Azure Stream Analytics:

Module 6, Lesson 6  
Real-time Fraud Detection Hands-On Lab

## Overview

In this lab, students will learn how to create an end-to-end solution for real-time fraud detection with Azure Stream Analytics. Students will bring events into an Azure event hub, write Stream Analytics queries for aggregation or alerting, and send the results to an output sink to gain insight over data with real-time processing. Real time anomaly detection for telecommunications as shown in this example technique is equally suited for other types of fraud detection such as credit card or identity theft scenarios

## Objectives

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

* Setup data to input into a pipeline of stages that will result in a new model
* Setup the pipeline and execute it
* Setup test data and run the new model against the test data

## Prerequisites

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

* A Web browser
* Download [TelcoGenerator.zip](http://download.microsoft.com/download/8/B/D/8BD50991-8D54-4F59-AB83-3354B69C8A7E/TelcoGenerator.zip) from the Microsoft Download Center
* Completion of [Module 6 Lessons 1-6](https://github.com/MSFTImagine/computerscience/tree/master/Complimentary%20Course%20Content/Module6/Lessons) and [Labs 4 and 5](https://github.com/MSFTImagine/computerscience/tree/master/Complimentary%20Course%20Content/Module6/Labs)

Note : The Azure portal is continually improved and changed. The steps in this exercise reflect the user interface of the Microsoft Azure portal at the time of writing, but may not match the latest design of portal.

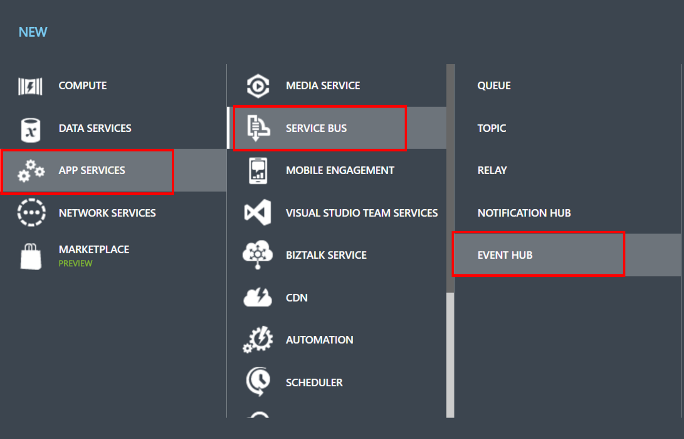
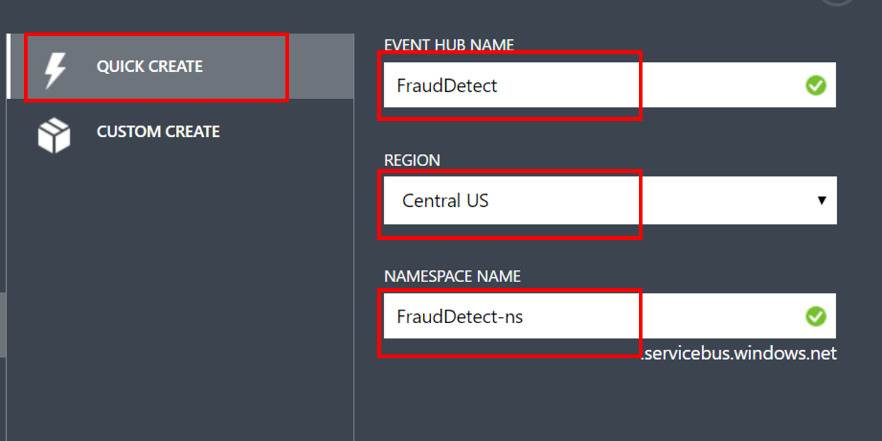
## Exercises

This hands-on lab includes the following exercises:

* Exercise 1: Create an Azure Event Hubs input
* Exercise 2: Configure and start event generator application
* Exercise 3: Create Stream Analytics Job
* Exercise 4: Create Stream Analytics Job Output

