**March 2018**



**Intelligent Data Lake Workshop**

*Lab 1 – Kinesis Platform*

Table of Contents

[Overview 3](#_Toc507008353)

[Create an Amazon Kinesis Stream 4](#_Toc507008354)

[Create an Amazon Kinesis Firehose Delivery Stream 5](#_Toc507008355)

[Load Data into the Stream 9](#_Toc507008356)

[Analyze Stream Data In Real-time With Amazon Kinesis Analytics 15](#_Toc507008357)

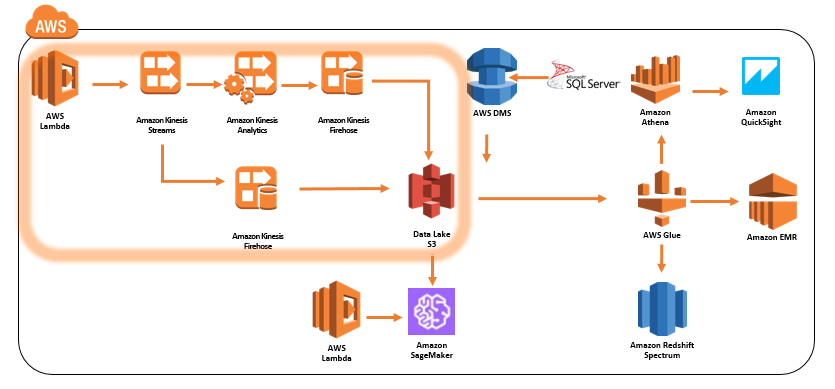
[Catalog Kinesis Analytics Results into Data Lake 22](#_Toc507008358)

[Conclusion 29](#_Toc507008359)

# Overview

In this lab, we will use Kinesis Streams to collect and store the streaming IoT sensor data, then use Kinesis Analytics to process and analyze the streaming data continuously. Specifically, we will use Amazon Kinesis Analytics’ built-in [RANDOM\_CUT\_FOREST](http://docs.aws.amazon.com/kinesisanalytics/latest/sqlref/random-cut-forest.html) function which is a machine learning algorithm to detect anomalies in the streaming data. Finally, we will use Amazon Kinesis Firehose to export both the raw and processed data into S3 for further analysis in the following labs. Diagram below depicts a high-level of what we are trying to accomplish in this lab. One advantage that Amazon Kinesis offers is multiple consumers can read the same stream independently, providing a lot of flexibility to process the data. In our example, we have two consumers, first the Kinesis Analytics will analyze the data in real-time and store the result in S3. The second consumer is Kinesis Firehose that will store the raw data in the stream directly in S3.

Diagram below with highlighted area depicts what you will be building in this lab.



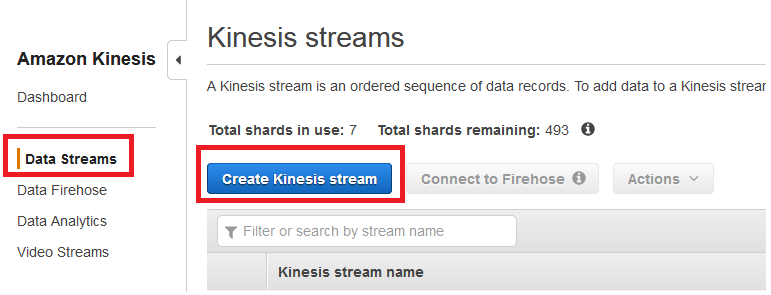
# Create an Amazon Kinesis Stream

1. Please use Chrome or Firefox browser to ensure smooth lab experience.
2. Sign into the AWS Management Console <https://console.aws.amazon.com/>.
3. In the upper-right corner of the AWS Management Console, confirm you are in the desired AWS region (e.g., N. Virginia).

Note, be sure to select a region such as N. Virgina or Oregon that offers Amazon Kinesis Analytics. Also, make sure you are doing all the labs in the same region.

1. Click on **Kinesis** from the list of all services. This will bring you to the Amazon Kinesis dashboard page.
2. On the Kinesis Dashboard, click **Data Stream** on the left panel and then click **Create** **Kinesis Stream**. If you do not see the panel but a welcome page, go ahead and click “**Get Started**”.

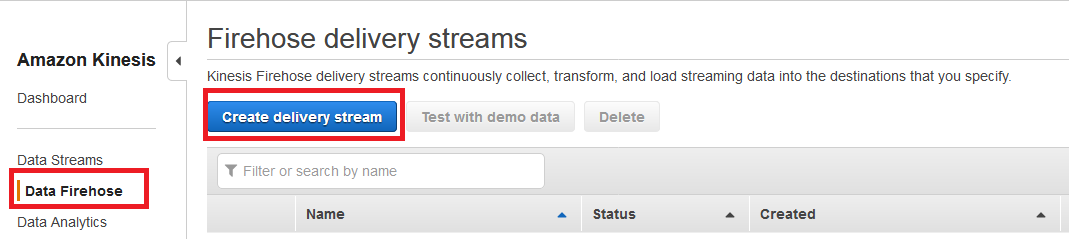




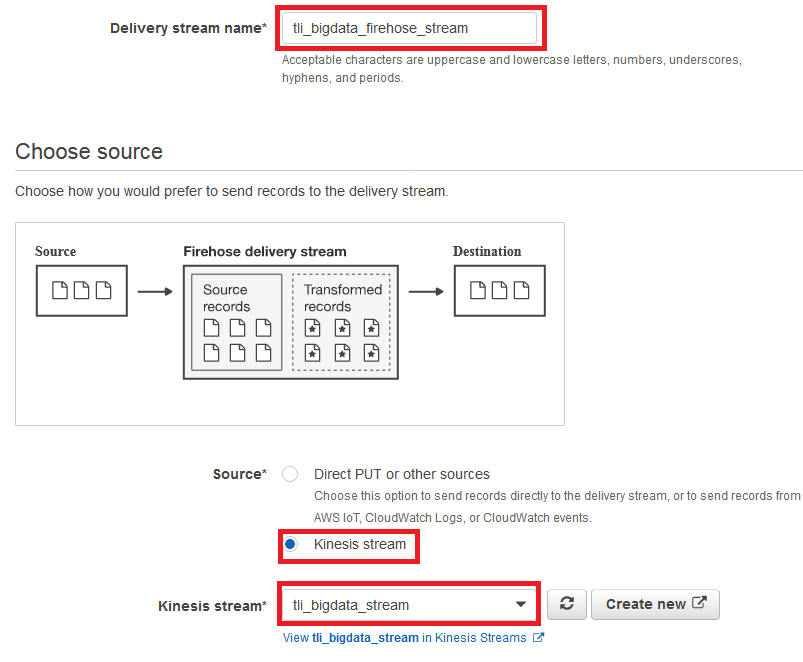
1. For Kinesis stream name, enter **YourInitials\_bigdata\_stream**. Enter **1** for **Number of Shards**.
2. Click **Create Kinesis Stream** in the lower-right corner of the screen.
3. Verify the stream is in the **CREATING** Status and wait for the stream status becomes **ACTIVE.**

# Create an Amazon Kinesis Firehose Delivery Stream

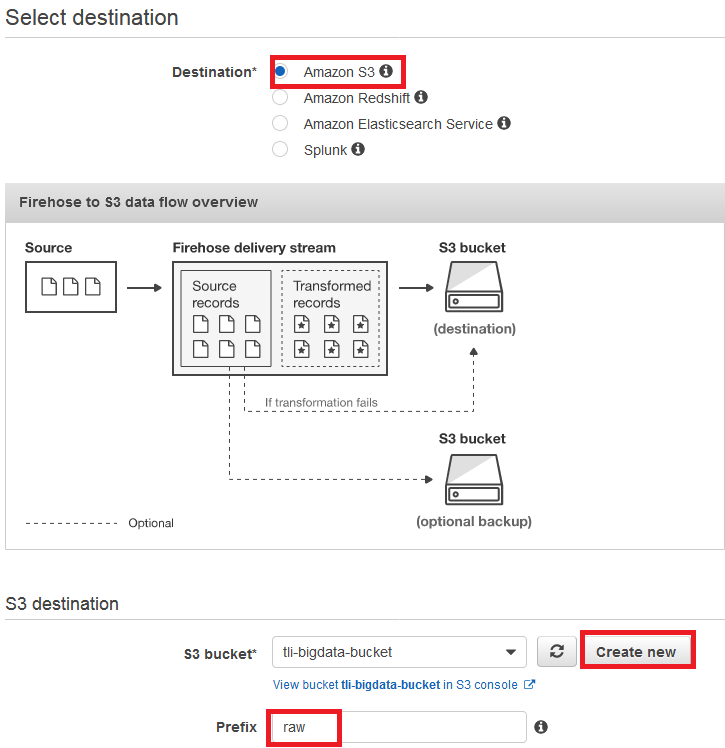
1. Sign into the AWS Management Console <https://console.aws.amazon.com/>.
2. In the upper-right corner of the AWS Management Console, confirm you are in the desired AWS region (e.g., N. Virginia).
3. Click on **Kinesis** from the list of all services. This will bring you to the Amazon Kinesis dashboard page.
4. On the Kinesis Dashboard, click **Data Firehose** on the left panel and then click **Create** **delivery stream**



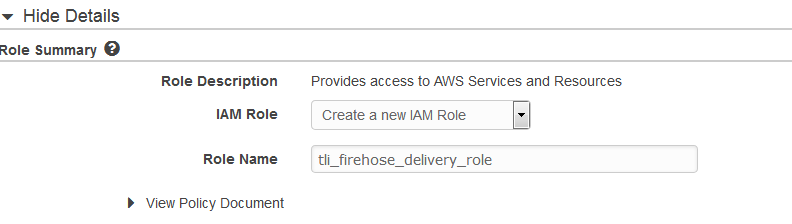
1. For Delivery stream name, enter **YourInitials\_bigdata\_firehose\_stream**. In the Source option field, choose **Kinesis stream** and in the Kinesis stream drop down, select the stream created in previous section.



