**Lab - Performing Real-Time Analytics with Stream Analytics**

**Lab objectives**

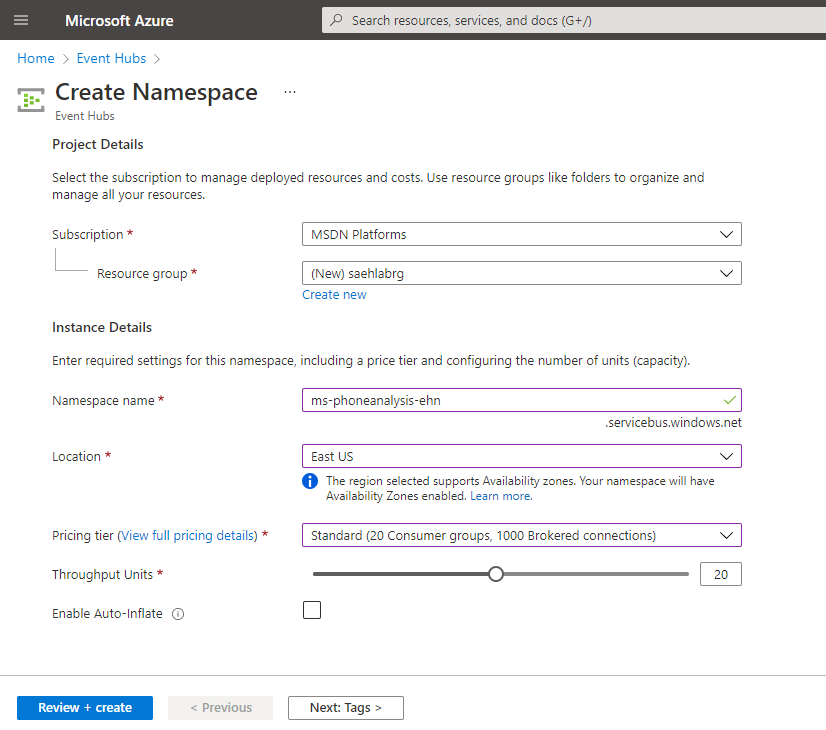
After completing this lab, you will be able to:

1. Explain data streams and event processing
2. Ingest data with Event Hubs
3. Initiate a data generation application
4. Process Data with a Stream Analytics Jobs

**Exercise 1: Data Ingestion with Event Hubs.**

**Task 1: Create and configure an Event Hub Namespace.**

1. In the Azure portal, click on the **Home** hyperlink at the top left of the screen.
2. In the Azure portal, click on the **+ Create a resource** icon , type **Event Hubs**, and then select **Event Hubs** from the resulting search. In the Event Hubs screen, click **Create**.
3. In the Create Namespace blade, type out the following options:
   * **Subscription**: **Your subscription**
   * **Resource group**: **saehlabrg**
   * **Namespace Name**: **xx-phoneanalysis-ehn**, where xx are your initials
   * **Location**: select the location closest to you
   * **Pricing Tier**: **Standard**
   * **Throughput Units**: **20**
   * Leave other options to their default settings

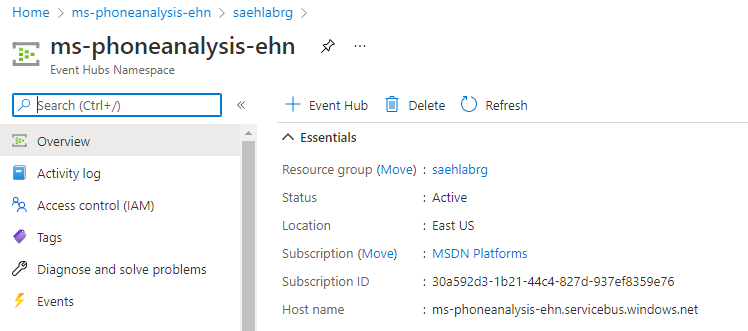


1. Then click **Review + Create** and then click **Create**

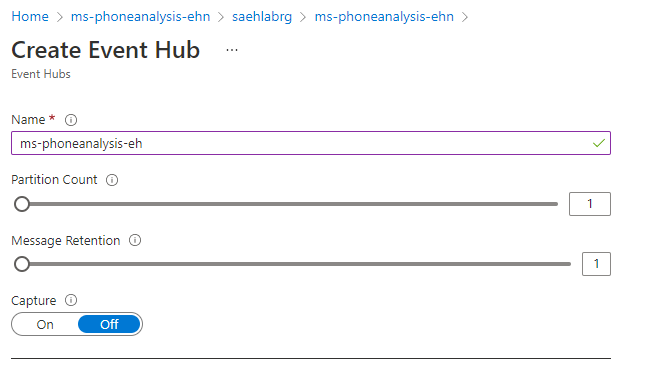
**Note**: The creation of the Event Hub Namespace takes approximately 1 minute.

**Task 2: Create and configure an Event Hub**

1. In the Azure portal, click on the **Home** hyperlink at the top left of the screen.
2. In the Azure portal, in the blade, click **Resource groups**, and then click **saehlabrg**
3. Click on **xx-phoneanalysis-ehn**, where **xx** are your initials.
4. In the **xx-phoneanalysis-ehn** screen, click on **+ Event Hubs**.



