MSc Business Analytics
Data Management & Business Intelligence

Azure Stream Analytics

LAKKAS-PYKNIS EVANGELOS
Registration number: f2822306
E-mail: eva.lakkaspyknis@aueb.gr

MESOLORA STAMATOULA-GERASIMOULA
Registration number: f2822308
E-mail: sta.mesolora@aueb.gr

Contents

Introduction

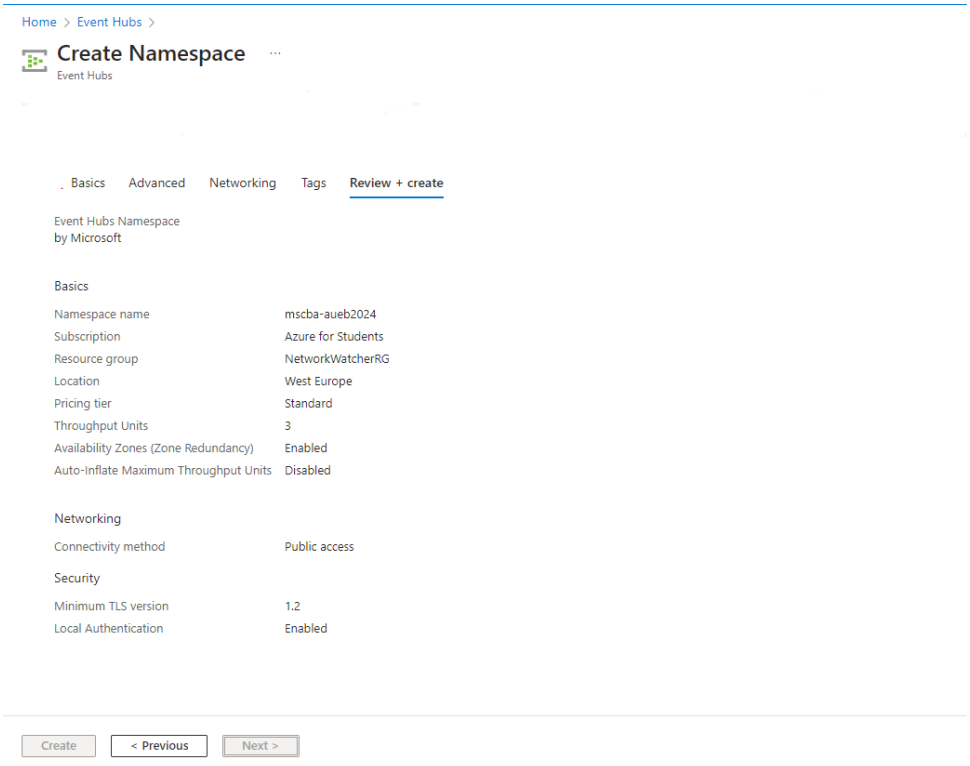
In this report, we delve into the realm of Azure Stream Analytics to harness its power in processing a continuous data stream composed of ATM transactions and addressing real-time stream queries. The stream schema, comprising vital transaction details such as ATMCode, CardNumber, Type, and Amount, forms the foundation of our analysis. The journey begins with the establishment of an Azure Student account, providing us with access to Azure services and a credit allowance to facilitate experimentation. Subsequently, we embark on a meticulous setup process involving the configuration of an Event Hub and the generation of a Security Access Signature using Windows terminal, enabling secure data access.

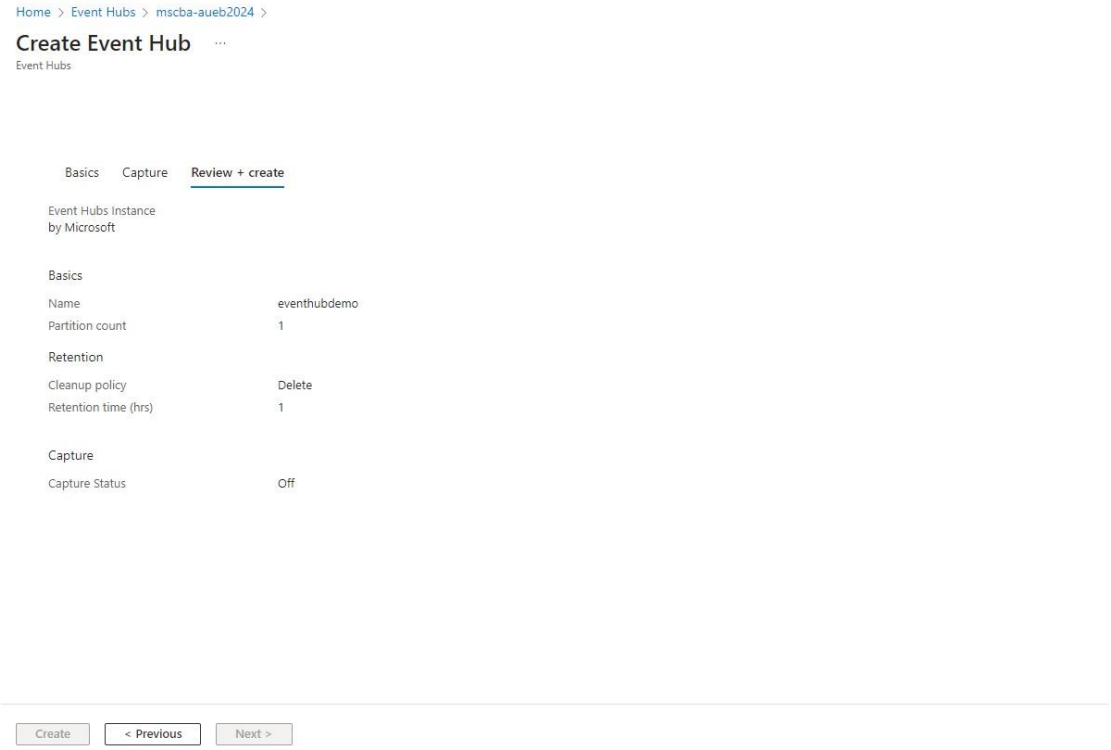
Method

We registered for an Azure Students account via the dedicated portal designed for students, granting us access to Azure services along with credits to explore cloud resources. To structure our cloud assets for the streaming analytics project, we initiated a Resource Group within Azure, ensuring a cohesive and efficient management approach.

Namespace & Eventhub creation

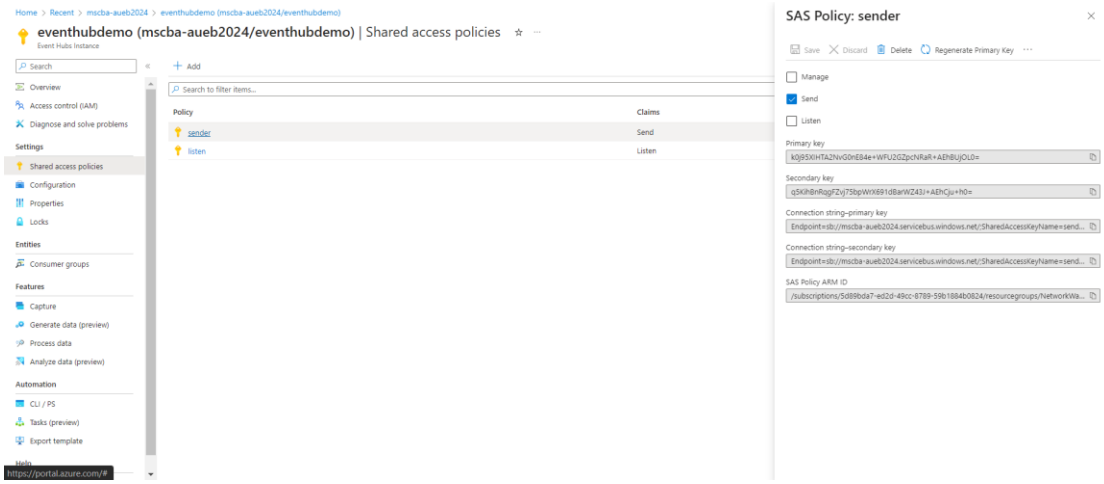
We first created a Namespace in order to create the event hub afterwards.