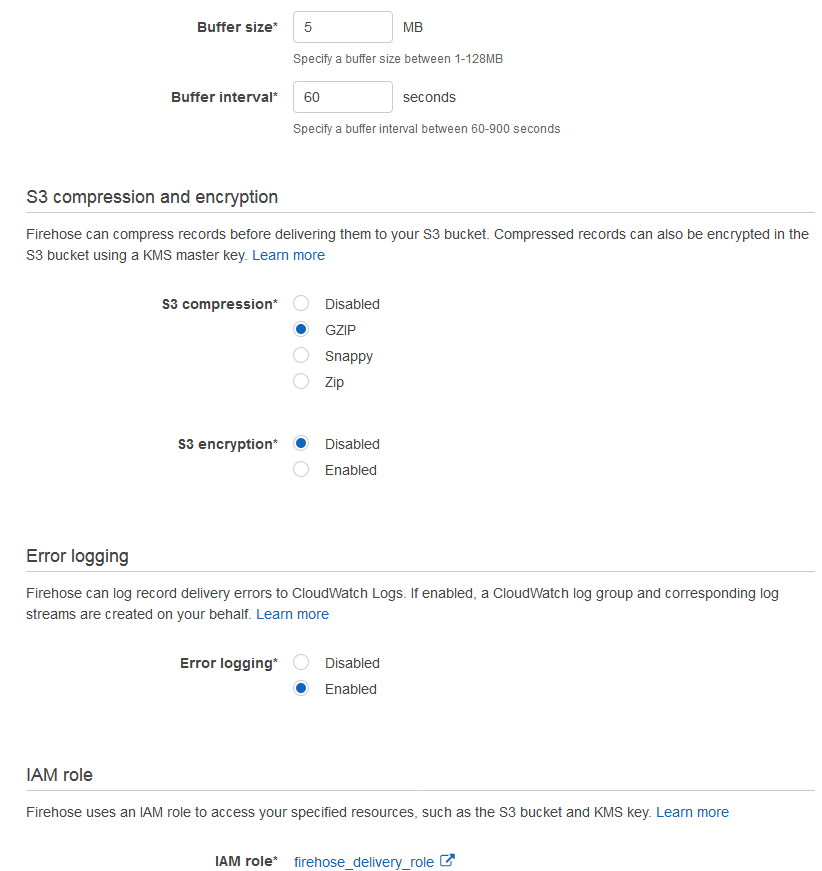
1. Click **Next** in the lower-right corner of the screen.
2. In the Transform records page, choose **Disabled**. Note that Amazon Kinesis Firehose provides the capability through Lambda to transform the source data before loading them into the destination datasource.
3. Click **Next** in the lower-right corner of the screen.
4. Choose **Amazon S3** as the Destination. Click **Create New** to create a new S3 bucket with the name **YourInitials-bigdata-bucket** in **lower** case**,** other wise you will receive an error. In the Prefix field, enter **raw**.



1. Click **Next** in the lower-right corner of the screen.
2. In the Configuration settings page and enter the following
   * Buffer size: 5 MB
   * Buffer interval: 60 seconds (This will shorten the time for data to reach S3)
   * S3 compression: GZIP (Compress data improves performance and save on storage fees)
   * S3 encryption: Disabled
   * Error logging: Enabled
   * IAM role: Click **Create new, or Choose** button to create a new IAM role. Enter **YourInitials\_firehose\_delivery\_role** for the Role Name. Click **Allow** to go back to the configuration settings page.



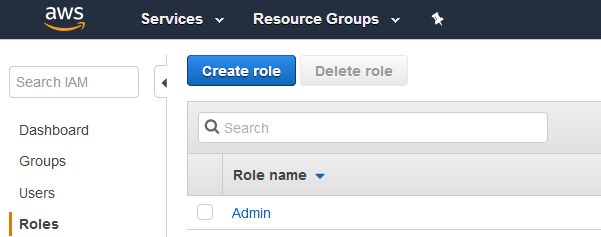
1. Final configuration should look similar to the following screenshot



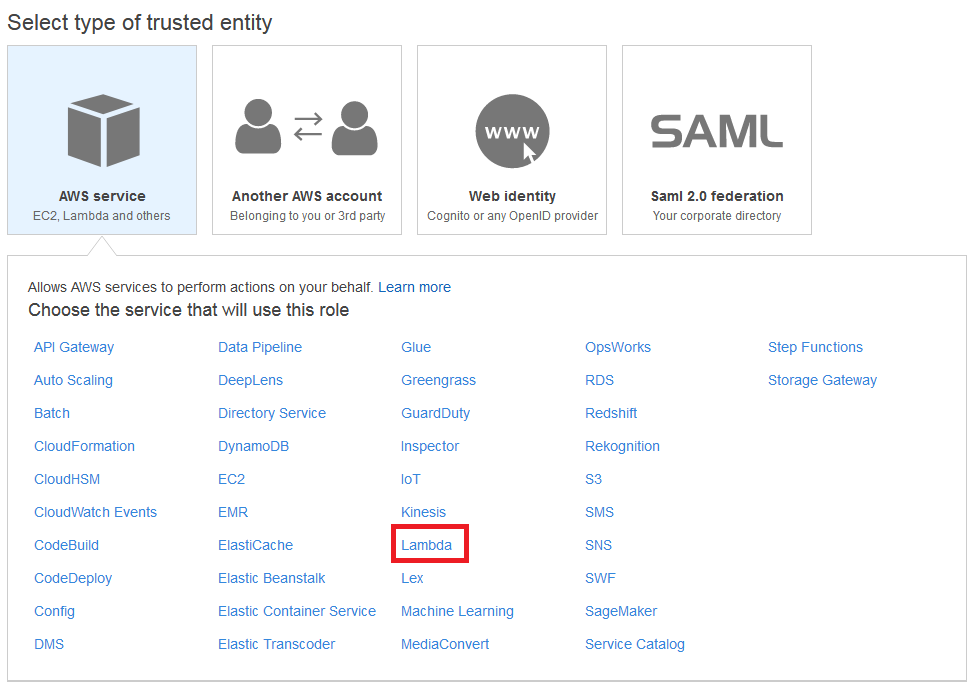
1. Click **Next** in the lower-right corner of the screen.
2. Review everything and once satisfied, click **Create delivery stream.**

# Load Data into the Stream

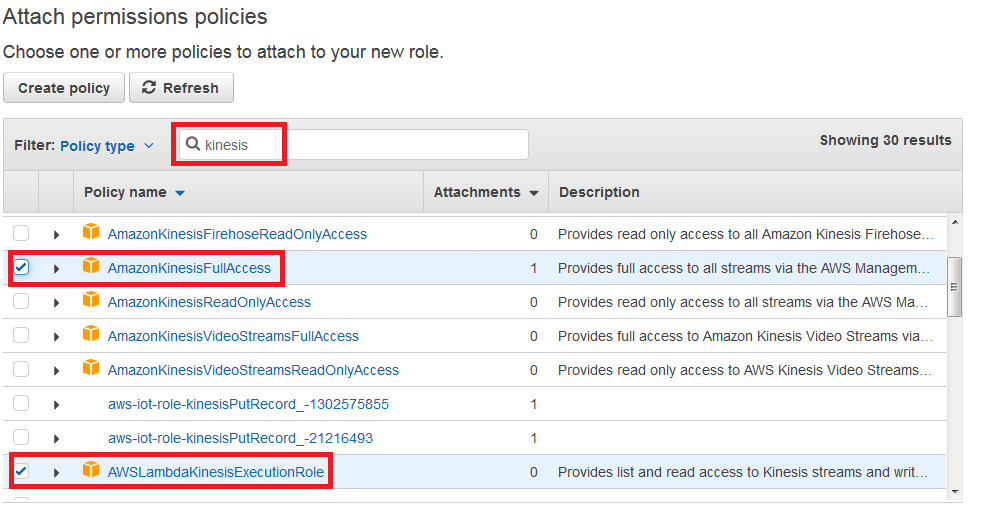
1. We will create a simple Lambda function as a simulator to put 50K records into the Kinesis stream but first we need to create a Role for the Lambda function to have proper permissions.
2. Sign into the AWS Management Console <https://console.aws.amazon.com/>.
3. In the upper-right corner of the AWS Management Console, confirm you are in the desired AWS region (e.g., N. Virginia).
4. Click on **IAM** from the list of all services. This will bring you to the IAM dashboard page.
5. Click **Roles** on the left hand panel and then click **Create role**



