

**ΟΙΚΟΝΟΜΙΚΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΑΘΗΝΩΝ**



**ATHENS UNIVERSITY
OF ECONOMICS
AND BUSINESS**



Athens University of Economics and Business

School of Business

Department of Management Science & Technology

Master of Science in Business Analytics

Program:	Full-time
Quarter:	1 st (Fall Quarter)
Course:	Data Management and Business Intelligence
Assignment №:	3
Students (Registration №):	Gkouma Konstantina (f2822208), Souflas Eleftherios-Efthymios (f2822217)

Data Management & Business Intelligence Assignment #3

Students: Gkouma Konstantina, Souflas Eleftherios-Efthymios

In this assignment we used Azure Stream Analytics to process a virtual data stream of ATM transactions and answer some stream queries. We used as a guideline the notes taken from the relevant Lab's lecture and of course the lecture's presentation guideline.

In order to use Azure's Stream Analytics, we created a student's account (Figure 1) with one of ours AUEB email account, using also our password for verification from <https://azure.microsoft.com/en-us/free/students/>.

Figure 1 - Azure Student Account

We then created a Namespace (Figure 2), creating on the same way a Resource Group where all our Resources (Event Hub Namespace, Storage Account, Stream Analytics job) will belong.

Figure 2 - Namespace creation

We then set up an Event Hub inside the Event Hub Namespace, as shown in Figure 3.

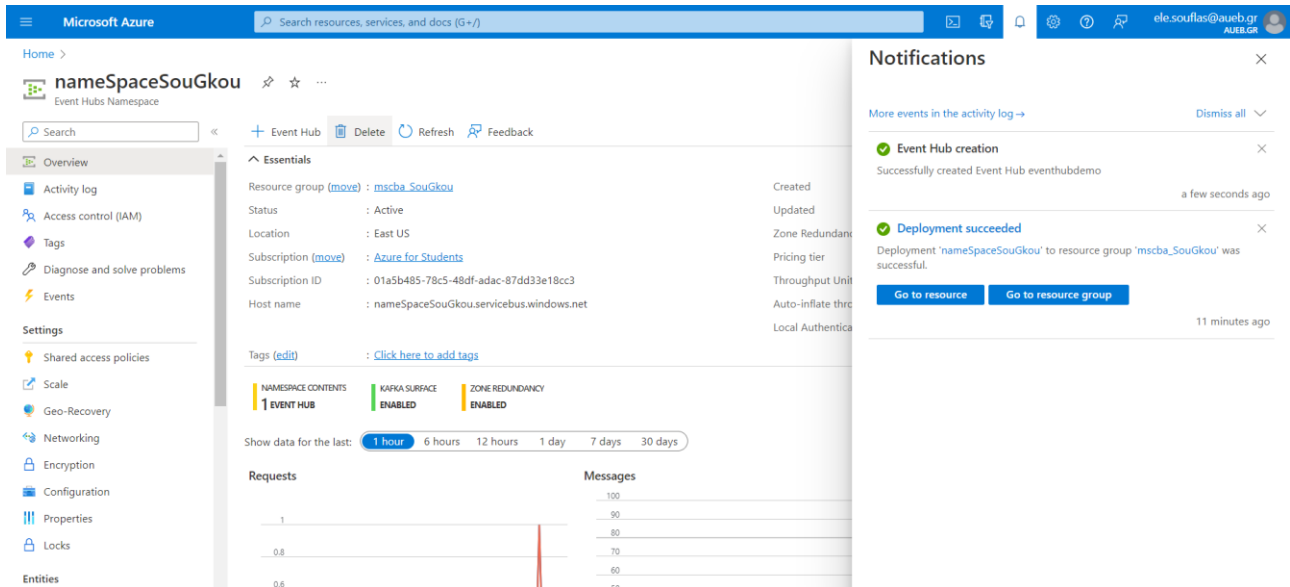


Figure 3 - Event Hub creation

After the Event Hub setup, we created a Shared access policy to manage both rights to send and receive messages to the entity, as shown in Figure 4. We named it “MySendPolicy” because firstly we created two policies (Send, Rec) to differentiate their uses, but as we could not manage to capture properly all the information, by the “MyRecPolicy”, sent by the Virtual Data Stream (ATM Transactions Generator), we did not use the latter at all and we altered the former to manage both rights.

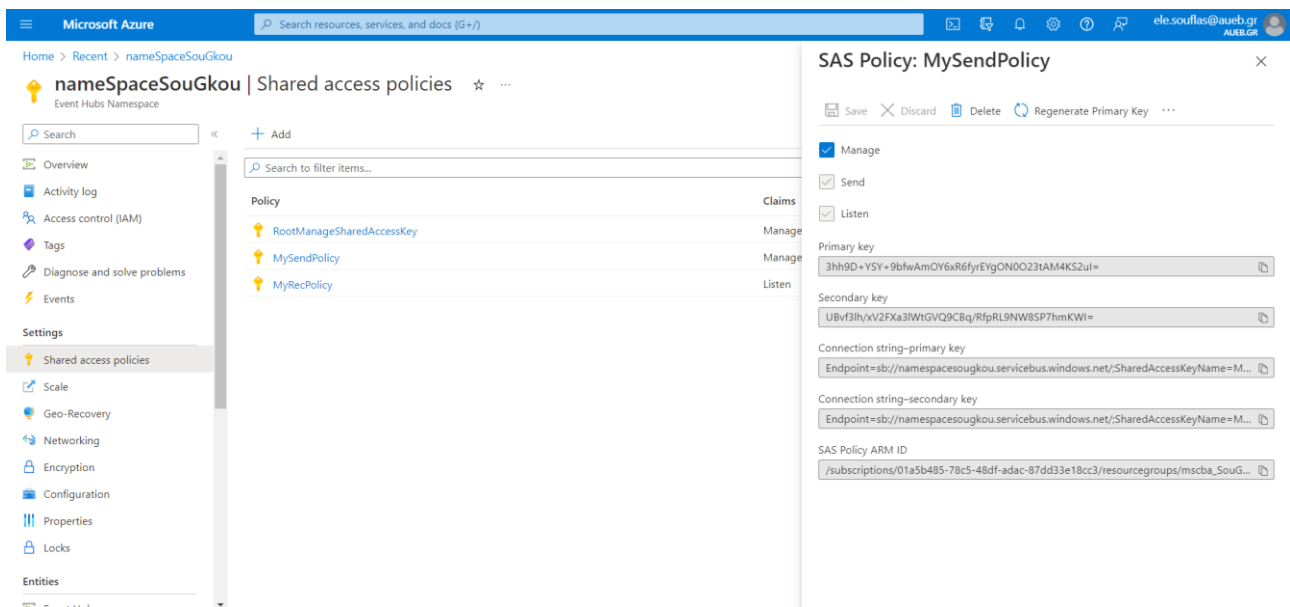


Figure 4 - SAS Policy creation

We copied the Primary Key and with the use of the Event Hubs – Signature Generator (<https://github.com/sandrinodimattia/RedDog/releases>), we generated a Security Access Signature (Figure 5).

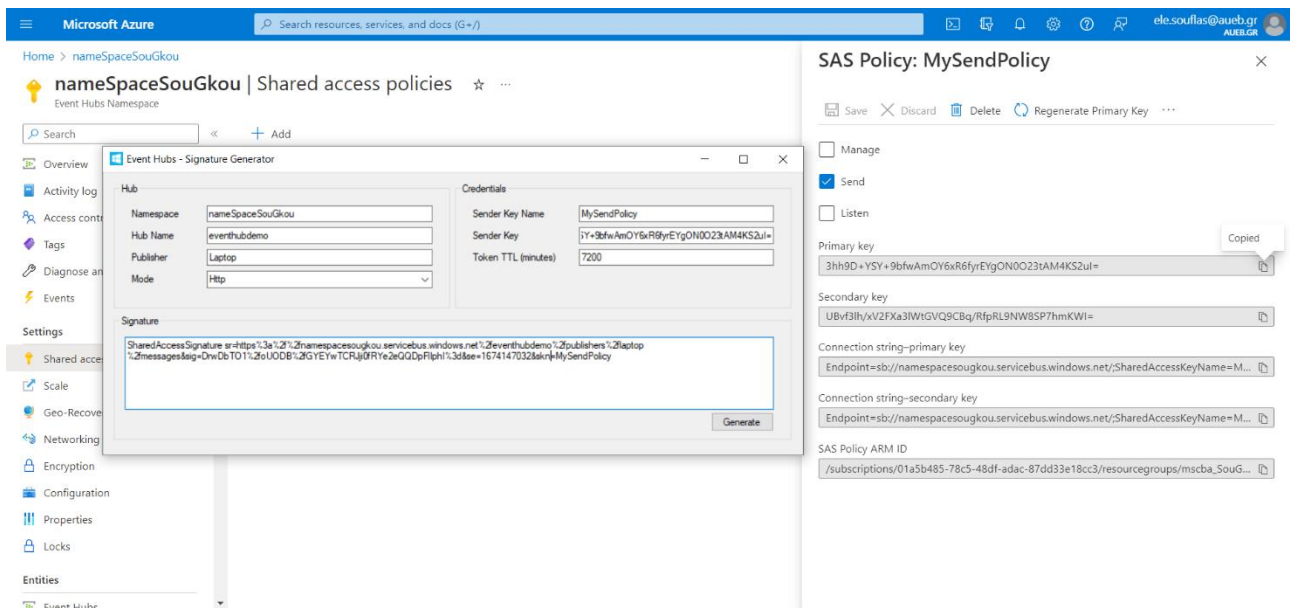


Figure 5 - Generation of SAS Signature

We then edited with Notepad++ the relevant configuration variables (green highlighted – lines 16, 17) of the Generator HTML file (Figure 6), opened the HTML file in a browser and pressed the “Send Data” button to begin the creation of virtual data stream of ATM transactions and feed the Event Hub with it (Figure 7).