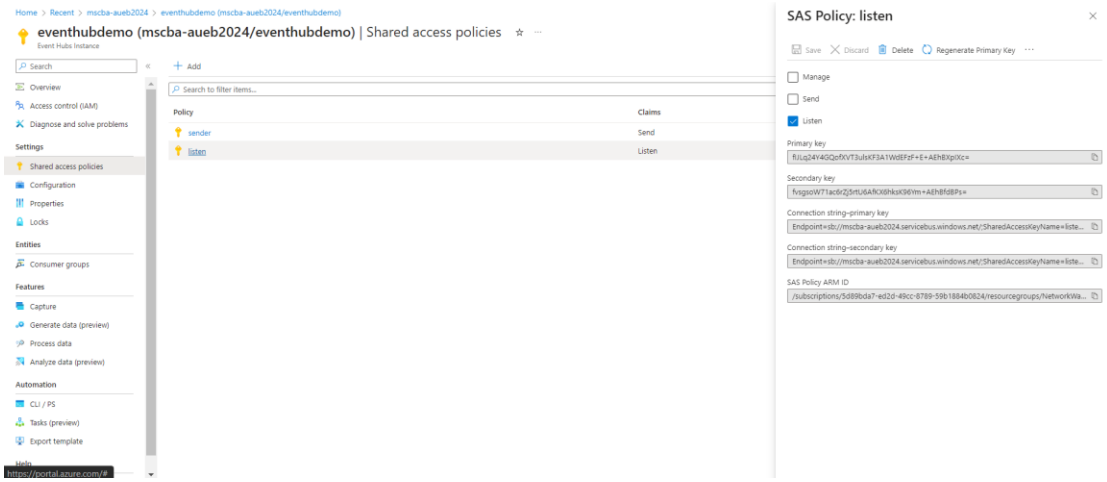




Shared access policies

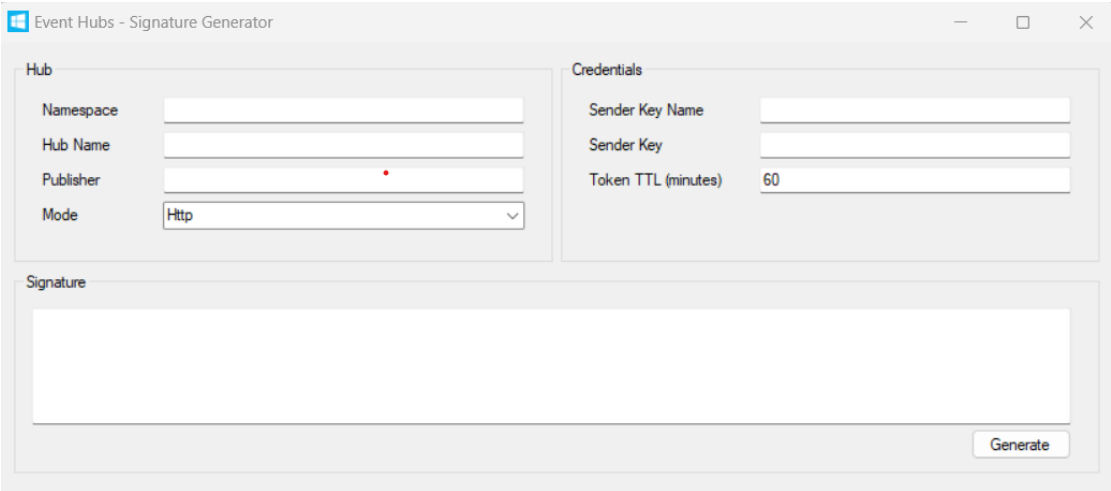
We created shared access policies in order to connect the EventHub with the data generator and the inputs for the stream analytics job.

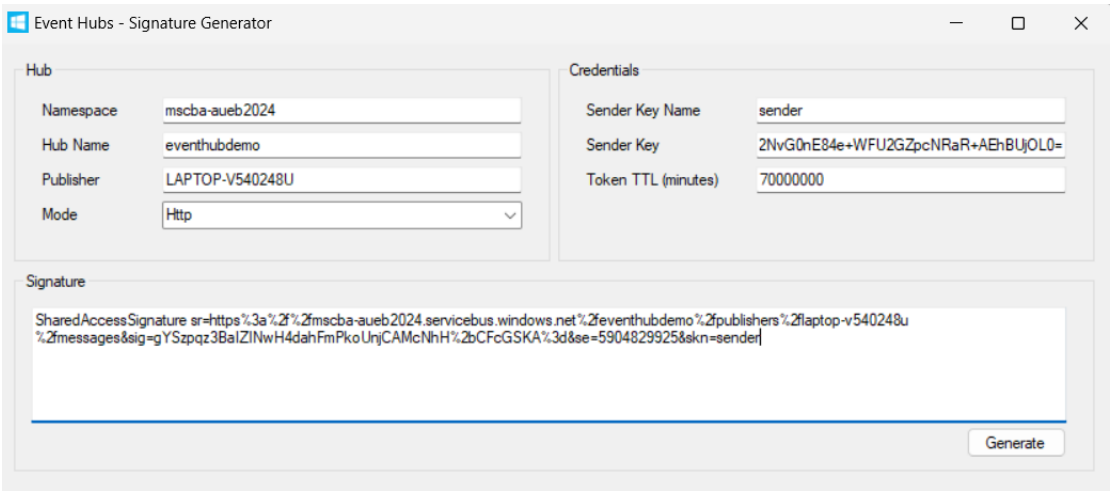


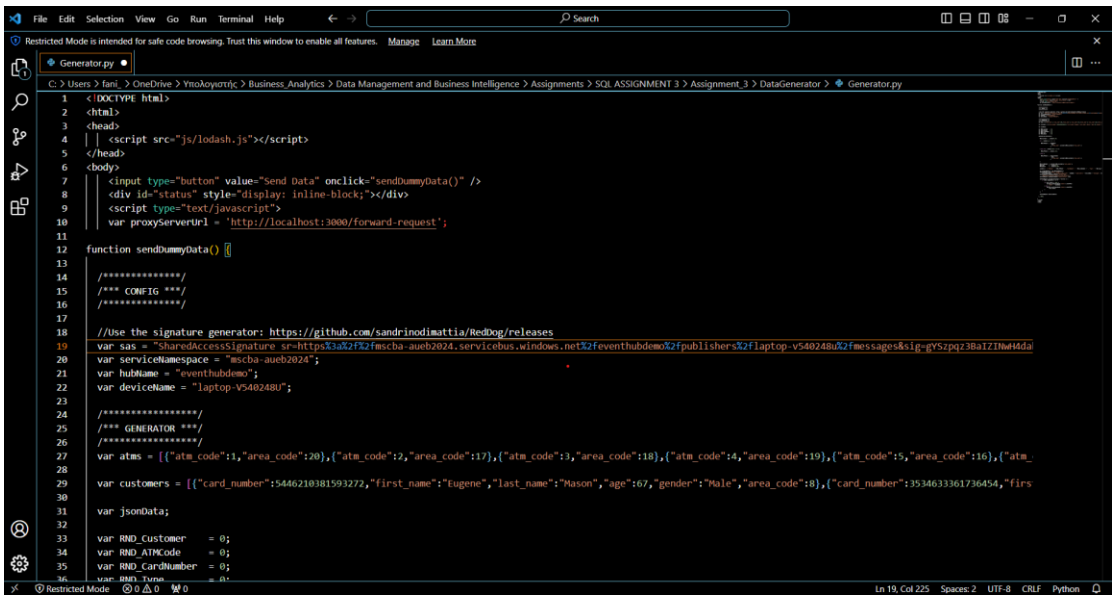


Signature generator:

