

Azure Internet of Things (IoT) Using Event Hubs and Stream Analytics

Overview

Azure Stream Analytics is a cloud-based service for ingesting high-velocity data emanating from devices, sensors, applications, Web sites, and other data sources and analyzing that data in real time (or near real time). It supports a SQL-like query language that works over dynamic data streams and makes analyzing constantly changing data no more difficult than performing queries on static data stored in traditional databases. With Azure Stream Analytics, you can set up jobs that analyze incoming data for anomalies or information of interest and record the results, present notifications on dashboards, or even fire off alerts to mobile devices. And all of it can be done at low cost and with a minimum of effort — frequently without writing a single line of code.

Scenarios for the application of real-time data analytics are legion and include fraud protection, identity-theft protection, optimizing the allocation of resources (think of an Uber-like transportation service that sends drivers to areas of increasing demand *before* that demand peaks out), click-stream analysis on Web sites, and countless others. Having the ability to process data *as it comes in* rather than waiting until after it has been aggregated offers a competitive advantage to businesses that are agile enough to make adjustments on the fly.

In this lab, you'll create an Azure Stream Analytics job and use it to analyze data streaming in from simulated Internet of Things (IoT) devices. And you'll see how utterly simple it is to monitor real-time data streams for information of significance to your research or business.

Objectives

In this hands-on lab, you will learn how to:

- Create an Azure event hub and use it as a Stream Analytics input
- Create a Stream Analytics job and test queries on sample data streams
- Run a Stream Analytics job and perform queries on live data streams
- Direct Stream Analytics output to Azure blobs

Prerequisites

The following is required to complete this hands-on lab:

- A Microsoft Azure subscription - [sign up for a free trial](#)
-

Exercises

This hands-on lab includes the following exercises:

- [Exercise 1: Create an event hub](#)
- [Exercise 2: Create a shared-access signature token](#)
- [Exercise 3: Send events to the event hub](#)
- [Exercise 4: Create a Stream Analytics job](#)
- [Exercise 5: Prepare queries and test with sample data](#)
- [Exercise 6: Analyze a live data stream](#)

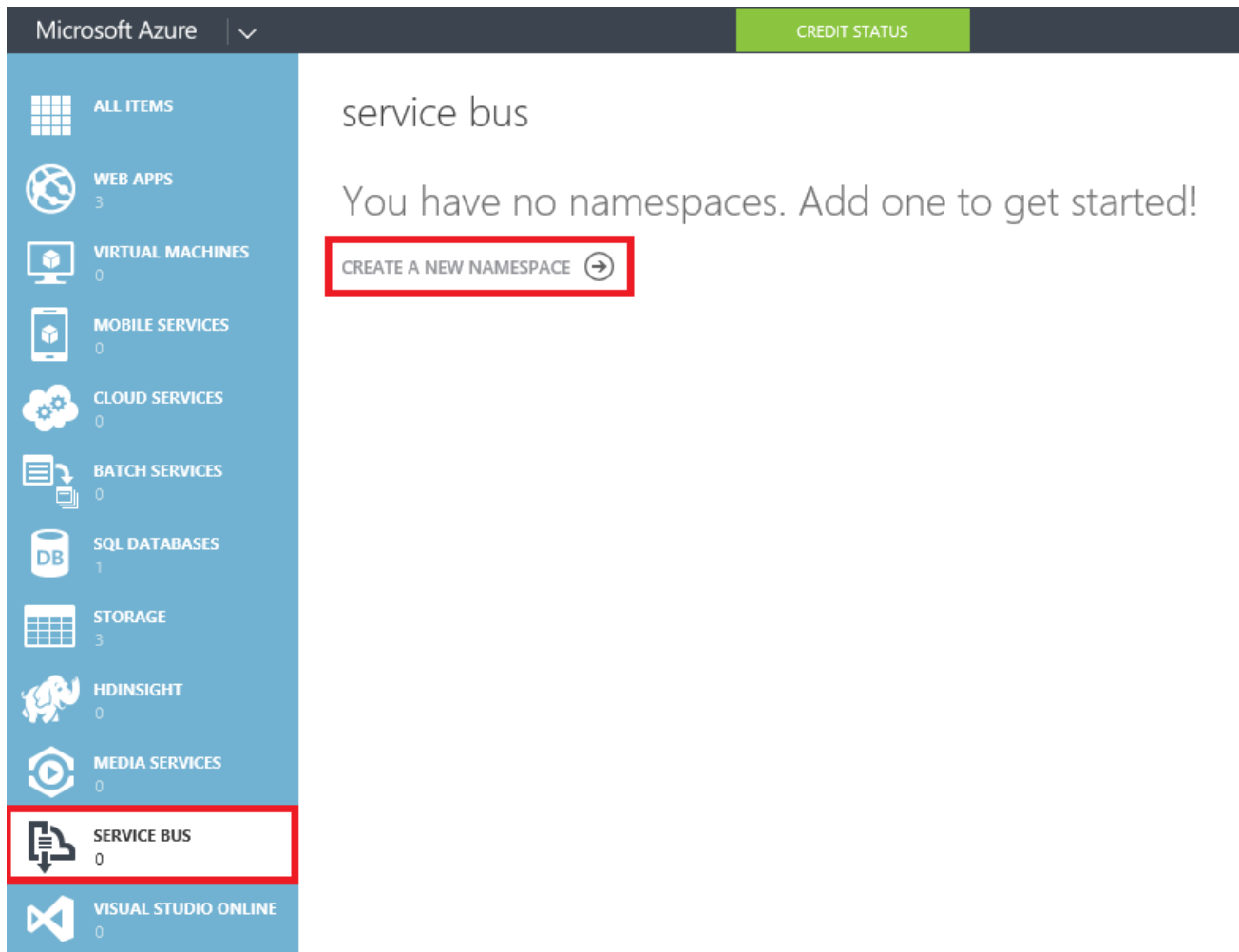
Estimated time to complete this lab: **90** minutes.

Exercise1: Create an event hub

Azure Stream Analytics supports two types of input: input from Azure blobs, and input from Azure event hubs. Of the two, the latter is typically more interesting because in the IoT world, data is easily transmitted to Azure event hubs through field gateways (for devices that are not IP-capable) or cloud gateways (for devices that *are* IP-capable), and a single Azure event hub can handle millions of events per second transmitted from devices spread throughout the world.

In this exercise, you'll create an Azure event hub to provide input to Azure Stream Analytics and configure it to so that it can be accessed safely and securely by IoT devices and gateways.

1. You can't (yet) create an event hub using the Azure Preview Portal, so you'll create it instead using the Classic Portal. Go to <https://manage.windowsazure.com> to open the Classic Portal, and click **Service Bus** in the ribbon on the left. Then click **CREATE A NEW NAMESPACE** to create a new service-bus namespace. (If you have already created one or more namespaces, click **+ NEW** in the lower-left corner of the page to create another one.)



Azure Service Bus

2. Type a namespace name into the **NAMESPACE NAME** box. The name must be unique within Azure, so you'll have to use something other than the name in the screen shot below. (A green check mark will appear in the box when the name you've entered is one that Azure will accept.) Optionally choose the region closest to you from the **REGION** drop-down. Then click the check mark in the lower-right corner of the dialog.

CREATE A NAMESPACE

Add a new namespace

NAMESPACE NAME

a4rlabs



.servicebus.windows.net

TYPE

MESSAGING

NOTIFICATION HUB

MESSAGING TIER

BASIC

STANDARD

PREMIUM

REGION

Central US



Creating a service-bus namespace

- Click the **+ NEW** button in the lower-left corner of the page. Then click **EVENT HUB**, followed by **QUICK CREATE**. Type "IoTEventHub" into the **EVENT HUB NAME** box (the name doesn't have to be unique within Azure). Optionally select the region closest to you, and make sure the namespace you created in the previous step is selected in the **NAMESPACE** box. Then click **CREATE A NEW EVENT HUB** in the lower-right corner.