Then, we created a Storage account (Figure 8) and uploaded the Reference Data files in a new container inside the Storage account, as shown in Figure 9.

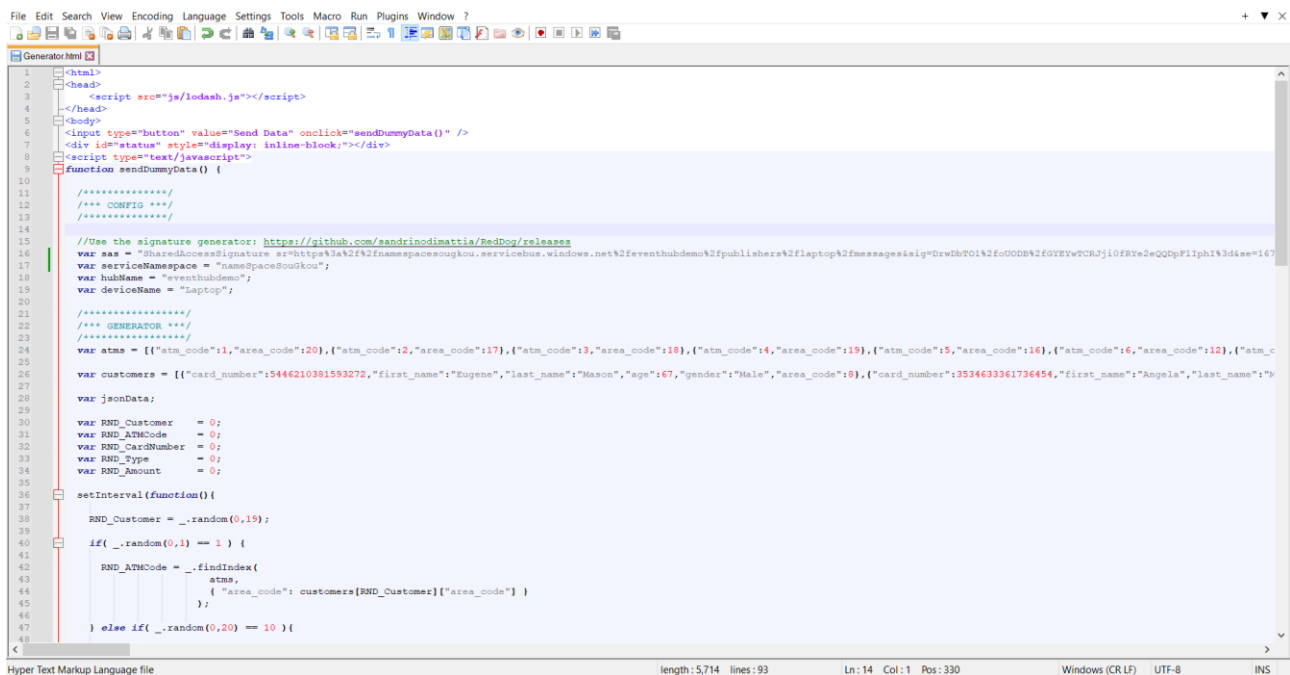


Figure 6 - Config variables update

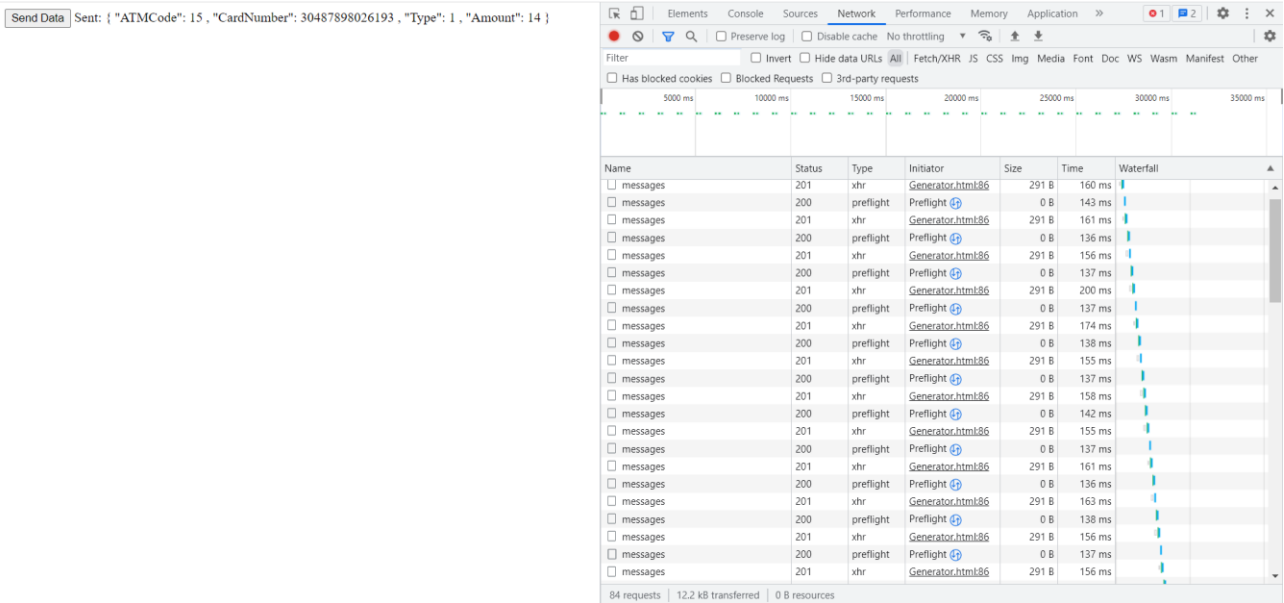


Figure 7 - ATM transactions data stream

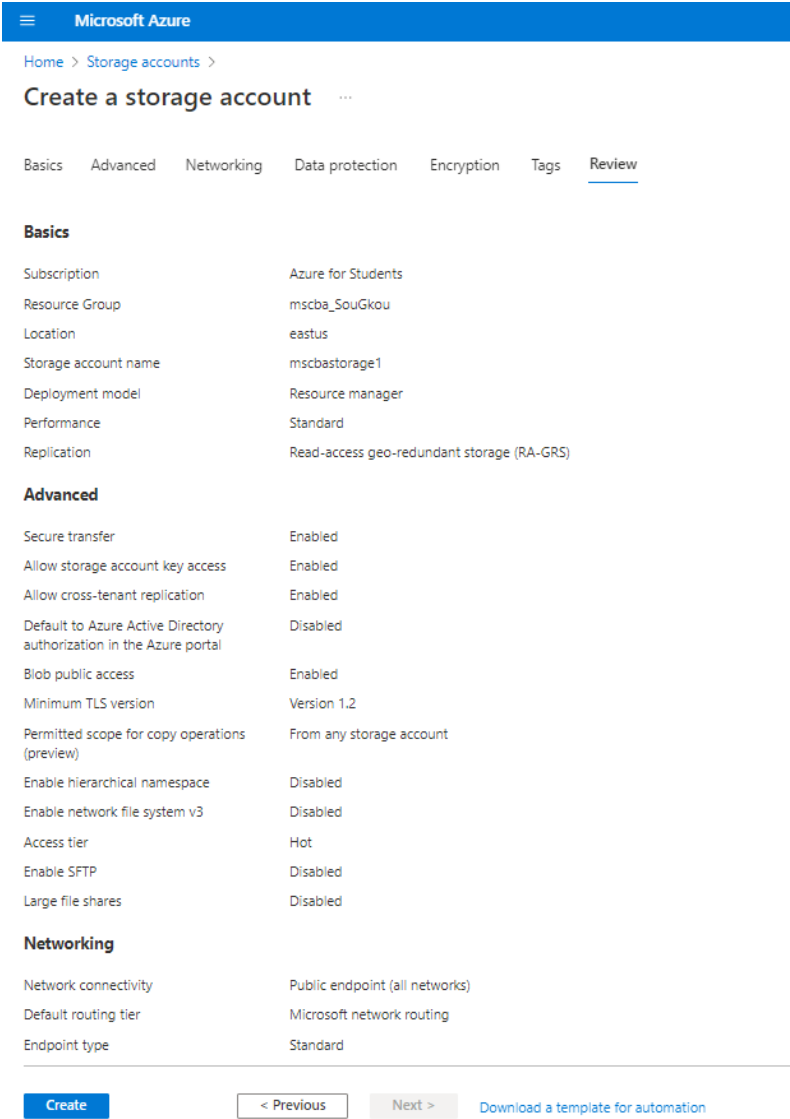


Figure 8 - Storage account setup

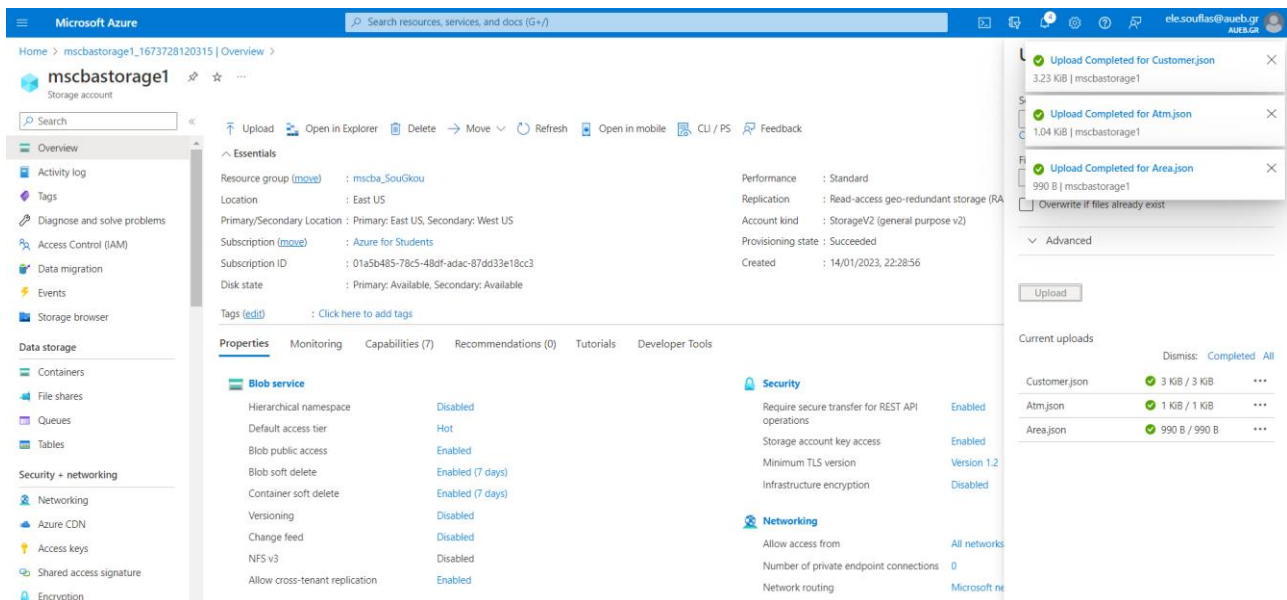


Figure 9 - Blob upload

The we created a Stream Analytics Job, which we will use for answering the Stream Queries (Figure 10). We used the Event Hub (Figure 11) and the Reference Data Files (Figure 13 and Figure 14) as a Stream input of the Job and a Blob Storage Container (we created a new container inside our Storage Account) as a Stream output (Figure 15) to store the results of the queries in JSON format. Before proceeding, we tested, by sampling data from the input source (Figure 12), that the Stream Input captures properly the data.