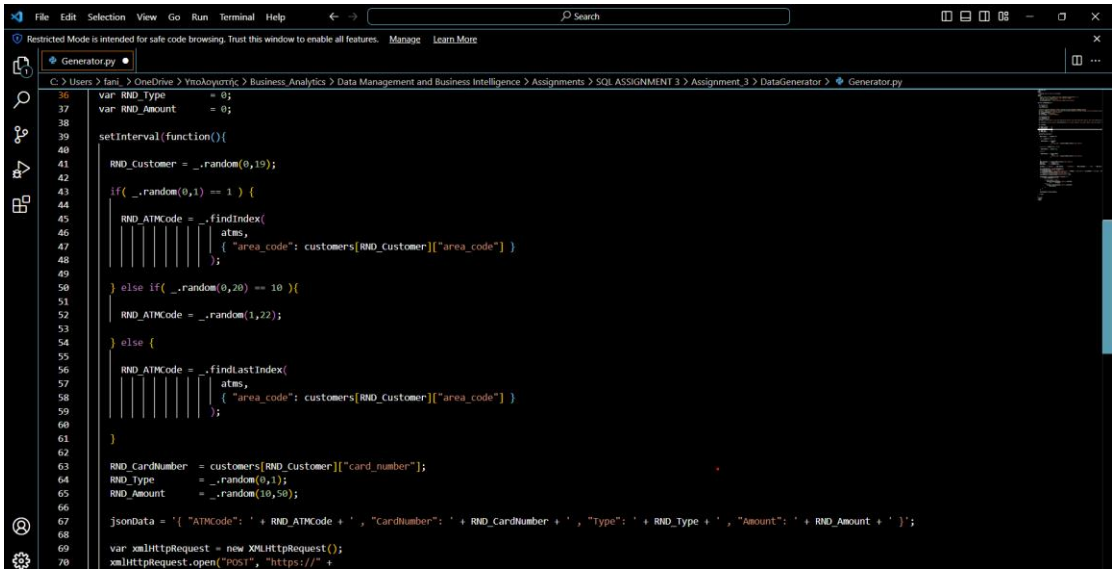
We generated a secure code in order to send data within our Azure Event Hub and modified the given Generator.html file by including the necessary scripts and data structure to enable the sending of data to our Event Hub. Then, we employed Generator.html to transmit simulated data to the Event Hub, effectively validating the data intake process.

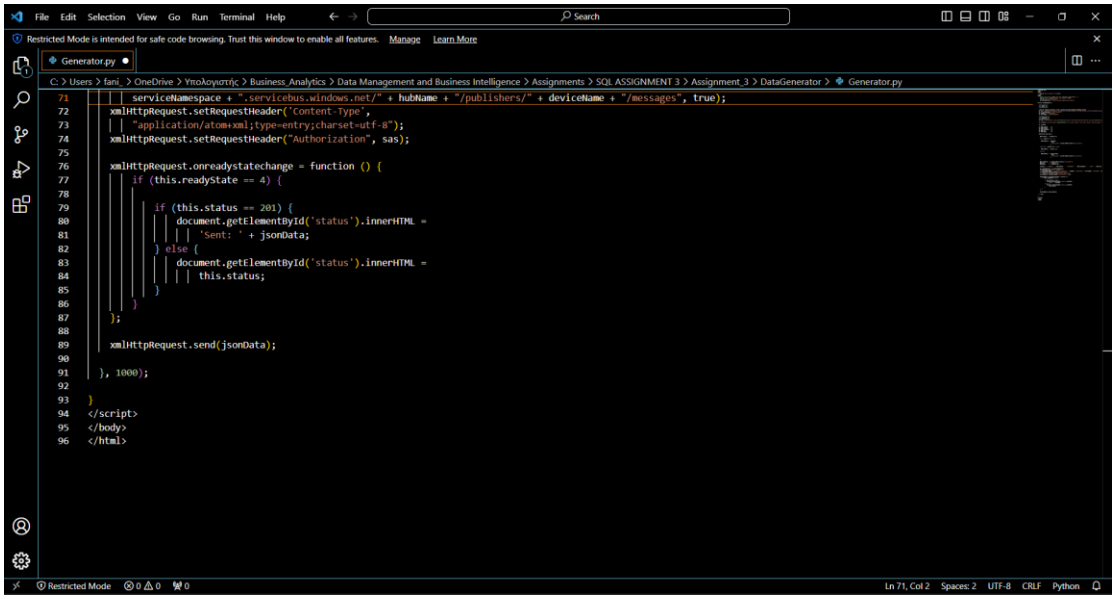



Event Hubs - Signature Generator
Hub
Namespace
mscba-aueb2024
Hub Name
eventhubdemo
Publisher
LAPTOP-V540248U
Mode
Http
Credentials
Sender Key Name
sender
Sender Key
2NvG0nE84e+WFU2GZpcNRaR+AEhBUjOL0=
Token TTL (minutes)
70000000
Signature
SharedAccessSignature sr=https%3a%2f%2fmscba-aueb2024.servicebus.windows.net%2feventhubdemo%2fpublishers%2flaptop-v540248u
%2fmessages&sig=gYSzpqz3BaIZINwH4dahFmPkoUnjCAMcNhH%2bCFcGSKA%3d&se=5904829925&skn=sender
Generate

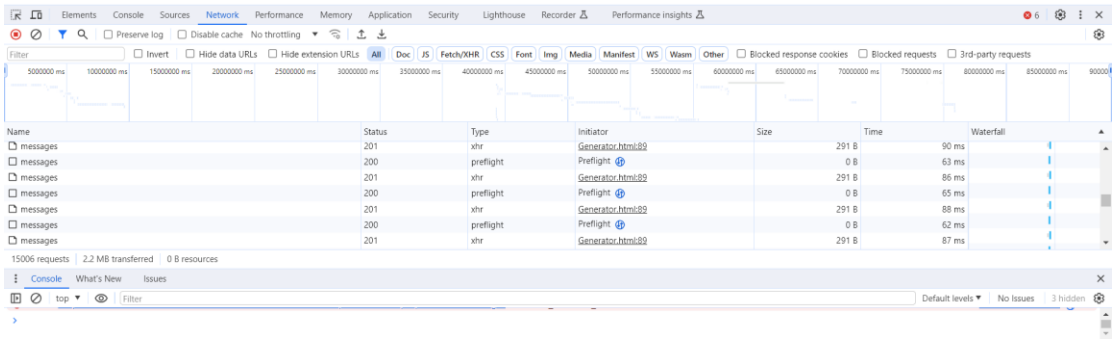

File Edit Selection View Go Run Terminal Help
Search
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
Generator.py
C: > Users > fani_ > OneDrive > Υπολογιστής > Business_Analytics > Data Management and Business Intelligence > Assignments > SQL ASSIGNMENT 3 > Assignment_3 > DataGenerator > Generator.py
<!DOCTYPE html>
<html>
<head>
<script src="js/lodash.js"></script>
</head>
<body>
<input type="button" value="Send Data" onclick="sendDummyData()" />