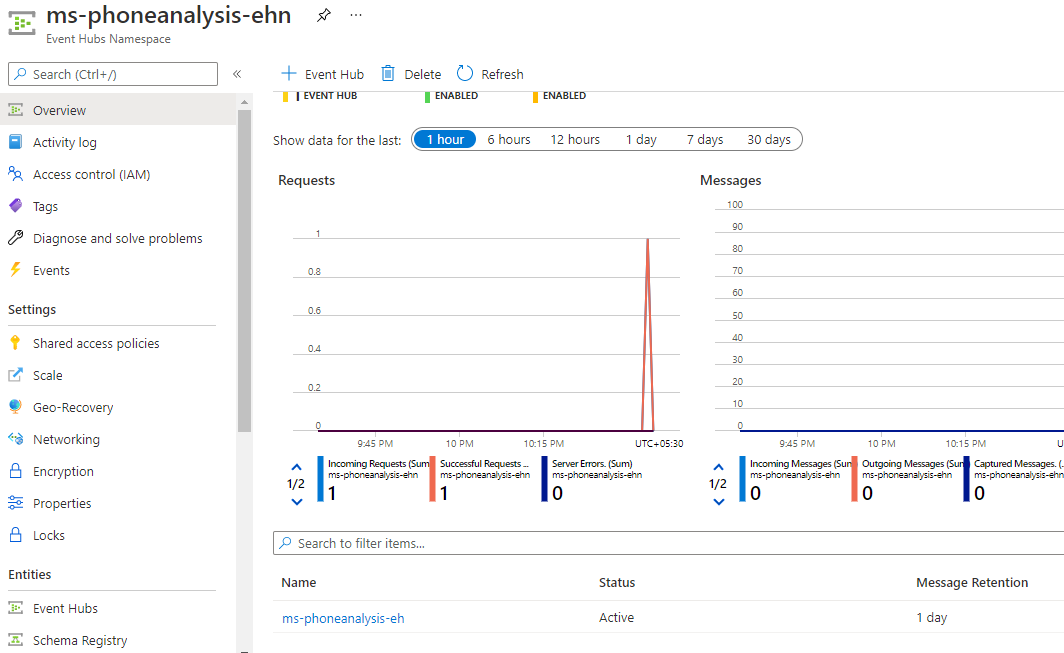
1. Provide the name **xx-phoneanalysis-eh**, leave the other settings to their default values and then select **Create**.



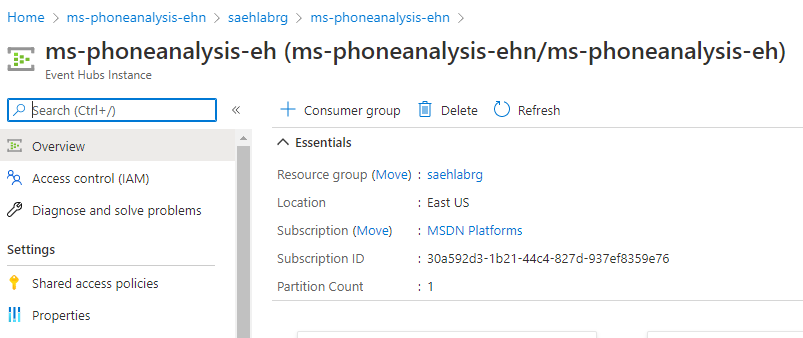
**Note**: You will receive a message stating that the Event Hub is created after about 10 seconds

**Task 3: Configure Event Hub security**

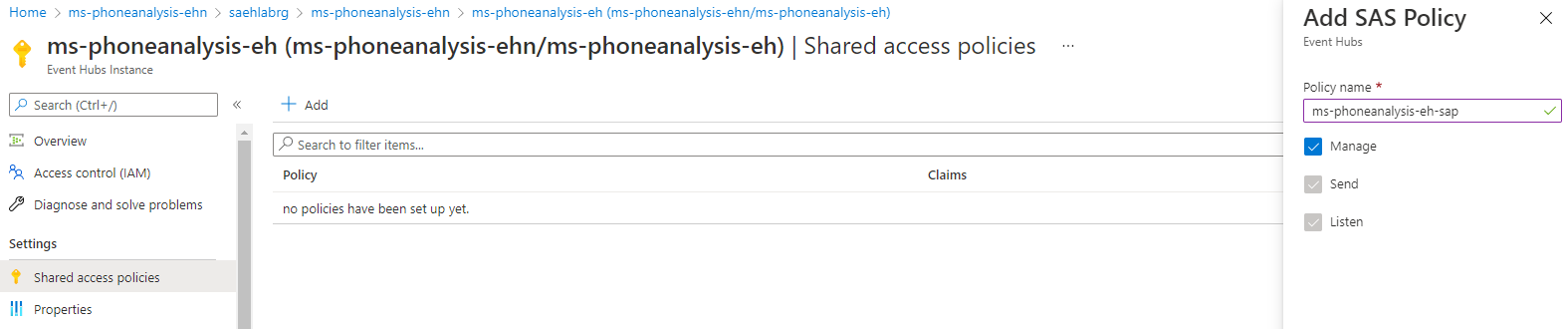
1. In the Azure portal, in the **xx-phoneanalysis-ehn** screen, where **xx** are your initials. Scroll to the bottom of the window, and click on **xx-phoneanalysis-eh** event hub.