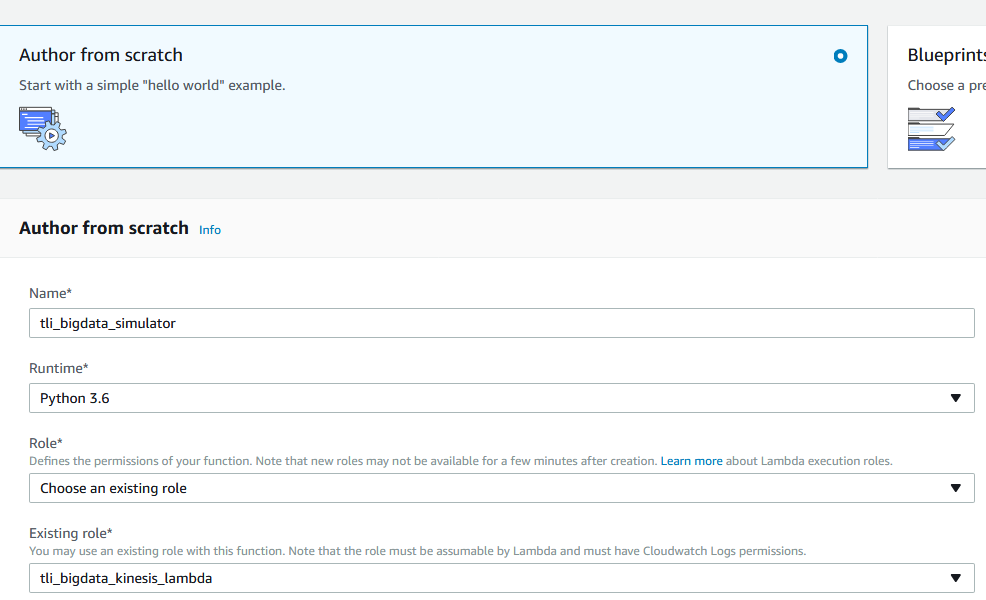
1. Select **Lambda**



1. Click **Next: Permissions**
2. In attach permission policies page, search for Kinesis and select **AmazonKinesisFullAccess** and **AWSLambdaKinesisExecutionRole**.



1. Click **Next: Review**
2. Enter **YourInitials\_bigdata\_kinesis\_lambda** for the Role Name and click **Create Role**
3. Go back to AWS Management Console <https://console.aws.amazon.com/>.
4. In the upper-right corner of the AWS Management Console, confirm you are in the desired AWS region (e.g., N. Virginia).
5. Click on **Lambda** from the list of all services. This will bring you to the AWS Lambda dashboard page.
6. On the Lambda Dashboard, click **Create Function**
7. Select **Author from scratch** and enter the following
   * Name: **YourInitials\_bigdata\_simulator**
   * Runtime: Python 3.6
   * Role: Choose an existing role
   * Exiting Role: **YourInitials\_bigdata\_kinesis\_lambda**



1. Click **Create function**
2. In code editor, copy and paste the following code, be sure to change variable **bigdataStreamName** to the stream created with your initials.

import json

import datetime

import random

import boto3

kinesis = boto3.client('kinesis', region\_name='us-east-1') #<--- change region if not in N.Virginia

def getData(sensorType, sensorName, lowVal, highVal):

data = {}

data['sensorType'] = sensorType

data['sensorName'] = sensorName

data['sensorValue'] = random.randint(lowVal, highVal)

return data

def lambda\_handler(event, context):

bigdataStreamName = **"YourInitials\_bigdata\_stream"**

count = 1

for \_ in range(0, 50000):

rnd = random.random()

if (rnd < 0.01):

if count==1:

data = json.dumps(getData(count, 'ShaftSpeedSensor', 18000, 20000))

kinesis.put\_record(StreamName=bigdataStreamName, Data=data, PartitionKey="ShaftSpeedSensor")

elif count==2:

data = json.dumps(getData(count, 'ChamberPressureSensor', 4200, 5000))

kinesis.put\_record(StreamName=bigdataStreamName, Data=data, PartitionKey="ChamberPressureSensor")

elif count==3:

data = json.dumps(getData(count, 'DischargeTempSensor', 3500, 3800))

kinesis.put\_record(StreamName=bigdataStreamName, Data=data, PartitionKey="DischargeTempSensor")

elif count==4:

data = json.dumps(getData(count, 'FuelOxidizerMixRatioSensor', 5, 10))

kinesis.put\_record(StreamName=bigdataStreamName, Data=data, PartitionKey="FuelOxidizerMixRatioSensor")

count = 1 if count>= 4 else count+1

else:

if count==1:

#13,800 rpm avg

data = json.dumps(getData(count, 'ShaftSpeedSensor', 13000, 14000))

kinesis.put\_record(StreamName=bigdataStreamName, Data=data, PartitionKey="ShaftSpeedSensor")

elif count==2:

#3,722 psi avg

data = json.dumps(getData(count, 'ChamberPressureSensor', 3500, 3800))

kinesis.put\_record(StreamName=bigdataStreamName, Data=data, PartitionKey="ChamberPressureSensor")

elif count==3:

#3,127c avg

data = json.dumps(getData(count, 'DischargeTempSensor', 3000, 3200))

kinesis.put\_record(StreamName=bigdataStreamName, Data=data, PartitionKey="DischargeTempSensor")

elif count==4:

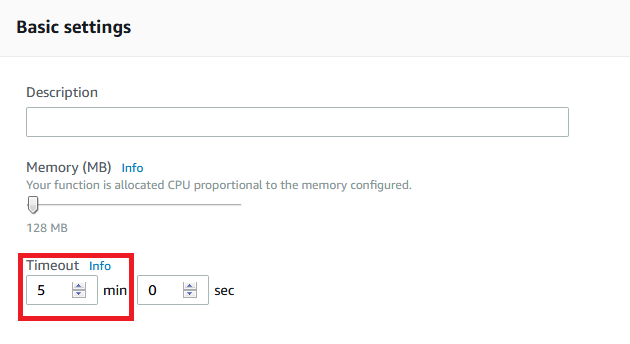
#3 avg

data = json.dumps(getData(count, 'FuelOxidizerMixRatioSensor', 2, 4))

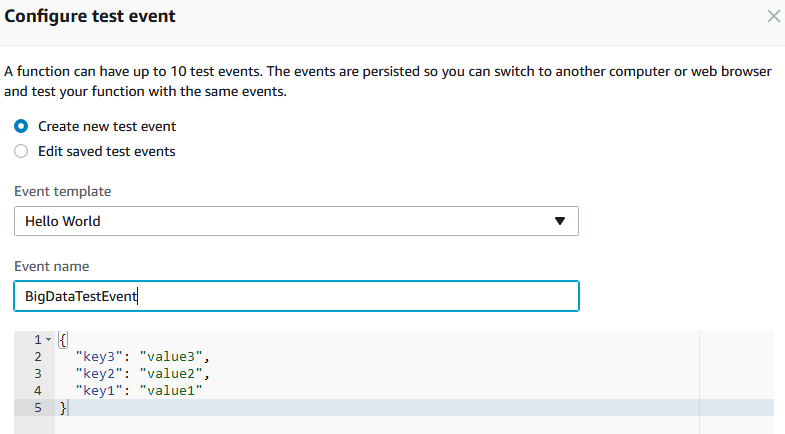
kinesis.put\_record(StreamName=bigdataStreamName, Data=data, PartitionKey="FuelOxidizerMixRatioSensor")

return "complete"

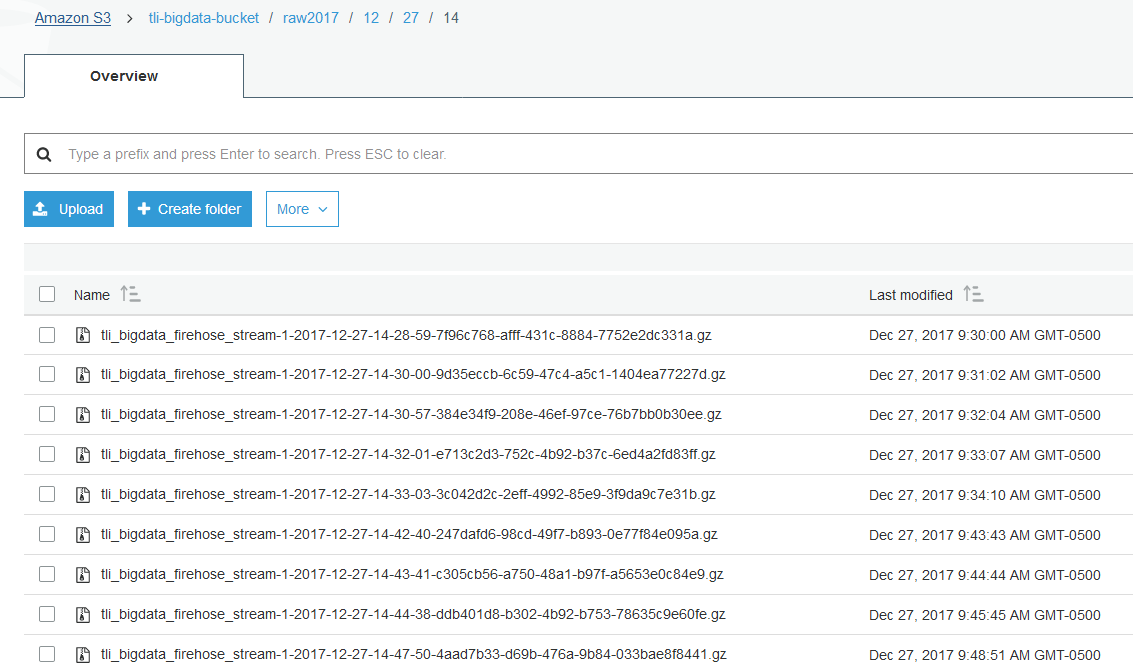
1. Leave everything on the page default except the Timeout value, change it from 3 seconds to 5 minutes



1. Click **Save** on the top right hand corner of the screen and then click **Test**. Since we are not providing any parameter or input values, leave everything default, give it a name, and click **Create**.