<script type="text/javascript">
var proxyServerUrl = 'http://localhost:3000/forward-request';
function sendDummyData() {
/*************/
/*** CONFIG ***/
/*************/
//Use the signature generator: https://github.com/sandrinodimattia/RedDog/releases
var sas = "SharedAccessSignature sr=https%3a%2f%2fmscba-aueb2024.servicebus.windows.net%2feventhubdemo%2fpublishers%2flaptop-v540248u%2fmessages&sig=gYSzpqz3BaIZINwH4da
var serviceNamespace = "mscba-aueb2024";
var hubName = "eventhubdemo";
var deviceName = "laptop-V540248U";
/****************/
/*** GENERATOR ***/
/****************/
var atms = [{"atm_code":1,"area_code":20},{"atm_code":2,"area_code":17},{"atm_code":3,"area_code":18},{"atm_code":4,"area_code":19},{"atm_code":5,"area_code":16},{"atm_
var customers = [{"card_number":5446210381593272,"first_name":"Eugene","last_name":"Mason","age":67,"gender":"Male","area_code":8},{"card_number":3534633361736454,"firs
var jsonData;
var RND_Customer = 0;
var RND_ATMCode = 0;
var RND_CardNumber = 0;
Restricted Mode
Ln 19, Col 225 Spaces: 2 UTF-8 CRLF Python


File Edit Selection View Go Run Terminal Help
Search
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
Generator.py
C: > Users > fani_ > OneDrive > Υπολογιστής > Business_Analytics > Data Management and Business Intelligence > Assignments > SQL ASSIGNMENT 3 > Assignment_3 > DataGenerator > Generator.py
var RND_Type = 0;
var RND_Amount = 0;
setInterval(function(){
RND_Customer = _.random(0,19);
if(_.random(0,1) == 1) {
RND_ATMCode = _.findIndex(
atms,
{ "area_code": customers[RND_Customer]["area_code"] }
);
} else if(_.random(0,20) == 10){
RND_ATMCode = _.random(1,22);
} else {
RND_ATMCode = _.findLastIndex(
atms,
{ "area_code": customers[RND_Customer]["area_code"] }
);
}
RND_CardNumber = customers[RND_Customer]["card_number"];
RND_Type = _.random(0,1);
RND_Amount = _.random(10,50);
jsonData = '{ "ATMCode": ' + RND_ATMCode + ' , "CardNumber": ' + RND_CardNumber + ' , "Type": ' + RND_Type + ' , "Amount": ' + RND_Amount + ' }';
var xmlHttpRequest = new XMLHttpRequest();
xmlHttpRequest.open("POST", "https://" +

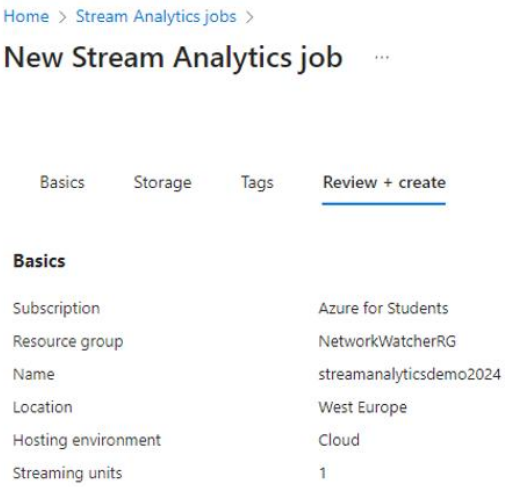


Send Data Sent: { "ATMCode": 19 , "CardNumber": 3554025590595485 , "Type": 0 , "Amount": 18 }

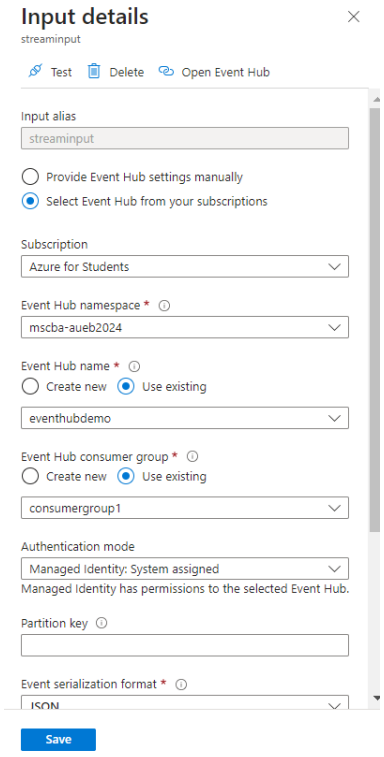


Stream analytics job

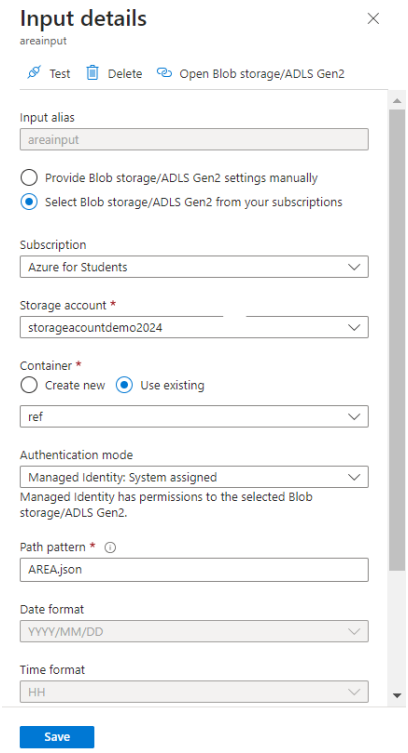
We set up a Stream Analytics Job, outlining where the data comes from and where it goes. This arrangement enabled ongoing analysis, handling live data streams to derive insights and generate actionable results. Afterwards, we incorporated the Event Hub and uploaded Reference Data files as the data sources in our Stream Analytics Job to ensure the job has the necessary information for processing.