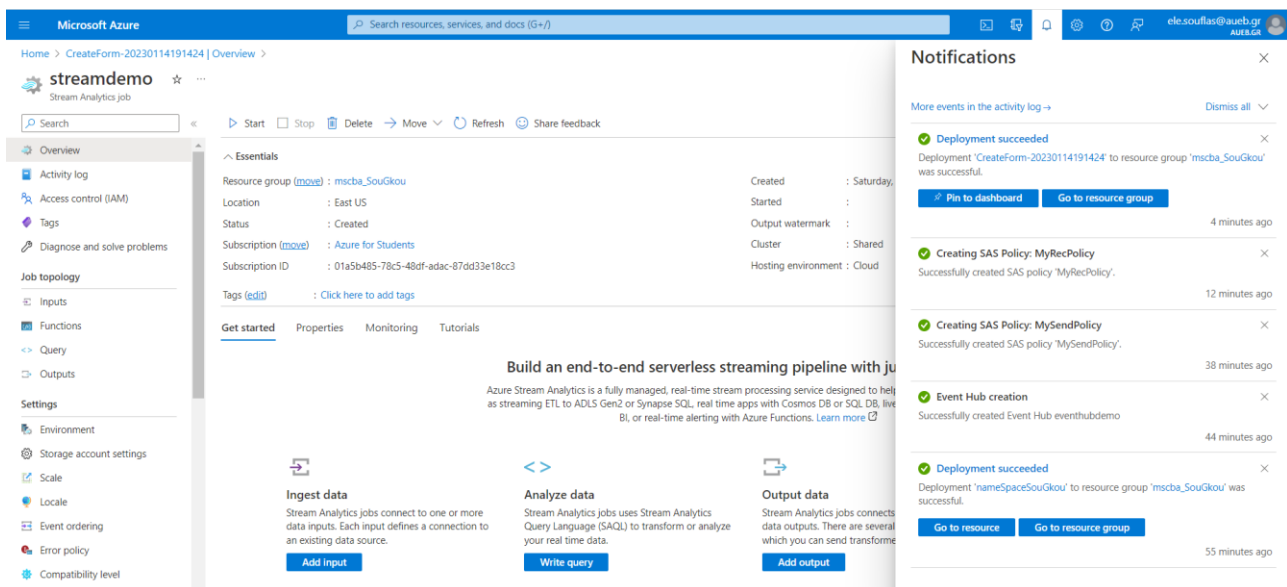


Figure 10 - Stream Analytics Job setup

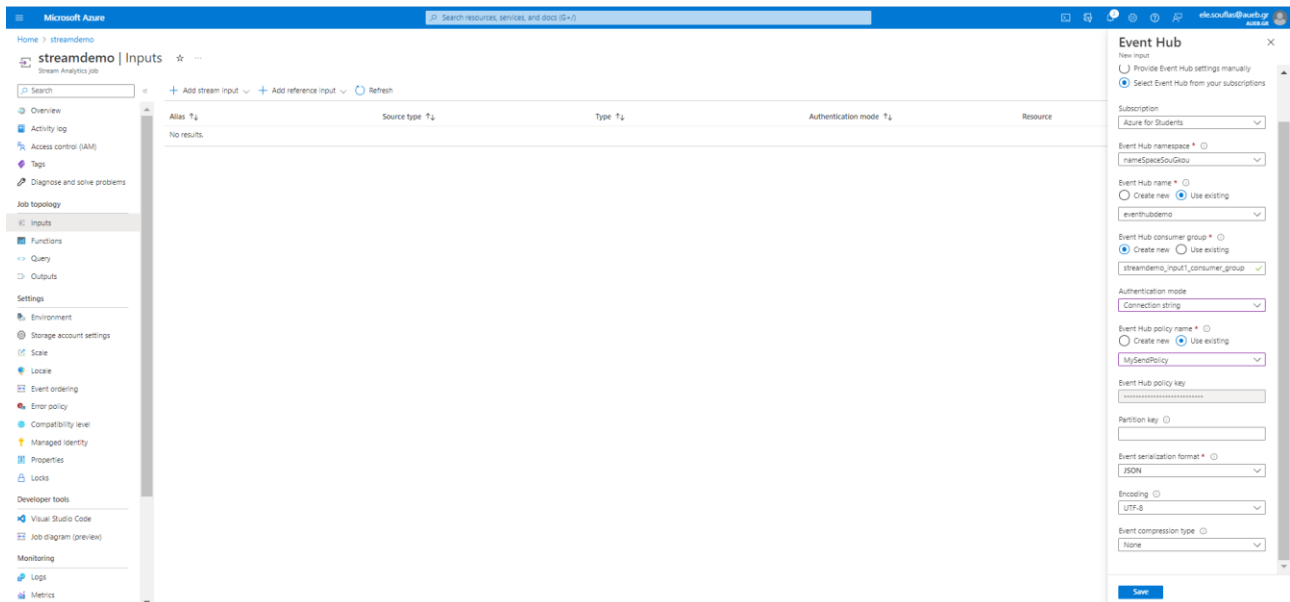


Figure 11 - Event Hub Stream Input

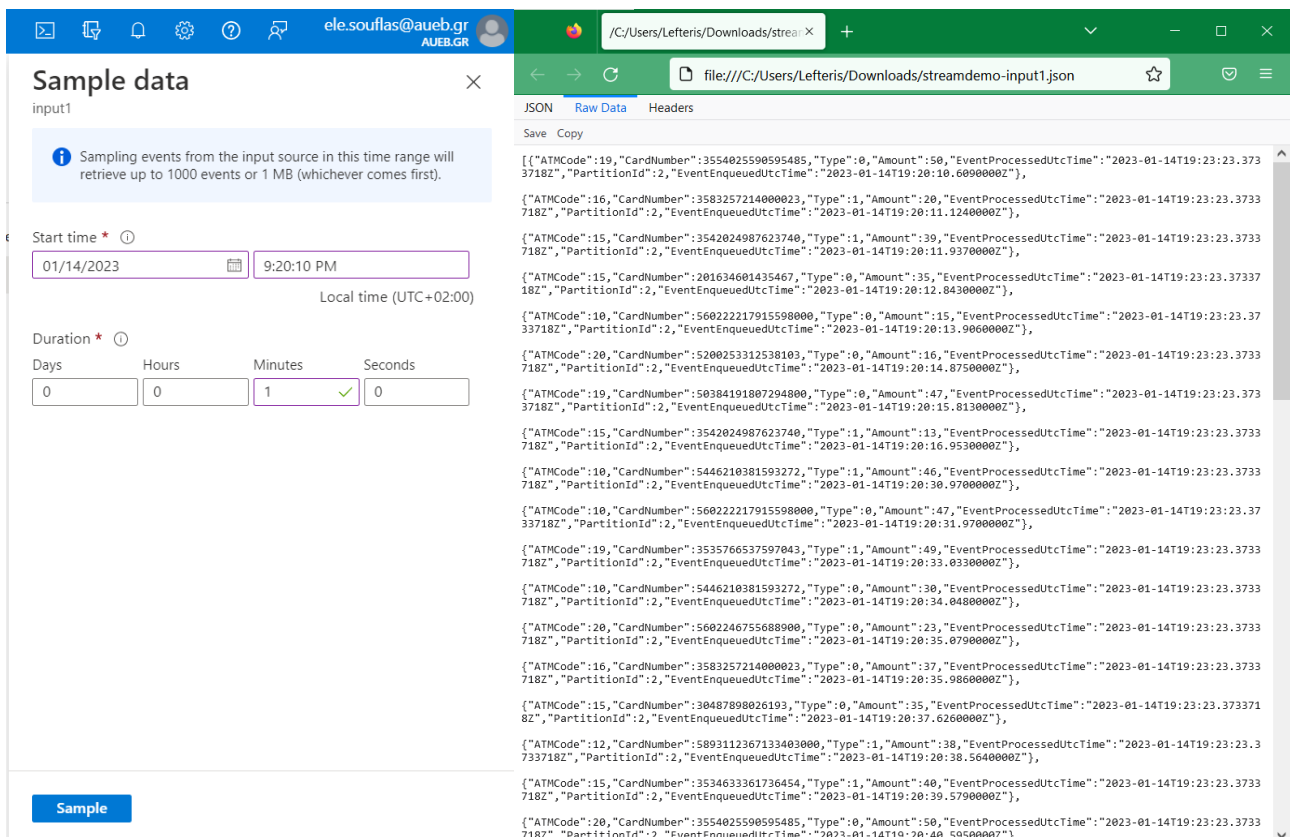


Figure 12 - Sample Input Data (JSON format)

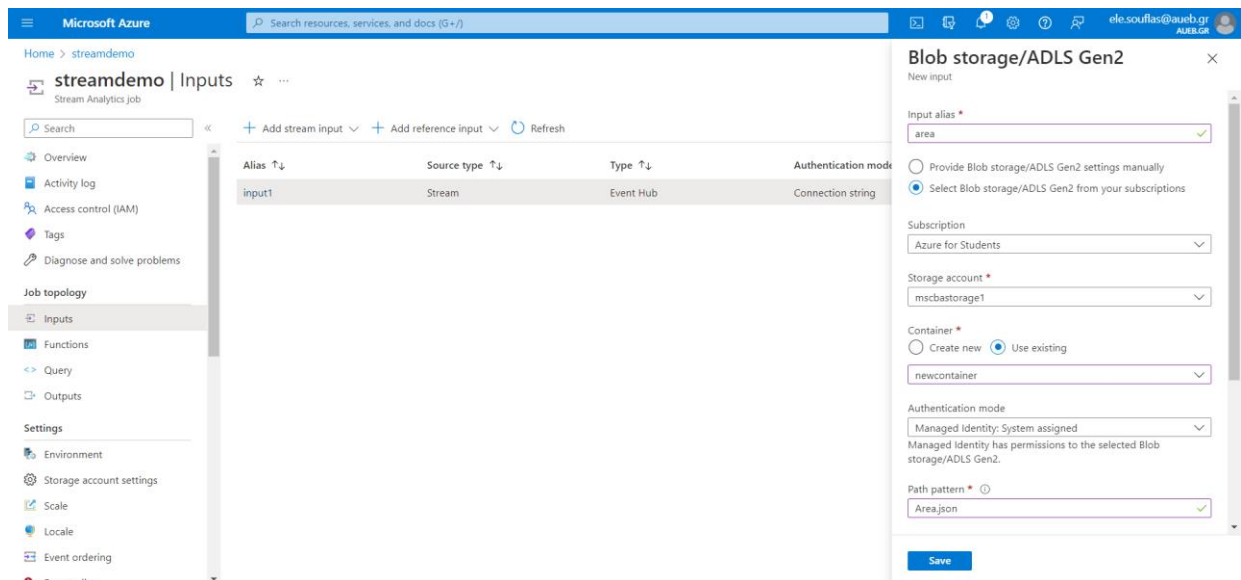


Figure 13 - Blob Storage Reference input creation

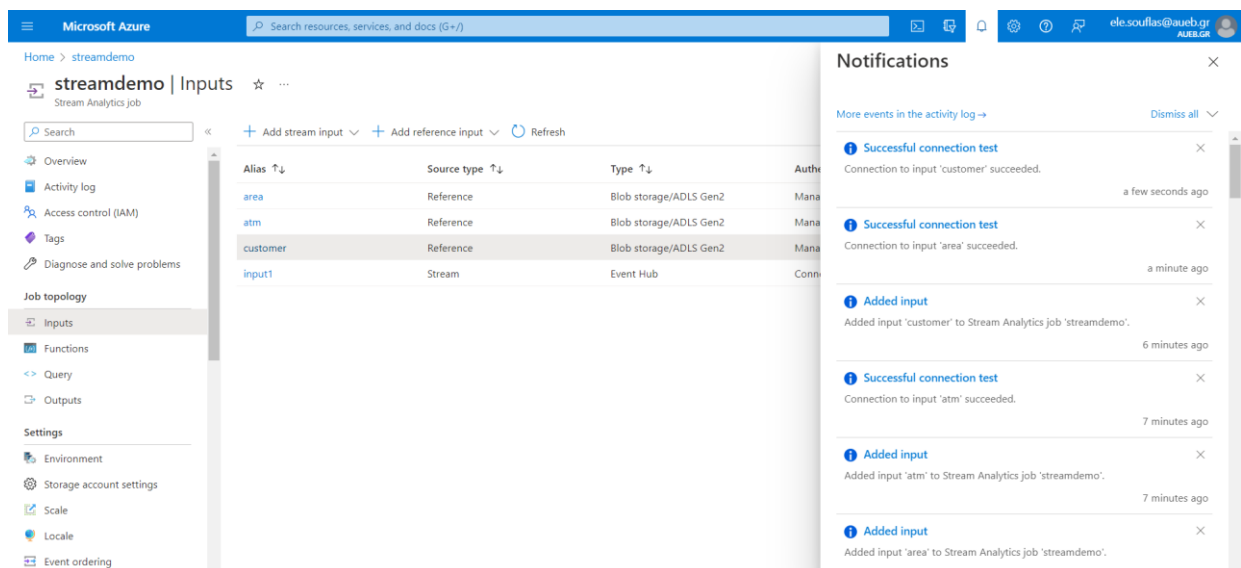