1. The function will run 5 minutes to put 50k records into the Kinesis stream. Keep this tab open as we may need to come back and generate more data later. Note you may get a timeout error, this is normal as the function timed out (5 mins) before it could generate all 50k records. Continue to next step.
2. So far, we have a Kinesis stream and we have created the Lambda function to put 50k records into the stream. We also setup Kinesis Firehose to retrieve the data in the stream and store them in compressed GZIP format in a S3 bucket. To verify everything is working, we can go to the S3 bucket and verify the data files exist. Note Kinesis Firehose stores data in a year/month/date folder.

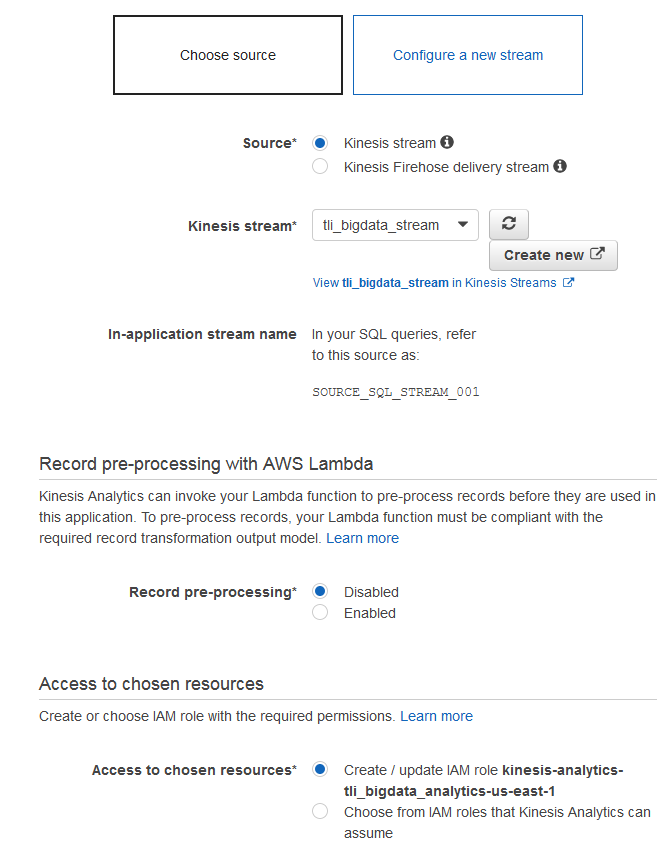


# Analyze Stream Data In Real-time With Amazon Kinesis Analytics

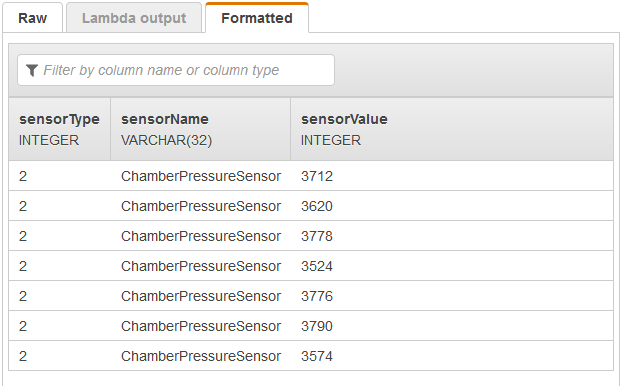
1. Sign into the AWS Management Console <https://console.aws.amazon.com/>.
2. In the upper-right corner of the AWS Management Console, confirm you are in the desired AWS region (e.g., N. Virginia).

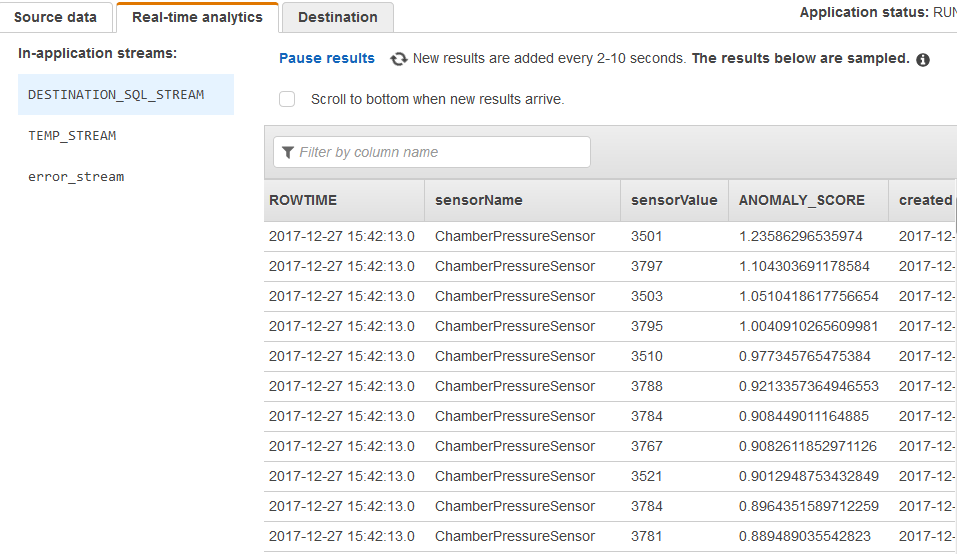
Note, be sure to select a region such as N. Virgina or Oregon that offers Amazon Kinesis Analytics. Also, make sure you are doing all the labs in the same region.

1. Click on **Kinesis** from the list of all services. This will bring you to the Amazon Kinesis dashboard page.
2. On the Kinesis Dashboard, click **Data Analytics** on the left panel and then click **Create** **application**
3. For application name, enter **YourInitials\_bigdata\_analytic** and click **Create application**
4. Click **Connect to a source**
5. Select **Choose source** option, **Kinesis stream** as source and select the big data stream created earlier. Leave **Record pre-processing** disabled. Choose **Create/Update IAM role** to create proper permissions to access the stream.



1. Click **Discover schema**. Note if you get an error stating there is not enough data in stream, go back to the Lambda tab/page, and click on Test again to generate some fresh data and click **Retry schema discovery**. You should now see the formatted data.



1. Click **Save and continue**
2. Click **Go to SQL editor**. If it prompts to start running the analytics application, click **Yes, start application**
3. Delete everything in the SQL editor, then copy and paste the following SQL into the editor. Click **Save and run SQL**.
4. You can now see the data are being processed real time by Kinesis Analytics. Note if you see no rows arriving in the analytic table, then go back to the Lambda tab/page, and click on Test again to generate some fresh data. The SQL script above uses built-in Random\_Forest\_Cut function to calculate a score based on the data streaming in for each sensor. As the Lambda generates a value that is higher than normal range, the score would be much higher as well. 

CREATE OR REPLACE STREAM "TEMP\_STREAM" (

"sensorName" varchar (40),

"sensorValue" integer,

"ANOMALY\_SCORE" DOUBLE);

-- Creates an output stream and defines a schema

CREATE OR REPLACE STREAM "DESTINATION\_SQL\_STREAM" (

"sensorName" varchar(40),

"sensorValue" integer,

"ANOMALY\_SCORE" DOUBLE,

"created" TimeStamp);

-- Compute an anomaly score for each record in the source stream

-- using Random Cut Forest

CREATE OR REPLACE PUMP "STREAM\_PUMP\_1" AS INSERT INTO "TEMP\_STREAM"

SELECT STREAM "sensorName", "sensorValue", ANOMALY\_SCORE FROM

TABLE(RANDOM\_CUT\_FOREST(

CURSOR(SELECT STREAM \* FROM "SOURCE\_SQL\_STREAM\_001" where "sensorType"=1)

)

);

CREATE OR REPLACE PUMP "STREAM\_PUMP\_2" AS INSERT INTO "TEMP\_STREAM"

SELECT STREAM "sensorName", "sensorValue", ANOMALY\_SCORE FROM

TABLE(RANDOM\_CUT\_FOREST(

CURSOR(SELECT STREAM \* FROM "SOURCE\_SQL\_STREAM\_001" where "sensorType"=2)

)

);

CREATE OR REPLACE PUMP "STREAM\_PUMP\_3" AS INSERT INTO "TEMP\_STREAM"

SELECT STREAM "sensorName", "sensorValue", ANOMALY\_SCORE FROM

TABLE(RANDOM\_CUT\_FOREST(

CURSOR(SELECT STREAM \* FROM "SOURCE\_SQL\_STREAM\_001" where "sensorType"=3)

)

);

CREATE OR REPLACE PUMP "STREAM\_PUMP\_4" AS INSERT INTO "TEMP\_STREAM"

SELECT STREAM "sensorName", "sensorValue", ANOMALY\_SCORE FROM

TABLE(RANDOM\_CUT\_FOREST(

CURSOR(SELECT STREAM \* FROM "SOURCE\_SQL\_STREAM\_001" where "sensorType"=4)

)

);