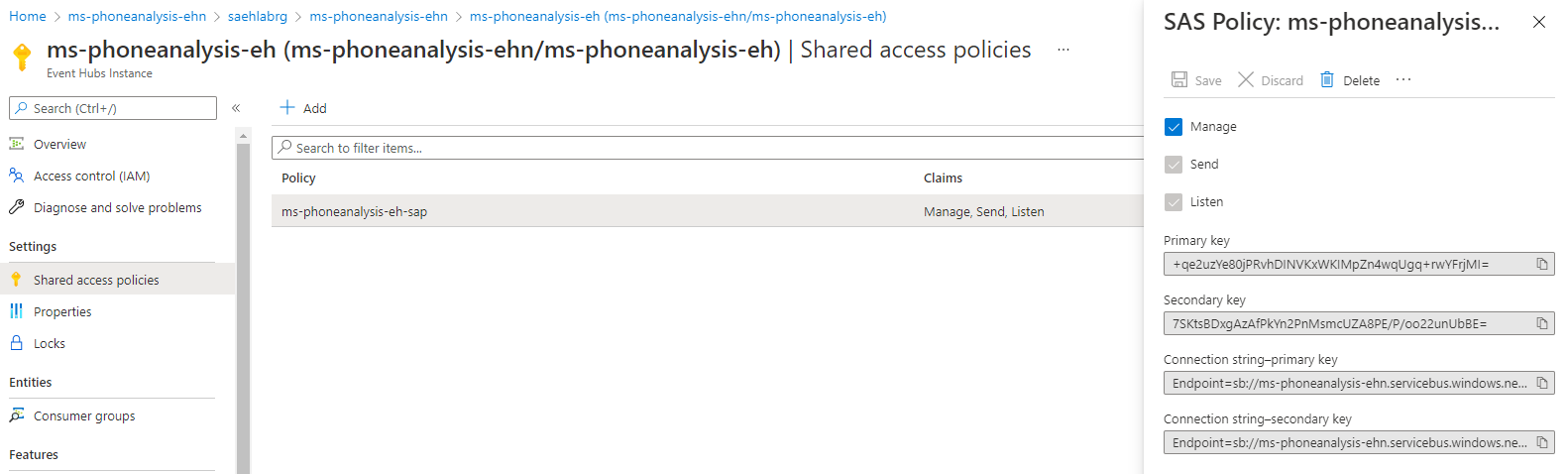
1. To grant access to the event hub, in the blade under the section **settings** on the left click **Shared access policies**.



1. Under the **xx-phoneanalysis-eh - Shared access policies** screen, create a policy with **Manage** permissions by selecting **+ Add**. Give the policy the name of **xx-phoneanalysis-eh-sap** , check **Manage**, and then click **Create**.



1. Click on your new policy **xx-phoneanalysis-eh-sap** after it has been created, and then select the copy button for the **CONNECTION STRING - PRIMARY KEY** and paste the CONNECTION STRING - PRIMARY KEY into Notepad, this is needed later in the exercise.



**CONNECTION STRING - PRIMARY KEY**

Endpoint=sb://ms-phoneanalysis-ehn.servicebus.windows.net/;SharedAccessKeyName=ms-phoneanalysis-eh-sap;SharedAccessKey=+qe2uzYe80jPRvhDINVKxWKIMpZn4wqUgq+rwYFrjMI=

**NOTE**: The connection string looks as follows:

Endpoint=sb://<Your event hub namespace>.servicebus.windows.net/;SharedAccessKeyName=<Your shared access policy name>;SharedAccessKey=<generated key>;EntityPath=<Your event hub name>

Notice that the connection string contains multiple key-value pairs separated with semicolons: Endpoint, SharedAccessKeyName, SharedAccessKey, and EntityPath.

1. Close down the Event hub screens in the portal

**Result**: After you completed this exercise, you have created an Azure Event Hub within an Event Hub Namespace and set the security for the Event Hub that can be used to provide access to the service.

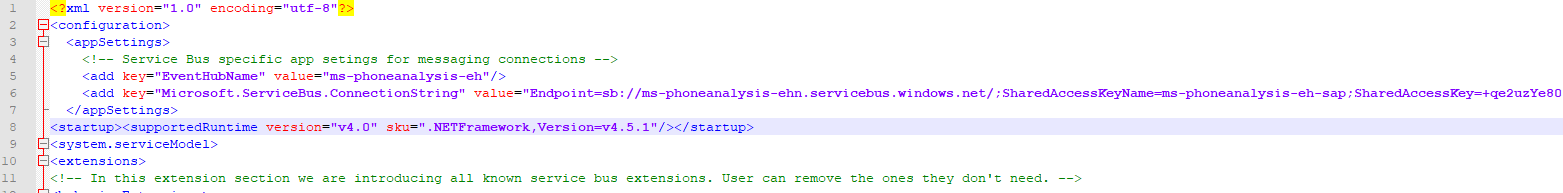
**Exercise 3: Starting the telecom event generator application**

The main tasks for this exercise are as follows:

1. Updates the application connection string
2. Run the application

**Task 1: Updates the application connection string.**

1. Browse to the location **\DataGenerator**
2. Open the **telcodatagen.exe.config** file in a text editor of your choice
3. Update the element in the config file with the following details:
   * Set the value of the **EventHubName** key to the value of the **EntityPath** in the connection string.
   * Set the value of the **Microsoft.ServiceBus.ConnectionString** key to the connection string **without the EntityPath value** (don't forget to remove the semicolon that precedes it).