The screenshot shows the Azure portal's 'NEW' page. On the left, the 'SERVICE BUS' category is selected, and 'EVENT HUB' is highlighted in the sub-menu. In the center, the 'QUICK CREATE' button is highlighted with a red box. On the right, the 'EVENT HUB NAME' field contains 'IoTEventHub' with a green checkmark. The 'REGION' dropdown is set to 'East US'. The 'NAMESPACE' dropdown is set to 'a4rlabs', with '.servicebus.windows.net' displayed below it. At the bottom right, the 'CREATE A NEW EVENT HUB' button is highlighted with a red box and a green checkmark.

Creating an event hub

- In the portal, click the event hub that you just created to display the event hub's dashboard.

Microsoft Azure | CREDIT STATUS

a4rlabs

ALL QUEUES TOPICS RELAYS **EVENT HUBS** SCALE CONFIGURE

NAME	STATUS
ioteventhub	✓ Active

IoTEventHub

- In the dashboard, click **CONFIGURE**.

Microsoft Azure | CREDIT STATUS

ioteventhub

DASHBOARD **CONFIGURE** CONSUMER GROUPS

✓ INCOMING MESSAGES ✓ INCOMING.THROUGHPUT ✓ INTERNAL SERVER ERRORS 6 MORE ▼

10:15	10:20	10:25	10:30	10:35	10:40	10:45	10:50

IoTEventHub dashboard

- In order to transmit events to the event hub from an application or device, you need to create a shared-access policy that includes Send permission. In the **shared access policies** section of the IoTEventHub configuration page, create a new policy named "SendPolicy" and check the **Send** box in the drop-down list under **PERMISSIONS**. Then click the **Save** button at the bottom of the page to save the new policy.

shared access policies

NAME	PERMISSIONS
<input type="text" value="SendPolicy"/>	<div>Send</div> <div><input type="checkbox"/> Manage</div> <div><input checked="" type="checkbox"/> Send</div> <div><input type="checkbox"/> Listen</div>
<input type="text" value="NEW POLICY NAME"/>	



Creating a policy with Send permission

7. In the **shared access key generator** section that appears underneath **shared access policies**, click the button at the right end of the **PRIMARY KEY** box to copy the key to the clipboard. Then temporarily save the key by pasting it into your favorite text editor. You'll need this key in the next exercise.

shared access policies

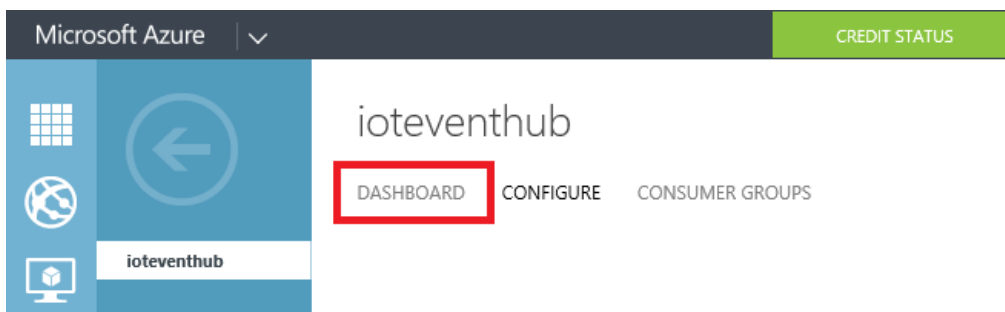
NAME	PERMISSIONS
<input type="text" value="SendPolicy"/>	<div>Send</div>
<input type="text" value="NEW POLICY NAME"/>	<div></div>

shared access key generator

POLICY NAME	<input type="text" value="SendPolicy"/>	
PRIMARY KEY	<div>V+A3j1c2qdKxWbe+3dPad4K1wNrx5RldnY5R1efA4mk=</div> <div></div>	<div>Regenerate</div>
SECONDARY KEY	<div>vd2SR80Fje2a7OGuOv/IsHILOgtq7Y2cTDqHeunxgvk=</div> <div></div>	<div>Regenerate</div>

Copying the primary key to the clipboard

8. Click **DASHBOARD** near the top of the page to return to the event hub's dashboard.



Returning to the dashboard

9. Under **quick glance** on the right side of the page, find **EVENT HUB URL** and copy the URL into your text editor. You'll need this URL, too, in the next exercise.

quick glance

 [View Connection String](#)

 [Download the sample solution for a Event Hub](#)

STATUS

Active

EVENT HUB URL

<https://a4rlabs.servicebus.windows.net/ioteventhub>

MANAGEMENT SERVICES

[Operation Logs](#)

Getting the event hub's URL

You have created an event hub that can ingest events and be used as the source of input to a Stream Analytics job. You have also created a policy that allows holders of that policy to send events to the event hub. The next step is to generate a security token that can be used to authenticate calls to the event hub.

Exercise 2: Create a shared-access signature token

Applications, devices, or gateways can send events to event hubs using the [Azure Event Hubs REST API](#). Each request transmitted via this API must include a valid [shared-access signature \(SAS\)](#) token in the Authorization header. SAS tokens are generated from the event hub's URL and the primary key associated with the policy used to communicate with the event hub — in this case, the policy named "SendPolicy" that you created in the previous exercise.

In this exercise, you will generate a shared-access signature token for the event hub created in [Exercise 1](#) and copy it, along with the event hub URL, into a Node.js application that will be used to send events to the event hub in Exercise 3.

1. Neither the Classic Portal nor the Preview Portal currently provides an interface for generating SAS tokens. Therefore, you will generate a token using a Node.js utility named `sas.js` provided with this lab. Begin by opening a terminal window.
2. Verify that Node.js is installed on your computer by executing the following command:

```
node -v
```

If Node.js is installed, you'll see the Node.js version number. If you don't see a version number, or if the **node** command didn't run at all, then you need to install Node.js. You'll find detailed instructions for installing it in Exercise 2 of the lab entitled "Azure Storage and the Azure CLI." **If you don't already have Node.js installed, install it now.**

You already have Node.js installed if you completed the **Azure Storage and the Azure CLI** lab because the Azure CLI requires Node.js.

3. At the command prompt, navigate to this lab's "resources" directory. Then execute the following command:

```
node sas.js
```

It is very important that you run this command from the lab's "resources" directory, because that directory contains subdirectories that contain components required by `sas.js`.

4. When prompted, enter the event-hub URL you saved in Exercise 1, Step 9. Then press Enter.
5. When prompted, enter the name ("SendPolicy") of the policy you created for the Azure event hub in Exercise 1, Step 6. Then press Enter.
6. When prompted, enter the key that you saved in Exercise 1, Step 7. Then press Enter.
7. The SAS token, which is highlighted with the red box below, will be output to the terminal window. Copy it to the clipboard.

```
Command Prompt
D:\Labs\Azure\Stream Analytics\resources>node sas.js
Azure Event Hub URL: https://a4rlabs.servicebus.windows.net/ioteventhub
Policy name: SendPolicy
Policy key: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX=
SharedAccessSignature sr=https%3A%2F%2Fa4rlabs.servicebus.windows.net%2Fioteventhub&sig=CzaMvUeDC1mmQazuJ6IgZY1LqiOx451E0MNV4i0yewI%3D&se=1443477127924&skn=SendPolicy
D:\Labs\Azure\Stream Analytics\resources>
```

Generating a SAS token

- Find the file named eventgen.js in the "resources" directory of this lab and open it in your favorite text editor. Then find the section at the top of the file labeled "KEY VARS:"

```
////////////////// KEY VARS ////////////////////
var sas = "Token";
var uri = "URL";
//////////////////
```

- Replace *Token* with the SAS token you copied to the clipboard in Step 7. **Important:** The SAS token must **not include line breaks**. It needs to appear on this line as one contiguous string, and it must begin and end with quotation marks. In addition, the line must end with a semicolon.
- Replace *URL* with the event-hub URL you saved in exercise 1, Step 9.
- Save the modified eventgen.js file. The modified "KEY VARS" section should look something like this:

```
////////////////// KEY VARS ////////////////////
var sas = "SharedAccessSignature sr=https%3A%2F%2Fa4rlabs.servicebus.windows.net%2Fioteventhub&sig=CzaMvUeDC1mmQazuJ6IgZY1LqiOx451E0MNV4i0yewI%3D&se=1443477127924&skn=SendPolicy";
var uri = "https://a4rlabs.servicebus.windows.net/ioteventhub";
//////////////////
```

Now that you've modified eventgen.js with information specific to your event hub, it's time to generate some events.

