Hortonworks Streaming Analytics Manager 3

# **Getting Started with Streaming Analytics**

Date of Publish: 2020-12-15



https://docs.cloudera.com/

# Contents

<b>Building</b> a	n End-to-End Stream Application	4
	tanding the Use Case	
	ce Architecture	
Prepare Y	our Environment	5
Deploy	ing Your Cluster	6
Registe	ring Schemas in Schema Registry	6
-	Create the Kafka Topics	6
	Register Schemas for the Kafka Topics	6
Setting	up an Enrichment Store, Creating an HBase Table, and Creating an HDFS Directory	7
Creating 4	a Dataflow Application	8
	roducer Application Generates Events	
	Server a Dataflow Application	
	NiFi Controller Services	
	NiFi Ingests the Raw Sensor Events	
	Publish Enriched Events to Kafka for Consumption by Analytics Applications	
	Start the NiFi Flow	
	Start ule 14111110w	
Pick your	Streaming Engine	14
Pick your	Streaming Engine	14
·		
Creating a	a Streaming Analytics Application with SAM	16
Creating a	a Streaming Analytics Application with SAM	<b>16</b>
Creating a	a Streaming Analytics Application with SAM g a Stream Analytics Application with SAM Two Options for Creating the Streaming Analytics Applications	<b>16</b> 
Creating a Creatin	a Streaming Analytics Application with SAM g a Stream Analytics Application with SAM Two Options for Creating the Streaming Analytics Applications Creating a Service Pool and Environment	<b>16</b> 
Creating a Creatin	a Streaming Analytics Application with SAM g a Stream Analytics Application with SAM Two Options for Creating the Streaming Analytics Applications	<b>16</b> 
Creating a Creatin	a Streaming Analytics Application with SAM g a Stream Analytics Application with SAM Two Options for Creating the Streaming Analytics Applications Creating a Service Pool and Environment Creating Your First Application Creating and Configuring the Kafka Source Stream	<b>16</b> 16 16 17 18 19
Creating a Creatin	a Streaming Analytics Application with SAM g a Stream Analytics Application with SAM Two Options for Creating the Streaming Analytics Applications Creating a Service Pool and Environment Creating Your First Application	<b>16</b> 16 16 17 18 19
Creating a Creatin	a Streaming Analytics Application with SAM g a Stream Analytics Application with SAM Two Options for Creating the Streaming Analytics Applications Creating a Service Pool and Environment Creating Your First Application Creating and Configuring the Kafka Source Stream	<b>16</b> 16 16 17 18 19 20
Creating a Creatin	a Streaming Analytics Application with SAM g a Stream Analytics Application with SAM Two Options for Creating the Streaming Analytics Applications Creating a Service Pool and Environment Creating Your First Application Creating and Configuring the Kafka Source Stream Connecting Components Joining Multiple Streams Filtering Events in a Stream using Rules	<b>16</b> 16171819202123
Creating a Creatin	a Streaming Analytics Application with SAM g a Stream Analytics Application with SAM Two Options for Creating the Streaming Analytics Applications Creating a Service Pool and Environment Creating Your First Application Creating and Configuring the Kafka Source Stream Connecting Components Joining Multiple Streams	<b>16</b> 16171819202123
Creating a Creatin	a Streaming Analytics Application with SAM g a Stream Analytics Application with SAM Two Options for Creating the Streaming Analytics Applications Creating a Service Pool and Environment Creating Your First Application Creating and Configuring the Kafka Source Stream Connecting Components Joining Multiple Streams Filtering Events in a Stream using Rules	<b>16</b> 161617181920212324
Creating a Creatin	a Streaming Analytics Application with SAM g a Stream Analytics Application with SAM Two Options for Creating the Streaming Analytics Applications Creating a Service Pool and Environment Creating Your First Application Creating and Configuring the Kafka Source Stream Connecting Components Joining Multiple Streams Filtering Events in a Stream using Rules Using Aggregate Functions over Windows	<b>16</b> 16161718192021232424
Creating a Creatin	A Streaming Analytics Application with SAM g a Stream Analytics Application with SAM Two Options for Creating the Streaming Analytics Applications Creating a Service Pool and Environment Creating Your First Application Creating and Configuring the Kafka Source Stream Connecting Components Joining Multiple Streams Filtering Events in a Stream using Rules Using Aggregate Functions over Windows Implementing Business Rules on the Stream Transforming Data using a Projection Processor Streaming Alerts to an Analytics Engine for Dashboarding	<b>16</b> 161617181920212324242424242424242424
Creating a Creatin	A Streaming Analytics Application with SAM g a Stream Analytics Application with SAM Two Options for Creating the Streaming Analytics Applications Creating a Service Pool and Environment Creating Your First Application Creating and Configuring the Kafka Source Stream Connecting Components Joining Multiple Streams Filtering Events in a Stream using Rules Using Aggregate Functions over Windows Implementing Business Rules on the Stream Transforming Data using a Projection Processor	<b>16</b> 161617181920212324242424242424242424
Creating a Creatin	A Streaming Analytics Application with SAM g a Stream Analytics Application with SAM Two Options for Creating the Streaming Analytics Applications Creating a Service Pool and Environment Creating Your First Application Creating and Configuring the Kafka Source Stream Connecting Components Joining Multiple Streams Filtering Events in a Stream using Rules Using Aggregate Functions over Windows Implementing Business Rules on the Stream Transforming Data using a Projection Processor Streaming Alerts to an Analytics Engine for Dashboarding	<b>16</b> 1617181920212324242424262930
Creating a Creatin	A Streaming Analytics Application with SAM	<b>16</b> 16171819202123242426293032
Creating a Creatin	A Streaming Analytics Application with SAM	<b>16</b> 161718192021232424242424
Creating a Creatin	A Streaming Analytics Application with SAM	<b>16</b> 16161718
Creating a Creatin	A Streaming Analytics Application with SAM	<b>16</b> 161617181920212324242424243035353536
Creating a Creatin Deploy Advance	A Streaming Analytics Application with SAM	<b>16</b> 16161718192021232424242424263035353638
Creating a Creatin Deploy Advanc	A Streaming Analytics Application with SAM	$\begin{array}{c} \textbf{16} \\ \textbf{16} \\ \textbf{16} \\ \textbf{17} \\ \textbf{17} \\ \textbf{18} \\ \textbf{19} \\ \textbf{20} \\ \textbf{20} \\ \textbf{21} \\ \textbf{21} \\ \textbf{23} \\ \textbf{24} \\ \textbf{26} \\ \textbf{36} \\ \textbf{36} \\ \textbf{36} \\ \textbf{36} \\ \textbf{38} \\ \textbf{39} \\ \textbf{40} \\ \textbf{40}$
Creating a Creatin Deploy Advanc	A Streaming Analytics Application with SAM	$\begin{array}{c} \textbf{16} \\ \textbf{16} \\ \textbf{16} \\ \textbf{17} \\ \textbf{17} \\ \textbf{18} \\ \textbf{19} \\ \textbf{20} \\ \textbf{20} \\ \textbf{21} \\ \textbf{21} \\ \textbf{23} \\ \textbf{24} \\ \textbf{26} \\ \textbf{36} \\ \textbf{36} \\ \textbf{36} \\ \textbf{36} \\ \textbf{38} \\ \textbf{39} \\ \textbf{40} \\ \textbf{40}$
Creating a Creatin Deploy Advanc	A Streaming Analytics Application with SAM	<b>16</b> <ul> <li></li></ul>