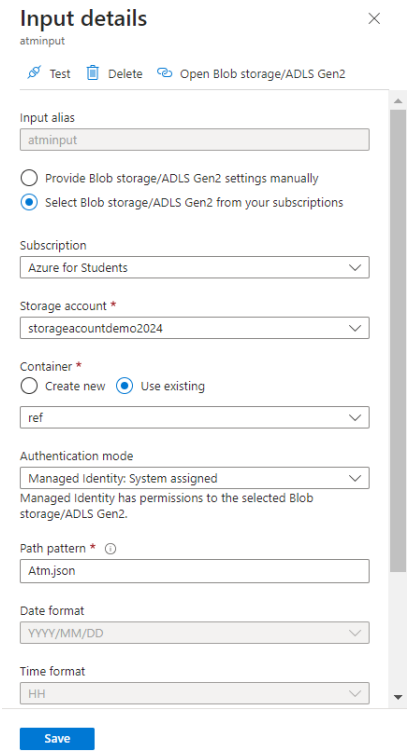
- Stream input



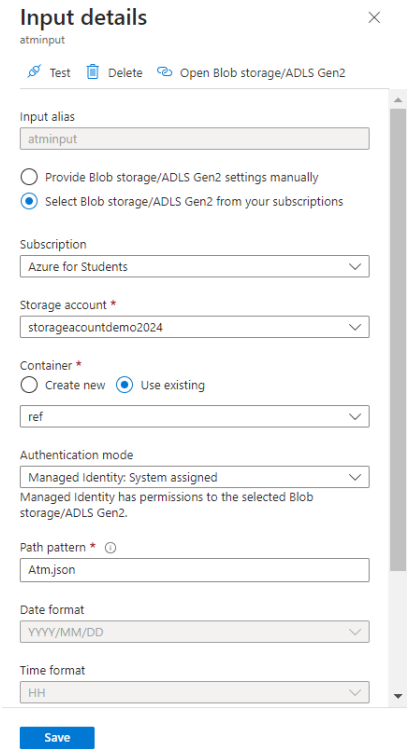
- areainput

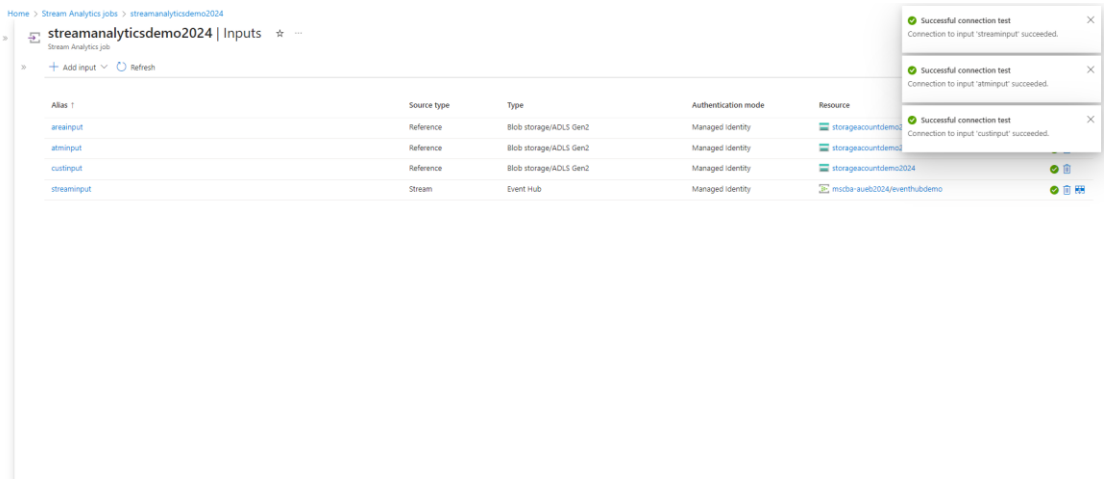


- atminput



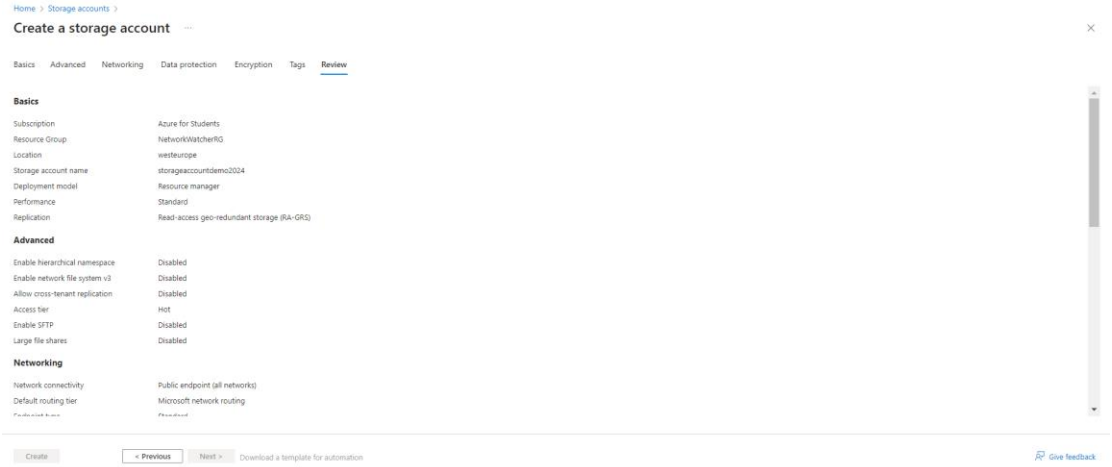
- Custinput

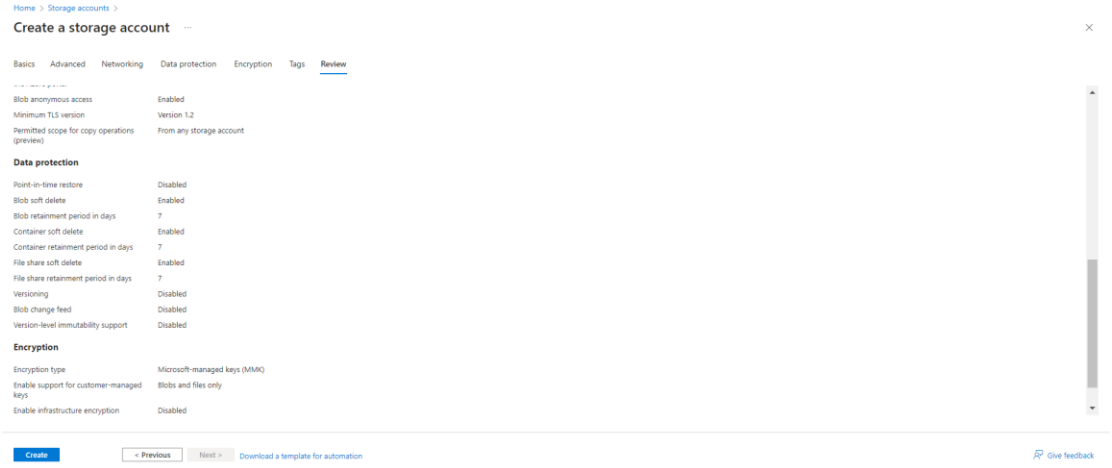




Storage account

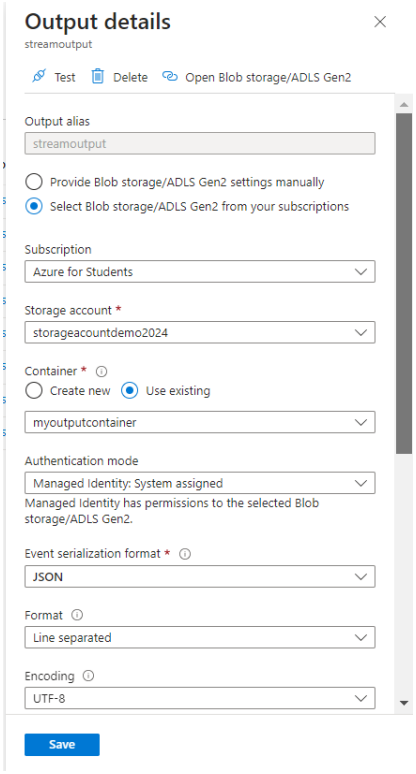
Creation of storage account in order to build a container which hosts the reference data files.

