BIG DATA MANAGEMENT SYSTEMS: PROJECT #4 – AZURE STREAM ANALYTICS

July 5, 2019

Zoe Kotti and Chryssa Nampouri
Department of Management Science and Technology
Athens University of Economics and Business
Athens, Greece
{t8150062, t8150096}@aueb.gr

Supervisor: Prof. Damianos Chatziantoniou

Contents

1 Project Description . . . 2
2 Azure Configuration . . . 3
3 Azure Configuration: Screenshots . . . 4
4 Azure Stream Analytics: Queries . . . 6

1 Project Description

In the context of this assignment a set of Reference Data is given that consists of three JSON files:

1. **Customer.json:** Personal data about customers who have made transactions. Each customer example is described by the following attributes:
    - *card_number* (integer)
    - *first_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)

The purpose of this project was to process a data stream of ATM transactions and answer stream queries. The schema of the stream is the following: (ATMCode, CardNumber, Type, Amount).

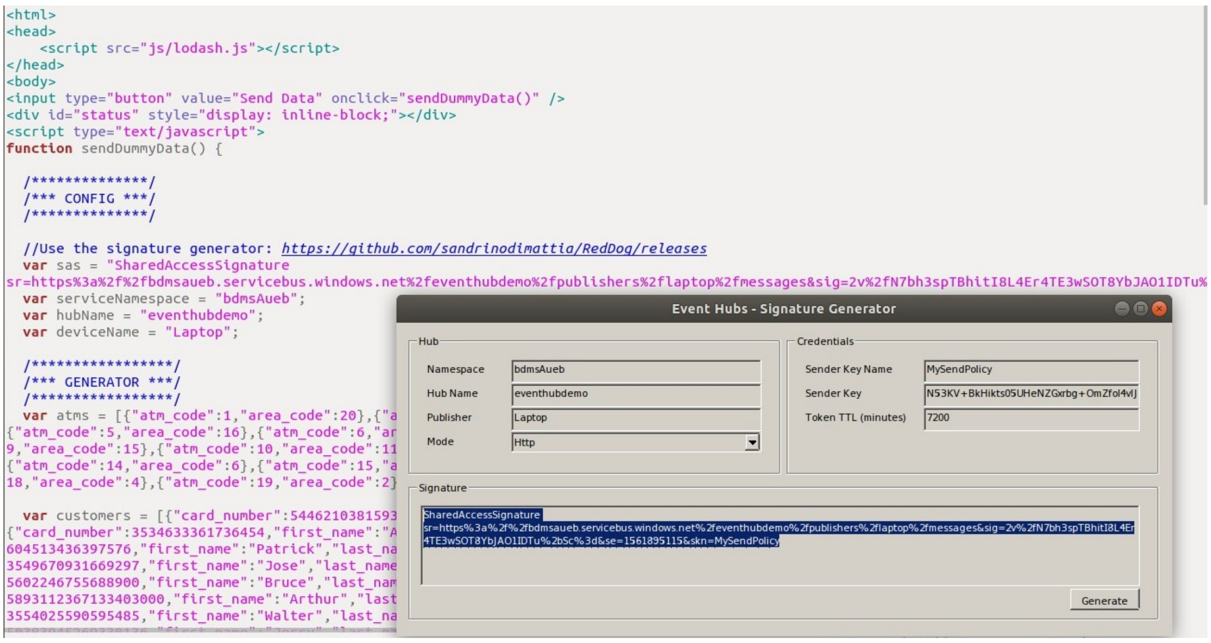
2 Azure Configuration

In order to execute the above process on Azure Stream Analytics Platform the following steps are required:

1. Create an Azure account.
2. Setup an Event Hub.
3. Generate a Security Access Signature [1].
4. Edit Generator.html and update the CONFIG variables with your security access signature.
5. Feed the Event Hub with streaming data by using the Generator.html.
6. Setup a Storage account.
7. Upload the Reference Data files to your storage account.
8. Setup a Stream Analytics Job.
9. Use the Event Hub and the Reference Data Files as Input.
10. Create a Blob Storage Output.
11. Run queries.