Streaming Split Join Pattern	
Score the Model Using the PMML Processor and Alert	
Creating Visualizations Using Superset	
Creating Insight Slices	61
Adding Insight Slices to a Dashboard	
SAM Test Mode	
Four Test Cases using SAM's Test Mode	
Creating Custom Sources and Sinks	
Cloud Use Case: Integration with AWS Kinesis and S3	
Registering a Custom Source in SAM for AWS Kinesis	
Registering a Custom Sink in SAM for AWS S3	
Implementing the SAM App with Kinesis Source and S3 Sink	
Stream Operations	
My Applications View	
Application Performance Monitoring	
Exporting and Importing Stream Applications	89
Troubleshooting and Debugging a Stream Application	91
Monitoring SAM Apps and Identifying Performance Issues	91
Identifying Processor Performance Bottlenecks	
Debugging an Application through Distributed Log Search	
Debugging an Application through Sampling	
Spark Streaming	
Running the Stream Simulator	103
Managing Kafka with Streams Messaging Manager	
SMM Overview	
Installing DataPlane Streams Messaging Manager	
Enabling Reference Application Cluster for SMM	
Monitoring Kafka with SMM	107

# **Building an End-to-End Stream Application**

A good way to get started using Hortonworks DataFlow (HDF) and all of its services like NiFi, Streams Messaging Manager, Streaming Analytics Manager, Schema Registry, etc is to imagine a real life use case, and to learn about the common HDF stream processing tasks and concepts through this use case. This guide sets up a fictional use case, and walks you through the deployment and common tasks you would perform while engaging in many of HDF's stream processing use cases.