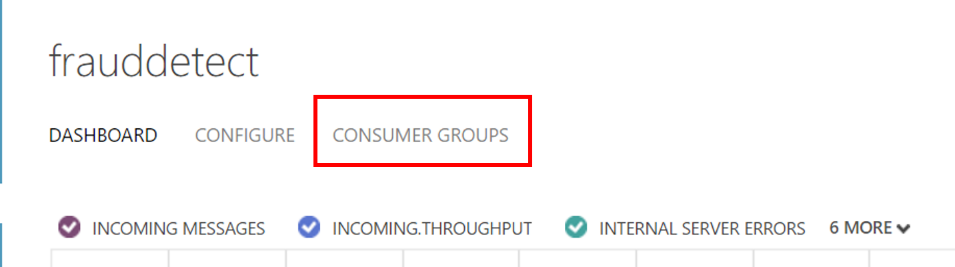
## Exercise 1: Create an Azure Event Hubs input

Students will generate events and push them to an Event Hub instance for real-time processing. Service Bus Event Hubs are the preferred method of event ingestion for Stream Analytics.

1. In the classic [Azure portal](https://manage.windowsazure.com/) (manage.windowsazure.com) click **New** > **App Services** > **Service Bus** > **Event Hub** > **Quick**

Provide a name, region, and new or existing namespace to create a new Event Hub.

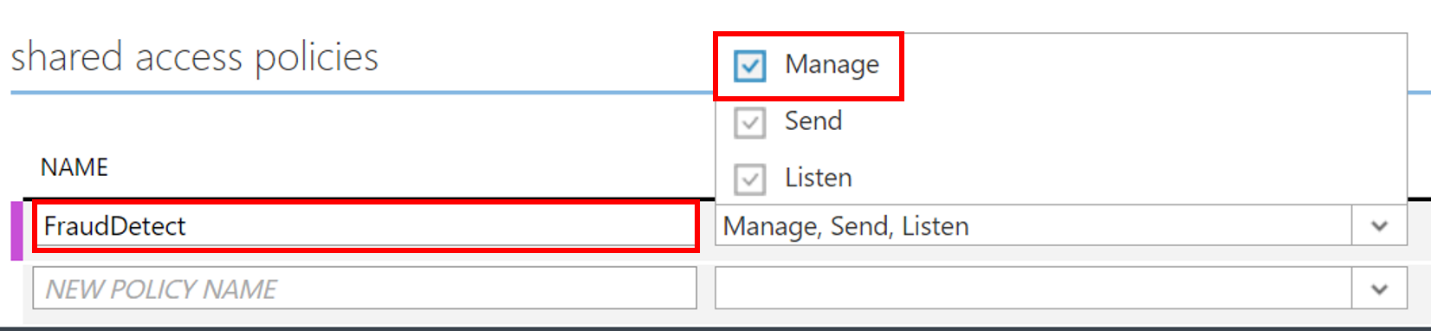
1. Create Consumer Group



As a best practice, each Stream Analytics job should read from a single Event Hub Consumer Group. To create a Consumer Group, navigate to the newly created Event Hub and click the **Consumer Groups** tab, then click **Create** on the bottom of the page and provide a name for the Consumer Group.

1. Grant access to Event Hub

A shared access policy needs to be created to grant access to the Event Hub. Under **Configure >** **Shared Access Policies**, create a new policy with **Manage** permissions.



After entering a policy name and manage permissions, click **Save** at bottom of page.

1. Copy **Connection Information**

Navigate to the **Dashboard** and click **Connection Information** at the bottom of the page, and then copy and save the connection information. Students can use Notepad to save the connection string to a text file.

## Exercise 2: Configure and start event generator application

For this lab, students will use a client application that will generate sample incoming call metadata and push it to Event Hub. Follow the steps below to set up this application.

1. Download the [TelcoGenerator.zip file](http://download.microsoft.com/download/8/B/D/8BD50991-8D54-4F59-AB83-3354B69C8A7E/TelcoGenerator.zip). Then unzip it to a directory.