Figure 14 - All Blob Storage Reference inputs

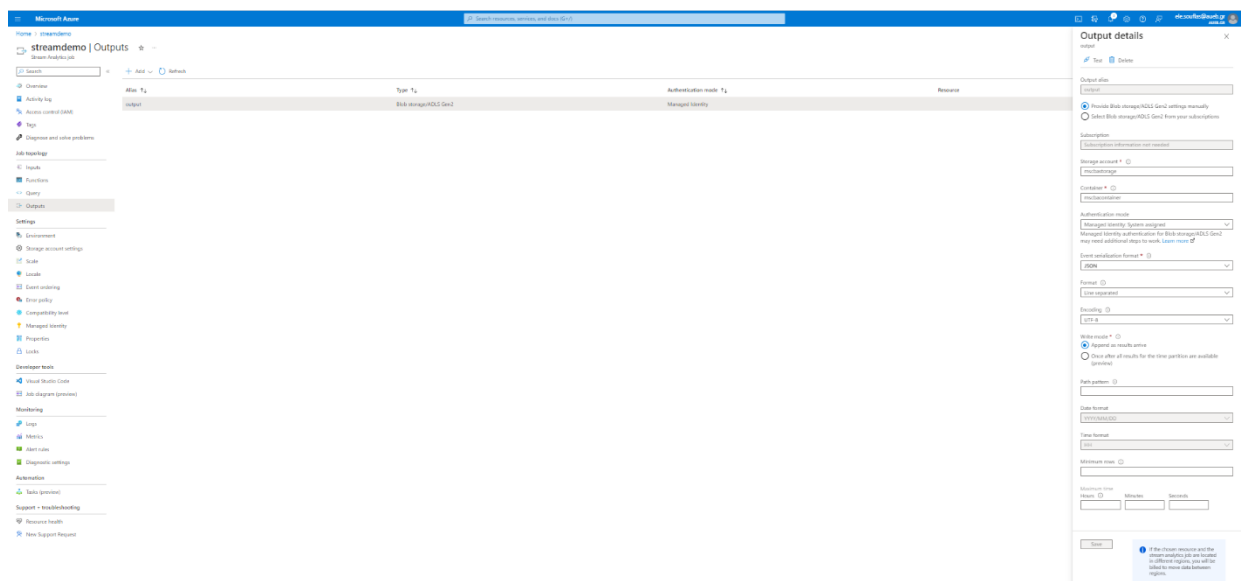


Figure 15 - Blob Storage Stream Output

Finally, we created a query that includes all possible input joins to test that all components used in the Stream Analytics Job work properly. From Figure 16, we can see that everything works as designed and expected.