Use this guide as a tutorial to get you started with NiFi, SMM, SAM and Schema Registry and Spark Streaming. All the resources required to complete the tasks are provided in line.

# **Understanding the Use Case**

To build a complex streaming analytics application from scratch, we will work with a fictional use case. A trucking company has a large fleet of trucks, and wants to perform real-time analytics on the sensor data from the trucks, and to monitor them in real time. Their analytics application has the following requirements:

# Procedure

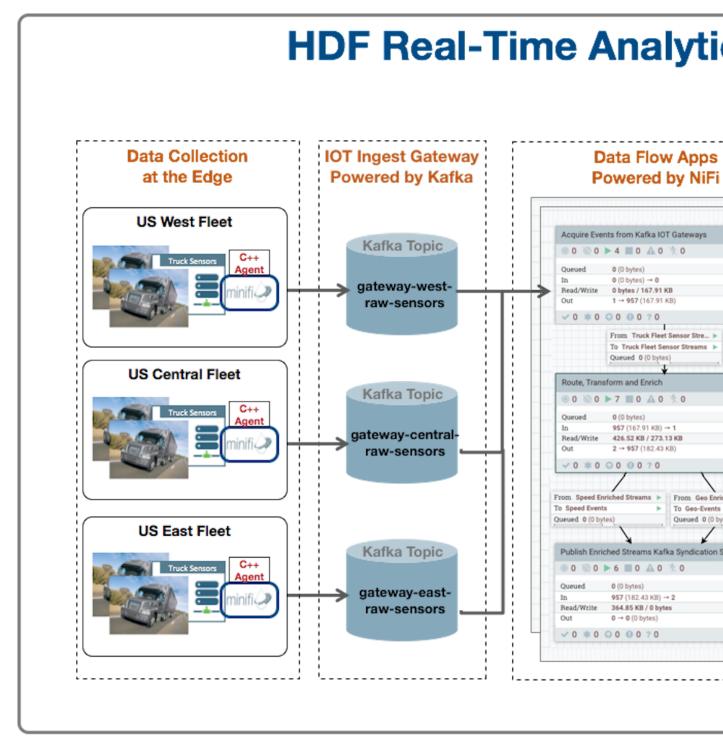
- 1. Outfit each truck with two sensors that emit event data such as timestamp, driver ID, truck ID, route, geographic location, and event type.
  - The geo event sensor emits geographic information (latitude and longitude coordinates) and events such as excessive braking or speeding.
  - The speed sensor emits the speed of the vehicle.
- 2. Stream the sensor events to an IoT gateway. The data producing app (e.g: a truck) will send CSV events from each sensor to one of three gateway topics (gateway-west-raw-sensors, gateway-east-raw-sensors or gateway-central-raw-sensors). Each event will pass the schema name for the event as a Kafka event header.
- **3.** Use NiFi to consume the events from the Kafka topic, and then route, transform, enrich, and deliver the data from the gateways to two syndication topics (e.g: syndicate-geo-event-avro, syndicate-speed-event-avro, syndicate-geo-event-avro, syndicate-speed-event-json, syndicate-speed-event-json ) that various downstream analytics applications can subscribe to.
- 4. Connect to the two streams of data to perform analytics on the stream.
- **5.** Join the two sensor streams using attributes in real-time. For example, join the geo-location stream of a truck with the speed stream of a driver.
- 6. Filter the stream on only events that are infractions or violations.
- 7. All infraction events need to be available for descriptive analytics (dash-boarding, visualizations, or similar) by a business analyst. The analyst needs the ability to perform analysis on the streaming data.
- **8.** Detect complex patterns in real-time. For example, over a three-minute period, detect if the average speed of a driver is more than 80 miles per hour on routes known to be dangerous.
- 9. When each of the preceding rules fires, create alerts, and make them instantly accessible.
- **10.** Execute a logistical regression Spark ML model on the events in the stream to predict if a driver is going to commit a violation. If violation is predicted, then generate an alert.
- 11. Monitor and manage the entire application using Streams Messaging Manager and Stream Operations.

# What to do next

The following sections walk you through how to implement all ten requirements. Requirements 1-3 are performed using NiFi and Schema Registry. Requirements 4 through 10 are implemented using the new Streaming Analytics Manager.

# **Reference Architecture**

This reference architecture diagram gives you a general idea of how to build an HDF cluster for your trucking use case. Review this suggested architecture before you begin deployment.



# **Prepare Your Environment**

# **Deploying Your Cluster**

Now that you have reviewed the reference architecture and planned the deployment of your trucking application, you can begin installing HDF according to your use case specifications. To fully build the trucking application as described in this *Getting Started with Stream Analytics* document, use the following steps.