Exercise 3: Send events to the event hub

In this exercise, you will send events to the event hub you created in [Exercise 1](#). To do that, you'll use Node.js to run eventgen.js, which in turn transmits secure requests to the event hub using the [Azure Event Hubs REST API](#). eventgen.js generates simulated events representing ATM withdrawals. Each event contains relevant information such as the card number used for the withdrawal, the time and amount of the withdrawal, and a unique identifier for the ATM machine used.

- At the command prompt, navigate to the "resources" directory of this lab if you aren't there already.
- Now execute the following command:

```
node eventgen.js
```

You should see output similar to the following. Each line represents one event sent to the event hub, and events will probably roll by at a rate of about 2 to 3 per second. (Rates will vary depending on your connection speed.) **Confirm that each request returns the HTTP status code 201.** This indicates that the event hub received and accepted the request.

```
[1000] Event sent (status code: 201)
```

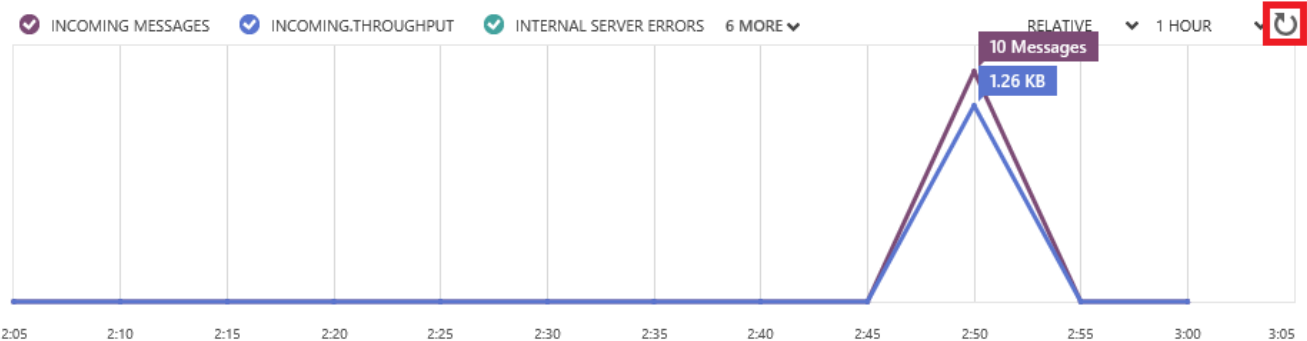
```
[1001] Event sent (status code: 201)
[1002] Event sent (status code: 201)
[1003] Event sent (status code: 201)
[1004] Event sent (status code: 201)
[1005] Event sent (status code: 201)
[1006] Event sent (status code: 201)
[1007] Event sent (status code: 201)
[1008] Event sent (status code: 201)
[1009] Event sent (status code: 201)
```

It is very important that you run this command in the lab's "resources" directory, because the "resources" directory contains subdirectories that contain components required by eventgen.js.

- After 10 to 20 events have been sent, press Ctrl+C (or whatever key combination your operating system supports for terminating an application running in a terminal window) to stop the flow of events. **Leave the terminal window open so you can return to it later.**
- Return to the [Classic Portal](#) and open the dashboard for the event hub you created in [Exercise 1](#). Wait a few minutes, and then click the **Refresh Metrics** button in the upper-right corner of the chart at the top of the page (the button highlighted in red below). Confirm that the chart shows several messages have been received.

ioteventhub

DASHBOARD CONFIGURE CONSUMER GROUPS



Messages received by the event hub

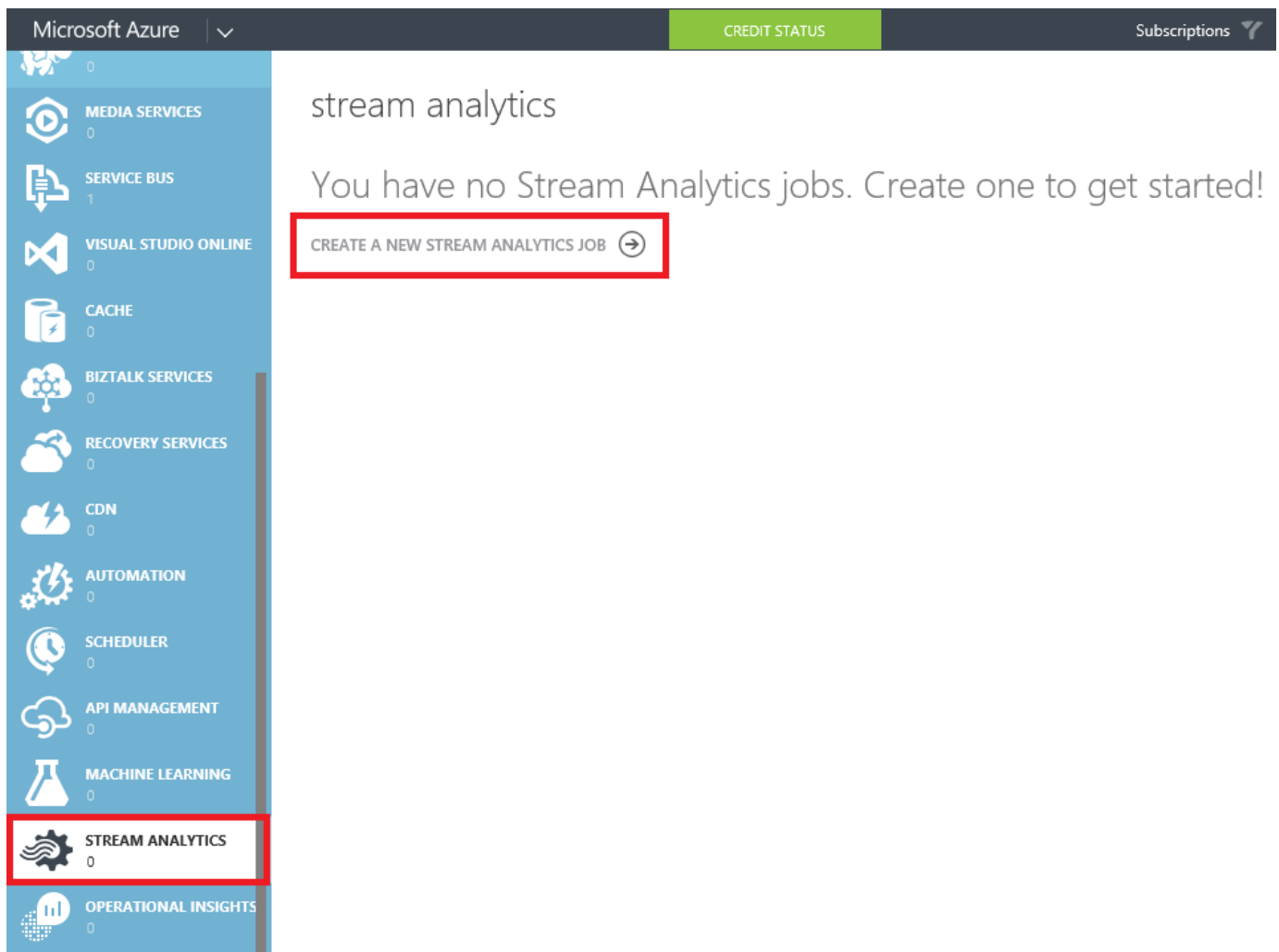
The dashboard doesn't show events in real time. An event typically doesn't appear in the chart until 5 to 10 minutes after it is received. While you're waiting, take a few moments to peruse the code in eventgen.js. In particular, notice the Authorization header sent in each request, and the URL that the request is directed to.

If you'd rather not wait for the events to appear in the dashboard, feel free to move on to the next exercise. But if you are unable to generate sample data in the Stream Analytics job in the next exercise, return to the event-hub dashboard and verify that the event hub received the events.