The screenshot shows the Microsoft Azure portal interface for a Stream Analytics job named 'streamdemo'. The 'Query' tab is active, displaying a SQL query that performs multiple INNER JOINs between input streams: 'area', 'atm', 'customer', and 'input1'. The query selects various fields including area_city, first_name, last_name, age, gender, area_code, card_number, and UTC time. Below the query, the 'Test results' tab is shown, displaying a table with 43 rows of data. The table columns are: ATM area, first_name, last_name, age, gender, customer area, Type, Amount, and UTC Time. The data includes entries for locations like Schaumburg, Vancouver, Springfield, Canton, Memphis, and Omaha, with associated customer information and timestamps.

Figure 16 - Test Query with all possible joins

We are now ready to edit our 8 queries and collect our output data. To answer each query we followed the following procedure:

- We stopped previously started Stream Analytics Job.
- We edited the query.
- We tested it, refreshing data from input stream and uploading a sample input for each reference input used.
- We tested the connection and sampled data from input stream to test that everything worked fine for our inputs.
- We tested the connection to our output blob storage container.
- We started the Stream Analytics Job.
- We observed the output watermark datetime of the Stream Analytics Job Essentials (information) to be informed about the time our JSON output will be available in the Blob Storage Container.
- We clicked on the "Edit/View" of the JSON output to observe our output.

The screenshot shows the Microsoft Azure portal interface for the 'streamdemo' Stream Analytics job. The 'Overview' tab is selected, displaying job details such as Resource group (msbhe_soufou), Location (East US), Status (Running), Subscription (Azure for Students), and Subscription ID. A notification at the top right states 'Streaming job started successfully. Started streaming job 'streamdemo' successfully.' Below the job details, the 'Key metrics' section is visible, showing four charts: Resource utilization, Events count, Watermark delay, and Backlogged input events. The charts display data over a 24-hour period, with the x-axis labeled 'UTC-12:00'.

Figure 17 - Starting Stream Analytics Job

Now, we will provide screenshots of all answered queries.

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).

The screenshot shows the Microsoft Azure Stream Analytics Query Editor interface. The query is named "streamdemo | Query". The query text is as follows:

```
1 SELECT
2   Type, ATMCode, SUM(Amount)
3 INTO
4   [output1]
5 FROM
6   [input1]
7 WHERE
8   Type = 0 and ATMCode = 21
9 GROUP BY Type, ATMCode, SlidingWindow(minute,10)
```

The "Test results" tab is selected, showing 1 row from "output1". The results are displayed in a table with the following columns: Type, ATMCode, and SUM. The data row shows Type: 0, ATMCode: 21, and SUM: 81.

Type	ATMCode	SUM
0	21	81

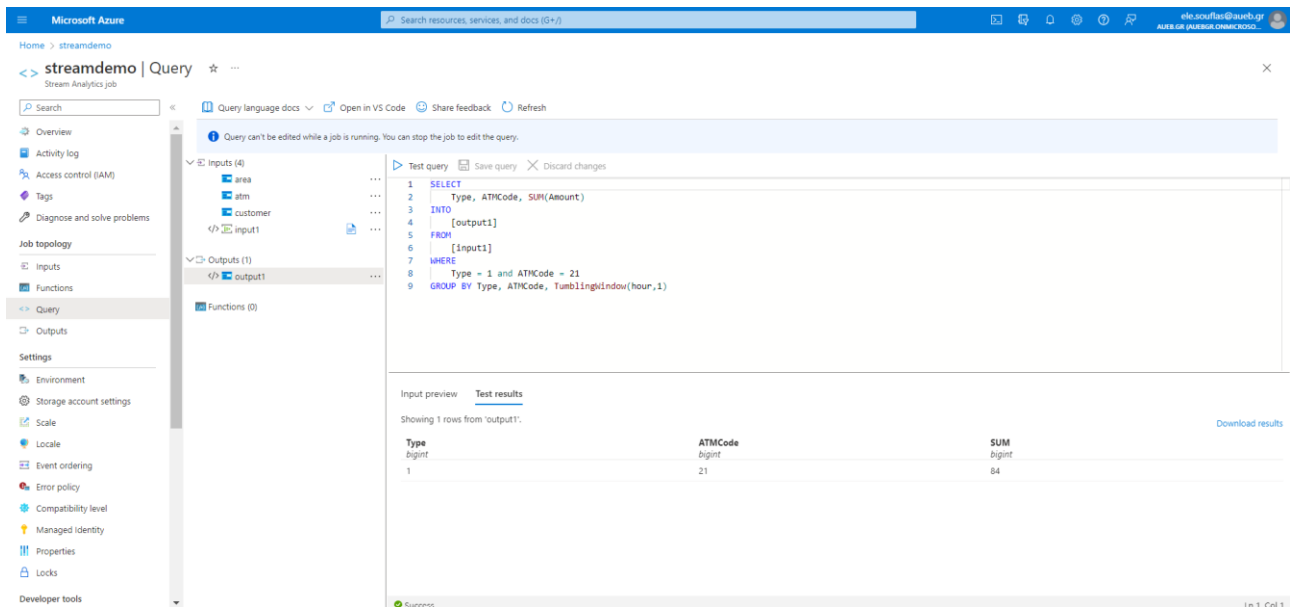
Figure 18 - Query 1 Test Results

The screenshot shows the Microsoft Azure Storage Explorer interface. The selected blob is named "0_3c12d81da6b44eb49eff1115ebc6ee1c_1.json". The blob content is displayed in a JSON format, showing a single array element with the following structure:

```
1 [{"Type":0,"ATMCode":21,"SUM":48.0}]
```

Figure 19 - Query 1 JSON Output

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).



The screenshot shows the Microsoft Azure Stream Analytics Query Editor interface. The query is as follows:

```

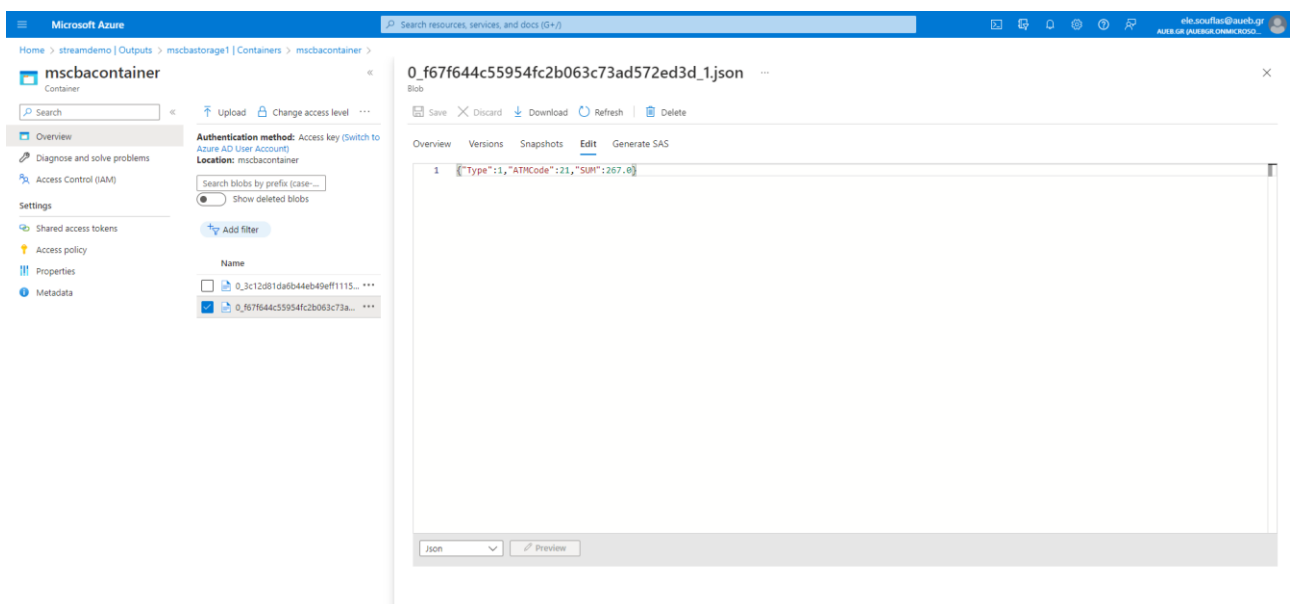
1 SELECT
2   Type, ATMCode, SUM(Amount)
3 INTO
4   [output1]
5 FROM
6   [input1]
7 WHERE
8   Type = 1 and ATMCode = 21
9 GROUP BY Type, ATMCode, TumblingWindow(hour,1)

```

The test results show the following output:

Type	ATMCode	SUM
1	21	84

Figure 20 - Query 2 Test Results



The screenshot shows the Microsoft Azure Storage Explorer interface. The blob '0_f67f644c55954fc2b063c73ad572ed3d_1.json' is selected in the 'mscbscontainer' container. The JSON output is displayed as follows:

```

1 [{"Type":1,"ATMCode":21,"SUM":267.0}]

```

Figure 21 - Query 2 JSON Output

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).