### Procedure

- 1. Install Ambari 2.7.0.
- 2. Install HDP 3.0.0.
- 3. Install the HDF 3.2.0 Management Pack onto the HDP cluster.

### What to do next

You can find more information about your HDF and HDP cluster deployment options in Planning Your Deployment.

You can find more information about which versions of HDP and Ambari you should use with your version of HDF in the *HDF Support Matrices*.

# **Registering Schemas in Schema Registry**

The trucking application streams CSV events from the truck sensors to a IOT gateway powered by a set of Kafka gateway topics. NiFi consumes the events from these topics, and then routes, enriches, and delivers them to a set of syndication Kafka topics.

To do this, you must perform the following tasks:

- Create the Kafka gateway and syndication topics
- Register a set of Schemas in Schema Registry

# **Create the Kafka Topics**

Kafka topics are categories or feed names to which records are published.

# Procedure

- 1. Log into the node where Kafka broker is running.
- 2. Download script createKafkaTopics.sh from here: https://raw.githubusercontent.com/georgevetticaden/sam-trucking-data-utils/hdf-3-2/src/main/resources/scripts/smm/createKafkaTopics.sh
- **3.** Execute the script by passing as arg ZK:Port

For example:

./createKafkaTopics.sh ZK\_HOST:2181

# **Register Schemas for the Kafka Topics**

Register schemas for the IOT gateway and syndication Kafka topics .Registering the Kafka topic schema is beneficial in several ways. Schema Registry provides a centralized schema location, allowing you to stream records into topics without having to attach the schema to each record.

### Procedure

- 1. Log into the node where you have access to the Schema Registry Server.
- 2. Download the Data-Loader and unzip the contents.

3. Navigate to the Data-Loader directory:

cd Data-Loader

**4.** Execute the following:

```
java -cp \
stream-simulator-jar-with-dependencies.jar \
hortonworks.hdf.sam.refapp.trucking.simulator.schemaregistry.TruckSchemaRegistryLoade
\
$SCHEMA_REGISTRY_ENDPOINT_URL
E.g: SCHEMA_REGISTRY_ENDPOINT_URL = http://SR_HOST::7788/api/v1
```

# Setting up an Enrichment Store, Creating an HBase Table, and Creating an HDFS Directory

To prepare to perform predictive analytics on streams, you need some HBase and Phoenix tables. This section gives you instructions on setting up the HBase and Phoenix tables timesheet and drivers, loading them with reference data, and downloading the custom UDFs and processors to perform the enrichment and normalization.

Install HBase/Phoenix and download the sam-extensions

- 1. If HBase is not installed, install/add an HBase service.
- 2. Ensure that Phoenix is enabled on the HBase Cluster.
- 3. Download the Sam-Custom-Extensions.zip and save it to your local machine.
- 4. Unzip the contents. Name the unzipped folder \$SAM\_EXTENSIONS.

Steps for Creating Phoenix Tables and Loading Reference Data

- 1. Copy the \$SAM\_EXTENSIONS/custom-processor/scripts.tar.gz to a node where HBase/Phoenix client is installed.
- 2. On that node, untar the scripts.tar.gz. Name the directory \$SCRIPTS.

tar -zxvf scripts.tar.gz

**3.** Navigate to the directory where the phoenix script is located which will create the phoenix tables for enrichment and load it with reference data.

cd \$SCRIPTS/phoenix

- 4. Open the file phoenix\_create.sh and replace <ZK\_HOST> with the FQDN of your ZooKeeper host.
- 5. Make the phoenix\_create.sh script executable and execute it. Make sure you add the script to JAVA\_HOME.

./phoenix\_create.sh

Steps for Verifying Data has Populated Phoenix Tables

1. Start up the sqlline Phoenix client.

cd /usr/hdp/current/phoenix-client/bin ./sqlline.py \$ZK\_HOST:2181:/hbase-unsecure

**2.** List all the tables in Phoenix.

!tables

3. Query the drivers and timesheet tables.

select \* from drivers;

select \* from timesheet;

Steps for Starting HBase and Creating an HBase Table

- 1. This can be easily done by adding the HDP HBase Service using Ambari.
- 2. Create a new HBase table by logging into an node where Hbase client is installed then execute the following commands:

```
cd /usr/hdp/current/hbase-client/bin
/hbase shell
create 'violation_events', {NAME=> 'events', VERSIONS => 3};
```

Steps for Creating an HDFS Directory

Create the following directory in HDFS and give it access to all users.