<add key="EventHubName" value="ms-phoneanalysis-eh"/>

<add key="Microsoft.ServiceBus.ConnectionString" value="Endpoint=sb://ms-phoneanalysis-ehn.servicebus.windows.net/;SharedAccessKeyName=ms-phoneanalysis-eh-sap;SharedAccessKey=+qe2uzYe80jPRvhDINVKxWKIMpZn4wqUgq+rwYFrjMI="/>

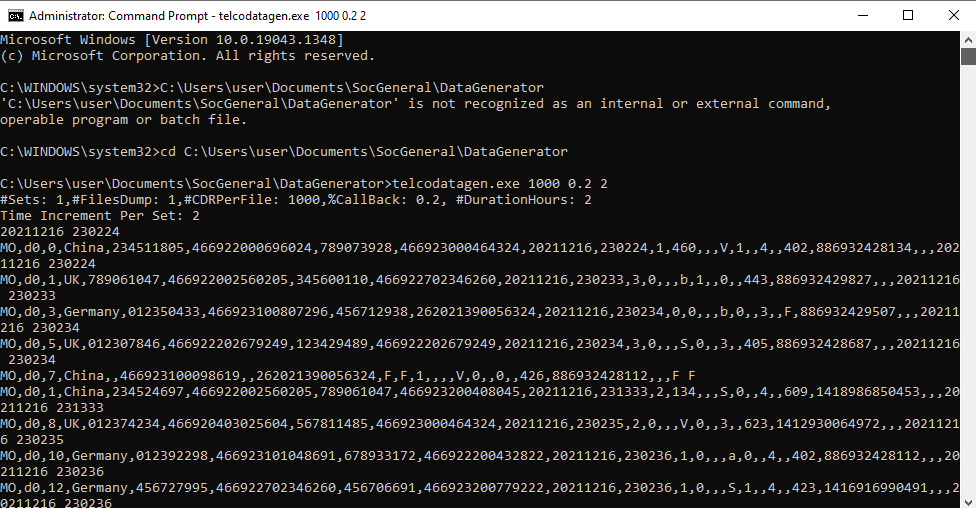
1. Save the file.

**Task 2: Run the application.**

1. Click on **Start**, and type **CMD**
2. Right click **Command Prompt**, click **Run as Administer**, and in the User Access Control screen, click **Yes**
3. In Command Prompt, browse to the location **\DataGenerator**
4. Type in the following command:

telcodatagen.exe 1000 0.2 2

NOTE: This command takes the following parameters: Number of call data records per hour. Percentage of fraud probability, which is how often the app should simulate a fraudulent call. The value 0.2 means that about 20% of the call records will look fraudulent. Duration in hours, which is the number of hours that the app should run. You can also stop the app at any time by ending the process (Ctrl+C) at the command line.



After a few seconds, the app starts displaying phone call records on the screen as it sends them to the event hub. The phone call data contains the following fields:

| **Record** | **Definition** |
| --- | --- |
| CallrecTime | The timestamp for the call start time. |
| SwitchNum | The telephone switch used to connect the call. For this example, the switches are strings that represent the country/region of origin (US, China, UK, Germany, or Australia). |
| CallingNum | The phone number of the caller. |
| CallingIMSI | The International Mobile Subscriber Identity (IMSI). It's a unique identifier of the caller. |
| CalledNum | The phone number of the call recipient. |
| CalledIMSI | International Mobile Subscriber Identity (IMSI). It's a unique identifier of the call recipient. |

1. Minimize the command prompt window.

**Result**: After you completed this exercise, you have configured an application to generate data to minimum phone calls received by a call center.

**Exercise 4: Processing Data with Stream Analytics Jobs**

Estimated Time: 15 minutes

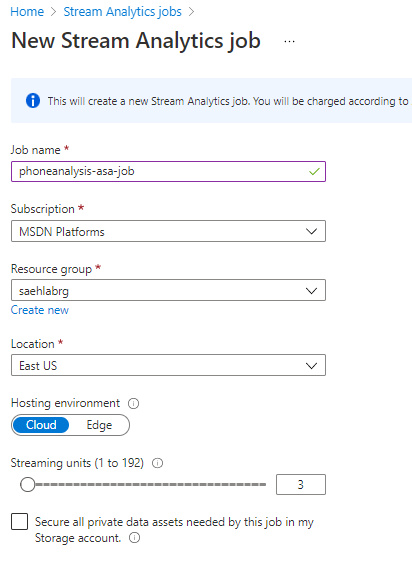
Individual exercise

The main tasks for this exercise are as follows:

1. Provision a Stream Analytics job.
2. Specify the a Stream Analytics job input.
3. Specify the a Stream Analytics job output.
4. Defining a Stream Analytics query.
5. Start the Stream Analytics job.
6. Validate streaming data is collected

**Task 1: Provision a Stream Analytics job.**

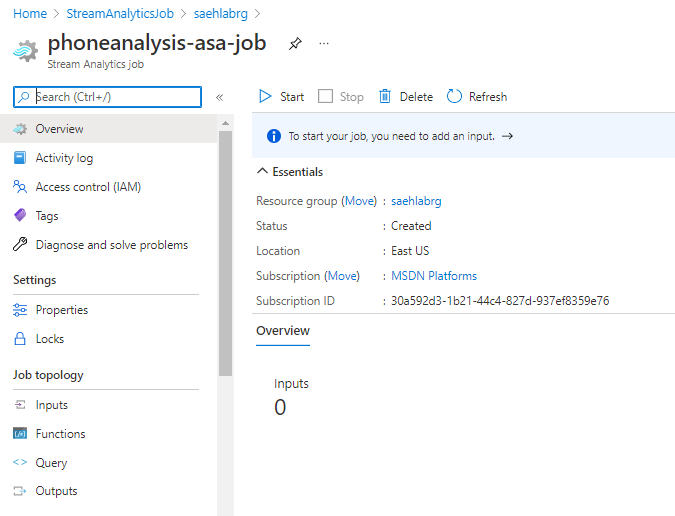
1. Go back to the Azure portal, navigate and click on the **+ Create a resource** icon, type **Stream analytics**, and then click the **Stream Analytics Job**, and then click **Create**.
2. In the **New Stream Analytics job** screen, fill out the following details and then click on **Create**:
   * **Job name**: phoneanalysis-asa-job.
   * **Subscription**: select your subscription
   * **Resource group**: saehlabrg
   * **Location**: choose a location nearest to you.
   * Leave other options to their default settings
   * Click **Create**