Exercise 4: Create a Stream Analytics job

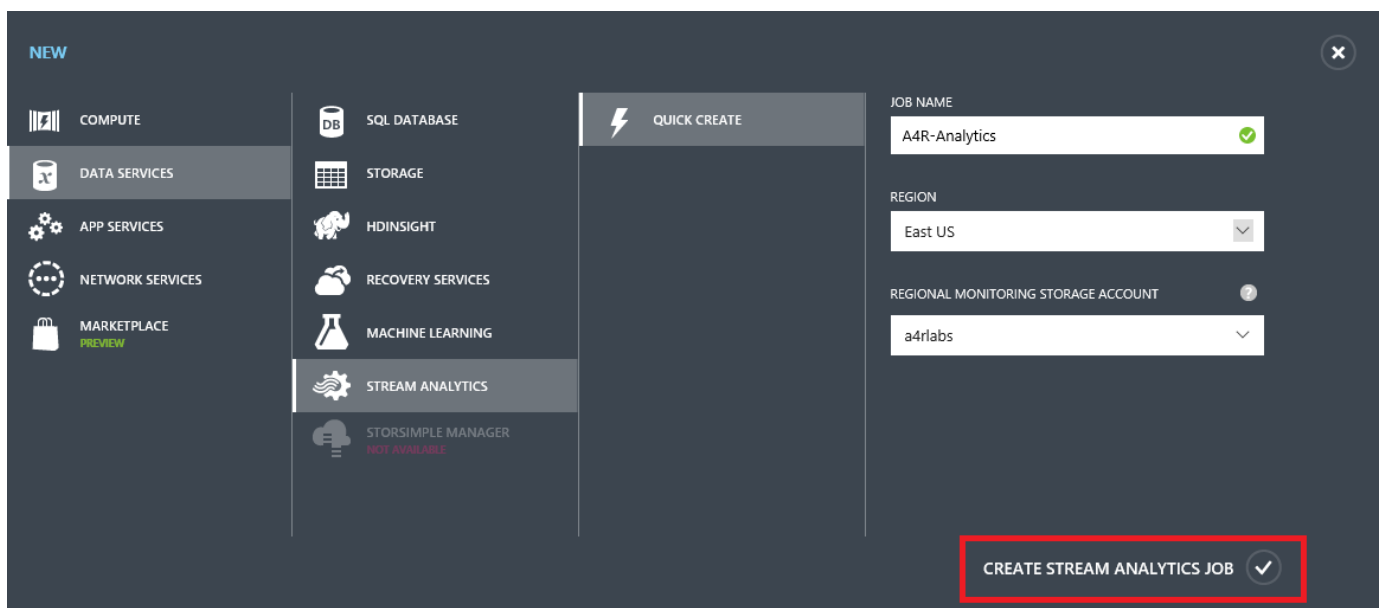
You now have software that sends events to an Azure event hub, and an event hub that ingests the data. In this exercise, you'll use the Microsoft Azure [Classic Portal](#) to create a Stream Analytics job and connect it to the event hub. You'll also capture the raw data being presented to Stream Analytics by the event hub and examine its structure.

- Open the [Classic Portal](#) in your browser if it isn't already open. Click **STREAM ANALYTICS** in the ribbon on the left, and then click **CREATE A NEW STREAM ANALYTICS JOB**.



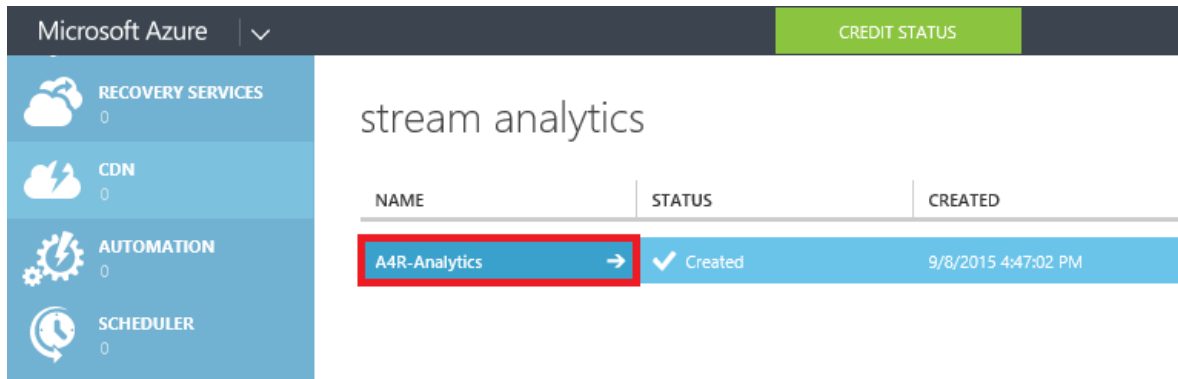
Azure Stream Analytics

2. Type "A4R-Analytics" into the **JOB NAME** box. Optionally select the region nearest you in the **REGION** box. Under **REGIONAL MONITORING STORAGE ACCOUNT**, either select an existing storage account or select **Create new storage account** from the drop-down list and enter a name for the new storage account. (If you choose to create a new storage account, recall that storage-account names can be 3 to 24 characters in length, can only contain numbers and lowercase letters, and must be unique within Azure. A green check mark next to the name indicates that it meets all these criteria.) When you're done, click **CREATE STREAM ANALYTICS JOB** in the lower-right corner.



Creating a Stream Analytics job

- After a few moments, the Stream Analytics job you created will appear in the portal. Click it to go to the page devoted to the job.



The screenshot shows the Microsoft Azure portal interface. On the left is a navigation pane with icons for Recovery Services, CDN, Automation, and Scheduler. The main area displays the 'stream analytics' section with a table of jobs. The table has columns for NAME, STATUS, and CREATED. One job, 'A4R-Analytics', is listed with a status of 'Created' and a creation time of '9/8/2015 4:47:02 PM'. A red box highlights the 'A4R-Analytics' name and the right-pointing arrow icon next to it.

NAME	STATUS	CREATED
A4R-Analytics	✓ Created	9/8/2015 4:47:02 PM

The new Stream Analytics job

- Click **INPUTS** near the top of the page.

a4r-analytics

[DASHBOARD](#) [MONITOR](#) [INPUTS](#) [QUERY](#) [OUTPUTS](#) [SCALE](#) [CONFIGURE](#)



Your Stream Analytics job has been created!
Here are a few options to help you get started.

☐ Skip Quick Start the next time I visit

A4R-Analytics page

- Click **ADD AN INPUT**.

a4r-analytics

[DASHBOARD](#) [MONITOR](#) [INPUTS](#) [QUERY](#) [OUTPUTS](#) [SCALE](#) [CONFIGURE](#)

You have no inputs. Add one to get started!

[ADD AN INPUT](#) ➔

Adding an input

- Make sure **Data stream** is selected, and then click the right-arrow in the lower-right corner of the dialog.

ADD AN INPUT

Add an input to your job

Each job requires at least one data stream input. Adding reference data input is optional.

☒ Data stream

A continuous sequence of data or events to be consumed and transformed by a Stream Analytics job.

☐ Reference data

Auxiliary data used for correlation and lookups. Reference data is static or slow-changing.



Specifying an input type

7. Make sure **Event Hub** is selected, and then click the right-arrow.

ADD A DATA STREAM

×

Add a data stream to your job

☒ Event Hub ?

☐ Blob storage ?

☐ IoT Hub **PREVIEW** ?

Missing an input source? [Let us know!](#) (Powered by [UserVoice](#) - [Privacy Policy](#))

1

←→3

Specifying a data-stream type

IoT hubs are a relatively recent addition to Azure; their primary purpose is to enable two-way communications between IoT devices. You chose **Event Hub** because you're connecting simulated IoT devices to a Stream Analytics job, not to other devices.