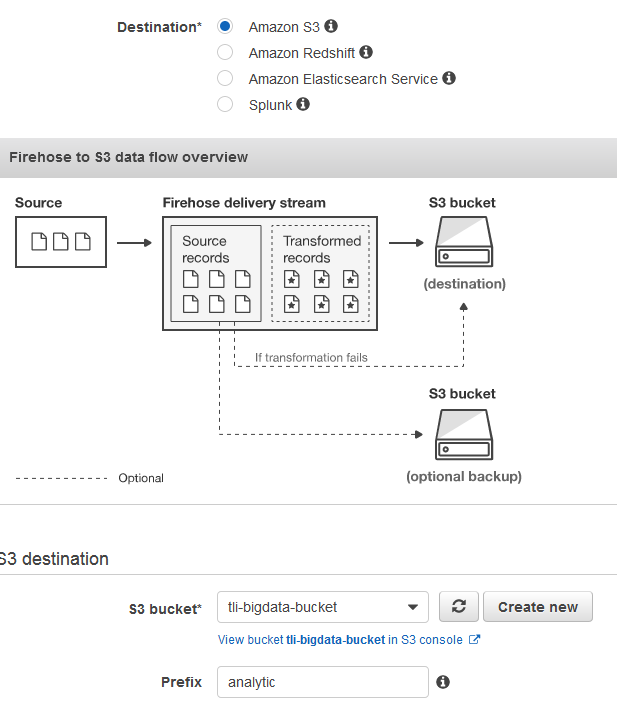
-- Sort records by descending anomaly score, insert into output stream

CREATE OR REPLACE PUMP "OUTPUT\_PUMP" AS INSERT INTO "DESTINATION\_SQL\_STREAM"

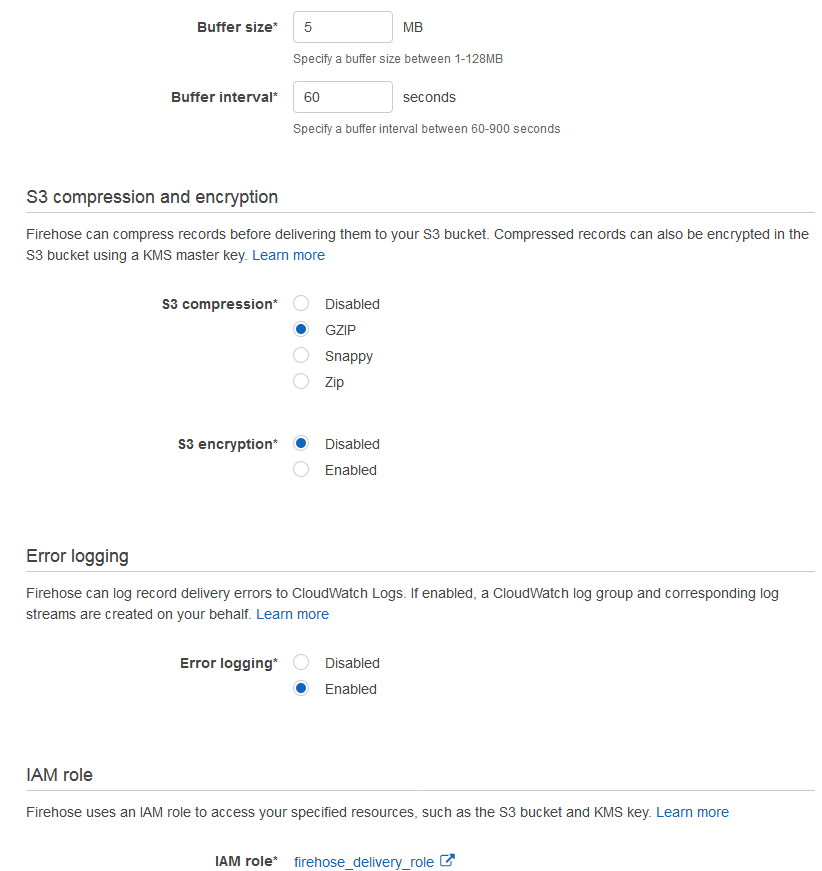
SELECT STREAM "sensorName", "sensorValue", ANOMALY\_SCORE, ROWTIME FROM "TEMP\_STREAM"

ORDER BY FLOOR("TEMP\_STREAM".ROWTIME TO SECOND), ANOMALY\_SCORE DESC;

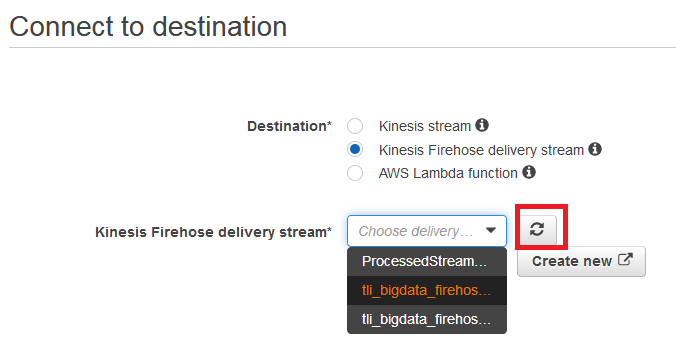
1. Click **Exit (done editing)**
2. Click **Connect to a destination**
3. Choose **Kinesis Firehose delivery stream** option and click **Create new**
4. For Delivery stream name, enter **YourInitials\_bigdata\_firehose\_analytic**
5. For Source, choose **Direct Put or other sources**
6. Click **Next**
7. For Record transformation, leave it as **Disabled**
8. Click **Next**
9. For Destination, choose **Amazon S3**. Select the same big data S3 bucket created earlier with Prefix **analytic**



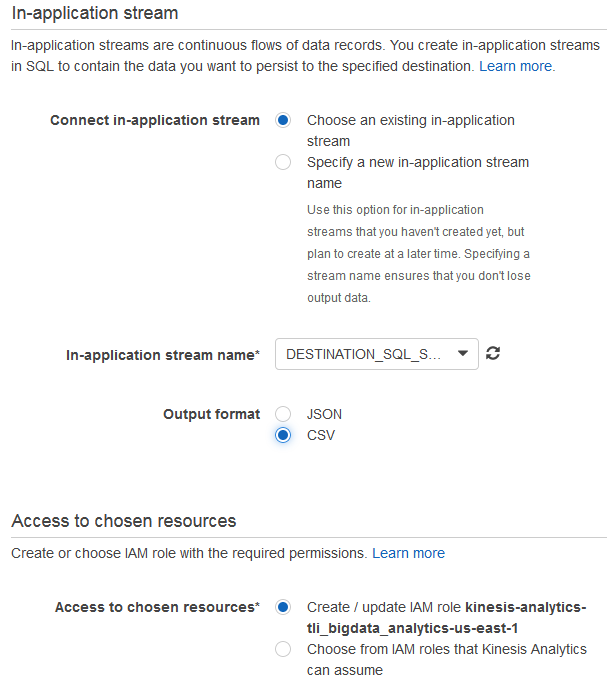
1. Click **Next**
2. In the Configuration settings page and enter the following
   * Buffer size: 5 MB
   * Buffer interval: 60 seconds (This will shorten the time for data to reach S3)
   * S3 compression: GZIP (Compress data improves performance and save on storage fees)
   * S3 encryption: Disabled
   * Error logging: Enabled
   * IAM role: Click **Create new, or Choose** button to create a new IAM role. Use the same role created earlier and click **Allow** to go back to the configuration settings page.
3. Final configuration should look similar to the following screenshot



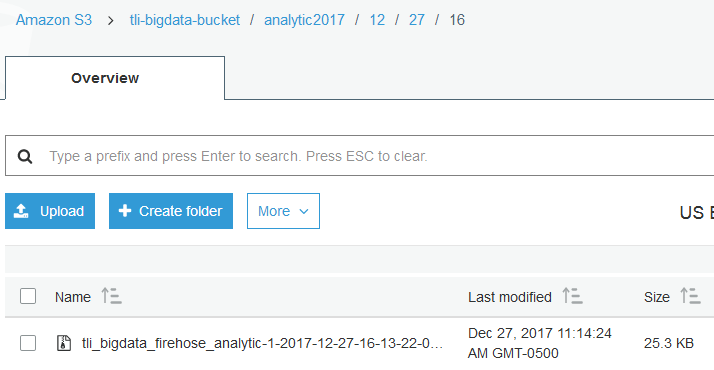
1. Click **Next** in the lower-right corner of the screen.
2. Review everything and once satisfied, click **Create delivery stream.**
3. Close the tab or page to go back to the Data Analytics application by selecting it from the dashboard and clicking on ‘Application Details’ button. Then under the Connect to destination page, click on the refresh button and select the newly created Firehose stream, **YourInitials\_bigdata\_firehose\_analytic**.



1. For Connect in-application stream, **choose an exiting in-application stream**. For In-application stream name, select **DESTINATION\_SQL\_STREAM**. For Output format, choose **CSV**.



1. Click **Save and continue**
2. The Kinesis Analytics application is now setup and running. It will monitoring the incoming Kinesis stream and analyze the data in real-time, then utilize Kinesis Firehose to save the processed output with ANOMALY\_SCORE as CSV to the S3 bucket.
3. Click **Exit to Kinesis Analytics applications**
4. Go back to the Lambda tab/page, and click on Test again to generate some fresh data. Wait for a few minutes and verify that the processed data are saved in the S3 bucket, under the ‘analytic’ prefix.