Windows may block the downloaded zip file. Right click the file and select properties. If the message "This file came from another computer and might be blocked to help protect this computer." then tick the "Unblock" box and click apply on the zip file.

1. Replace the Microsoft.ServiceBus.ConnectionString and EventHubName values in **telcodatagen.exe** with the Event Hub connection string and name.

The connection string copied from the Azure portal places the name of the connection at the end.

1. Start the telcodatagen application.

In PowerShell navigate to the TelcoGenerator directory; the usage is as follows:

telcodatagen.exe [#NumCDRsPerHour] [SIM Card Fraud Probability] [#DurationHours]

telcodatagen.exe 1000 .2 2

Students should see data being sent to the Event Hub. Some key fields that we will be using in this real-time fraud detection application are defined here:

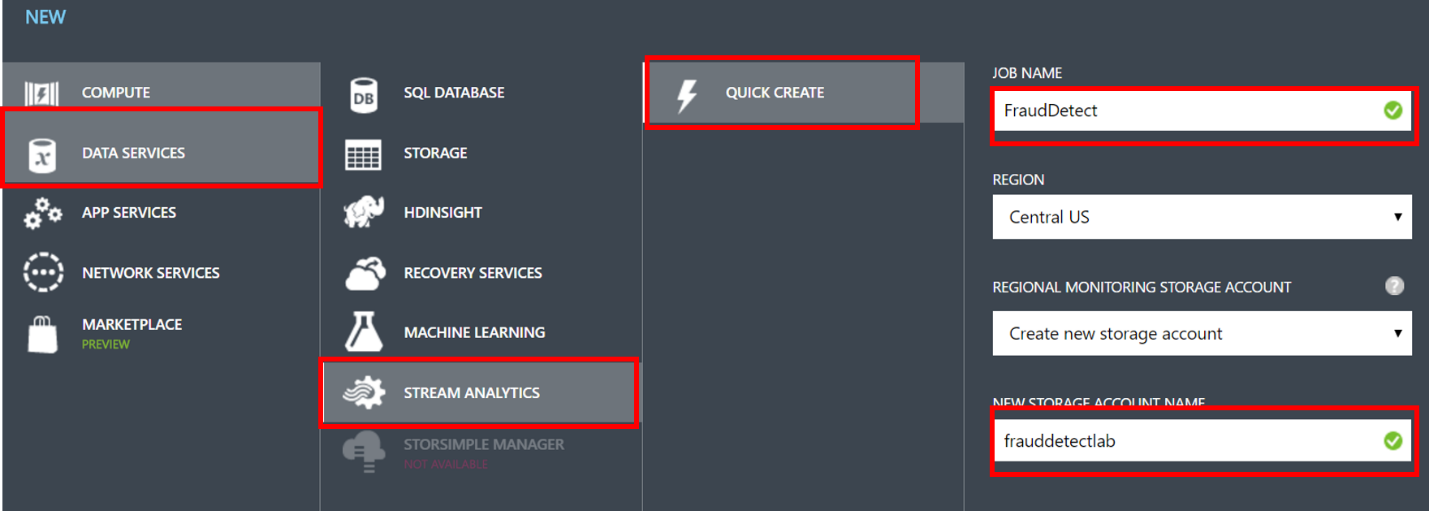
| **Record** | **Definition** |
| --- | --- |
| CallrecTime | Timestamp for the call start time. |
| SwitchNum | Telephone switch used to connect the call. |
| CallingNum | Phone number of the caller. |
| CallingIMSI | International Mobile Subscriber Identity (IMSI). Unique identifier of the caller. |
| CalledNum | Phone number of the call recipient. |
| CalledIMSI | International Mobile Subscriber Identity (IMSI). Unique identifier of the call recipient. |

## Exercise 3: Create Stream Analytics Job

Students have just created an event hub where a stream of telecommunications events are being sent. The next step is to set up a Stream Analytics job to analyze these events in real time.