8. Enter "Withdrawals" as a friendly alias for the input in the **INPUT ALIAS** box. In the **CHOOSE A NAMESPACE** and **CHOOSE AN EVENTHUB** boxes, select the namespace and event hub that you created in [Exercise 1](#). Leave **EVENT HUB POLICY NAME** set to **RootManageSharedAccessKey** (that's a default policy that's created automatically when you create an event hub; it grants permission to manage the event hub, send events, and receive events) and **CHOOSE A CONSUMER GROUP** set to **\$Default**. Then click the right-arrow in the lower-right corner.

ADD A SERVICE BUS EVENT HUB

Event Hub settings

INPUT ALIAS

Withdrawals

SUBSCRIPTION

Use Event Hub from Current Subscription

CHOOSE A NAMESPACE ?

a4rlabs (East US)

CHOOSE AN EVENTHUB ?

ioteventhub

EVENT HUB POLICY NAME ?

RootManageSharedAccessKey

CHOOSE A CONSUMER GROUP ?

\$Default

1 2

4

Specifying event-hub settings

9. Make sure **JSON** is selected under **EVENT SERIALIZATION FORMAT** (the Node.js application that sends events to the event hub indeed sends JSON data), and **UTF8** is selected under **ENCODING**. Then click the check mark in the lower-right corner to finish adding the input.

ADD A SERVICE BUS EVENT HUB

Serialization settings

EVENT SERIALIZATION FORMAT ?

JSON

ENCODING ?

UTF8

1 2 3

⏪ ✓

Specifying a serialization format

10. After a few moments, the new input — "Withdrawals" — appears in the list of inputs for the Stream Analytics job. Go back to the terminal window you left open at the end of the previous exercise and run eventgen.js again by executing the following command:

```
node eventgen.js
```

11. Allow eventgen.js to run for a minute or two. Then press Ctrl+C (or the equivalent) to stop it, and return to the portal open in your browser.
12. Click the **SAMPLE DATA** button at the bottom of the page to sample data from the event hub.

Microsoft Azure | CREDIT STATUS | Subscriptions

a4r-analytics

DASHBOARD MONITOR INPUTS QUERY OUTPUTS SCALE CONFIGURE

NAME	SOURCE TYPE	TYPE	DIAGNOSIS
Withdrawals	Data stream	Event Hub	✓ OK

NEW START ADD INPUT TEST CONNECTION DELETE SAMPLE DATA

Sampling input data

- Click the check mark in the lower-right corner of the ensuing dialog to sample any data transmitted to the event hub in the last 10 minutes. (This is why you ran eventgen.js again: to make sure there is data to sample, even if more than 10 minutes have elapsed since you completed Exercise 3.)

Sample Data

START TIME ?

8/9/2015 09 : 56 : 44 AM

LOCAL TIME (UTC-04:00)

DURATION (HH:MM:SS) ?

00 : 10 : 00



Specifying start time and duration

- Wait until the sample is completed. Then click the button in the lower-right corner of the page that indicates the operation has completed.

Microsoft Azure | CREDIT STATUS | Subscriptions

a4r-analytics

DASHBOARD MONITOR INPUTS QUERY OUTPUTS SCALE CONFIGURE

NAME	SOURCE TYPE	TYPE	DIAGNOSIS
Withdrawals	Data stream	Event Hub	✓ OK

+ NEW | START | ADD INPUT | TEST CONNECTION | DELETE | SAMPLE DATA | 1 ?

Data sampling completed

- When a ribbon appears that says "Successfully sampled data from Withdrawals," click the **Details** button on the right.

Microsoft Azure | CREDIT STATUS | Subscriptions

a4r-analytics

DASHBOARD MONITOR INPUTS QUERY OUTPUTS SCALE CONFIGURE

NAME	SOURCE TYPE	TYPE	DIAGNOSIS
Withdrawals	Data stream	Event Hub	✓ OK

✓ Successfully sampled data from "Withdrawals". | DETAILS | OK | ✓

+ NEW | START | ADD INPUT | TEST CONNECTION | DELETE | SAMPLE DATA | 1 ?

Data sampling succeeded

- Click **Click here** to download the data sampled from the event hub. Save the JSON file that is downloaded to a location where you can easily find it. Then click **OK** to dismiss the ribbon.

Microsoft Azure | CREDIT STATUS | Subscriptions

a4r-analytics

DASHBOARD MONITOR INPUTS **QUERY** OUTPUTS SCALE CONFIGURE

NAME	SOURCE TYPE	TYPE	DIAGNOSIS
Withdrawals	Data stream	Event Hub	✓ OK

← Back to progress operations

[Click here](#) to download the sample file that was generated from 'Withdrawals'. Please note that this link will expire in a few minutes.

DETAILS ⓘ OK ✓

+ NEW

START ADD INPUT TEST CONNECTION DELETE SAMPLE DATA

Downloading sample data

- Open the JSON file you downloaded in your favorite text editor and take a moment to examine its contents. How many rows (events) are represented in the sample data? What is the structure of each row — that is, what fields does each row contain?

You have connected a Stream Analytics job to an event hub and demonstrated that data is passed from one to the other. You have also examined the structure of that data. The next step is to do something with it — specifically, to bring the power of Azure Stream Analytics to bear on the data.

Exercise 5: Prepare queries and test with sample data

Now that your job is set up, there's much more you can do with Stream Analytics than simply view the raw data presented to it. The whole point of Stream Analytics is being able to perform queries on the data, even though the data is dynamic rather than static. In this exercise, you'll use the [Stream Analytics Query Language](#) to query a sample data set for potentially fraudulent ATM transactions. It is always a good idea to test your queries against sample data before deploying them against live data streams, because with sample data, you can verify that a known set of inputs produces the expected set of outputs.

To flag potentially fraudulent withdrawals from ATMs, you will query for transactions performed with the same ATM card at different ATM machines within a specified time window (60 seconds). In real life, you would probably use a larger time window and perhaps even factor in the distance between ATM machines. However, a narrower time window is useful in a lab environment because it allows you to perform meaningful experiments in minutes rather than hours.

- Begin by returning to the Stream Analytics job in the portal and clicking **QUERY** at the top of the page.

Microsoft Azure | CREDIT STATUS

a4r-analytics

DASHBOARD MONITOR INPUTS **QUERY** OUTPUTS SCALE CONFIGURE

NAME	SOURCE TYPE	TYPE
Withdrawals	Data stream	Event Hub

Navigating to the Query page

2. Enter the following query into the **query** field, and then click the **Test** button.

```
SELECT * FROM Withdrawals
```

Where did the name "Withdrawals" come from? That's the alias you assigned to the event-hub input in the previous exercise. If you named it differently, you'll need to replace "Withdrawals" with the alias name you used.

The screenshot shows the Microsoft Azure a4r-analytics web interface. The top navigation bar includes "Microsoft Azure", "CREDIT STATUS", "Subscriptions", and a user profile icon. The left sidebar contains various icons for navigation, with "A4R-Analytics" selected. The main content area is titled "a4r-analytics" and includes tabs for "DASHBOARD", "MONITOR", "INPUTS", "QUERY", "OUTPUTS", "SCALE", and "CONFIGURE". Below the tabs, there is a help link: "Need help with your query? Check out some of the most common Stream Analytics query patterns [here](#)." The "query" field is active, showing the text "1 SELECT * FROM Withdrawals". Below the query field, there is a link: "Missing some language constructs? [Let us know!](#) (Powered by UserVoice - [Privacy Policy](#))". At the bottom of the query field, there are two buttons: "Test" (highlighted with a red box) and "Rerun". The bottom status bar includes a "NEW" button, "START", "SAVE", "DISCARD", and a "1" with a question mark icon.

Testing a query

3. In the ensuing dialog, click **BROWSE FOR FILE**. Select the file named Withdrawals.json provided in the "resources" directory of this lab. Then OK the selection by clicking the check mark in the dialog's lower-right corner.

The reason you're using a file provided for you (rather than the one you captured in the previous exercise) is to make sure everyone who is doing this exercise gets the same results. eventgen.js uses JavaScript's `Math.random()` function to randomize results, and `Math.random()` does not produce repeatable sequences of pseudo-random numbers.



Test 'A4R-Analytics'

Test Data

Input

Sample File

withdrawals



BROWSE FOR FILE...