- 1. Log into a node where HDFS client is installed.
- 2. Execute the following commands:

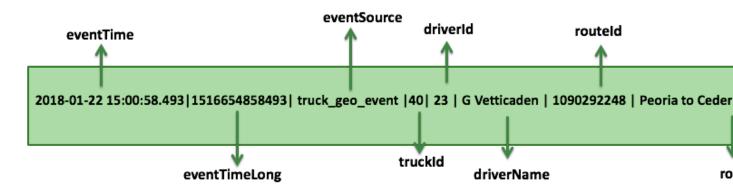
```
su hdfs
hadoop fs -mkdir /apps/trucking-app
hadoop fs -chmod 777 /apps/trucking-app
```

# **Creating a Dataflow Application**

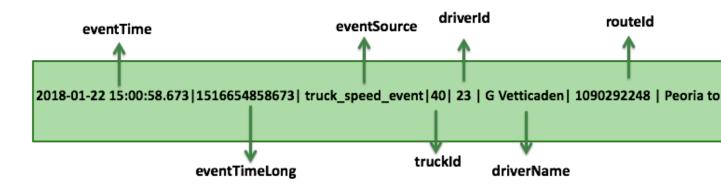
# **Data Producer Application Generates Events**

The following is a sample of a raw truck event stream generated by the sensors.

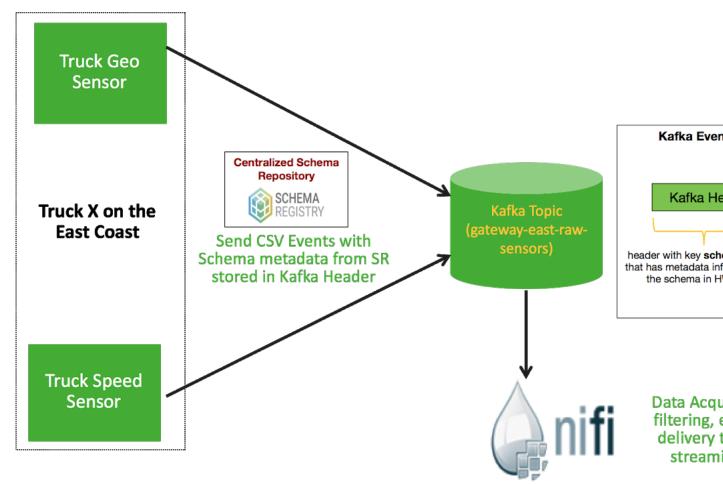
Geo Sensor Stream



Speed Sensor Stream



The date producing application or data simulator publishes CSV events with schema name in the Kafka event header (leveraging the Kafka Header feature in Kafka 1.0). The following diagram illustrates this:

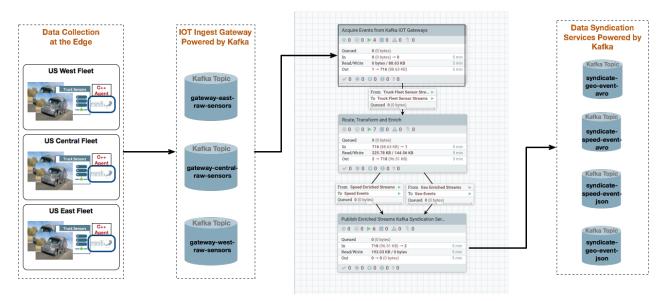


Use NiFi's Kafka 1.X ConsumeKafkaRecord and PublishKafkaRecord processors using record-based processing to do the following:

- 1. Grab the schema name from Kafka Header and store in flow attribute called schema.name
- 2. Use the schema name to look up the Schema in HWX SR
- 3. Use the schema from HWX SR to convert to ProcessRecords
- 4. Route events by event source (speed or geo) using SQL
- 5. Filter, Enrich, Transform

6. Deliver the data to downstream syndication topics

The below diagram illustrates the entire flow and the subsequent sections walks through how to setup this data flow.



# NiFi: Create a Dataflow Application

To make things easier to setup, import the NiFi Template for this flow by downloading it to this Github location. After importing, select IOT Trucking Fleet - Data Flow process group. The following instructions are with respect to that flow.

# **NiFi Controller Services**

# Procedure

- 1. Click on Flow Configuration Settings icon, select Controller Services tab, and select Hortonworks Schema Registry Controller Service.
- 2. Click on Flow Configuration Settings icon and select Controller Services tab.

You will see the HWX Schema Registry controller service. Edit the properties to configure the Schema Registry URL based on your environment. You can find this value in the Streaming Analytics Manager Service in Ambari for the configuration property called registry.url. An example of what the URL looks similar to http:// \$REGISTRY\_SERVER:7788/api/v1.

