Time":"2023-06-01T09:55:33.2740000Z"}
{"ATMCode":15,"Total_Amount":180,"Time":"2023-06-01T09:55:37.3220000Z"}
{"ATMCode":10,"Total_Amount":142,"Time":"2023-06-01T09:55:42.2760000Z"}
{"ATMCode":21,"Total_Amount":44,"Time":"2023-06-01T09:55:45.2760000Z"}
{"ATMCode":15,"Total_Amount":212,"Time":"2023-06-01T09:55:46.2760000Z"}
{"ATMCode":20,"Total_Amount":68,"Time":"2023-06-01T09:55:47.2760000Z"}
{"ATMCode":18,"Total_Amount":55,"Time":"2023-06-01T09:55:48.3230000Z"}
{"ATMCode":12,"Total_Amount":133,"Time":"2023-06-01T09:55:51.2770000Z"}
{"ATMCode":19,"Total_Amount":87,"Time":"2023-06-01T09:55:53.2920000Z"}
{"ATMCode":21,"Total_Amount":92,"Time":"2023-06-01T09:55:56.2930000Z"}
{"ATMCode":12,"Total_Amount":152,"Time":"2023-06-01T09:55:57.6830000Z"}
```

```
{"ATMCode":10,"Total_Amount":172,"Time":"2023-06-01T09:56:00.2930000Z"}
{"ATMCode":15,"Total_Amount":241,"Time":"2023-06-01T09:56:01.3080000Z"}
{"ATMCode":12,"Total_Amount":173,"Time":"2023-06-01T09:56:02.3550000Z"}
{"ATMCode":15,"Total_Amount":267,"Time":"2023-06-01T09:56:04.3240000Z"}
{"ATMCode":19,"Total_Amount":104,"Time":"2023-06-01T09:56:06.3240000Z"}
{"ATMCode":20,"Total_Amount":115,"Time":"2023-06-01T09:56:08.2770000Z"}
{"ATMCode":15,"Total_Amount":291,"Time":"2023-06-01T09:56:10.3710000Z"}
{"ATMCode":18,"Total_Amount":81,"Time":"2023-06-01T09:56:11.3560000Z"}
{"ATMCode":17,"Total_Amount":14,"Time":"2023-06-01T09:56:12.9020000Z"}
{"ATMCode":21,"Total_Amount":112,"Time":"2023-06-01T09:56:14.2930000Z"}
```

Query 5

Show the total `Amount` of `Type = 1` transactions per `Area Code` of the last hour. Repeat once every hour (use a tumbling window).

```
SELECT
    [inputAtm].[area_code],
    sum(CAST([input].[Amount] AS BIGINT)) as Total_Amount,
    System.Timestamp() AS Time
INTO [output]
FROM
    [input]
INNER JOIN [inputAtm]
    ON [input].[ATMCode] = [inputAtm].[atm_code]
WHERE [input].[Type] = 1
GROUP BY [inputAtm].[area_code],
         TumblingWindow(hour, 1)
```

Output:

```
{"area_code":5,"Total_Amount":419,"Time":"2023-06-01T10:00:00.0000000Z"}
{"area_code":1,"Total_Amount":472,"Time":"2023-06-01T10:00:00.0000000Z"}
{"area_code":3,"Total_Amount":199,"Time":"2023-06-01T10:00:00.0000000Z"}
{"area_code":2,"Total_Amount":336,"Time":"2023-06-01T10:00:00.0000000Z"}
{"area_code":7,"Total_Amount":83,"Time":"2023-06-01T10:00:00.0000000Z"}
{"area_code":11,"Total_Amount":427,"Time":"2023-06-01T10:00:00.0000000Z"}
{"area_code":4,"Total_Amount":128,"Time":"2023-06-01T10:00:00.0000000Z"}
{"area_code":10,"Total_Amount":65,"Time":"2023-06-01T10:00:00.0000000Z"}
{"area_code":9,"Total_Amount":311,"Time":"2023-06-01T10:00:00.0000000Z"}
```

Query 6

Show the total `Amount` per ATM's `City` and Customer's `Gender` of the last hour. Repeat once every hour (use a tumbling window).