Loading test data

4. Scroll down the page and confirm that you see the output pictured below. The test data contains 607 rows. Each row has fields named TRANSACTIONID, TRANSACTIONTIME, DEVICEID, CARDNUMBER, and AMOUNT. DEVICEID is the ID of the ATM machine at which the transaction took place. AMOUNT is the amount of cash withdrawn from the ATM.

Microsoft Azure

CREDIT STATUS

Subscriptions

A4R-Analytics

Summary

Processed Events From:

Generated The Following:

OutputStep

TRANSACTIONID	TRANSACTIONTIME	DEVICEID	CARDNUMBER	AMOUNT
1446	Thu, 10 Sep 2015 14:38:46 GMT	42173	757296432	340
1453	Thu, 10 Sep 2015 14:38:48 GMT	33583	210383513	320
1454	Thu, 10 Sep 2015 14:38:48 GMT	38178	924020807	60
1455	Thu, 10 Sep 2015 14:38:48 GMT	32275	164373956	400
1450	Thu, 10 Sep 2015 14:38:47 GMT	24826	130090691	160
1456	Thu, 10 Sep 2015 14:38:49 GMT	17781	592188538	260
1452	Thu, 10 Sep 2015 14:38:48 GMT	99897	504146179	20
1459	Thu, 10 Sep 2015 14:38:49 GMT	46038	762131204	160
1460	Thu, 10 Sep 2015 14:38:50 GMT	73486	873140686	140
1462	Thu, 10 Sep 2015 14:38:50 GMT	72775	977315986	400
1457	Thu, 10 Sep 2015 14:38:49 GMT	58243	576357605	160
1458	Thu, 10 Sep 2015 14:38:49 GMT	42146	566111934	320
1463	Thu, 10 Sep 2015 14:38:50 GMT	94574	175799682	100

Missing some language constructs? [Let us know!](#) (Powered by UserVoice - Privacy Policy)

Test Rerun

Summary

Processed Events From:

- withdrawals containing 607 events.

Generated The Following:

- outputStep with 607 rows.

OutputStep

TRANSACTIONID	TRANSACTIONTIME	DEVICEID	CARDNUMBER	AMOUNT
1446	Thu, 10 Sep 2015 14:38:46 GMT	42173	757296432	340
1453	Thu, 10 Sep 2015 14:38:48 GMT	33583	210383513	320
1454	Thu, 10 Sep 2015 14:38:48 GMT	38178	924020807	60
1455	Thu, 10 Sep 2015 14:38:48 GMT	32275	164373956	400
1450	Thu, 10 Sep 2015 14:38:47 GMT	24826	130090691	160
1456	Thu, 10 Sep 2015 14:38:49 GMT	17781	592188538	260
1452	Thu, 10 Sep 2015 14:38:48 GMT	99897	504146179	20
1459	Thu, 10 Sep 2015 14:38:49 GMT	46038	762131204	160
1460	Thu, 10 Sep 2015 14:38:50 GMT	73486	873140686	140
1462	Thu, 10 Sep 2015 14:38:50 GMT	72775	977315986	400
1457	Thu, 10 Sep 2015 14:38:49 GMT	58243	576357605	160
1458	Thu, 10 Sep 2015 14:38:49 GMT	42146	566111934	320
1463	Thu, 10 Sep 2015 14:38:50 GMT	94574	175799682	100

+ NEW

START SAVE DISCARD ?

*Output from SELECT **

5. Suppose you only wanted to view transactions for amounts between 200 and 300, inclusive. Furthermore, suppose you wanted to clean up the output by assigning your own column names and excluding the TRANSACTIONID column. Enter the following query and click the **Rerun** button to test it. (**Rerun** executes the query against the test data already loaded. If you wanted to load a different test file, you would click the **Test** button again.)

```
SELECT TransactionTime as [Time of Transaction],
       DeviceID as [ATM],
       CardNumber as [Card Number],
       Amount as [Amount]
FROM Withdrawals
WHERE Amount >= 200 and Amount <= 300
```

6. Scroll down and confirm that the query generated the following output:

Microsoft Azure

CREDIT STATUS

Subscriptions

A4R-Analytics

Summary

Processed Events From:

Generated The Following:

OutputStep

Test

Rerun

TIME OF TRANSACTION

ATM

CARD NUMBER

AMOUNT

Thu, 10 Sep 2015 14:39:10 GMT

101126

178545727

240

Thu, 10 Sep 2015 14:39:10 GMT

60864

294189682

300

Thu, 10 Sep 2015 14:39:12 GMT

17971

777915980

220

Thu, 10 Sep 2015 14:39:11 GMT

80329

822163942

220

Thu, 10 Sep 2015 14:39:12 GMT

33309

153202956

220

Thu, 10 Sep 2015 14:39:14 GMT

35076

968595451

240

Thu, 10 Sep 2015 14:39:15 GMT

59792

889543099

280

Thu, 10 Sep 2015 14:39:15 GMT

38333

481371515

280

Thu, 10 Sep 2015 14:39:18 GMT

32481

733728642

200

Thu, 10 Sep 2015 14:39:19 GMT

62608

801493791

240

Thu, 10 Sep 2015 14:39:20 GMT

28229

529827286

260

Thu, 10 Sep 2015 14:39:20 GMT

19550

296724176

300

Thu, 10 Sep 2015 14:39:21 GMT

60855

697063943

260

+ NEW

START

SAVE

DISCARD

?

7. Now it's time to query the test data for potentially fraudulent transactions — transactions involving the same ATM card but different ATM machines that take place within 60 seconds of each other. *This is the query you will use in the next exercise against a live data stream.*

```
SELECT W1.CardNumber as [Card Number],
       W1.DeviceID as [ATM 1], W2.DeviceID as [ATM 2],
       W1.TransactionTime as [Time 1], W2.TransactionTime as [Time 2]
FROM Withdrawals W1 TIMESTAMP BY TransactionTime
JOIN Withdrawals W2 TIMESTAMP BY TransactionTime
ON W1.CardNumber = W2.CardNumber
AND DATEDIFF(ss, W1, W2) BETWEEN 0 and 60
WHERE W1.DeviceID != W2.DeviceID
```

8. This time the output should contain just three rows, each representing two transactions performed with one ATM card at two different locations within 60 seconds of each other:

CARD NUMBER	ATM 1	ATM 2	TIME 1	TIME 2
770522925	18842	82592	2015-09-10T14:37:16.000Z	2015-09-10T14:37:30.000Z
673004400	15981	32565	2015-09-10T14:38:24.000Z	2015-09-10T14:38:25.000Z
448603354	89307	98866	2015-09-10T14:38:56.000Z	2015-09-10T14:39:26.000Z

Potentially fraudulent transactions

- Click the **SAVE** button at the bottom of the page to save the query. Then click **YES** when asked to confirm.

The screenshot shows the Microsoft Azure Stream Analytics portal for a job named 'a4r-analytics'. The left sidebar contains navigation icons for various services. The main area displays a SQL query in a text editor. Below the query, there are buttons for 'Test' and 'Rerun'. At the bottom of the portal, there is a row of action buttons: '+ NEW', 'START', 'SAVE' (highlighted with a red box), and 'DISCARD'. The 'SAVE' button is a disk icon. The query text is as follows:

```

1 SELECT W1.CardNumber as [Card Number],
2       W1.DeviceID as [ATM 1], W2.DeviceID as [ATM 2],
3       W1.TransactionTime as [Time 1], W2.TransactionTime as [Time 2]
4 FROM Withdrawals W1 TIMESTAMP BY TransactionTime
5 JOIN Withdrawals W2 TIMESTAMP BY TransactionTime
6 ON W1.CardNumber = W2.CardNumber
7 AND DATEDIFF(ss, W1, W2) BETWEEN 0 and 60
8 WHERE W1.DeviceID != W2.DeviceID

```

Below the query, there is a link: "Missing some language constructs? Let us know! (Powered by UserVoice - Privacy Policy)".

Saving the query

With the query now formulated, tested against a set of sample data, and saved, it's time to deploy it against a live data stream to produce a running record of potentially fraudulent transactions.