The screenshot shows the Microsoft Azure Stream Analytics Query Editor interface. The query is as follows:

```

1 SELECT
2   Type, ATMCode, SUM(Amount)
3 INTO
4   [output1]
5 FROM
6   [input1]
7 WHERE
8   Type = 1 and ATMCode = 21
9 GROUP BY Type, ATMCode, HoppingWindow(minute,60,30)
10

```

The test results show the following data:

Type	ATMCode	SUM
1	21	84
1	21	84

Figure 22 - Query 3 Test Results

The screenshot shows the Microsoft Azure Blob Storage interface. The blob is named `0_0b48bdcd7884d798fb1f9e717e5cc7e_1.json`. The JSON content is as follows:

```

1 [{"Type":1,"ATMCode":21,"SUM":2854.0}]

```

Figure 23 - Query 3 JSON Output

Query 4: Show the total “Amount” of “Type = 1” transactions per “ATM Code” of the last one hour (use a sliding window).

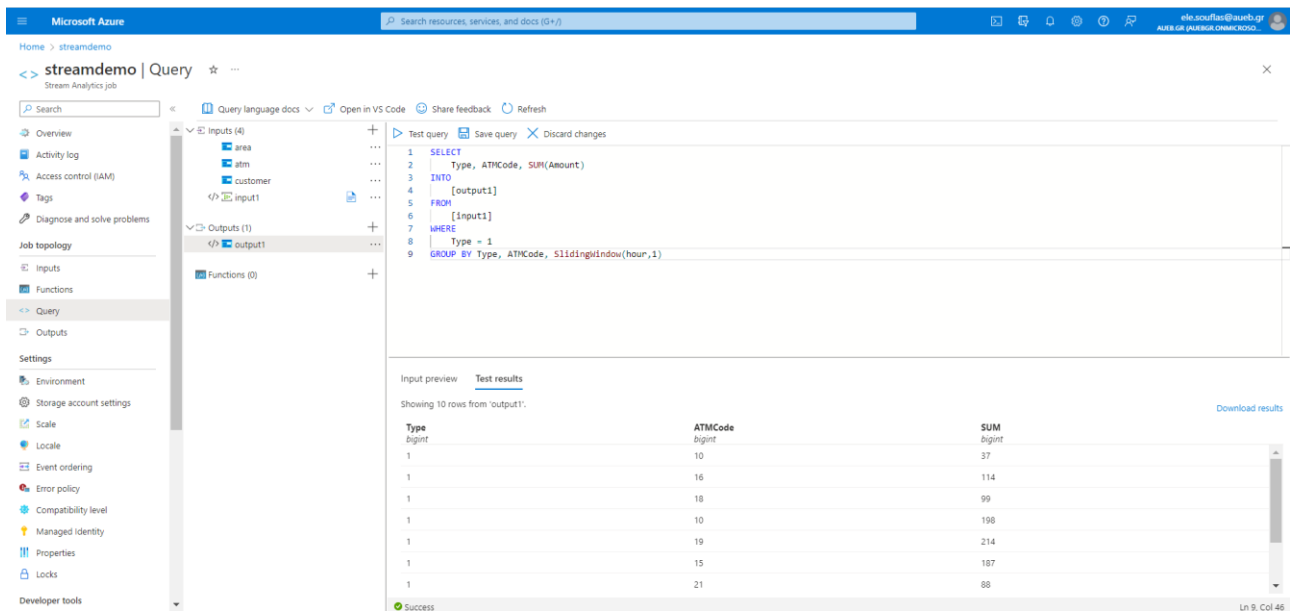


Figure 24 - Query 4 Test Results

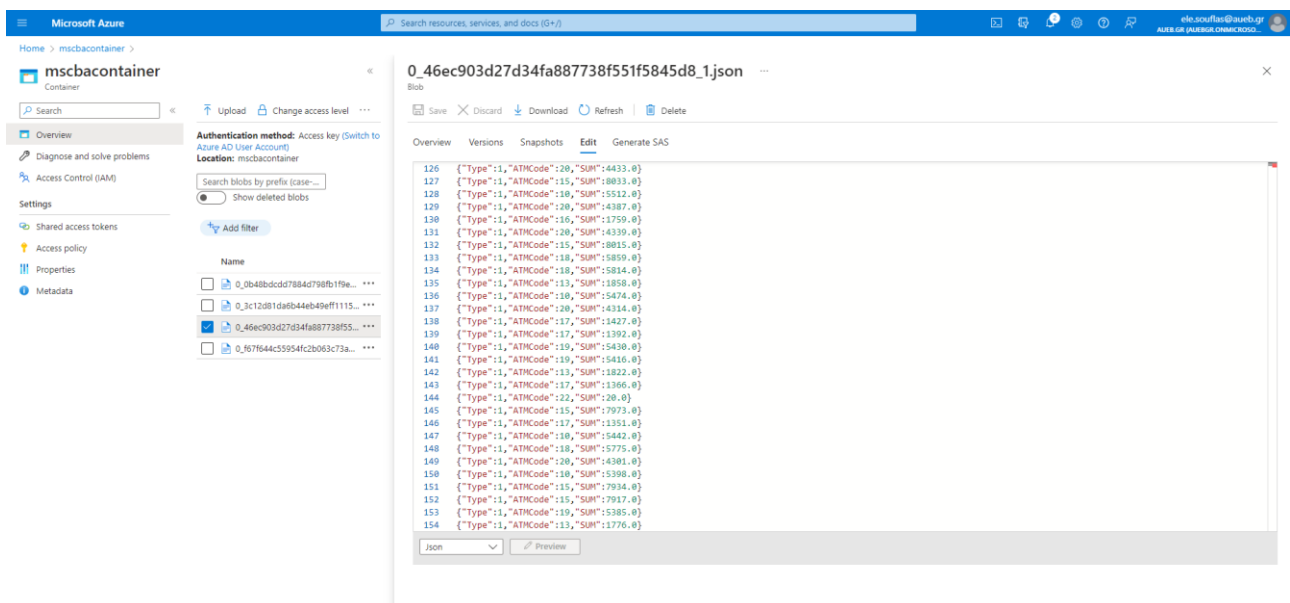


Figure 25 - Query 4 JSON Output

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).