1. Provision a Stream Analytics Job

In the Azure portal, click **New > Data Services > Stream Analytics > Quick Create**

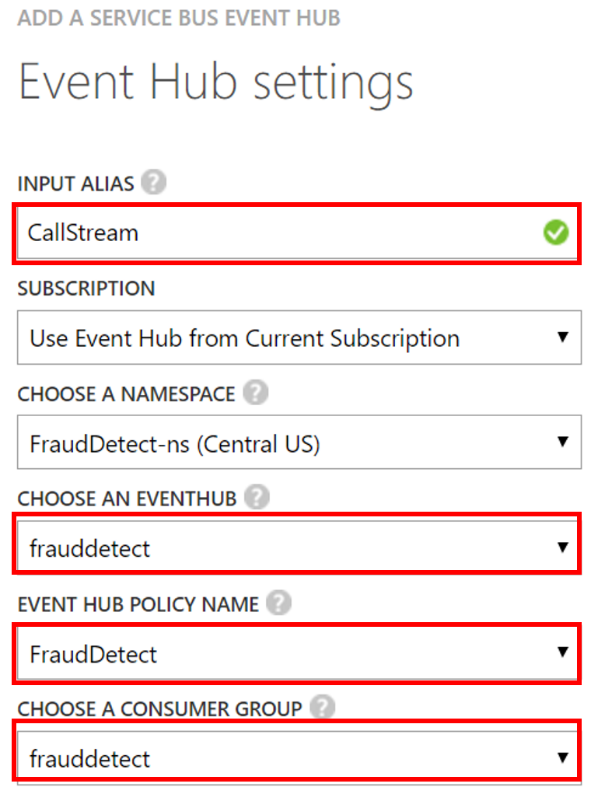


Specify the Job Name. Select a region where the job will run. It is best to choose the same region as where the Event Hub created in the previous exercise. This will ensure that data is not being transferred between regions at an additional cost. Choose or create a storage account where the monitoring data for the Stream Analytics job will be saved. Finally, click on the **Create** button on the bottom to create a new Stream Analytics job.

Click on **Stream Analytics** to review the newly created job. Notice that the start button is disabled. This will be enabled once the job input, output, and query is specified.

1. Specify Job Input

From the Stream Analytics job created in the previous step, click **Inputs** from the top of the page, and then click **Add Input**. The dialog box that opens will walk students through a number of steps to set up the input.



Select **Data Stream** and click right arrow button on bottom left.

Select **Event Hub** and click right arrow button on bottom left.

In **Input Alias**, type in any friendly name. This name will be used later in queries. In **Choose and EventHub**, select the Event Hub created in the previous step. For **Event Hub Policy Name** and **CHOOSE A ConsumeR Group**, select the policy and consumer group created in exercise 1.

Click right arrow button on bottom left.

Next, chose JSON for **Event Serializer Format** and UTF8 for **Encoding.**

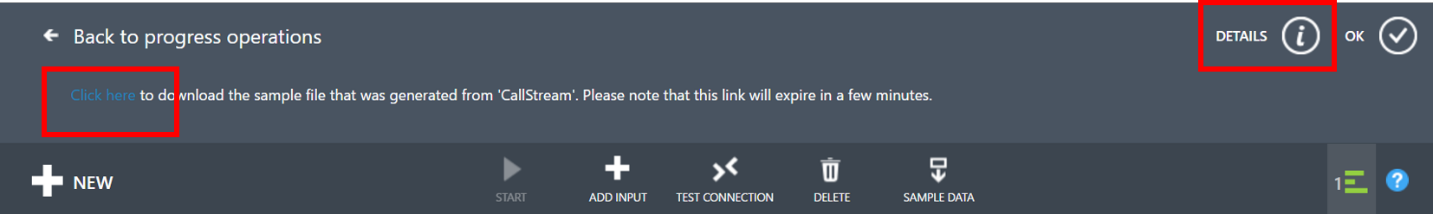
Click the check button to add the specified input source and verify that Stream Analytics can connect to the specified Event Hub.