Exercise 6: Analyze a live data stream

Being able to run your queries and see the results in the Azure Portal is great for testing, but when a query is deployed against a live data stream, you will most likely want to capture the results in a persistent data store. Azure Stream Analytics supports a variety of output types, including blobs, Azure SQL databases, and even event hubs. Imagine a scenario in which a Stream Analytics job receives data from an event hub, transforms it, and sends the results to another event hub, which itself serves as the input to another Stream Analytics job. Jobs can be chained this way to create rich analytic paths. Another reason for using event hubs for output is that software can subscribe to events from event hubs, enabling developers to build custom applications that show Stream Analytics output in near real time.

In this exercise, you'll configure the Stream Analytics job to store output in storage blobs. Then you'll run the job against a live data stream and check the results by inspecting the blob that was generated.

- Return to the Stream Analytics job in your browser and click **OUTPUTS**.

Microsoft Azure | CREDIT STATUS | Subscriptions

a4r-analytics

DASHBOARD MONITOR INPUTS QUERY **OUTPUTS** SCALE CONFIGURE

Need help with your query? Check out some of the most common Stream Analytics query patterns [here](#).

query

```
1 SELECT W1.CardNumber as [Card Number],
2     W1.DeviceID as [ATM 1], W2.DeviceID as [ATM 2],
3     W1.TransactionTime as [Time 1], W2.TransactionTime as [Time 2]
4 FROM Withdrawals W1 TIMESTAMP BY TransactionTime
5 JOIN Withdrawals W2 TIMESTAMP BY TransactionTime
6 ON W1.CardNumber = W2.CardNumber
7 AND DATEDIFF(ss, W1, W2) BETWEEN 0 and 60
8 WHERE W1.DeviceID != W2.DeviceID
```

Navigating to the Outputs page

2. Click **ADD AN OUTPUT** to add an output to the job.

Microsoft Azure | CREDIT STATUS | Subscriptions

a4r-analytics

DASHBOARD MONITOR INPUTS QUERY **OUTPUTS** SCALE CONFIGURE

You have no output. Add one to get started!

ADD AN OUTPUT →

Adding an output

3. Select **Blob storage** as the output type. Then click the right-arrow in the lower-right corner of the dialog.

ADD AN OUTPUT

Add an output to your job

- ☐ SQL Database ?
- ☒ Blob storage ?
- ☐ Event Hub ?
- ☐ Power BI **PREVIEW** ?
- ☐ Table storage ?
- ☐ Service Bus Queue ?
- ☐ Service Bus Topic ?
- ☐ DocumentDB ?

Missing an output sink? [Let us know!](#) (Powered by [UserVoice](#) - [Privacy Policy](#))



2

Specifying the output type

4. Type "Flagged-Withdrawals" into the **OUTPUT ALIAS** box. Select the storage account you want to use for the output blobs (feel free to create a new account if you prefer). Make sure **Create a new container** is selected to create a new blob container to hold the output, and type "a4r-analytics" into the **CONTAINER** box. Type "withdrawals/{date}/{time}" into the **PATH PREFIX PATTERN** box. Then click the right-arrow in the lower-right corner.

Each time you run a Stream Analytics job configured with a blob output, a new blob with a unique name is created. The purpose of **PATH PREFIX PATTERN** is to allow you to embed meaningful information, such as the time and date the job was executed, in the blob's name.

ADD A BLOB STORAGE

×

Blob Storage Settings

OUTPUT ALIAS

Flagged-Withdrawals

✓

SUBSCRIPTION

Use Storage Account from Current Subscription

▼

CHOOSE A STORAGE ACCOUNT

a4rlabs (East US)

▼

STORAGE ACCOUNT KEY

.....

CHOOSE A STORAGE CONTAINER ?

Create a new container

▼

CONTAINER

a4r-analytics

PATH PREFIX PATTERN ?

withdrawals/{date}/{time}

Example: cluster1/loqs/{date}/{time}

1

2

3

←

→

Specifying blob storage settings

5. Make sure **EVENT SERIALIZATION FORMAT**, **ENCODING**, and **FORMAT** are set as shown below, and then finish up by clicking the check mark in the lower-right corner.

1
2

ADD A BLOB STORAGE

Serialization settings

EVENT SERIALIZATION FORMAT ?

JSON

ENCODING ?

UTF8

FORMAT ?

Line separated

←

✓

Specifying serialization settings

- Click the **START** button at the bottom of the page to start the Stream Analytics job.

Microsoft Azure

CREDIT STATUS

Subscriptions

←

A4R-Analytics

NAME

TYPE

DIAGNOSIS

Flagged-Withdrawals	→ Blob storage	✓ OK
---------------------	----------------	------

+

NEW

▶

START

+

ADD OUTPUT

↔

TEST CONNECTION

🗑

DELETE

4

?

Starting the Stream Analytics job

- Make sure **JOB START TIME** is selected, and then click the check mark. **JOB START TIME** means that the job will begin sampling output from the input source the moment the job is started. Event hubs retain events for a specified period of time (the default is 1 day), so if you wanted, you could select **CUSTOM TIME** and start sampling output before the job's start time.

A4R-Analytics

START OUTPUT ?

JOB START TIME CUSTOM TIME

Specify when the job will start generating output. Note that the job may read the input data before this time if necessary to generate correct results for the specified query. To resume a stopped job without data loss, select Last Stopped Time (not available when running the job for the first time).



Starting the output

8. Return to the terminal window in which you ran eventgen.js and execute the following command to run it again:

```
node eventgen.js
```

9. Wait 5 to 10 minutes to give the job time to start and eventgen.js time to transmit several hundred events. Then terminate eventgen.js and return to the browser window.

If you'd like, you can open several terminal windows and run eventgen.js in each one to increase the volume of events.

10. Click the **STOP** button at the bottom of the page to stop the Stream Analytics job. Then click **YES** when asked if you're sure you want to stop the job.

Microsoft Azure | CREDIT STATUS Subscriptions

a4r-analytics

DASHBOARD MONITOR INPUTS QUERY OUTPUTS SCALE CONFIGURE

Output can't be edited while a job is running. You can stop the job to edit the output.

NAME	TYPE	DIAGNOSIS
Flagged-Withdrawals	Blob storage	✓ OK

+ NEW START **STOP** TEST CONNECTION ?

Stopping the Stream Analytics job

- Wait until the job is stopped, which might take a minute or two. Then click the **Storage** button in the ribbon on the left to go the portal's Storage page.

Microsoft Azure | CREDIT STATUS Subscriptions

a4r-analytics

DASHBOARD MONITOR INPUTS QUERY OUTPUTS SCALE CONFIGURE

NAME	TYPE	DIAGNOSIS
Flagged-Withdrawals	Blob storage	✓ OK

+ NEW START ADD OUTPUT TEST CONNECTION DELETE 1 ?

The Storage button

- Click the storage account that you designated to receive output from the Stream Analytics job.

Microsoft Azure | CREDIT STATUS | Subscriptions

STORAGE
4

HDINSIGHT 0
MEDIA SERVICES 0
SERVICE BUS 1
VISUAL STUDIO ONLINE 0
CACHE 0
BIZTALK SERVICES 0
RECOVERY SERVICES 0
CDN 0
AUTOMATION

storage

NAME	STATUS	LOCATION	SUBSCRIPTION
a4rlabs	✓ Online	East US	Visual Studio Ultimate with MS...
contoso2015	✓ Online	East US	Visual Studio Ultimate with MS...
contoso2016	✓ Online	East US	Visual Studio Ultimate with MS...
mycomix	✓ Online	East US	Visual Studio Ultimate with MS...

+ NEW | MANAGE ACCESS KEYS | MANAGE DOMAIN | DELETE | 1 ?

Storage accounts

- Click **CONTAINERS** to view the storage containers associated with this account.