3. Edit the properties to configure the Schema Registry URL based on your environment.

You can find this value in the Streaming Analytics Manager Service in Ambari for the configuration property called registry.url. The URL should look similar to the following: http://\$REGISTRY\_SERVER:7788/api/v1.

**4.** Enable the HWX Schema Registry and the other controller services. You should have 5 controller services enabled like the following.

	Name	Туре	Bun.
8	HWX Schema Registry	HortonworksSchema	org
8	Enrich- ReverseGeoCodeLookupService	ScriptedLookupServic	org
8	CSV Writer - HWX SR Registry	CSVRecordSetWriter 1	org
8	CSV Reader - HWX SR Registry	CSVReader 1.5.0.3.1.0	org
8	Avro Writer - HWX SR Registry - HWX Content Enc	AvroRecordSetWriter	org

# NiFi Ingests the Raw Sensor Events

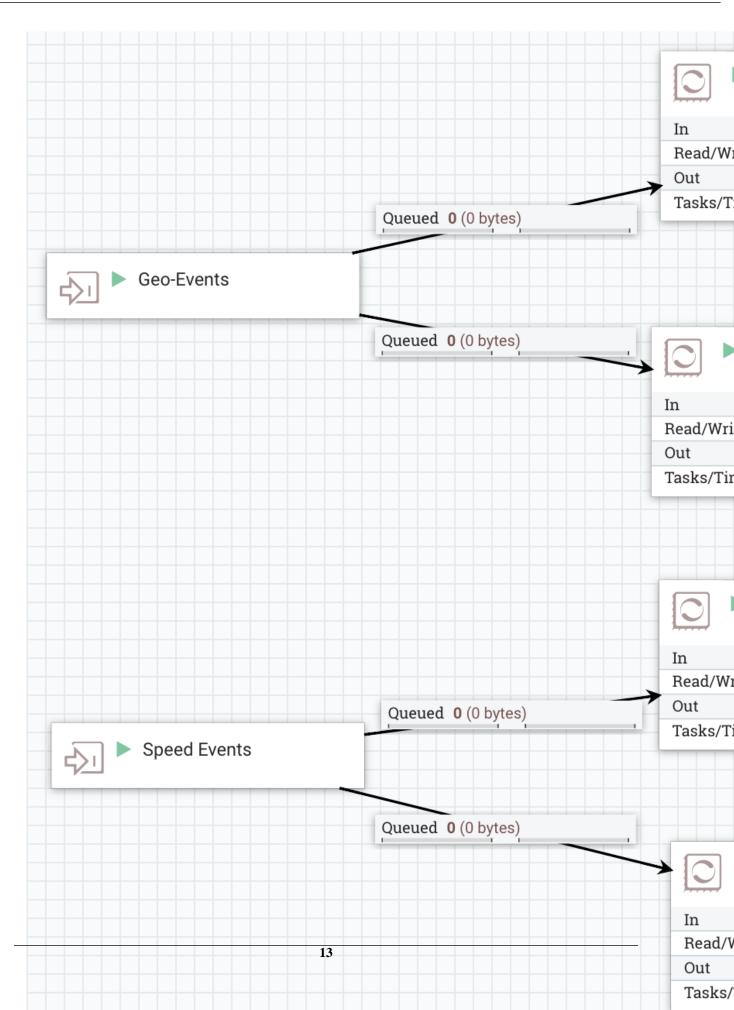
In the IOT Trucking Fleet - Data Flow process group, go into the "Acquire Events from Kafka IOT Gateways" process group. The first step in the NiFi flow is to ingest the csv events from the three IOT gateway topics. We will use the new Kafka 1.0 ConsumerKafkaRecord processor for this.

Upstream app is sen Use ConsumeKafkaf 1. Grab the schema 2. Use the schema n 3. Use the schema fi 4. Route events by e	Record do the name from Ka ame to look u rom HWX SR t	followi afka He Ip the S to conv	ng ader and store in chema in HWX S ert to ProcessRe	n Attribute called SR ecords	-
US West Truck Fleet - Kafk ConsumeKafkaRecord_1_0 1.7 org.apache.nifi - nifi-kafka-1-0-nar			US Central Truck Fleet ConsumeKafkaRecord_1_ org.apache.nifi - nifi-kafka-1-0-n	0 1.7.0.3.2.0.0-485	0
In 0 (0 bytes)	5 min	In	<b>0</b> (0 bytes)	5 min	In
Read/Write 0 bytes / 40.35 KB	5 min	Read/Wi	ite 0 bytes / 27.82 KB	5 min	Read
Out         326 (40.35 KB)           Tasks/Time         1,052 / 00:00:12.521	5 min 5 min	Out Tasks/Ti	230 (27.82 KB) me 1,000 / 00:00:11.451	5 min 5 min	Out Task
	Name succe	955	success	Name success	
	Queued <b>0</b> (0	bytes)	<b>d 0</b> (0 bytes)	Queued 0 (0 bytes)	
			Truck Fleet Sensor Streams		

Configure the 'Kafka Brokers' value for each of the three ConsumeKafkaRecord\_1\_0 processors based on your cluster.