1. Specify Job Query

Stream Analytics supports a simple, declarative query model for describing transformations for real-time processing. To learn more about the language, see the [Azure Stream Analytics Query Language Reference](https://msdn.microsoft.com/library/dn834998.aspx). This tutorial will help students author and test several queries over the real-time stream of call data.

1. Create sample input data

The **Sample Data** feature can be used to extract sample events on which users can validate queries. This feature creates a JSON file.

Select the Event Hub Input and click on **Sample Data** at bottom of page. In the dialogue box that appears, specify a **START TIME** and **Duration**. Students should keep the default start time and adjust the Duration to about 2 minutes. Click on the check button to start sampling event data. Once the process is completed, click **Details** and download the JSON file created. 

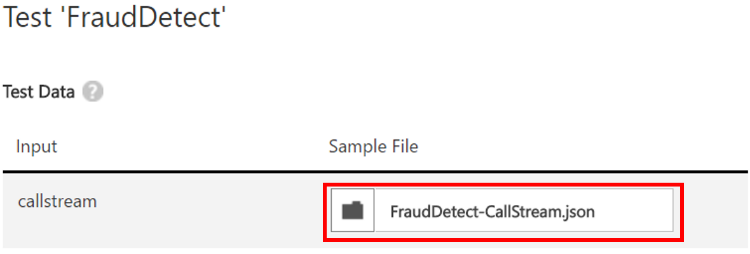
1. Create a passthrough query

All events can be archived using a passthrough query to read all the fields of the event.

Click **Query** from the top of the Stream Analytics job page and replace the query with the following code.

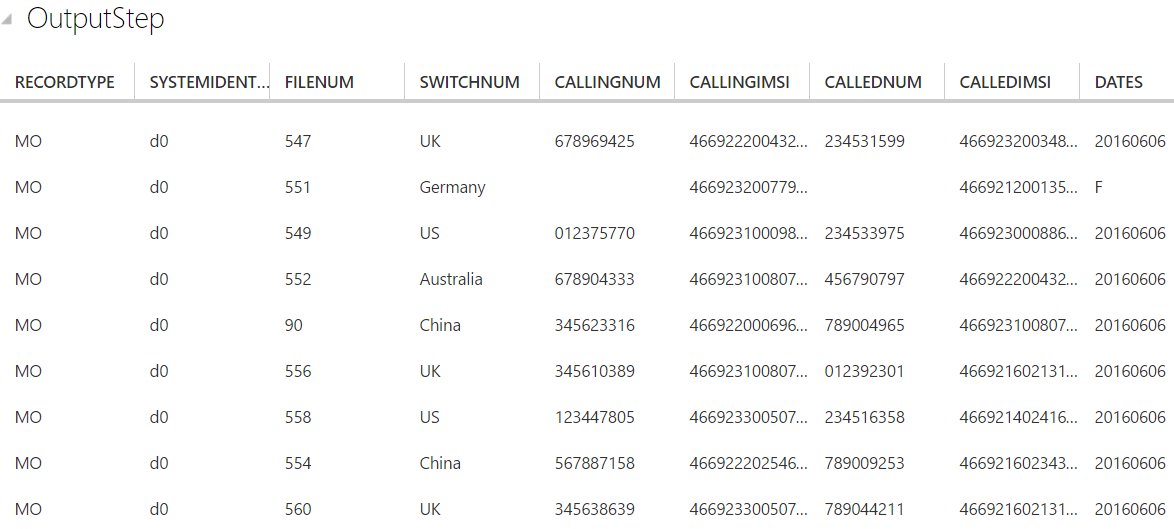
Select \* from *CallStream*

The example shows *CallStream*, however, students should substitute the INPUT ALIAS declared in the previous step.



Click **TEST** under the query editor and specify the JSON file created in the previous step as the test file.

Click the check button to see the results of query displayed by scrolling down..