Microsoft Azure | CREDIT STATUS | Subscriptions

a4rlabs

DASHBOARD | MONITOR | CONFIGURE | **CONTAINERS** | IMPORT/EXPORT

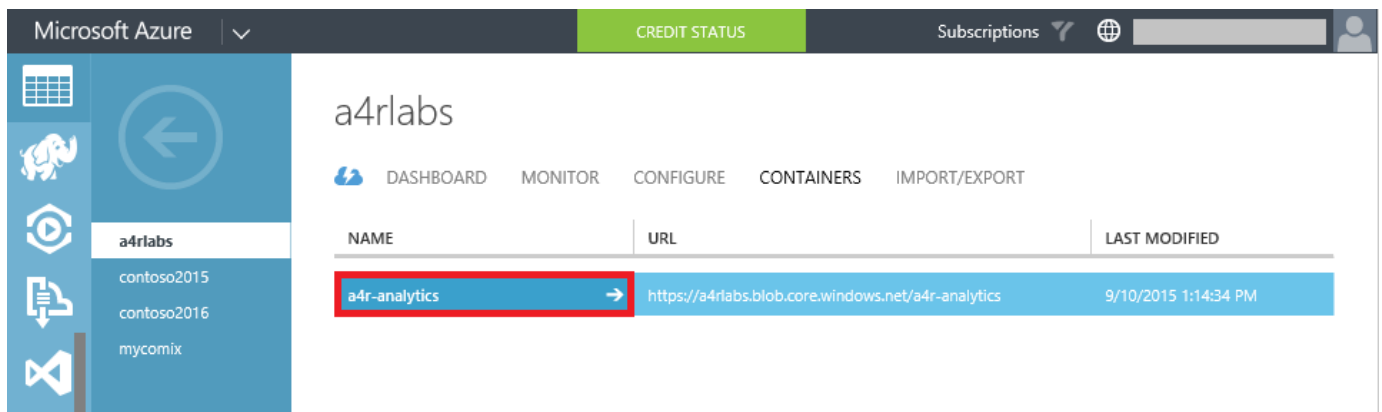
←

a4rlabs
contoso2015
contoso2016
mycomix

Your storage account has been created!
Here are a few options to help you get started.
☐ Skip Quick Start the next time I visit

a4rlabs account

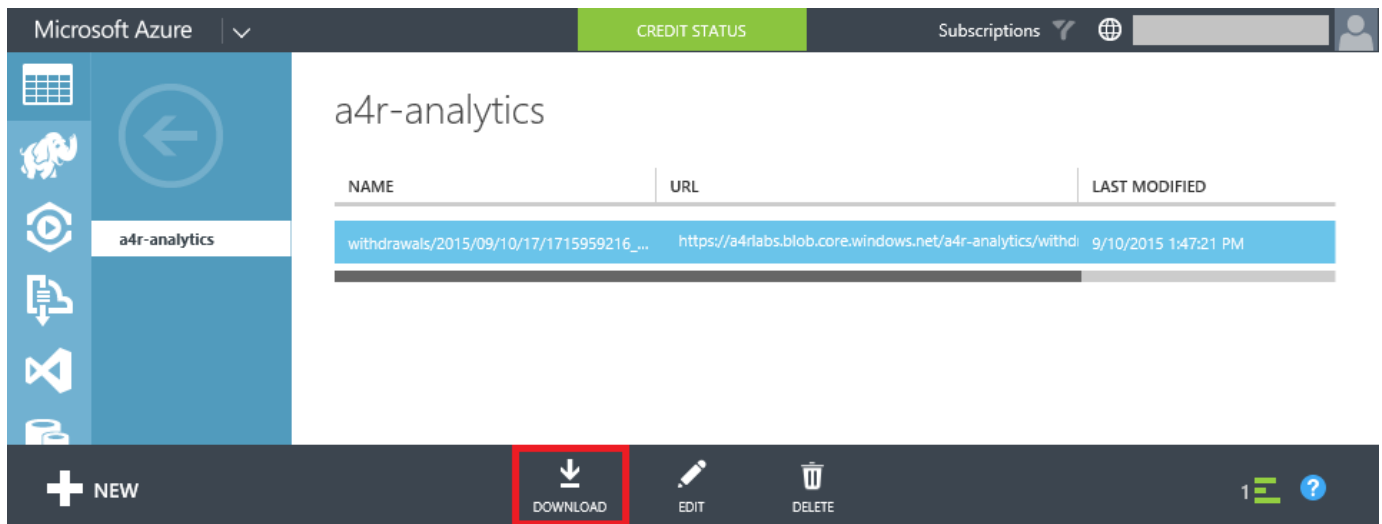
- Click **a4r-analytics** to view the contents of the container you created when you added an output to the Stream Analytics job.



a4rlabs containers

- Click the **DOWNLOAD** button at the bottom of the page to download the blob containing the output from the Stream Analytics job.

If there are no blobs in the container, wait a few minutes and check the container again. Sometimes a blob created by a Stream Analytics job appears immediately, and at other times, it may take 10 minutes or more to show up.



Downloading the blob containing the job output

- Open the downloaded JSON file in your favorite text editor. Each object (row) in the output represents a potentially fraudulent transaction. Note that **the number of rows and the content of each row will vary from machine to machine as well as from one run to another**.

```
{ "card number":179811479,"atm 1":60809,"atm 2":49279,"time 1":"2015-09-10T17:42:52.000000Z","time 2":"2015-09-10T17:43:03.000000Z"}
{"card number":566973417,"atm 1":75220,"atm 2":16356,"time 1":"2015-09-10T17:43:27.000000Z","time 2":"2015-09-10T17:43:32.000000Z"}
{"card number":337616401,"atm 1":34099,"atm 2":31717,"time 1":"2015-09-10T17:44:46.000000Z","time 2":"2015-09-10T17:45:00.000000Z"}
{"card number":384709097,"atm 1":21043,"atm 2":28896,"time 1":"2015-09-10T17:45:04.000000Z","time 2":"2015-09-10T17:45:23.000000Z"}
{"card number":904183575,"atm 1":90126,"atm 2":34762,"time 1":"2015-09-10T17:47:12.000000Z","time 2":"2015-09-10T17:47:15.000000Z"}
```

JSON job output

Currently, the data output from your Stream Analytics job is stored in a blob. In real life, you might prefer to view the output in a more convenient form, such as in a chart that's updated in real time. You could accomplish that by writing an application that monitors the blob and charts the data, or, better yet, by directing the output to an event hub and writing an application that subscribes to events from the event hub.

Microsoft recognizes that not everyone wants to write applications, and has provided an alternative in the form of **Microsoft Power BI**. With Power BI, you can create dashboards that render output from Stream Analytics jobs without writing any code. The connection between Azure and Power BI is currently offered only as a preview and is subject to certain limitations, but soon the two will be making beautiful music together in the hands of data scientists.

Summary

Azure Stream Analytics is a powerful tool for analyzing live data streams from IoT devices or anything else that's capable of transmitting data. In

this lab, you got a first-hand look at Stream Analytics as well as Azure event hubs. Among other things, you learned how to:

- Create an Azure event hub and use it as a Stream Analytics input
- Create a shared access signature that allows event hubs to be called securely using REST APIs
- Create a Stream Analytics job and test queries on sample data streams
- Run a Stream Analytics job and perform queries on live data streams
- Create a rule (query) that detects anomalies in streaming data
- Use that rule to record anomalies in Azure blobs

One drawback to hard-coding rules into Stream Analytics is that rules don't "learn" from the data streams, which can lead to false positives in anomaly detection. If this concerns you, read the article entitled [Anomaly Detection – Using Machine Learning to Detect Abnormalities in Time Series Data](#) in the Azure team's Machine Learning blog. In it, they present an anomaly detection service accessible through a REST API that uses Azure Machine Learning to learn from the data presented to it. Imagine combining the power of Stream Analytics to extract information from real-time data streams with the power of Azure Machine Learning to learn from that information and refine the analytics on the fly. This is precisely the type of solution that Microsoft Azure empowers you to build!

Copyright 2015 Microsoft Corporation. All rights reserved. Except where otherwise noted, these materials are licensed under the terms of the Apache License, Version 2.0. You may use it according to the license as is most appropriate for your project on a case-by-case basis. The terms of this license can be found in <http://www.apache.org/licenses/LICENSE-2.0>.