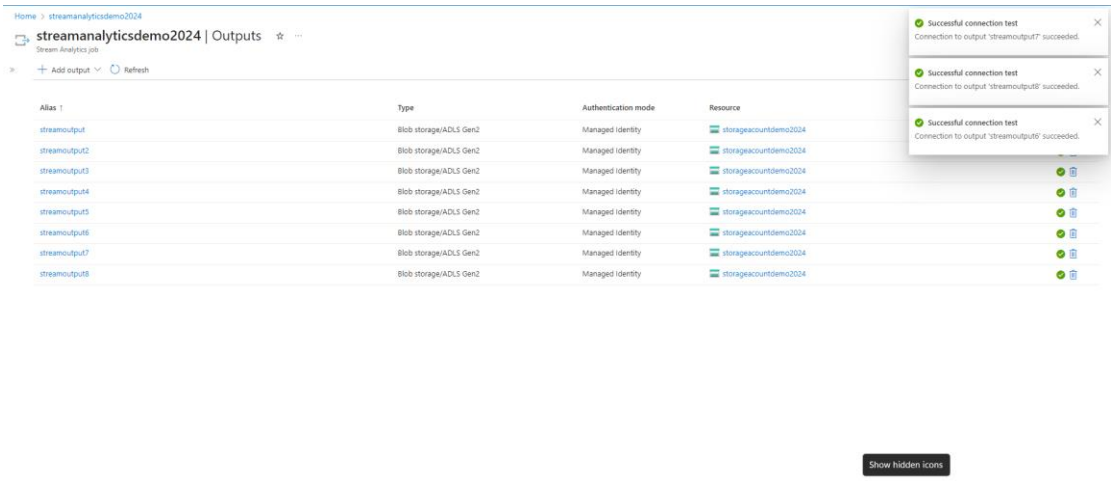




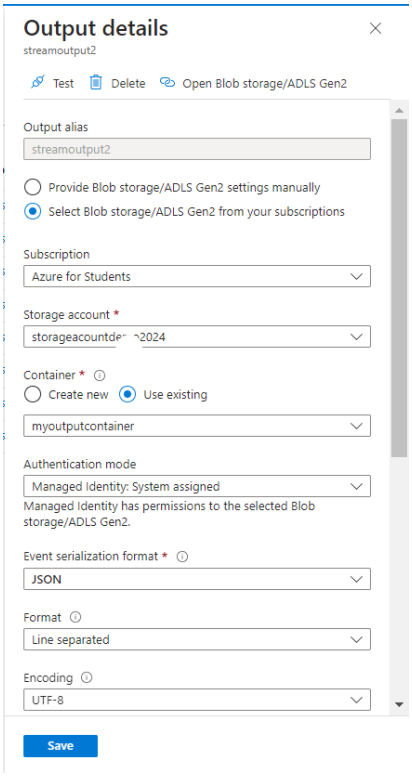
Stream output

We designated Blob Storage to collect the processed data from your Stream Analytics Job, storing it for future use or analysis. During the setup of our analytics framework, we faced hurdles concerning the configuration of streaming units. To overcome these challenges, we adapted our settings to utilize a single processing streaming unit and established eight storage outputs for each query, enabling concurrent processing of all tasks.


Output details
streamoutput
Test
Delete
Open Blob storage/ADLS Gen2
Output alias
streamoutput
Provide Blob storage/ADLS Gen2 settings manually
Select Blob storage/ADLS Gen2 from your subscriptions
Subscription
Azure for Students
Storage account *
storageacountdemo2024
Container *
Create new
Use existing
myoutputcontainer
Authentication mode
Managed Identity: System assigned
Managed Identity has permissions to the selected Blob storage/ADLS Gen2.
Event serialization format *
JSON
Format
Line separated
Encoding
UTF-8
Save

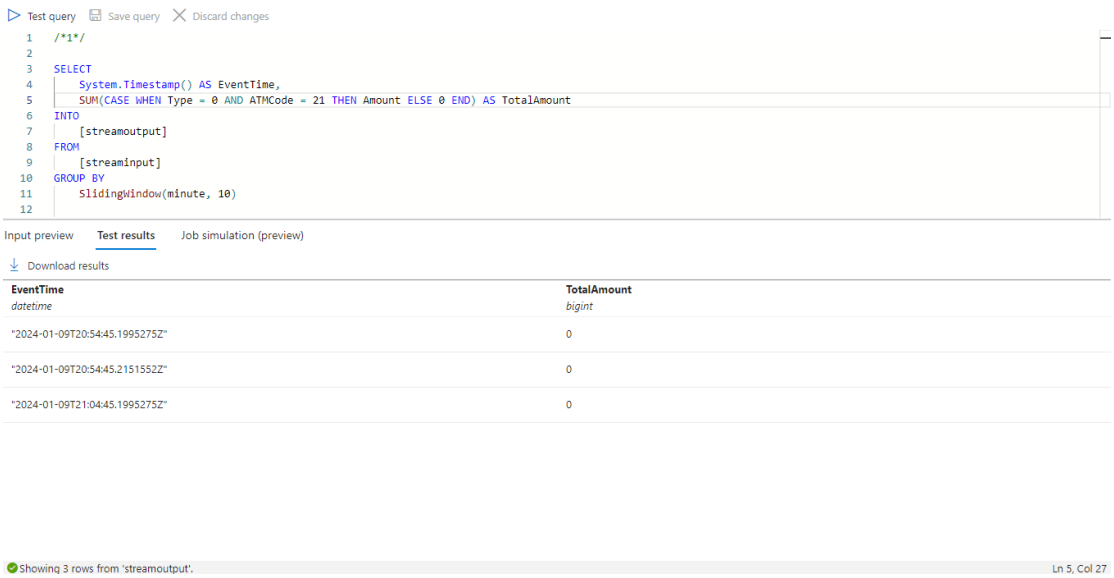

Home > streamanalyticsdemo2024
streamanalyticsdemo2024 | Outputs
Stream Analytics job
Add output
Refresh
Alias
Type
Authentication mode
Resource
streamoutput
streamoutput2
streamoutput3
streamoutput4
streamoutput5
streamoutput6
streamoutput7
streamoutput8
Blob storage/ADLS Gen2
Managed Identity
storageacountdemo2024
Successful connection test
Connection to output 'streamoutput7' succeeded.
Successful connection test
Connection to output 'streamoutput8' succeeded.
Successful connection test
Connection to output 'streamoutput6' succeeded.