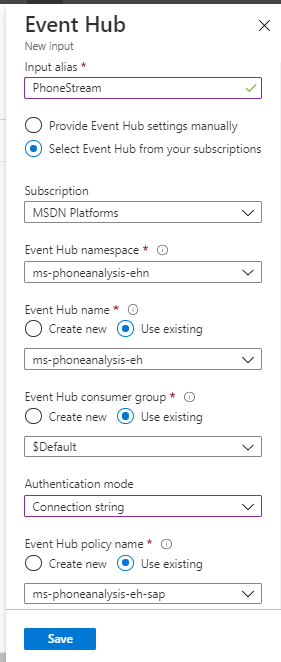
1. **Note**: You will receive a message stating that the Stream Analytics job is created after about 10 seconds. It may take a couple of minutes to update in the Azure portal.
2. **Note**: Before going to Task 2, create a blob storage and container name as phoneclaim

**Task 2: Specify the Stream Analytics job input.**

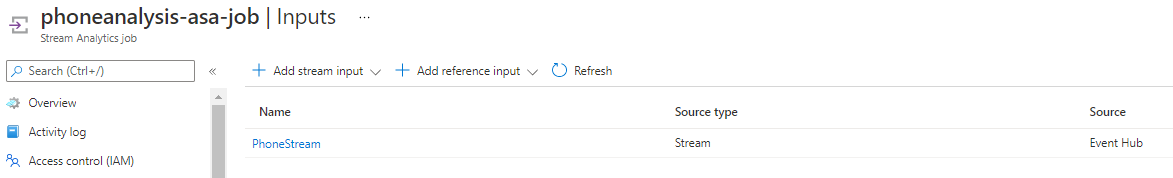
1. In the Azure portal, in the blade, click **Resource groups**, and then click **saehlabrg**
2. Click on **phoneanalysis-asa-job**.
3. In your **phoneanalysis-asa-job** Stream Analytics job window, in the left hand blade, under **Job topology**, click **Inputs**.



1. In the **Inputs** screen, click **+ Add stream input**, and then click **Event Hub**.
2. In the Event Hub screen, type in the following values and click the **Save** button.
   * **Input alias**: Enter a name for this job input as **PhoneStream**.
   * **Select Event Hub from your subscriptions**: checked
   * **Subscription**: Your subscription name
   * **Event Hub Namespace**: xx-phoneanalysis-ehn
   * **Event Hub Name**: Use existing named xx-phoneanalysis-eh
   * **Event Hub Consumer Group**: Use existing
   * **Authentication Method**: Connection string
   * **Event Hub Policy Name**: use existing named xx-phoneanalysis-eh-sap
   * Leave the rest of the entries as default values. Finally, click **Save**\*.

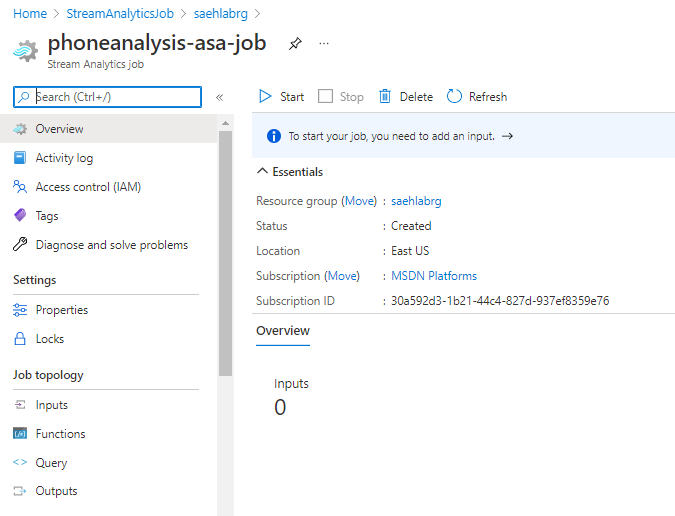


1. Once completed, the **PhoneStream** Input job will appear under the input window. Close the input widow to return to the Resource Group Page