[1] https://github.com/sandrinodimattia/RedDog/releases

3 Azure Configuration: Screenshots



```
<html>
<head>
    <script src="js/lodash.js"></script>
</head>
<body>
<input type="button" value="Send Data" onclick="sendDummyData()" />
<div id="status" style="display: inline-block;"></div>
<script type="text/javascript">
function sendDummyData() {

  /**************/
  /*** CONFIG ***/
  /**************/

  //Use the signature generator: https://github.com/sandrinodimattia/RedDog/releases
  var sas = "SharedAccessSignature
sr=https%3a%2f%2fbdmsaueb.servicebus.windows.net%2feventhubdemo%2fpublishers%2flaptop%2fmessages&sig=2v%2fN7bh3spTBhitI8L4Er4TE3wSOT8YbJAO1IDTu%
  var serviceNamespace = "bdmsAueb";
  var hubName = "eventhubdemo";
  var deviceName = "Laptop";

  /*****************/
  /*** GENERATOR ***/
  /*****************/
  var atms = [{"atm_code":1,"area_code":20},{"a
{"atm_code":5,"area_code":16},{"atm_code":6,"ar
9,"area_code":15},{"atm_code":10,"area_code":11
{"atm_code":14,"area_code":6},{"atm_code":15,"a
18,"area_code":4},{"atm_code":19,"area_code":2}

  var customers = [{"card_number":5446210381593
{"card_number":3534633361736454,"first_name":"A
604513436397576,"first_name":"Patrick","last_na
3549670931669297,"first_name":"Jose","last_name
5602246755688900,"first_name":"Bruce","last_nam
5893112367133403000,"first_name":"Arthur","last
3554025590595485,"first_name":"Walter","last_na
```

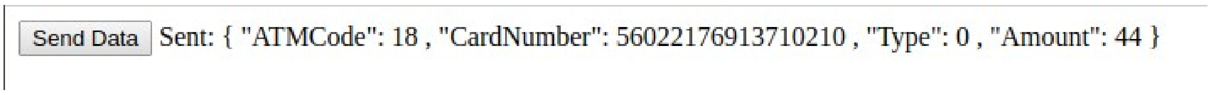
Hub	
Namespace	bdmsAueb
Hub Name	eventhubdemo
Publisher	Laptop
Mode	Http

Credentials	
Sender Key Name	MySendPolicy
Sender Key	N53KV+BkHikts05UHeNZGxrbg+OmZfol4vlJ
Token TTL (minutes)	7200

Signature

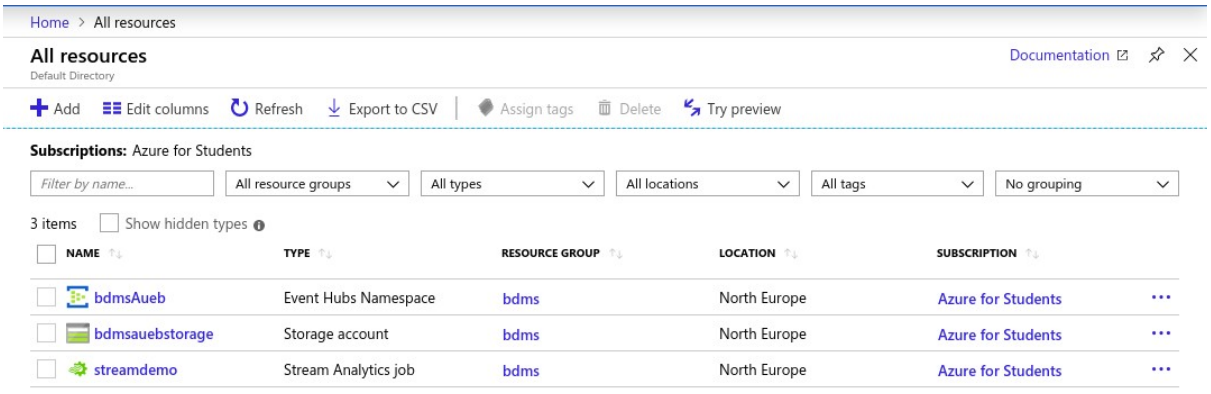
```
SharedAccessSignature
sr=https%3a%2f%2fbdmsaueb.servicebus.windows.net%2feventhubdemo%2fpublishers%2flaptop%2fmessages&sig=2v%2fN7bh3spTBhitI8L4Er
4TE3wSOT8YbJAO1IDTu%2bSc%3d&se=1561895115&skn=MySendPolicy
```