Show hidden icons

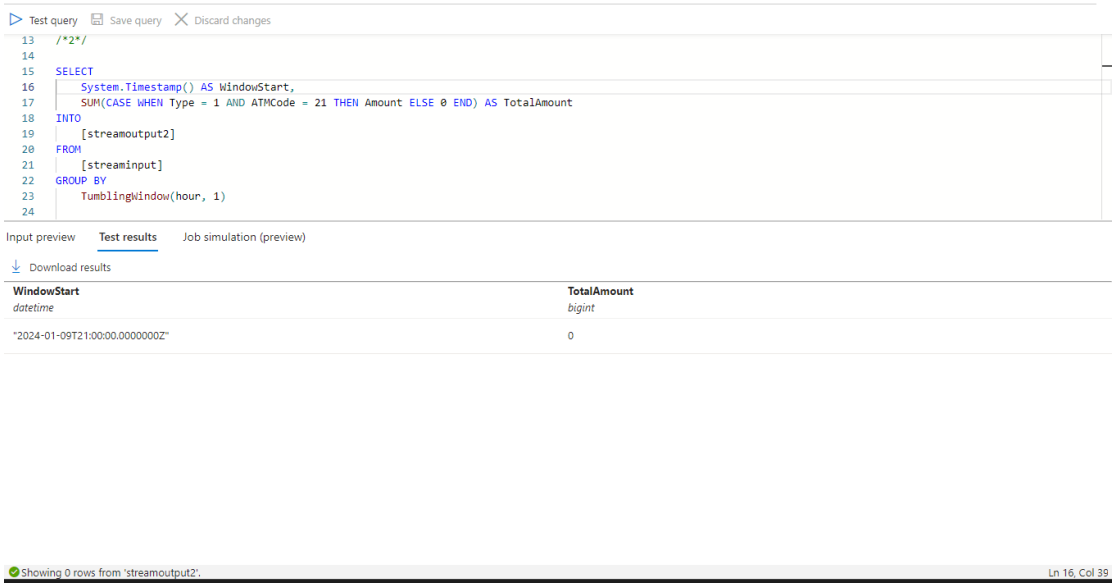


SQL Queries

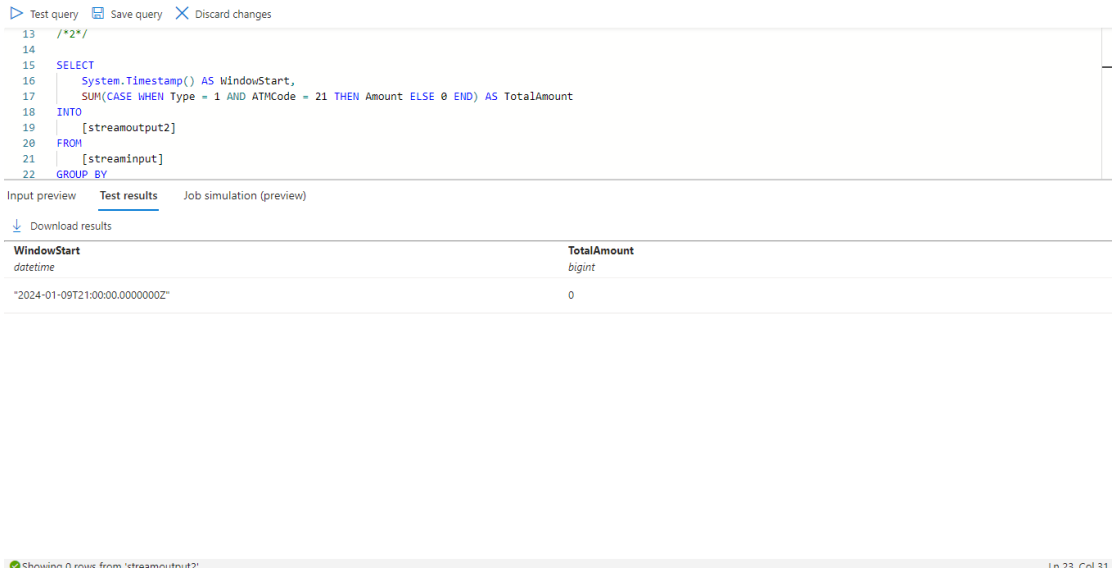
Query1



Query 2



An hour later:



Query 3

```
/*3*/

SELECT
    MIN(System.Timestamp()) AS WindowStart,
    SUM(CASE WHEN Type = 1 AND ATMCode = 21 THEN Amount ELSE 0 END) AS TotalAmount
INTO
    [streamoutput3]
FROM
    [streaminput]
GROUP BY
    HoppingWindow(minute, 60, 30)
```

Input preview | **Test results** | Job simulation (preview)

Download results

WindowStart *datetime*	TotalAmount *bigint*
"2024-01-09T21:00:21.2812542Z"	0
"2024-01-09T21:00:21.2812542Z"	0

Showing 0 rows from 'streamoutput3'. Ln 35, Col 34

An hour later:

Test query | Save query | Discard changes

```
/*3*/

SELECT
    MIN(System.Timestamp()) AS WindowStart,
    SUM(CASE WHEN Type = 1 AND ATMCode = 21 THEN Amount ELSE 0 END) AS TotalAmount
INTO
    [streamoutput3]
FROM
    [streaminput]
GROUP BY
```

Input preview | **Test results** | Job simulation (preview)

Download results

WindowStart *datetime*	TotalAmount *bigint*
"2024-01-09T21:01:12.8313297Z"	0
"2024-01-09T21:01:12.8313297Z"	0
"2024-01-09T21:01:12.8313297Z"	0
"2024-01-09T21:01:12.8313297Z"	0

Showing 0 rows from 'streamoutput3'. Ln 36, Col 1

Query 4

Test query Save query Discard changes

```
37  /*4*/
38
39  SELECT
40      System.Timestamp() AS WindowStart,
41      ATMCode,
42      SUM(CASE WHEN Type = 1 THEN Amount ELSE 0 END) AS TotalAmount
43  INTO
44      [streamoutput4]
45  FROM
46      [streaminput]
47  GROUP BY
48      SlidingWindow(hour, 1), ATMCode
```

Input preview **Test results** Job simulation (preview)

Download results

WindowStart *datetime*	ATMCode *bigint*	TotalAmount *bigint*
"2024-01-09T21:02:07.6314985Z"	20	22
"2024-01-09T21:02:07.6471352Z"	10	45
"2024-01-09T21:02:07.6471352Z"	13	46
"2024-01-09T21:02:07.6471352Z"	15	0
"2024-01-09T21:02:07.6471352Z"	16	17

Showing 0 rows from 'streamoutput4'. Ln 40, Col 23

An hour later:

Test query Save query Discard changes

```
37  /*4*/
38
39  SELECT
40      System.Timestamp() AS WindowStart,
41      ATMCode,
42      SUM(CASE WHEN Type = 1 THEN Amount ELSE 0 END) AS TotalAmount
43  INTO
44      [streamoutput4]
45  FROM
46      [streaminput]
47  GROUP BY
```

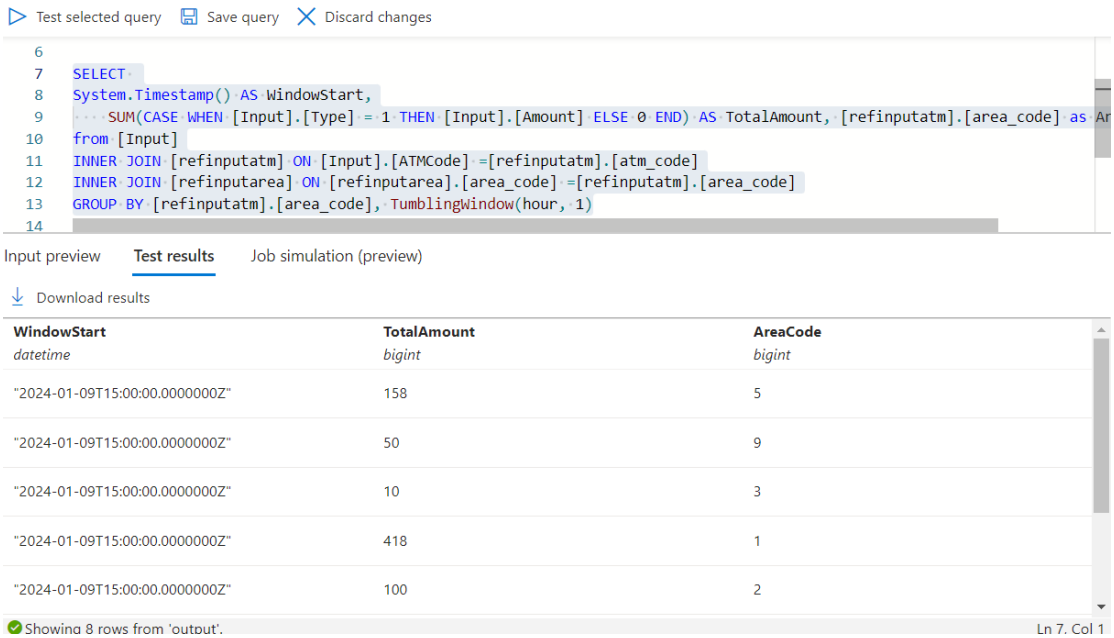
Input preview **Test results** Job simulation (preview)