# Publish Enriched Events to Kafka for Consumption by Analytics Applications

After NiFi completes the routing, transforms, and enrichment, NiFi publishes the enriched events into a set of syndication Kafka topics. This is done in the Publish Enriched Streams Kafka Syndication Services process group. This process group uses the PublishKafkaRecord processor to publish the events into 4 syndication topics.



Ensure that for the PublishKafkaRecord, you change the Kafka Brokers property value to your cluster settings.

# Start the NiFi Flow

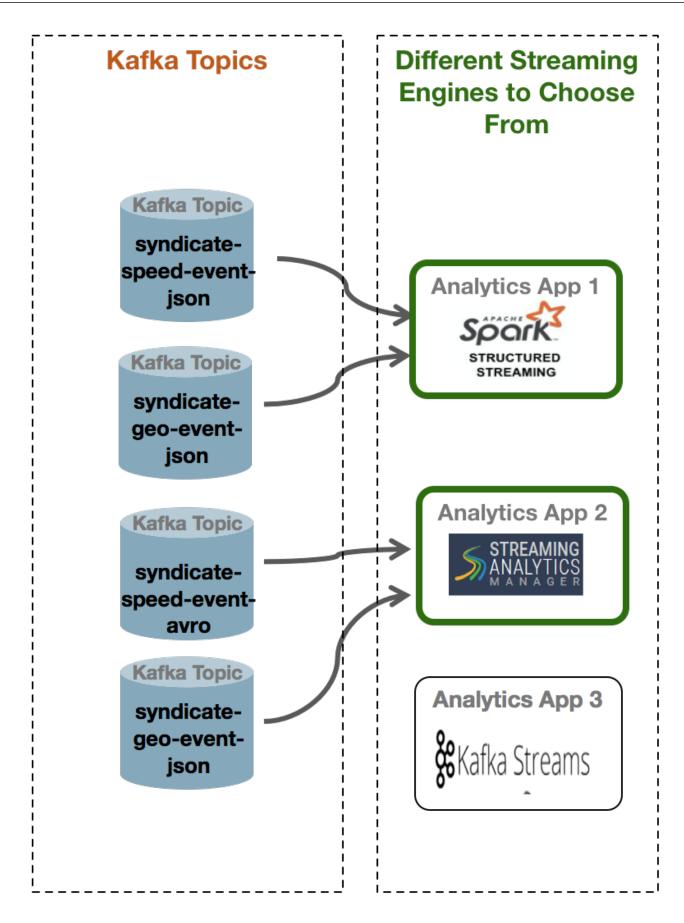
Start the Process Group called IOT Trucking Fleet - Data Flow.

# **Pick your Streaming Engine**

Hortonworks supports a number of powerful Streaming Engines including:

- Spark Streaming (Currently Supported in HDP)
- Streaming Analytics Manager (SAM) using Apache Storm (Currently Supported in HDF)
- Apache Kafka Streams (Will be supported in future HDP/HDF release)

Hortonworks provides you the flexibility to pick the streaming engine of your choice to build streaming analytics application.



If your organization has not standardized on a streaming engine and is looking for guidance on choosing an engine, use the below table to help guide your selection.

Requirement	Streaming Engine to Use
You want to build streaming applications with as little code as possible and want to use ETL like Tooling with a drag and drop paradigm to build streaming apps	Streaming Analytics Manager (SAM) with Storm
You want the ability to build an app that you can deploy in batch and/ or streaming mode	Spark Streaming
You plan to develop streaming applications using scala and java and want a clean easy to use API	Spark Streaming
You want the ability to execute SQL against the stream	Spark Streaming
You want process event one at a time (no microbatching)	Streaming Analytics Manager (SAM) with Storm

The below sections walk you through implementing the streaming analytics requirements with these different tools/ engines.