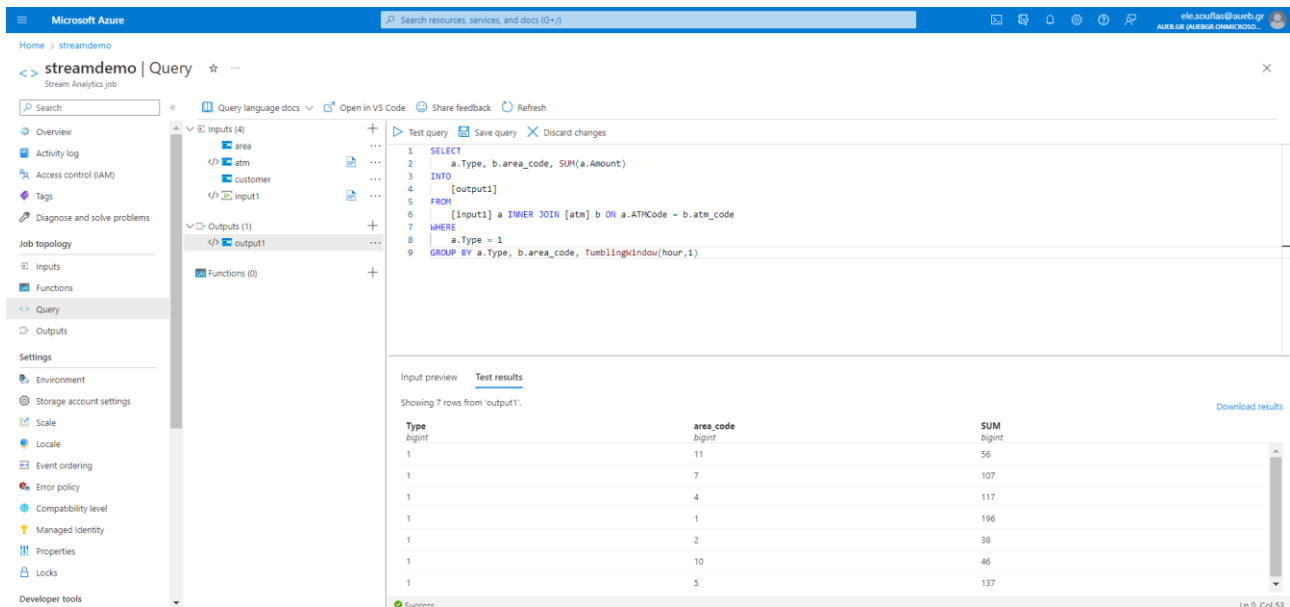


Figure 26 - Query 5 Test Results

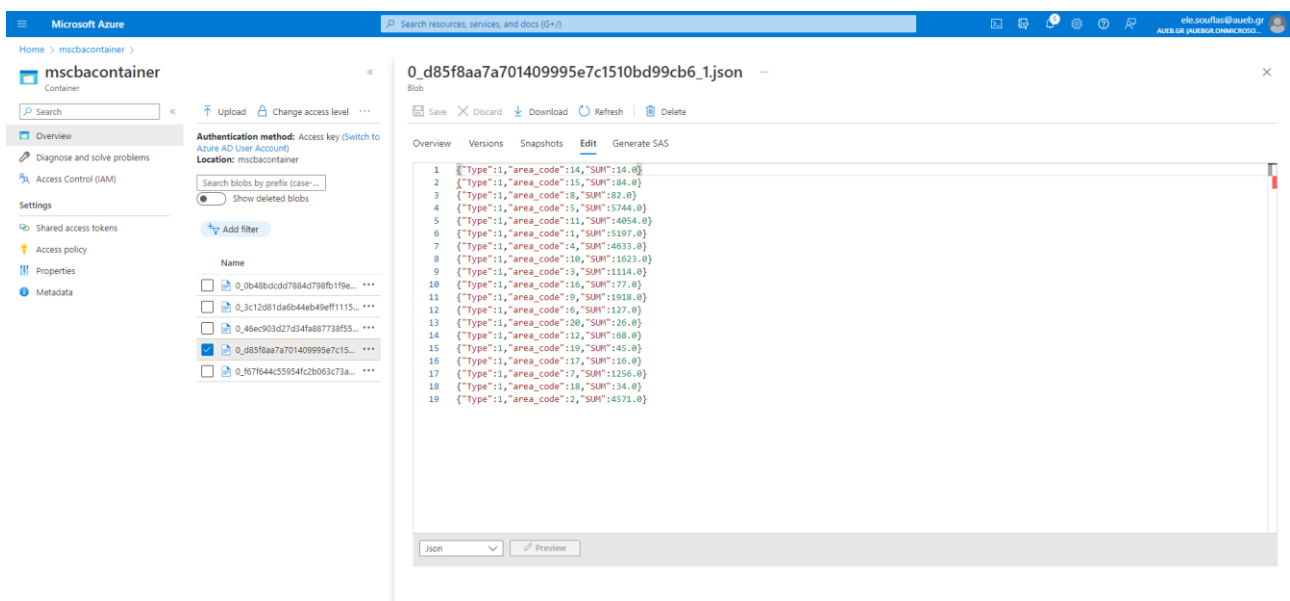
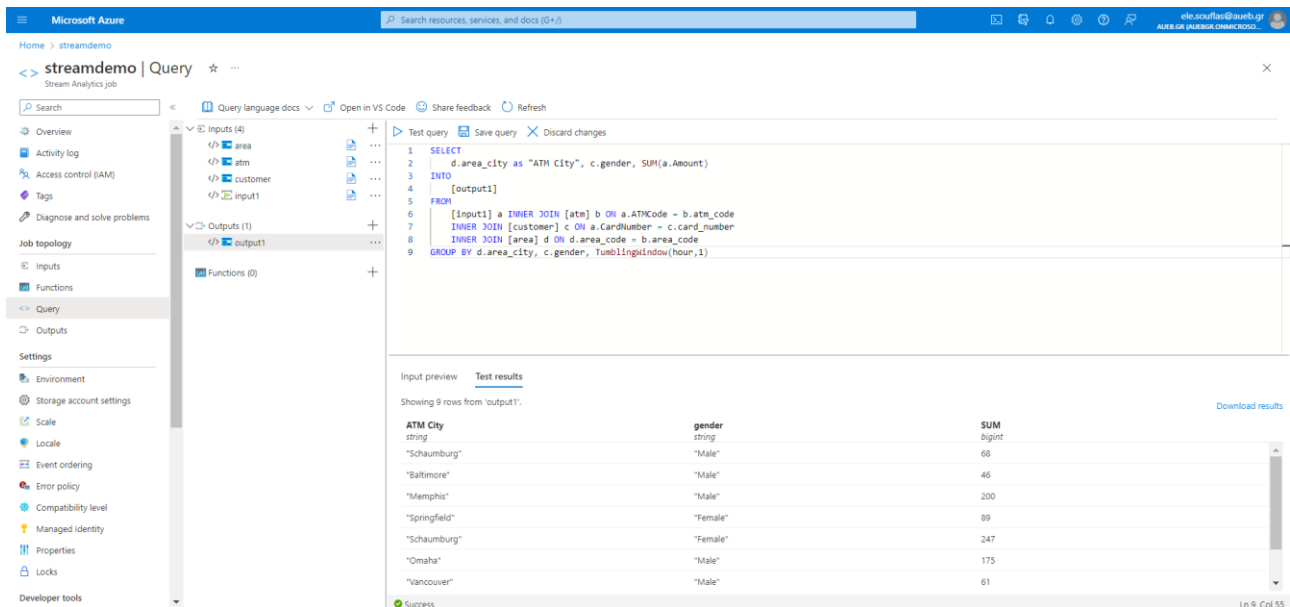


Figure 27 - Query 5 JSON Output

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).



The screenshot shows the Microsoft Azure Stream Analytics Query Editor interface. The query is as follows:

```

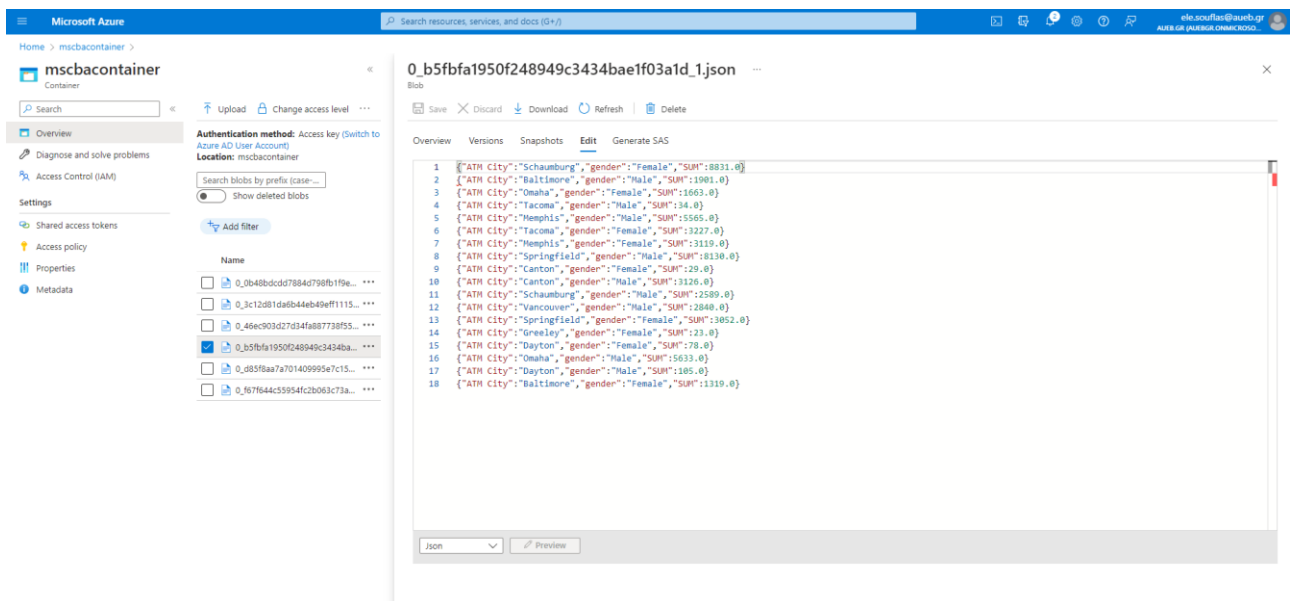
1 SELECT
2   d.area_code as "ATH City", c.gender, SUM(a.Amount)
3 INTO
4   [output1]
5 FROM
6   [input1] a INNER JOIN [atm] b ON a.ATMCode = b.atm_code
7   INNER JOIN [customer] c ON a.CardNumber = c.card_number
8   INNER JOIN [area] d ON d.area_code = b.area_code
9 GROUP BY d.area_code, c.gender, TumblingWindow(hour,1)

```

The test results show the following data:

ATH City	gender	SUM
"Schaumburg"	"Male"	68
"Baltimore"	"Male"	46
"Memphis"	"Male"	200
"Springfield"	"Female"	89
"Schaumburg"	"Female"	247
"Omaha"	"Male"	175
"Vancouver"	"Male"	61

Figure 28 - Query 6 Test Results



The screenshot shows the Microsoft Azure Blob Storage interface. The blob is named "0_b5fbfa1950f248949c3434baef03a1d_1.json". The JSON output is as follows:

```

1 [{"ATH City": "Schaumburg", "gender": "Female", "SUM": 8831.0}]
2 [{"ATH City": "Baltimore", "gender": "Male", "SUM": 1901.0}]
3 [{"ATH City": "Omaha", "gender": "Female", "SUM": 1663.0}]
4 [{"ATH City": "Tacoma", "gender": "Male", "SUM": 134.0}]
5 [{"ATH City": "Memphis", "gender": "Male", "SUM": 5565.0}]
6 [{"ATH City": "Tacoma", "gender": "Female", "SUM": 3227.0}]
7 [{"ATH City": "Memphis", "gender": "Female", "SUM": 3119.0}]
8 [{"ATH City": "Springfield", "gender": "Male", "SUM": 18130.0}]
9 [{"ATH City": "Canton", "gender": "Female", "SUM": 129.0}]
10 [{"ATH City": "Canton", "gender": "Male", "SUM": 3126.0}]
11 [{"ATH City": "Schaumburg", "gender": "Male", "SUM": 2589.0}]
12 [{"ATH City": "Vancouver", "gender": "Male", "SUM": 2840.0}]
13 [{"ATH City": "Springfield", "gender": "Female", "SUM": 13052.0}]
14 [{"ATH City": "Grevelly", "gender": "Female", "SUM": 133.0}]
15 [{"ATH City": "Dayton", "gender": "Female", "SUM": 78.0}]
16 [{"ATH City": "Omaha", "gender": "Male", "SUM": 5633.0}]
17 [{"ATH City": "Dayton", "gender": "Male", "SUM": 185.0}]
18 [{"ATH City": "Baltimore", "gender": "Female", "SUM": 1319.0}]

```

Figure 29 - Query 6 JSON Output

Query 7: Alert (Do a simple SELECT “1”) if a Customer has performed two transactions of “Type = 1” in a window of an hour (use a sliding window).