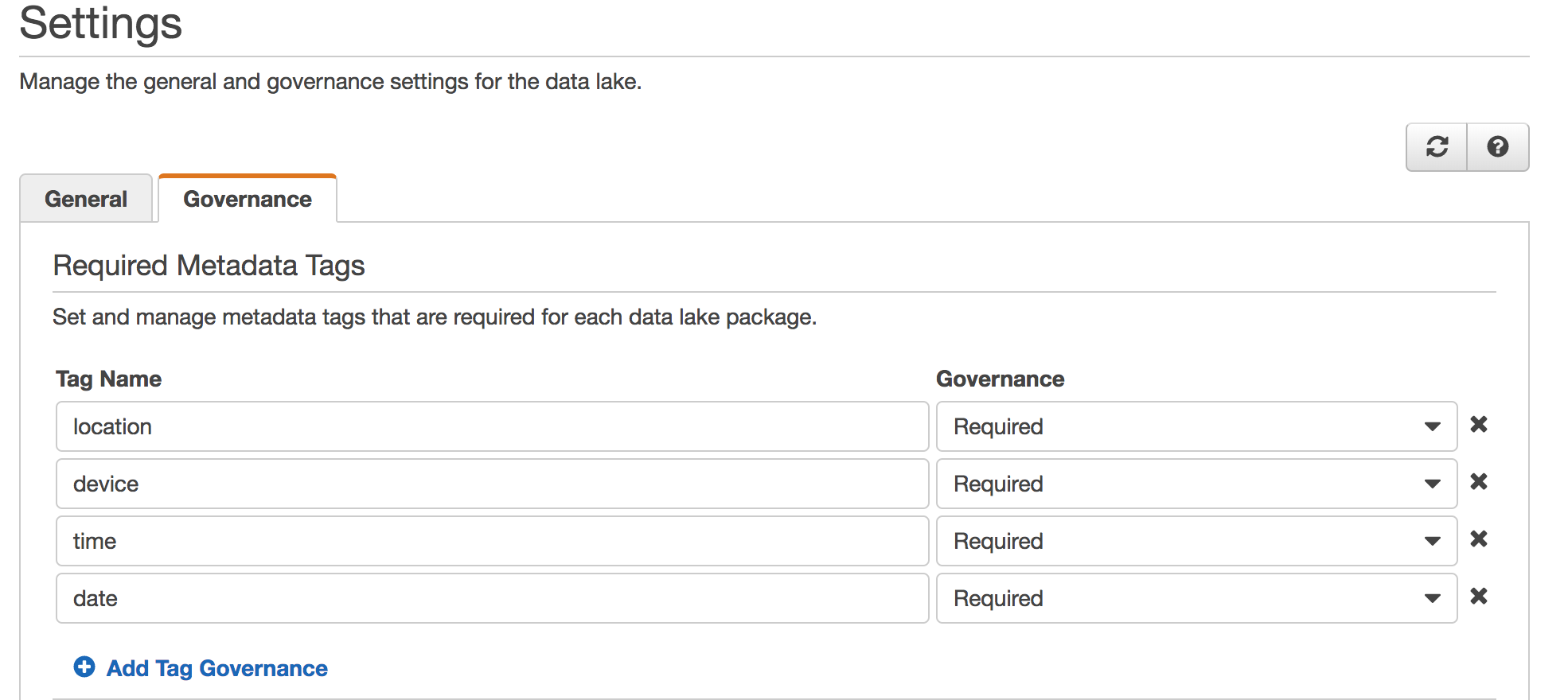
# Catalog Kinesis Analytics Results into Data Lake

In the first lab, you have created a Data lake in your AWS account. In this section, we will use the Data lake UI console to create a package to store the results of the Kinesis Analytics, tag them with some appropriate keys and search the package using the search keywords. This way, different users of the data lake will be able to search the Data lake and download the content of their interest across the organization.

Log into your Data lake console as described in the first lab.

**Create Tags for Governance:**

1. From the Dashboard (home page) of the Data lake console, click on ‘**Settings**’ under Administration from the left hand side menu. On the governance tab, create four tags: location, device, time, date. For each tag, select ‘Required’ in the Governance column.



**Create a Package:**

1. Click on ‘**Create a Package**’ link from the left hand side menu. The tags which you made required in the previous step appear as ‘required’ inputs on this screen. Enter the following:

**Package Name:** IOT-Sensor-Data

**Description:** Data coming from different IOT sensors across different geo locations

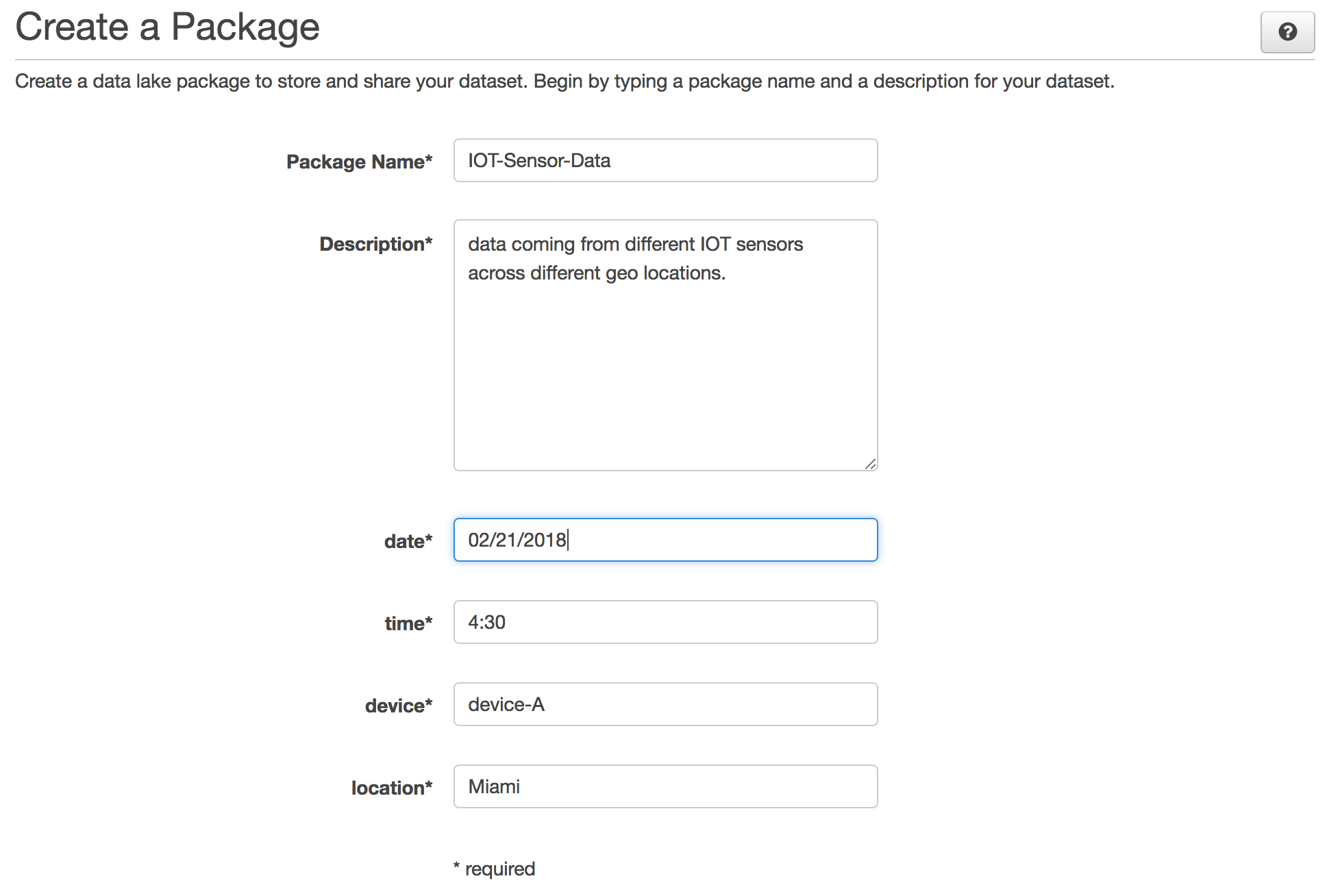
**date:** (today’s date in the format *MM/DD/YY*)