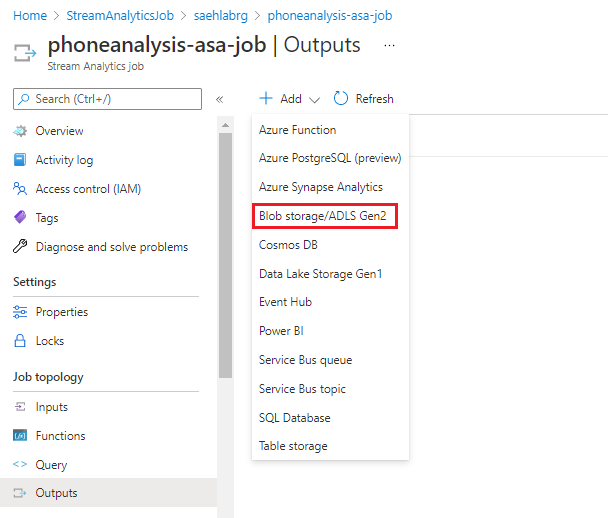


**Task 3: Specify the Stream Analytics job output.**

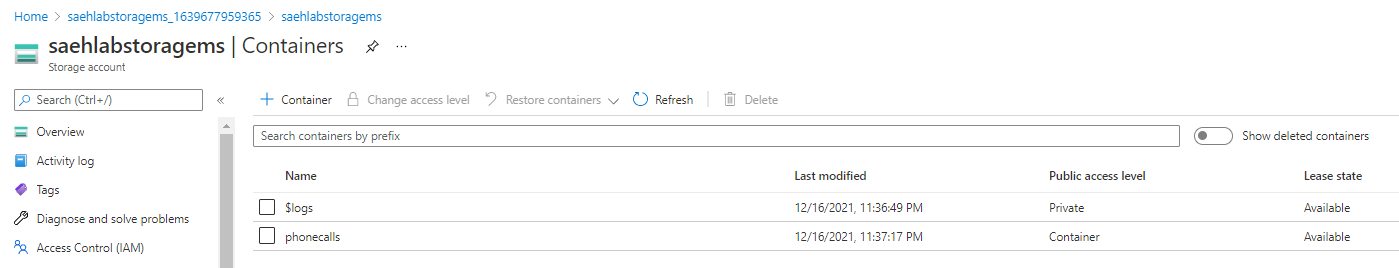
1. Click on **phoneanalysis-asa-job**.
2. In your **phoneanalysis-asa-job** Stream Analytics job window, in the left hand blade, under **Job topology**, click **Outputs**.



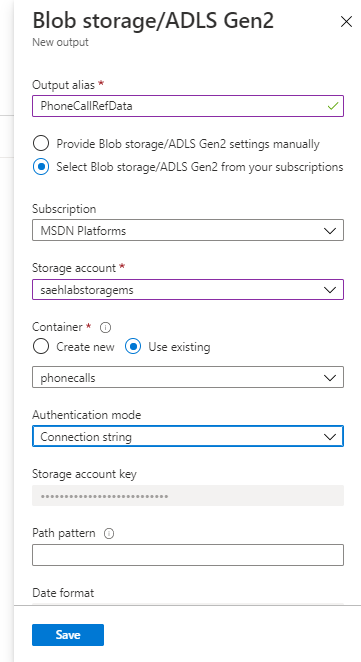
1. In the **Outputs** screen, click **+ Add**, and then click **Blob storage/ADLS Gen2**.



**Note:** Before Selecting **Blob storage/ADLS Gen2**, create a Storage Account as **saehlabstoragems (your unique storage account)** and Container as **phonecalls**



1. In the **Blob storage/ADLS Gen2** window, type or select the following values in the pane:
   * **Output alias**: **PhoneCallRefData**
   * **Select Blob storage/ADLS Gen2 from your subscriptions**: checked
   * **Subscription**: Your subscription name
   * **Storage account**: **saehlabstoragems**
   * **Container**: **Use existing** and select **phonecalls**
   * **Authentication mode**: select **Connection string**
   * Leave the rest of the entries as default values. Finally, click **Save**.



1. Close the output screen to return to the Resource Group page

**Task 4: Defining a Stream Analytics query.**

1. Click on **phoneanalysis-asa-job**.
2. In your **phoneanalysis-asa-job** window, in the **Query** screen in the middle of the window, click on **Edit query**
3. Replace the following query in the code editor:

SELECT

\*

INTO

[YourOutputAlias]

FROM

[YourInputAlias]

Replace with

SELECT System.Timestamp AS WindowEnd, COUNT(\*) AS FraudulentCalls

INTO "PhoneCallRefData"

FROM "PhoneStream" CS1 TIMESTAMP BY CallRecTime

JOIN "PhoneStream" CS2 TIMESTAMP BY CallRecTime

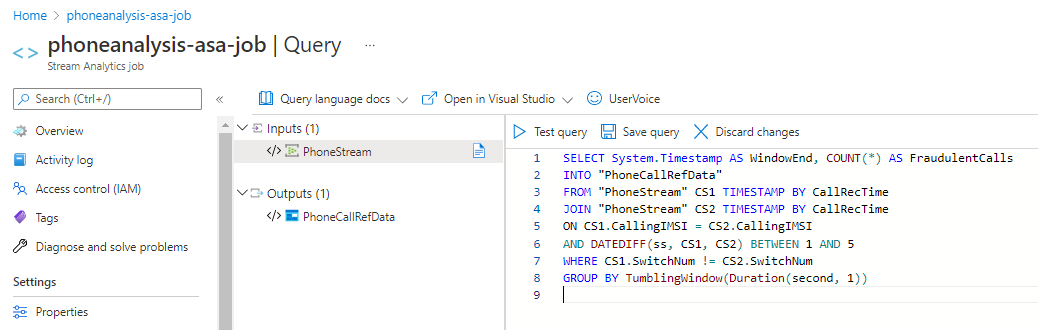
ON CS1.CallingIMSI = CS2.CallingIMSI

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

WHERE CS1.SwitchNum != CS2.SwitchNum

GROUP BY TumblingWindow(Duration(second, 1))