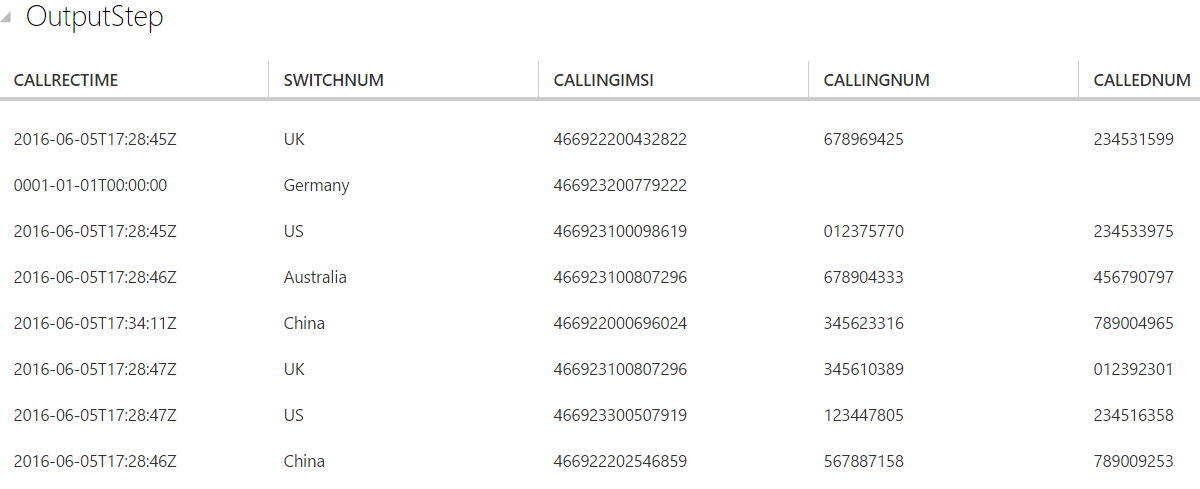


1. Create a column projection

Change the query to the code below and click **Rerun** to see results of query.

SELECT CallRecTime, SwitchNum, CallingIMSI, CallingNum, CalledNum

FROM CallStream



1. Create tumbling window with aggregation query

Create a query to count number of incoming calls by region. To compare the amount of incoming calls per region, students will use a tumbling window to get the count of incoming calls grouped by SwitchNum every 5 seconds.

Change the query to the code below and click **Rerun** to see results of query.

SELECT System.Timestamp as WindowEnd, SwitchNum, COUNT(\*) as CallCount

FROM CallStream TIMESTAMP BY CallRecTime

GROUP BY TUMBLINGWINDOW(s, 5), SwitchNum

This query uses the Timestamp By keyword to specify a timestamp field in the payload to be used in the temporal computation. If this field wasn't specified, the windowing operation would be performed using the time each event arrived at Event Hub.



1. Create SIM fraud detection with a Self-Join

To identify potentially fraudulent usage we'll look for calls originating from the same user but in different locations in less than 5 seconds. We join the stream of call events with itself to check for these cases.

Change the query to the code below and click **Rerun** to see results of query. Be sure to save your query before moving on (it must be saved in order to run a job).

SELECT System.Timestamp as Time, CS1.CallingIMSI, CS1.CallingNum as CallingNum1,

CS2.CallingNum as CallingNum2, CS1.SwitchNum as Switch1, CS2.SwitchNum as Switch2