Figure 1: Security Access Signature (*Step 3,4).*



```
Sent: { "ATMCode": 18 , "CardNumber": 56022176913710210 , "Type": 0 , "Amount": 44 }
```

Figure 2: Random Data Generator (*Step 5).*

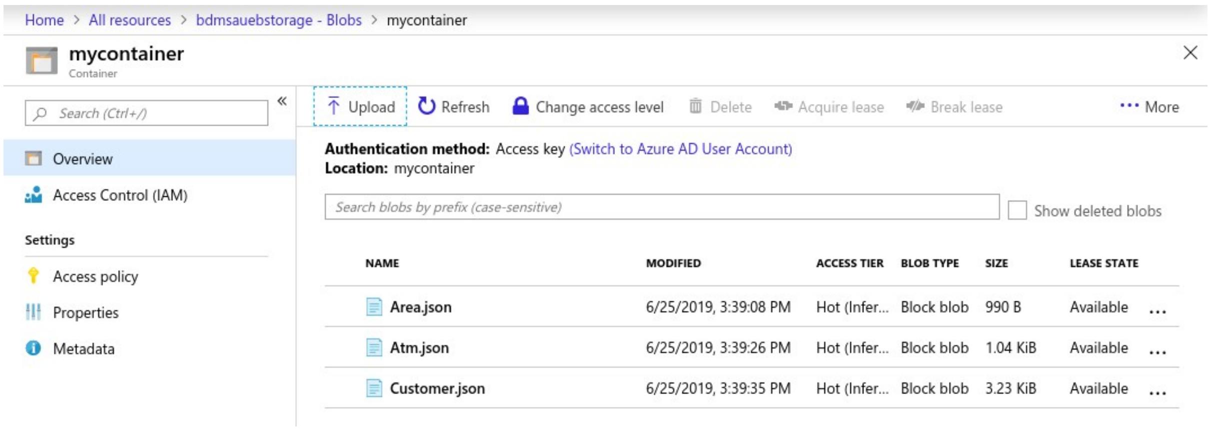


Subscriptions: Azure for Students

3 items ☐ Show hidden types

NAME	TYPE	RESOURCE GROUP	LOCATION	SUBSCRIPTION
bdmsAueb	Event Hubs Namespace	bdms	North Europe	Azure for Students
bdmsauebstorage	Storage account	bdms	North Europe	Azure for Students
streamdemo	Stream Analytics job	bdms	North Europe	Azure for Students

Figure 3: All Resources (*Step 2, 6, 8).*


Home > All resources > bdmsauebstorage - Blobs > mycontainer

mycontainer
Container

Upload | Refresh | Change access level | Delete | Acquire lease | Break lease | More

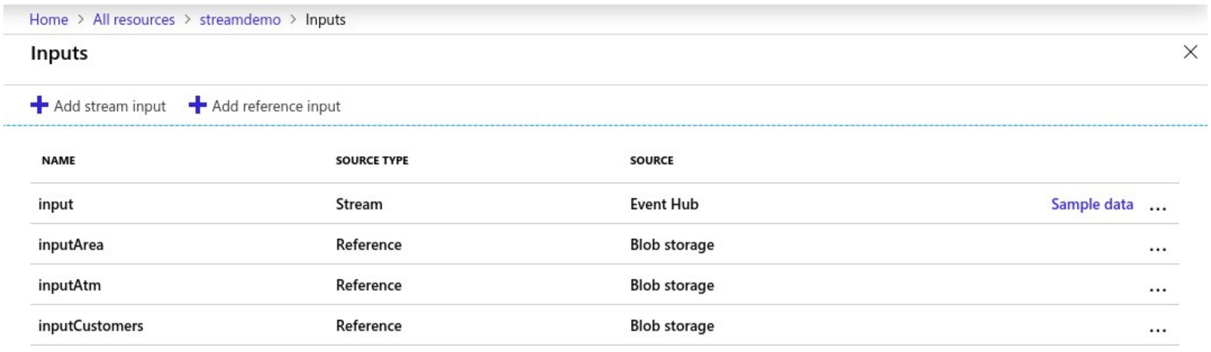
Overview
Access Control (IAM)
Settings
Access policy
Properties
Metadata