```
SELECT
    [inputArea].[area_city],
    [inputCustomers].[gender],
    sum(CAST([input].[Amount] AS BIGINT)) as Total_Amount,
    System.Timestamp() AS Time
INTO [output]
FROM
    [input]
INNER JOIN [inputAtm]
    ON [input].[ATMCode] = [inputAtm].[atm_code]
INNER JOIN [inputArea]
    ON [inputAtm].[area_code] = [inputArea].[area_code]
INNER JOIN [inputCustomers]
    ON [input].[CardNumber] = [inputCustomers].[card_number]
GROUP BY [inputArea].[area_city],
         [inputCustomers].[gender],
         TumblingWindow(hour, 1)
```

Output:

```
{"area_city":"Schaumburg","gender":"Female","Total_Amount":751,"Time":"2023-06-01T1
0:00:00.0000000Z"}
{"area_city":"Baltimore","gender":"Male","Total_Amount":177,"Time":"2023-06-01T10:0
0:00.0000000Z"}
{"area_city":"Omaha","gender":"Female","Total_Amount":183,"Time":"2023-06-01T10:00:
00.0000000Z"}
{"area_city":"Tacoma","gender":"Male","Total_Amount":28,"Time":"2023-06-01T10:00:00
.0000000Z"}
{"area_city":"Memphis","gender":"Male","Total_Amount":360,"Time":"2023-06-01T10:00:
00.0000000Z"}
{"area_city":"Tacoma","gender":"Female","Total_Amount":276,"Time":"2023-06-01T10:00
:00.0000000Z"}
{"area_city":"Memphis","gender":"Female","Total_Amount":76,"Time":"2023-06-01T10:00
:00.0000000Z"}
{"area_city":"Vancouver","gender":"Female","Total_Amount":43,"Time":"2023-06-01T10:
00:00.0000000Z"}
```

```
{"area_city":"Springfield","gender":"Male","Total_Amount":722,"Time":"2023-06-01T10:00:00.0000000Z"}
{"area_city":"Canton","gender":"Male","Total_Amount":435,"Time":"2023-06-01T10:00:00.0000000Z"}
{"area_city":"Schaumburg","gender":"Male","Total_Amount":206,"Time":"2023-06-01T10:00:00.0000000Z"}
{"area_city":"Vancouver","gender":"Male","Total_Amount":340,"Time":"2023-06-01T10:00:00.0000000Z"}
{"area_city":"Springfield","gender":"Female","Total_Amount":182,"Time":"2023-06-01T10:00:00.0000000Z"}
{"area_city":"Greeley","gender":"Female","Total_Amount":38,"Time":"2023-06-01T10:00:00.0000000Z"}
{"area_city":"Omaha","gender":"Male","Total_Amount":418,"Time":"2023-06-01T10:00:00.0000000Z"}
{"area_city":"Baltimore","gender":"Female","Total_Amount":155,"Time":"2023-06-01T10:00:00.0000000Z"}
```

Query 7

Alert (SELECT “1”) if a Customer has performed two transactions of `Type = 1` in a window of an hour (use a sliding window).

```
SELECT
    [inputCustomers].[first_name],
    [inputCustomers].[last_name],
    [input].[CardNumber] AS Card_Number,
    COUNT (*) AS Transactions,
    System.Timestamp AS Time
INTO
    [output]
FROM
    [input]
INNER JOIN [inputCustomers]
    ON [inputCustomers].[card_number] = [input].[CardNumber]
WHERE [input].[Type] = 1
GROUP BY [inputCustomers].[first_name],
         [inputCustomers].[last_name],
         [input].[CardNumber],
         SlidingWindow(hour, 1)
HAVING Transactions = 2
```

Output:

```
{"first_name":"Kathy","last_name":"Jordan","Card_Number":30487898026193,"Transactio
ns":2,"Time":"2023-06-01T09:55:25.3210000Z"}
{"first_name":"Martha","last_name":"Day","Card_Number":3535766537597043,"Transactio
ns":2,"Time":"2023-06-01T09:55:53.2920000Z"}
{"first_name":"Jesse","last_name":"Bradley","Card_Number":3542024987623740,"Transac
tions":2,"Time":"2023-06-01T09:56:04.3240000Z"}
{"first_name":"Richard","last_name":"Russell","Card_Number":5200253312538103,"Trans
actions":2,"Time":"2023-06-01T09:56:08.2770000Z"}
{"first_name":"Jerry","last_name":"Hansen","Card_Number":50383945269330136,"Transac
tions":2,"Time":"2023-06-01T09:56:14.2930000Z"}
{"first_name":"Jose","last_name":"Snyder","Card_Number":3549670931669297,"Transacti
ons":2,"Time":"2023-06-01T09:56:16.3090000Z"}
{"first_name":"Walter","last_name":"Stone","Card_Number":3554025590595485,"Transact
ions":2,"Time":"2023-06-01T09:56:21.2930000Z"}
{"first_name":"Bruce","last_name":"Morrison","Card_Number":5602246755688900,"Transa
ctions":2,"Time":"2023-06-01T09:56:27.3560000Z"}
{"first_name":"Brenda","last_name":"Carroll","Card_Number":560222217915598000,"Tran
sactions":2,"Time":"2023-06-01T09:56:29.2940000Z"}
{"first_name":"Angela","last_name":"Moreno","Card_Number":3534633361736454,"Transac
tions":2,"Time":"2023-06-01T09:56:31.3090000Z"}
{"first_name":"Lisa","last_name":"Perez","Card_Number":56022176913710210,"Transacti
ons":2,"Time":"2023-06-01T09:56:38.2790000Z"}
{"first_name":"Aaron","last_name":"Mitchell","Card_Number":5602238863017460,"Transa
ctions":2,"Time":"2023-06-01T09:56:57.2800000Z"}
{"first_name":"Julia","last_name":"Fuller","Card_Number":5610827137784218,"Transact
ions":2,"Time":"2023-06-01T09:57:07.2960000Z"}
{"first_name":"Gerald","last_name":"Young","Card_Number":50384191807294800,"Transac
tions":2,"Time":"2023-06-01T09:57:13.5930000Z"}
{"first_name":"Ruth","last_name":"Sims","Card_Number":3583257214000023,"Transaction
s":2,"Time":"2023-06-01T09:57:16.5930000Z"}
```

Query 8

Alert (SELECT “1”) if the `Area Code` of the ATM of the transaction is not the same as the “Area Code” of the `Card Number` (Customer’s Area Code) - (use a sliding window)