Download results

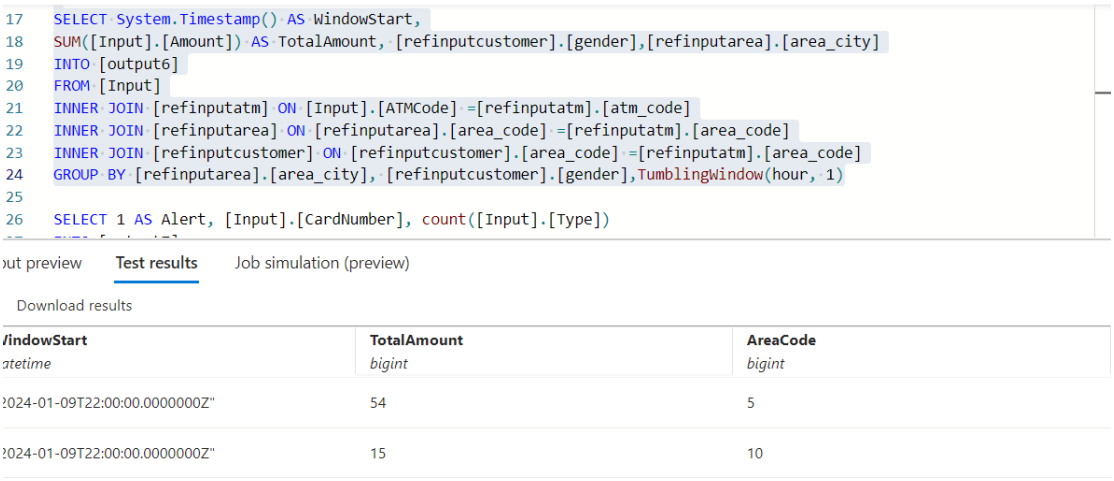
WindowStart *datetime*	ATMCode *bigint*	TotalAmount *bigint*
"2024-01-09T21:02:52.2603000Z"	20	22
"2024-01-09T21:02:52.2759173Z"	10	45
"2024-01-09T21:02:52.2759173Z"	13	46
"2024-01-09T21:02:52.2759173Z"	15	0
"2024-01-09T21:02:52.2759173Z"	16	17

Showing 0 rows from 'streamoutput4'. Ln 43, Col 5

Query 5



An hour later:



Query 6

```
SELECT System.Timestamp() AS WindowStart,
SUM([Input].[Amount]) AS TotalAmount, [refinputcustomer].[gender],[refinputarea].[area_city]
INTO [output6]
FROM [Input]
INNER JOIN [refinputatm] ON [Input].[ATMCode] =[refinputatm].[atm_code]
INNER JOIN [refinputarea] ON [refinputarea].[area_code] =[refinputatm].[area_code]
INNER JOIN [refinputcustomer] ON [refinputcustomer].[area_code] =[refinputatm].[area_code]
GROUP BY [refinputarea].[area_city], [refinputcustomer].[gender],TumblingWindow(hour, 1)
```

nput preview | **Test results** | Job simulation (preview)

Download results

WindowStart	TotalAmount	gender	area_city
datetime	*bigint*	*string*	*string*
"2024-01-09T18:00:00.0000000Z"	54	"Male"	"Tacoma"
"2024-01-09T18:00:00.0000000Z"	162	"Female"	"Tacoma"
"2024-01-09T18:00:00.0000000Z"	158	"Male"	"Omaha"
"2024-01-09T18:00:00.0000000Z"	79	"Female"	"Omaha"
"2024-01-09T18:00:00.0000000Z"	476	"Male"	"Springfield"

Showing 7 rows from 'output6'. Ln 23, Col 89

An hour later:

```
SELECT System.Timestamp() AS WindowStart,
SUM([Input].[Amount]) AS TotalAmount, [refinputcustomer].[gender],[refinputarea].[area_city]
INTO [output6]
FROM [Input]
INNER JOIN [refinputatm] ON [Input].[ATMCode] =[refinputatm].[atm_code]
INNER JOIN [refinputarea] ON [refinputarea].[area_code] =[refinputatm].[area_code]
INNER JOIN [refinputcustomer] ON [refinputcustomer].[area_code] =[refinputatm].[area_code]
GROUP BY [refinputarea].[area_city], [refinputcustomer].[gender],TumblingWindow(hour, 1)
```

nput preview | **Test results** | Job simulation (preview)

Download results

WindowStart	TotalAmount	gender	area_city
datetime	*bigint*	*string*	*string*
"2024-01-09T22:00:00.0000000Z"	15	"Male"	"Baltimore"

Showing 0 rows from 'output6'. Ln 25, Col 1

Query 7

```
SELECT 1 AS Alert, [Input].[CardNumber], count([Input].[Type])
INTO [output7]
FROM [Input]
INNER JOIN refinputcustomer
ON [Input].[CardNumber]=[refinputcustomer].[card_number]
WHERE [Input].[Type]=1
GROUP BY [Input].[CardNumber], [Input].[Type],SlidingWindow(hour, 1)
HAVING count([Input].[Type])=2
```

Query 8

```
33
34  SELECT 1 AS Alert, [Input].[ATMCode], [refinputcustomer].[area_code]
35  INTO [output8]
36  FROM [Input]
37  INNER JOIN refinputcustomer
38  ON [Input].[CardNumber]=[refinputcustomer].[card_number]
39  WHERE [refinputcustomer].[area_code]!=[Input].[ATMCode]
40  GROUP BY [Input].[ATMCode], [refinputcustomer].[area_code], SlidingWindow(hour, 1)
```

Input preview | Test results | Job simulation (preview)

Showing data from uploaded file 'Atm.json'.

View in JSON | Table | Raw | Reset | Upload sample input | Download sample data

atm_code *bigint*	area_code *bigint*
1	20
2	17
3	18
4	19

Success

An hour later:

```
35  SELECT 1 AS Alert, [Input].[ATMCode], [refinputcustomer].[area_code]
36  INTO [output8]
37  FROM [Input]
38  INNER JOIN refinputcustomer
39  ON [Input].[CardNumber]=[refinputcustomer].[card_number]
40  WHERE [refinputcustomer].[area_code]!=[Input].[ATMCode]
41  GROUP BY [Input].[ATMCode], [refinputcustomer].[area_code], SlidingWindow(hour, 1)
42
43  SELECT
```

Input preview | Test results | Job simulation (preview)

Download results

Alert *bigint*	ATMCode *bigint*	area_code *bigint*
1	13	6
1	15	7
1	10	8