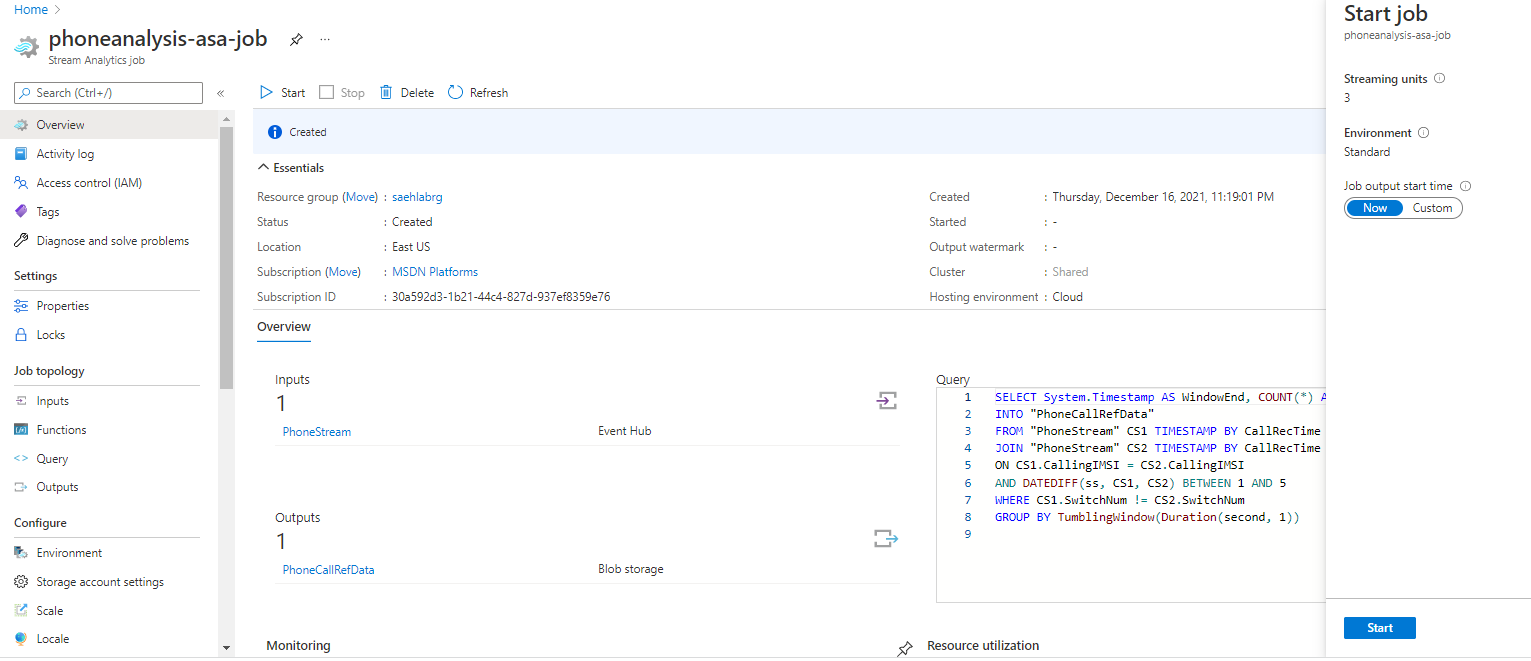
NOTE: This query performs a self-join on a 5-second interval of call data. To check for fraudulent calls, you can self-join the streaming data based on the CallRecTime value. You can then look for call records where the CallingIMSI value (the originating number) is the same, but the SwitchNum value (country/region of origin) is different. When you use a JOIN operation with streaming data, the join must provide some limits on how far the matching rows can be separated in time. Because the streaming data is endless, the time bounds for the relationship are specified within the ON clause of the join using the DATEDIFF function. This query is just like a normal SQL join except for the DATEDIFF function. The DATEDIFF function used in this query is specific to Stream Analytics, and it must appear within the ON...BETWEEN clause.



1. Select **Save Query**.
2. Close the Query window to return to the Stream Analytics job page.

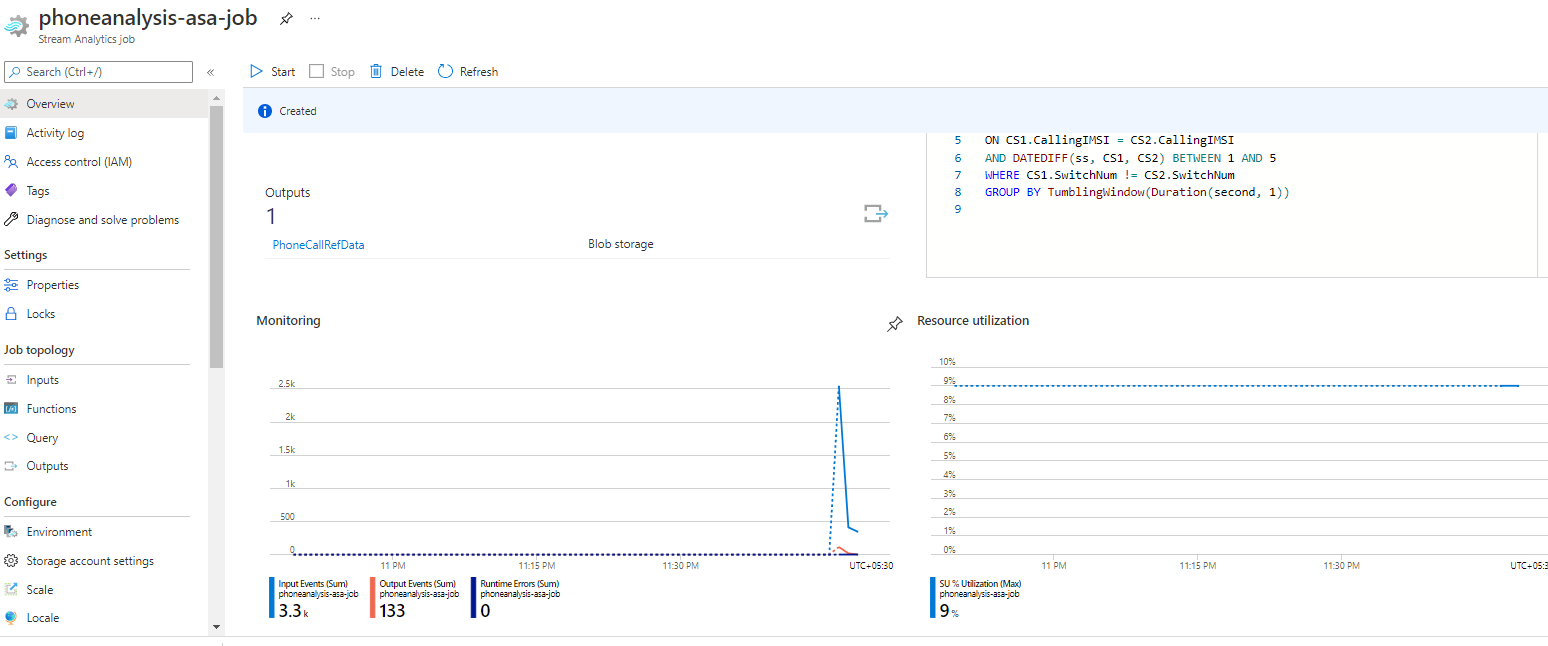
**Task 5: Start the Stream Analytics job**