```
SELECT
    [inputAtm].[area_code] AS Atm_Area_Code,
    [inputCustomers].[area_code] AS Customer_Area_Code,
    COUNT (*),
    System.Timestamp AS Time
```

```
INTO
    [output]
FROM
    [input]
INNER JOIN [inputCustomers]
    ON [inputCustomers].[card_number] = [input].[CardNumber]
INNER JOIN [inputAtm]
    ON [inputAtm].[atm_code] = [input].[ATMCode]
WHERE [inputAtm].[area_code] != [inputCustomers].[area_code]
GROUP BY [inputAtm].[area_code],
         [inputCustomers].[area_code],
         SlidingWindow(hour, 1)
```

Output:

```
{"Atm_Area_Code":5,"Customer_Area_Code":7,"COUNT":5,"Time":"2023-06-0
1T09:55:22.4460000Z"}
{"Atm_Area_Code":11,"Customer_Area_Code":8,"COUNT":5,"Time":"2023-06-
01T09:55:22.4770000Z"}
{"Atm_Area_Code":10,"Customer_Area_Code":6,"COUNT":2,"Time":"2023-06-
01T09:55:22.4770000Z"}
{"Atm_Area_Code":5,"Customer_Area_Code":7,"COUNT":6,"Time":"2023-06-0
1T09:55:22.5860000Z"}
{"Atm_Area_Code":5,"Customer_Area_Code":7,"COUNT":7,"Time":"2023-06-0
1T09:55:22.6170000Z"}
{"Atm_Area_Code":5,"Customer_Area_Code":7,"COUNT":8,"Time":"2023-06-0
1T09:55:25.3210000Z"}
{"Atm_Area_Code":4,"Customer_Area_Code":2,"COUNT":1,"Time":"2023-06-0
1T09:55:26.2580000Z"}
{"Atm_Area_Code":7,"Customer_Area_Code":3,"COUNT":1,"Time":"2023-06-0
1T09:55:27.7900000Z"}
{"Atm_Area_Code":11,"Customer_Area_Code":8,"COUNT":6,"Time":"2023-06-
01T09:55:28.2740000Z"}
{"Atm_Area_Code":5,"Customer_Area_Code":7,"COUNT":9,"Time":"2023-06-0
1T09:55:29.2740000Z"}
{"Atm_Area_Code":1,"Customer_Area_Code":6,"COUNT":1,"Time":"2023-06-0
1T09:55:31.3370000Z"}
{"Atm_Area_Code":7,"Customer_Area_Code":3,"COUNT":2,"Time":"2023-06-0
1T09:55:32.8370000Z"}
{"Atm_Area_Code":9,"Customer_Area_Code":10,"COUNT":3,"Time":"2023-06-
01T09:55:33.2740000Z"}
{"Atm_Area_Code":4,"Customer_Area_Code":2,"COUNT":2,"Time":"2023-06-0
1T09:55:34.2900000Z"}
{"Atm_Area_Code":10,"Customer_Area_Code":6,"COUNT":3,"Time":"2023-06-
01T09:55:35.2590000Z"}
{"Atm_Area_Code":2,"Customer_Area_Code":1,"COUNT":3,"Time":"2023-06-0
1T09:55:36.3060000Z"}
```