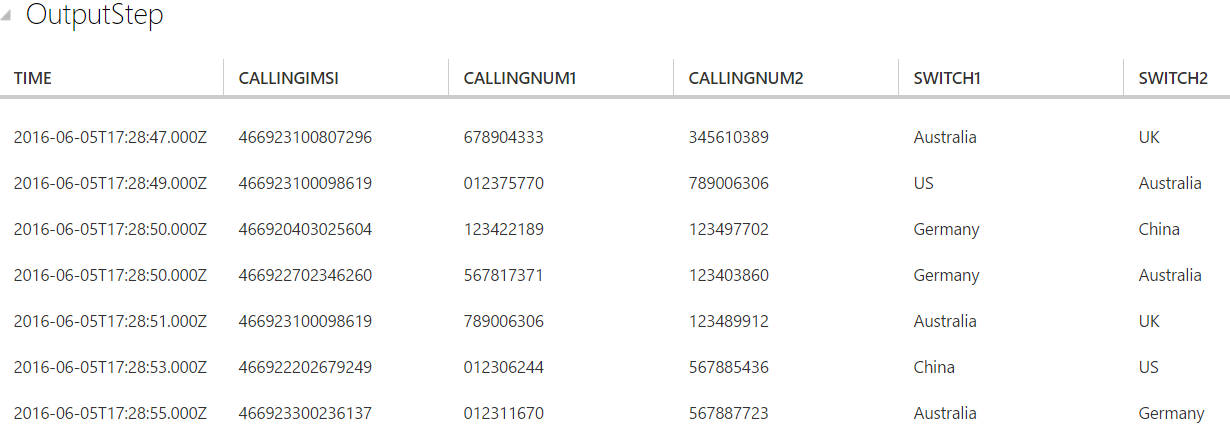
FROM CallStream CS1 TIMESTAMP BY CallRecTime

JOIN CallStream CS2 TIMESTAMP BY CallRecTime

ON CS1.CallingIMSI = CS2.CallingIMSI

AND DATEDIFF(ss, CS1, CS2) BETWEEN 1 AND 5

WHERE CS1.SwitchNum != CS2.SwitchNum



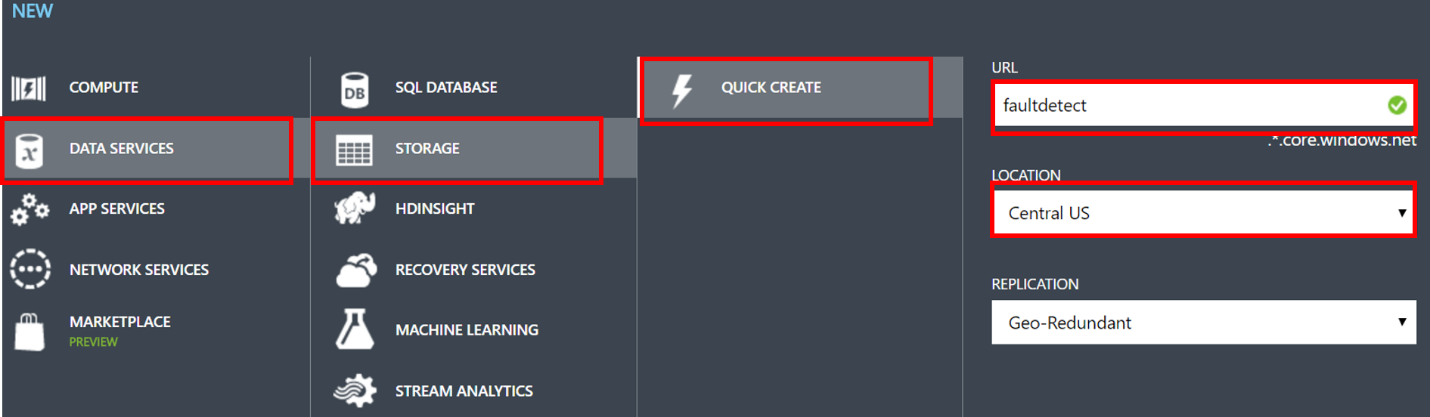
## Exercise 4: Create Stream Analytics Job Output

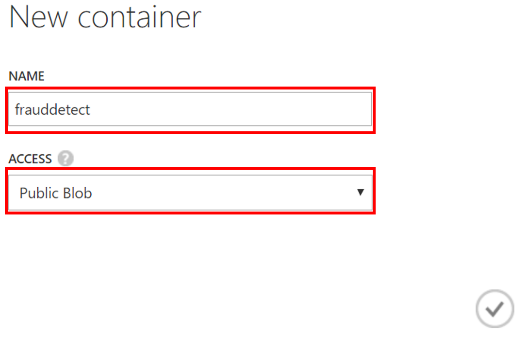
Students have defined an event stream, created an Event Hub input to ingest the events, and queries to perform transformation over the event stream. Now, the output of these Stream Analytics will be defined.