**time**: (the current time in the format *HH:MM*)

**device:** device-A

**location:** Miami

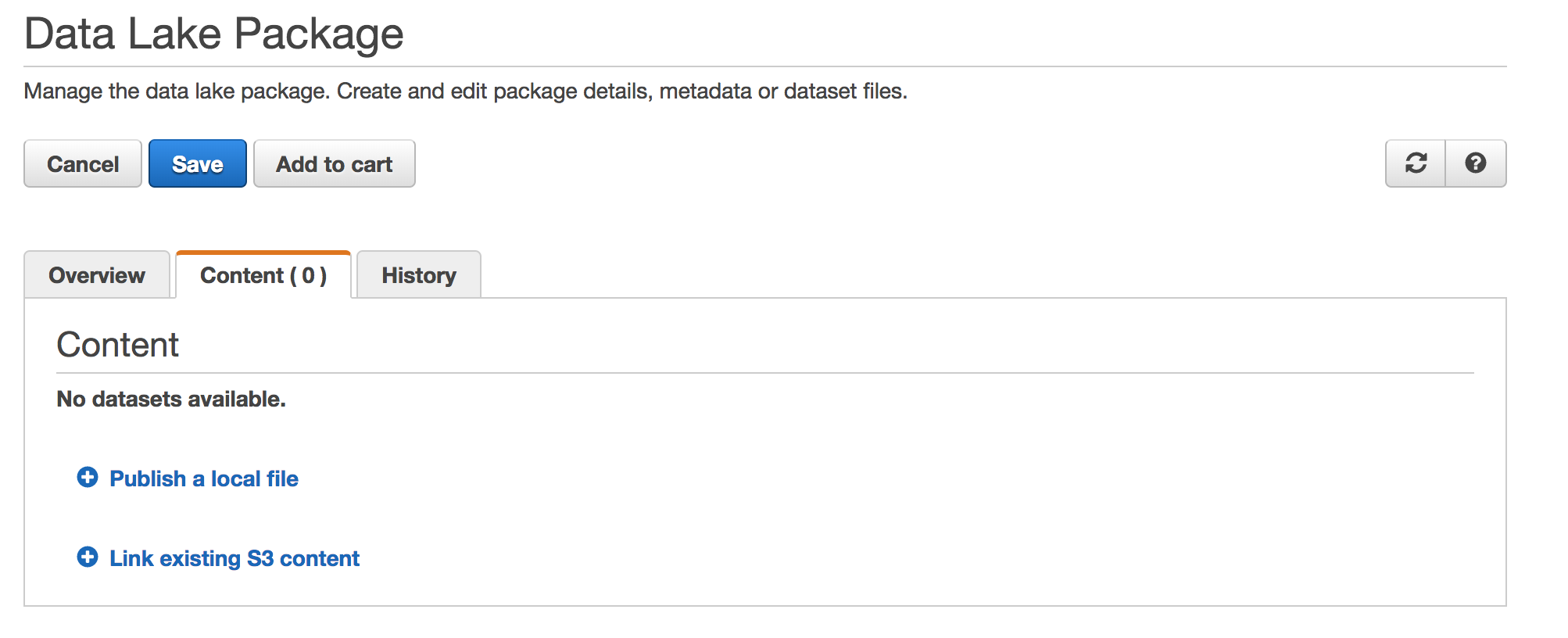
1. Click ‘**Create Package**’



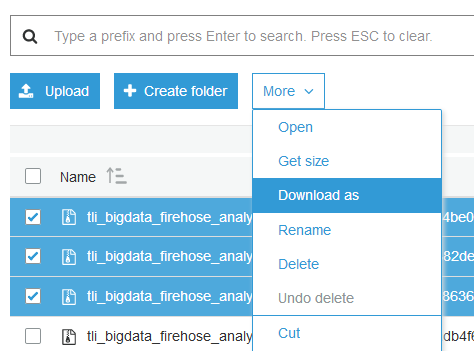
1. Go to the Content tab of your create package. You have two options to associate S3 data with this package –

* Publish a local file to S3
* Link existing S3 files to the Package

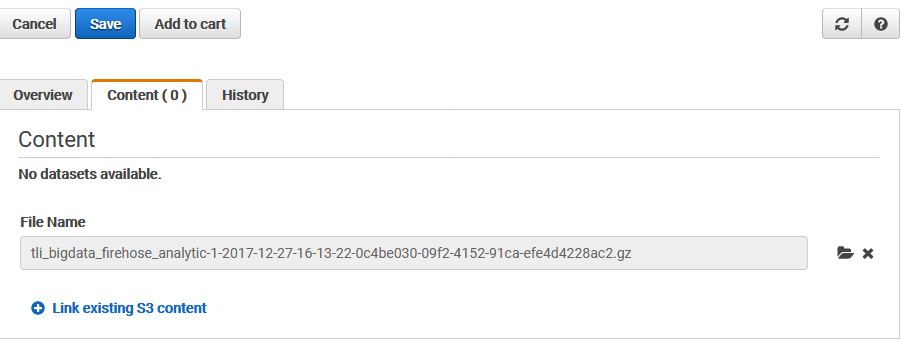
Since we already have our results from the Kinesis Analytics stream in S3 bucket, lets choose ‘Publish a loca file” option.



1. In order to publish a local file, you need download the S3 files that are generated by the Kinesis Analytics application.
2. Open a new tab and go to the S3 console, find **YourInitials-bigdata-bucket** and verify the analytics files exist. Note Kinesis Analytics stores data in a year/month/date folder in the analytics folder of your S3 bucket. Download a few files by selecting them first and then click **“More”** and then “**Download as**”.



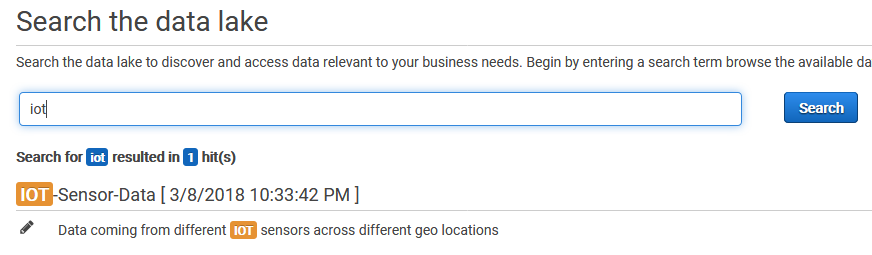
1. Follow the instruction in the pop up dialog and save the file on your workstation’s desktop.
2. Go back to the DataLake tab and click “**Publish a local file**”. Browse and upload the file you downloaded in step 5. Click “**Save**”



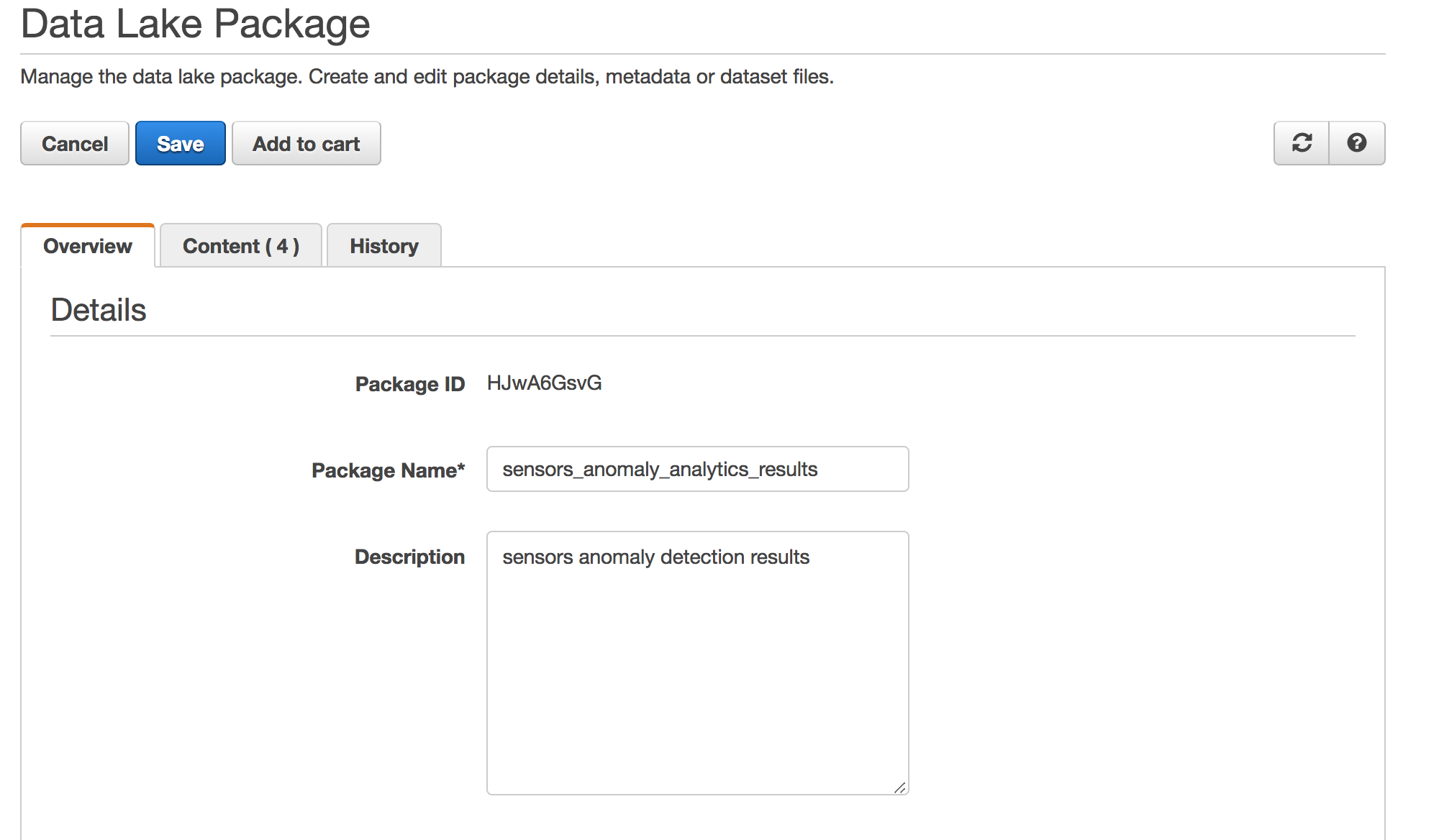
At this point, you have successfully cataloged with your analytic data in the Data lake.