Authentication method: Access key (Switch to Azure AD User Account)
Location: mycontainer

Search blobs by prefix (case-sensitive) — Show deleted blobs

NAME	MODIFIED	ACCESS TIER	BLOB TYPE	SIZE	LEASE STATE
Area.json	6/25/2019, 3:39:08 PM	Hot (Infer...	Block blob	990 B	Available
Atm.json	6/25/2019, 3:39:26 PM	Hot (Infer...	Block blob	1.04 KiB	Available
Customer.json	6/25/2019, 3:39:35 PM	Hot (Infer...	Block blob	3.23 KiB	Available

Figure 4: Storage Account (*Step 7*).

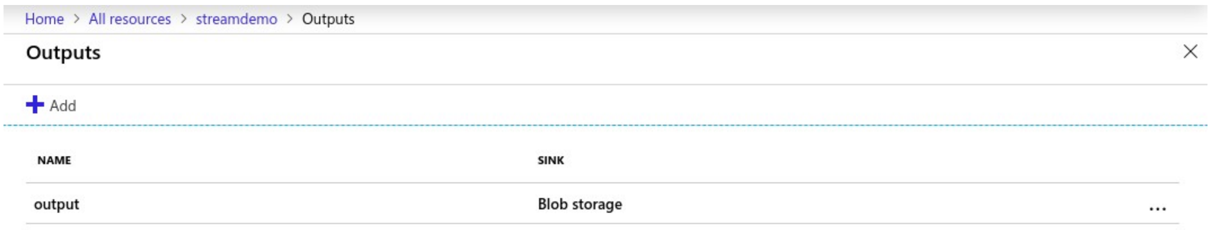

Home > All resources > streamdemo > Inputs

Inputs

Add stream input | Add reference input

NAME	SOURCE TYPE	SOURCE
input	Stream	Event Hub
inputArea	Reference	Blob storage
inputAtm	Reference	Blob storage
inputCustomers	Reference	Blob storage

Figure 5: Job Inputs (*Step 9*).


Home > All resources > streamdemo > Outputs

Outputs

Add

NAME	SINK
output	Blob storage

Figure 6: Job Output (*Step 10*).

4 Azure Stream Analytics: Queries

After specifying the input source of the streaming data and the output sink for the results, the following queries were expressed in a *SQL-like query language (T-SQL)*, in order to be executed on the Azure Platform:

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)

For each of the above queries we ran a stream analytics job and we present some indicative results in the report below. In some cases, we reduced the time limit of the windows, in order to precipitate the progress of the data transformation.

Query 1:

```sql
/*
    Q1: 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_1 (SlidingWindow(second, 10)):

```
[
  {
     "Total_Amount":21,
     "Event_Time":"2019-07-03T00:18:29.2580000Z"
  },
  {
     "Total_Amount":38,
     "Event_Time":"2019-07-03T00:18:39.2430000Z"
  },
  {
     "Total_Amount":17,
     "Event_Time":"2019-07-03T00:18:39.2580000Z"
  },
  {
     "Total_Amount":105,
     "Event_Time":"2019-07-03T00:18:40.2430000Z"
  },
  {
     "Total_Amount":88,
     "Event_Time":"2019-07-03T00:18:49.2430000Z"
  },
  {
     "Total_Amount":119,
     "Event_Time":"2019-07-03T00:18:49.2590000Z"
  }
]
```

Query 2:

```
/*
    Q2: 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_2 (TumblingWindow(minute, 1)):

```
[
  {
    "Total_Amount": 335,
    "Event_Time": "2019-06-29T17:36:00.0000000Z"
  },
  {
    "Total_Amount": 603,
    "Event_Time": "2019-06-29T17:37:00.0000000Z"
  },
  {
    "Total_Amount": 465,
    "Event_Time": "2019-06-29T17:38:00.0000000Z"
  }
]
```

Query 3:

```
/*
    Q3: 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_3 (HoppingWindow(second, 60, 30)):

```
[
  {
    "Total_Amount": 406,
    "Event_Time": "2019-06-29T17:48:00.0000000Z"
  },
  {
    "Total_Amount": 651,
    "Event_Time": "2019-06-29T17:48:30.0000000Z"
  },
  {
    "Total_Amount": 728,
    "Event_Time": "2019-06-29T17:49:00.0000000Z"
  }
]
```

Query 4:

```
/*
    Q4: 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_4 (SlidingWindow(minute, 1)):

```
[
  {
    "ATMCode":15
    "Total_Amount": 2519,
    "Event_Time": "2019-06-29T17:56:55.2870000Z"
  },
  {
    "ATMCode":18
    "Total_Amount": 1624,
    "Event_Time": "2019-06-29T17:56:55.2920000Z"
  },
  {
    "ATMCode":18
    "Total_Amount": 1660,
    "Event_Time": "2019-06-29T17:56:55.3020000Z"
  },
  {
    "ATMCode":19
    "Total_Amount": 2031,
    "Event_Time": "2019-06-29T17:56:55.3020000Z"
  }
]
```

Query 5:

```
/*
    Q5: 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_5 (TumblingWindow(minute, 1)):

```
[
  {
     "area_code":10,
     "Total_Amount":670,
     "Time":"2019-06-30T09:40:00.0000000Z"
  },
  {
     "area_code":14,
     "Total_Amount":43,
     "Time":"2019-06-30T09:40:00.0000000Z"
  },
  {
     "area_code":12,
     "Total_Amount":50,
     "Time":"2019-06-30T09:40:00.0000000Z"
  },
  {
     "area_code":2,
     "Total_Amount":1813,
     "Time":"2019-06-30T09:40:00.0000000Z"
  }
]
```

Query 6:

```sql
/*
    Q6: 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_6 (TumblingWindow(minute, 1)):

```json
[
  {
     "area_city":"Springfield",
     "gender":"Male",
     "Total_Amount":1066,
     "Time":"2019-06-30T20:12:00.0000000Z"
  },
  {
     "area_city":"Baltimore",
     "gender":"Male",
     "Total_Amount":384,
     "Time":"2019-06-30T20:12:00.0000000Z"
  }
]
```

Query 7:

```
/*
    Q7: 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_7 (SlidingWindow(hour, 1)):

```
[
  {
     "first_name":"Brenda",
     "last_name":"Carroll",
     "Card_Number":560222217915598000,
     "Transactions":2,
     "Time":"2019-06-30T12:49:59.7440000Z"
  },
  {
     "first_name":"Bruce",
     "last_name":"Morrison",
     "Card_Number":5602246755688900,
     "Transactions":2,
     "Time":"2019-06-30T12:50:02.6970000Z"
  }
]
```

Query 8:

```sql
/*
    Q8: 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_8 (SlidingWindow(hour, 1)):

```json
[
  {
     "Atm_Area_Code":5,
     "Customer_Area_Code":7,
     "COUNT":523,
     "Time":"2019-06-30T20:18:13.5610000Z"
  },
  {
     "Atm_Area_Code":2,
     "Customer_Area_Code":1,
     "COUNT":333,
     "Time":"2019-06-30T20:18:13.5760000Z"
  }
]
```

Bibliography

[1] D. Chatziantoniou and S. Safras, *Course Notes on Big Data Management Systems.*

[2] "Microsoft Azure: Stream Analytics Documentation." https://docs.microsoft.com/en-us/azure/stream-analytics/.