1. Create an output sink

An output sink will be defined to write events for fraudulent behavior to Blob storage.

Use an existing storage account or create a new storage account by clicking **NEW > DATA SERVICES > STORAGE > QUICK CREATE** and following the instructions.



Select the storage account from above and click **CONTAINERS** at the top of the page and click **ADD**. Specify a **NAME** for the container and set its **ACCESS** to Public Blob.

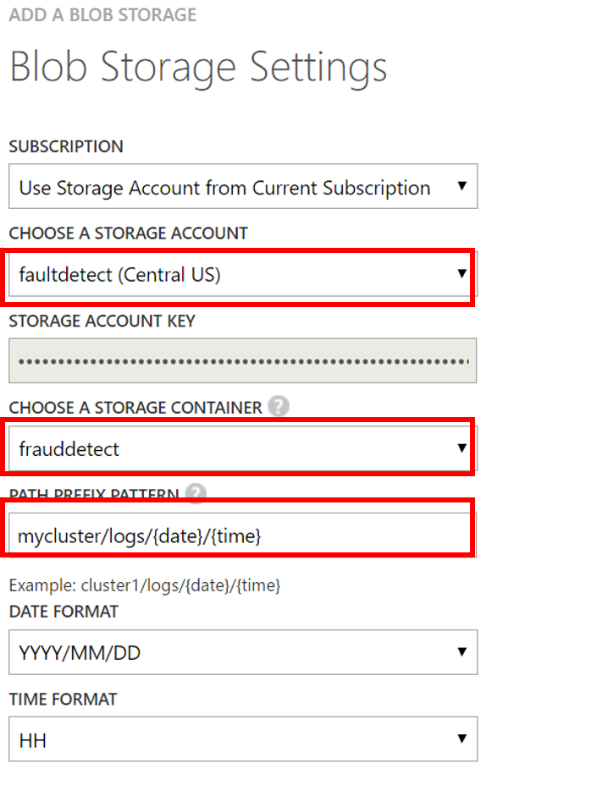
1. Specify job output

From the Stream Analytics job created in the previous exercises, click **OUTPUT** from the top of the page and click **ADD OUTPUT**. Students will be saving the output to the Blob storage created in the previous step. Select **BLOB STORAGE** as the output type and click the right button. Select the following values on the next page:

**OUTPUT ALIAS:**  Enter a friendly name for the job output.

**Subscription**: If the Blog Storage created in the previous step is in the same subscription as the Stream Analytics job, select **Use Storage Account from Current Subscription**. If it is in a different subscription, select **Use Storage Account from Another Subscription**, and manually enter information for the **STORAGE ACCOUNT**, **STORAGE KEY,** and **CONTAINER.**

**Storage Account**: Select the name of the storage account created in previous step.

**Container**: Select the name of the container created in previous step

**Filename Prefix**: Type in a file prefix to use when writing to Blob storage.

Click right arrow button on bottom left.

Next, chose JSON for **Event Serializer Format** and UTF8 for **Encoding.** Choose Line Separatedfor **format.**

Click the check button to add this source and to verify that Stream Analytics can successfully connect to the storage account.

1. Verify you have a valid query
2. Start job for real time processing

From the job dashboard, click start at the bottom of the page.

In the dialog box that appears, select **JOB START TIME** and then click the check button on the bottom of the dialog box. The job status will change to **Starting** and will shortly move to **Running**

## Summary

In this hands-on lab, you learned how to:

* Create a Stream Analytics Job by working through the 3 major steps:
  + Connecting to a stream of events as Input
  + Creating a query for transformation of the events
  + Creating an Output to save the results of the Stream Analytics job