**Search a Package –**

1. Click on the ‘Search’ link from the left hand side menu.
2. Specify some search keywords, for example ‘**iot**, to find a package in the Data lake –

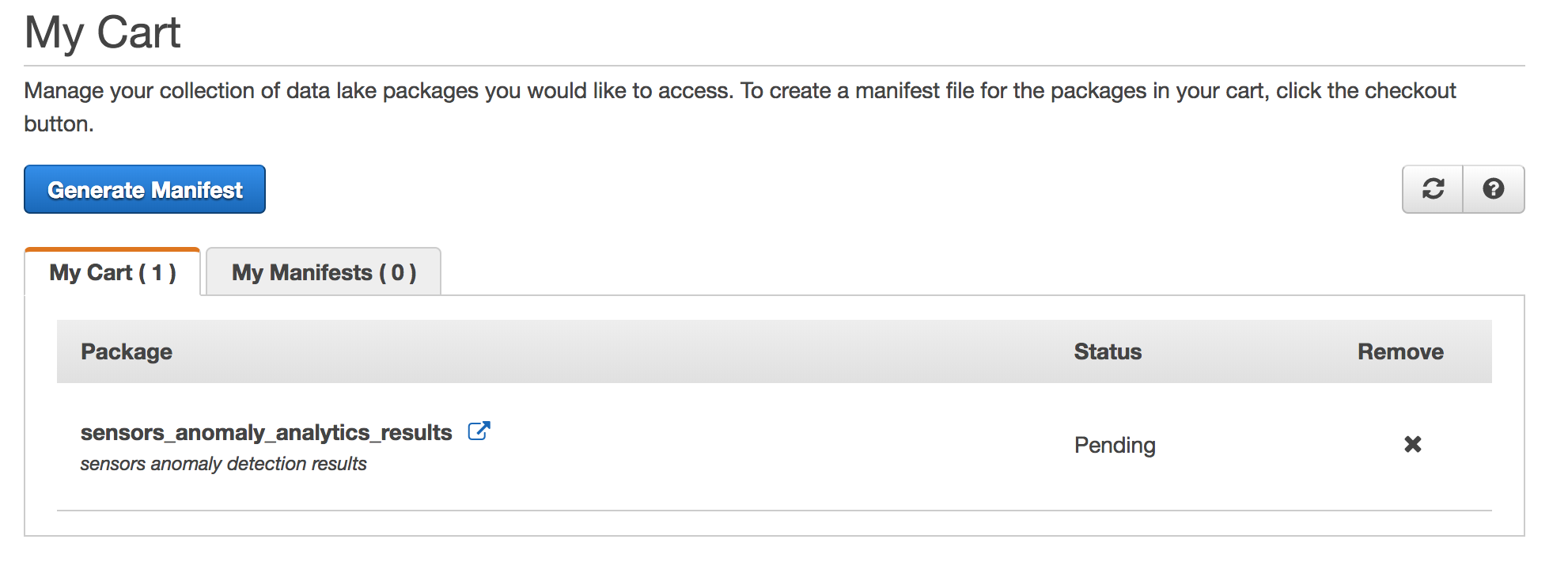


1. From the search results page, click on the pencil icon. This will take you to the package details where you can see the contents associated with this package. If this is the right package you were looking for, you can start the checkout process in order to download its contents.
2. Click on ‘Add to Cart’ button.

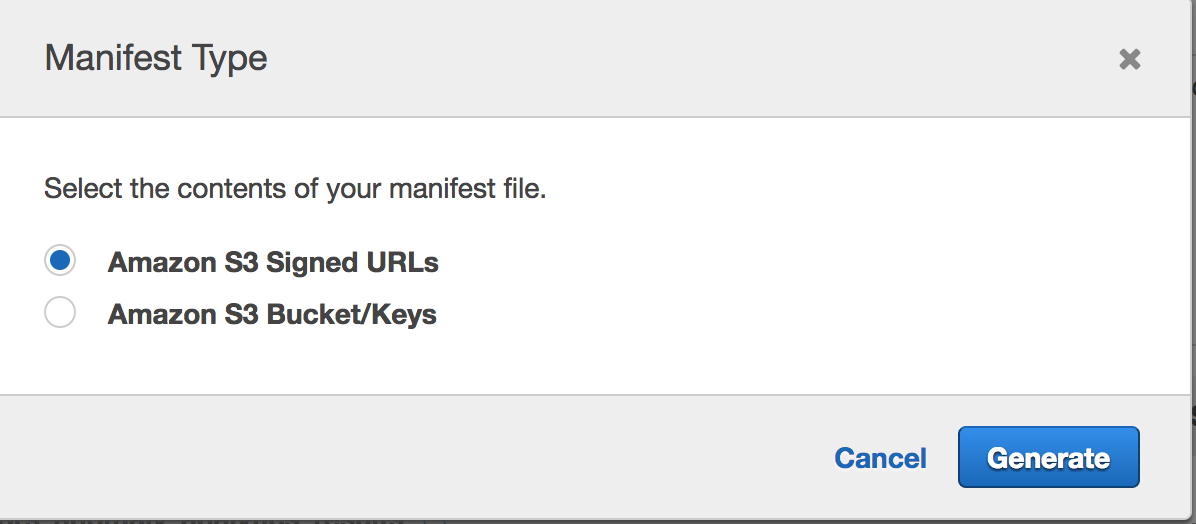


This will add this package to your cart.

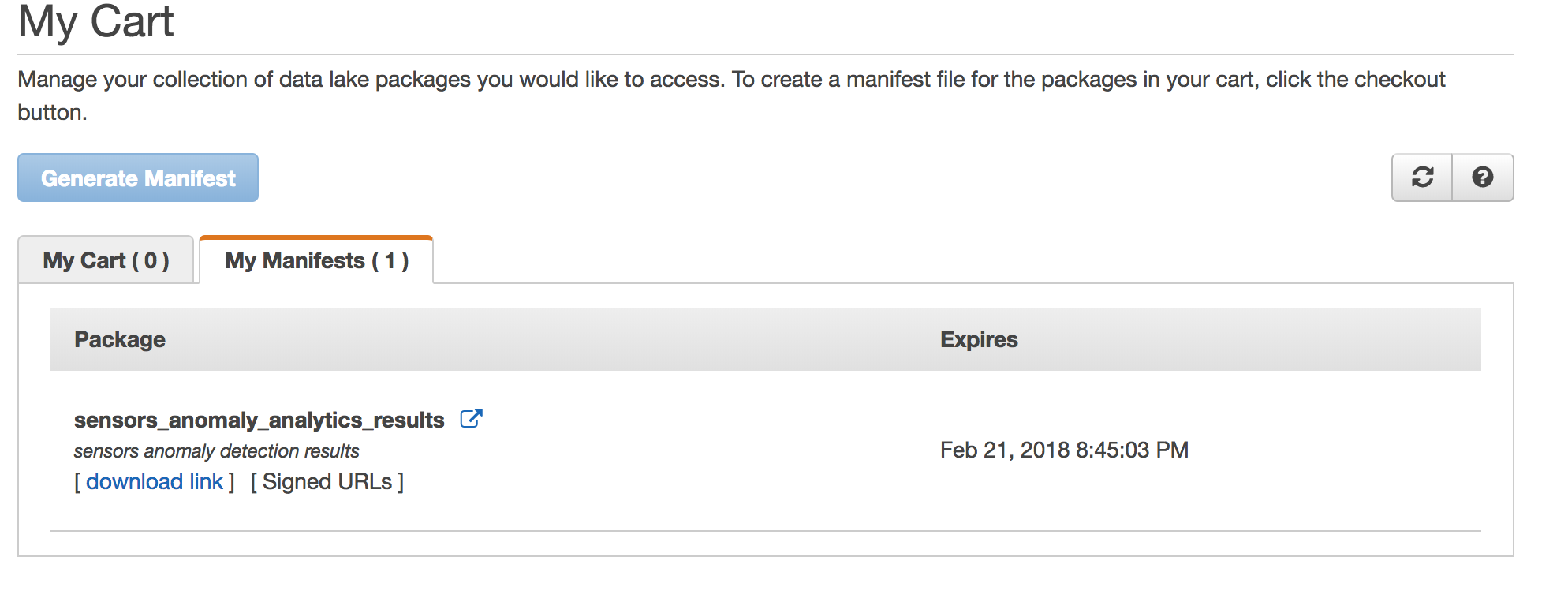
1. Go to your cart view either by clicking on the cart icon on top right corner or by clicking the ‘cart’ link from the left hand side menu. You will see one package in your cart.



1. Next, click on ‘Generate Manifest’ button. This will ask you to specify an option on how would you like the contents of your manifest file. Let’s choose S3 signed URLs for this exercise. Click on ‘Generate’.



1. Go to the Manifest tab on the same screen, and you should see one manifest file with ‘download link’.



1. Click on the ‘download link’ hyperlink and it will result in saving the manifest file on your local machine.
2. Open the manifest file and it should contain the pre-signed URLs for the S3 object associated with the package.
3. Copy and paste any of these pre-signed URLs in your browser and it should download that S3 object on your local machine.

Congratulations! You have successfully searched and downloaded your Data Lake contents.

# Conclusion

In this lab you have learned how to leverage Amazon Kinesis platform and Lambda to generate and collect streaming data, store the raw data on S3 for further processing and process the data in real-time then save the result on S3. You also learned how to use the Data Lake solution to catalog the Kinesis Analytics result and made it available for other users to search and retrieve. In the next lab, we will leverage other Amazon services such as Glue, Athena and QuickSight to perform further processing and analysis.