1. In your **phoneanalysis-asa-job** window, click on **Start**
2. In the **Start Job** dialog box that opens, click **Now**, and then click **Start**.



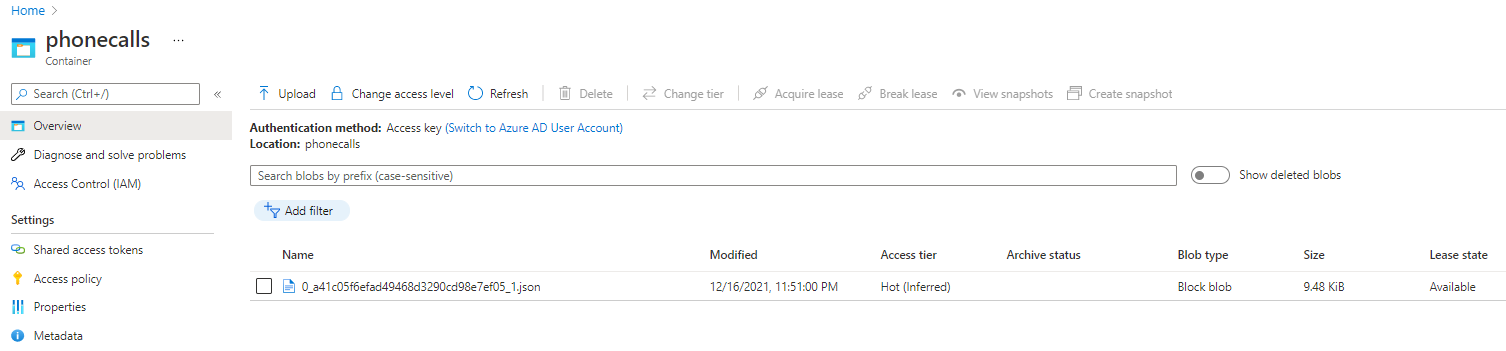
**Note**: In your **phoneanalysis-asa-job** window, a message appears after a minute that the job has started, and the started field changes to the time started

**Note**: Leave this running for 2 minutes so that data can be captured.

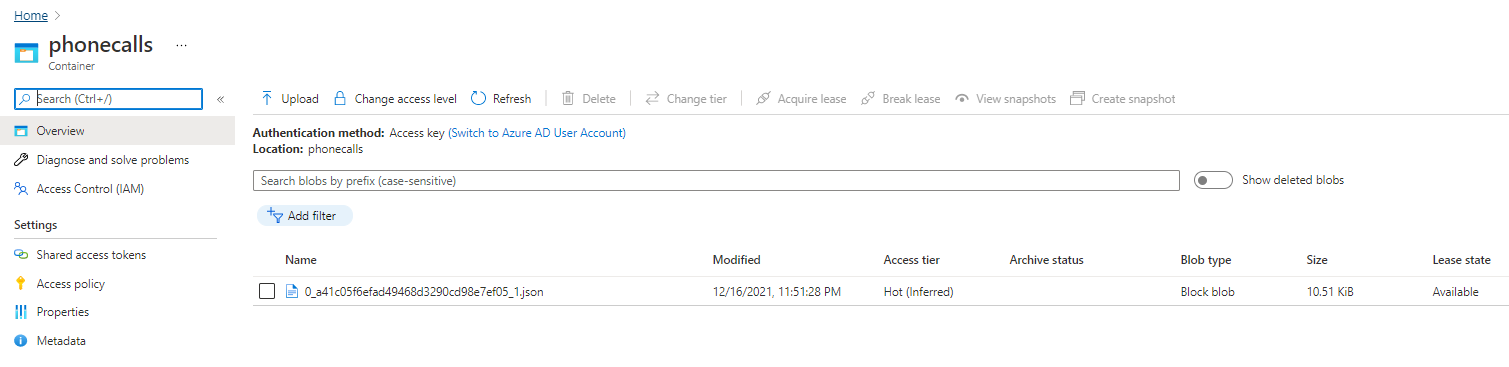


**Task 6: Validate streaming data is collected**

1. In the Azure portal, in the blade, click **Resource groups**, and then click **saehlabrg**, and then click on **saehlabstoragems.**
2. In the Azure portal, click **Containers** box, and then click on the container named **phonecalls**.
3. Confirm that a JSON file appears, and note the size column.



1. Refresh Microsoft Edge, and when the screen has refreshed note the size of the file



**Note**: You could download the file to query the JSON data, you could also output the data to Power BI.

**Result**: After you completed this exercise, you have configured Azure Stream Analytics to collect streaming data into an JSON file store in Azure Blob. You have done this with streaming phone call data.