The screenshot shows the Microsoft Azure Stream Analytics portal. The query editor displays the following SQL query:

```

1 SELECT
2     1 as "ALERT"
3 INTO
4     [output1]
5 FROM
6     [input1]
7 WHERE
8     Type = 1
9 GROUP BY Type, CardNumber, SlidingWindow(hour,1)
10 HAVING COUNT(Amount) >= 2

```

The test results section shows the output of the query, displaying 5 rows from 'output1'.

ALERT
1
1
1
1
1

Figure 30 - Query 7 Test Results (COUNT >= 2 has been used and not COUNT=2 because a customer with 3 transactions has also performed 2 transactions in the same sliding window)

The screenshot shows the Microsoft Azure Blob Storage portal. The selected blob is named "0_81f1b05184a4783b41e253712882b99_1.json". The JSON output of the query is displayed, showing 29 rows of data, all of which are "1".

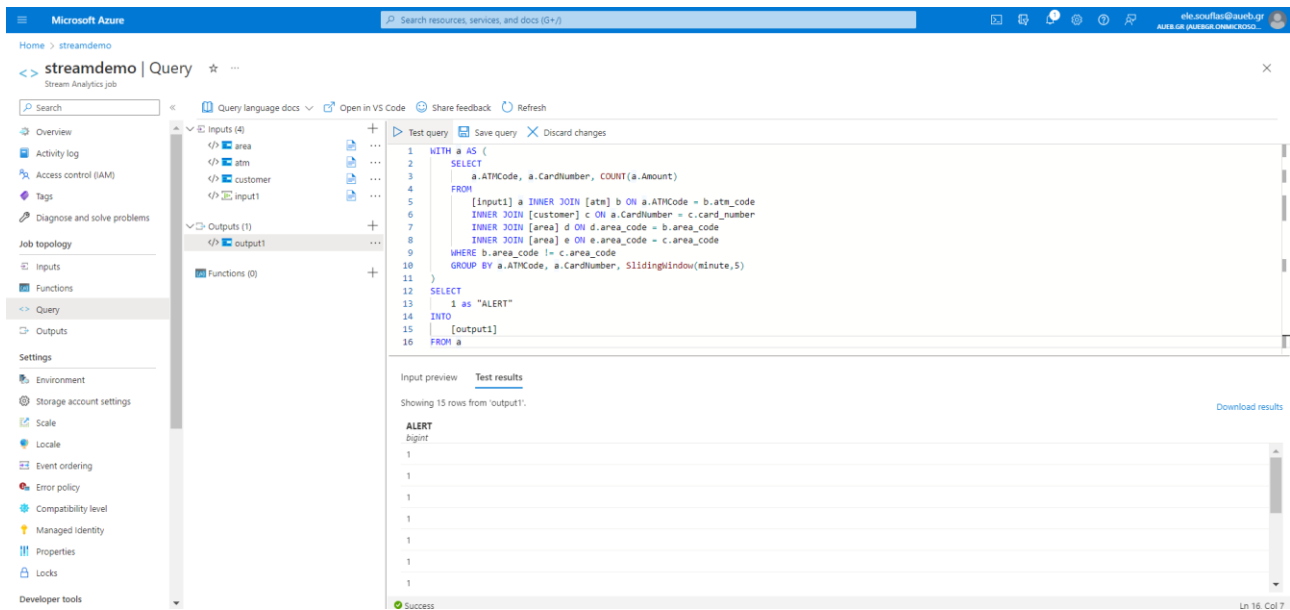
```

1 {"ALERT":1}
2 {"ALERT":1}
3 {"ALERT":1}
4 {"ALERT":1}
5 {"ALERT":1}
6 {"ALERT":1}
7 {"ALERT":1}
8 {"ALERT":1}
9 {"ALERT":1}
10 {"ALERT":1}
11 {"ALERT":1}
12 {"ALERT":1}
13 {"ALERT":1}
14 {"ALERT":1}
15 {"ALERT":1}
16 {"ALERT":1}
17 {"ALERT":1}
18 {"ALERT":1}
19 {"ALERT":1}
20 {"ALERT":1}
21 {"ALERT":1}
22 {"ALERT":1}
23 {"ALERT":1}
24 {"ALERT":1}
25 {"ALERT":1}
26 {"ALERT":1}
27 {"ALERT":1}
28 {"ALERT":1}
29 {"ALERT":1}

```

Figure 31 - Query 7 JSON Output

Query 8: Alert (Do a simple 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)



The screenshot shows the Microsoft Azure portal interface for a Stream Analytics job named 'streamdemo'. The 'Query' tab is active, displaying a SQL query that uses a sliding window to compare area codes of ATM transactions with card numbers. The query is as follows:

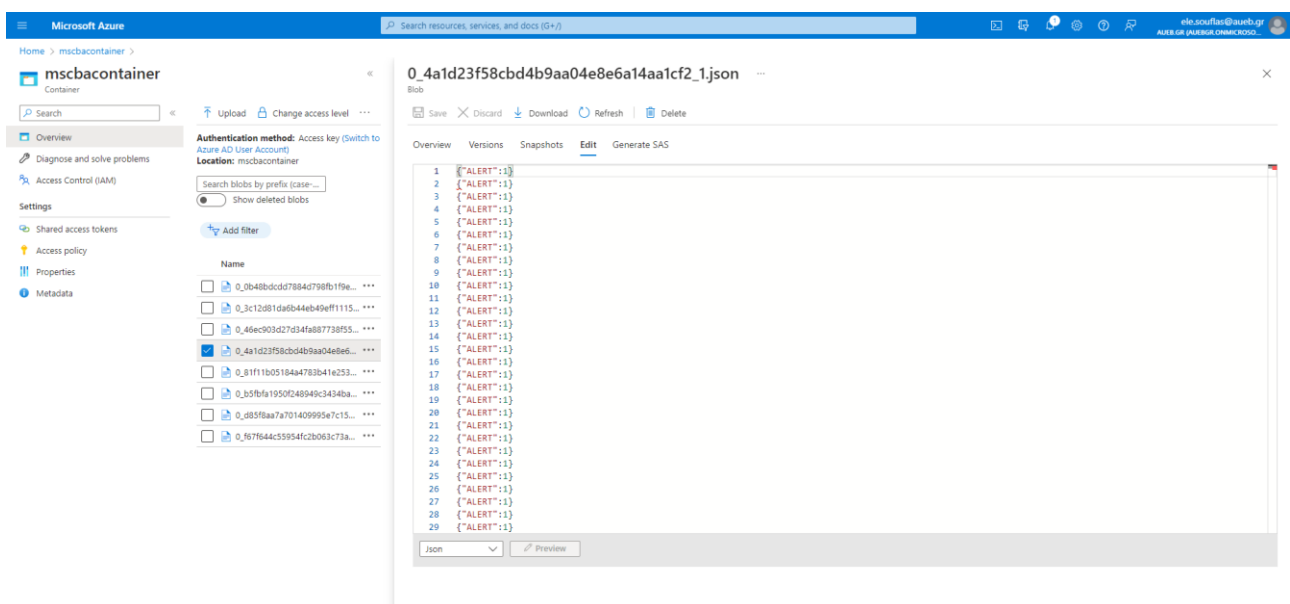
```

WITH a AS (
  SELECT
    a.ATMCode, a.CardNumber, COUNT(a.Amount)
  FROM
    [input1] a INNER JOIN [atn] b ON a.ATMCode = b.atm_code
    INNER JOIN [customer] c ON a.CardNumber = c.card_number
    INNER JOIN [area] d ON d.area_code = b.area_code
    INNER JOIN [area] e ON e.area_code = c.area_code
  WHERE b.area_code != c.area_code
  GROUP BY a.ATMCode, a.CardNumber, SlidingWindow(minute,5)
)
SELECT
  1 as "ALERT"
INTO
  [output1]
FROM a

```

The 'Test results' tab shows a preview of the output, which is a single column named 'ALERT' with values of '1'.

Figure 32 - Query 8 Test Results (a sliding window of 5 minutes has been arbitrarily used because it was not mentioned otherwise)



The screenshot shows the Microsoft Azure portal interface for a Blob container named 'mscbacontainer'. The '0_4a1d23f58cbd4b9aa04e8e6a14aa1cf2_1.json' file is selected, and the 'Edit' tab is active, displaying the JSON output of the query. The output is a single column named 'ALERT' with values of '1'.

Figure 33 - Query 8 JSON Output