# **Creating a Streaming Analytics Application with SAM**

# **Creating a Stream Analytics Application with SAM**

# **Two Options for Creating the Streaming Analytics Applications**

Over the next few sections, we walk through building up the stream analytics app to implement all the requirements of this use case. This will entail actions such as:

- Uploading Custom UDFs
- Uploading Custom Sources
- Uploading Custom Sinks
- Uploading a PMML model into Model Registry
- Uploading Custom Processors
- Creating Service Pools for HDP and HDF
- Create SAM Environment required for the Reference App
- Building the Reference application with streaming joins, filtering, aggregations over windows, dashboarding, execute PMML models, doing streaming split pattern, etc..
- Setting up Test Cases for the Reference App

There are two options to performing these actions:

- 1. Doing all of these steps manually as the subsequent sections will walk you through. This is recommended if you are new to SAM and want to build a complex app from scratch.
- 2. Running a utility that performs all of these actions for you.

## Setting up the Stream Analytics App using the TruckingRefAppEnvEnviornmentBuilder

Follow the below instructions if you want to set up the reference application using an utility. If not, skip this section and go through the next set of sections. Perform the below steps on the host where SAM is running.

### Procedure

1. Download the SAM\_EXTENSIONS zip file. Unzip the contents. Call the unzipped folder \$SAM\_EXTENSION

**2.** Navigate to the unzip location:

cd \$SAM\_EXTENSION/ref-app

- 3. Modify the trucking-advanced-ref-app.properties file based on your environment:
  - sam.rest.url = the REST url of the SAM instance in your env (e.g.: http://<SAM\_HOST>:<SAM\_PORT>/api/ v1)
  - sam.service.pool.hdp.ambari.url = The rest endpoint for the HDP/HDF cluster you installed (e.g.: http:/ <HDP\_AMBARI\_HOST>:<PORT>/api/v1/clusters/<cluster\_name>)
  - sam.service.pool.hdp.ambari.username = the username to log into Ambari
  - sam.service.pool.hdp.ambari.passwd = the password to log into Ambari
  - sam.schema.registry.url = The url of the Schema Registry service in SAM you installed as part of the HDF cluster (e.g.: http://SR\_HOST:SR\_PORT/api/v1)
- **4.** Run the following command:

```
java -cp sam-trucking-ref-app-shaded.jar
hortonworks.hdf.sam.refapp.trucking.env.SMMTruckingRefAppEnviornmentBuilderImpl
trucking-advanced-ref-app.properties
```

## Results

If script ran successfully, you should see output like the following (it will take about 3-5 minutes to finish):

```
Trucking Ref App environment creation time[367 seconds]
Trucking Ref App SAM URL: http://SAM_HOST:SAM_PORT/#/applications/78/view
```

## **Configuring and Deploying the Reference Application**

In SAM, go into the edit mode the Trucking Ref App using the url outputted in the logs and then follow the below steps to configure and deploy the reference application:

### Procedure

- 1. Double click on the TruckGeoEvent Kafka Source and configure it.
  - a. Select a cluster.
  - **b.** Select truck\_events\_avro for the kafka topic.
  - c. Select 1 for the Reader Schema Version.
  - **d.** Put an unique value for the consumer group id.
- 2. Double click on the TruckSpeedEvent Kafka Source and configure it.
  - **a.** Select a cluster.
  - **b.** Select truck\_speed\_events\_avro for the kafka topic.
  - c. Select 1 for the Reader Schema Version.
  - **d.** Put an unique value for the consumer group id.
- **3.** Open the configuration for the two Druid sinks (Alerts-Speeding-Driver-Cube and Dashboard-Predictions) and configure each one by selecting a cluster.
- **4.** Open the configuration for the ENRICH-HR and ENRICH-Timesheet configure the 'Phoenix ZooKeeper Connection Url" based on your cluster (e.g.: ZK\_HOST:ZK:PORT).
- 5. Open the configuration for the Hbase and HDFS sink and ensure that the cluster is selected.

# **Creating a Service Pool and Environment**

Before you create an application, you have to create a Service Pool and then an Environment that you associate with an application.

# **Creating Your First Application**

The Streaming Analytics Manager provides capabilities to the application developer for building streaming applications. You can go to the Stream Builder UI by selecting the **Streaming Analytics Manager** service in Ambari and under **Quick Links** select **SAM UI**.

## About this task

Creating a new stream application requires two steps: clicking the + icon, and then providing a unique name for the stream application and associating the application with an Environment.

## Procedure

- 1. Click the + icon on the My Applications dashboard and choose New Application.
- 2. Specify the name of the stream application and the environment that you want it to use to stream.



Note:

The name of the stream application cannot have any spaces.

Add Stream	×
NAME •	
Trucking-IOT-Stream-Analytics	
ENVIRONMENT .	
Dev	*
	Cancel Ok

### Results

SAM displays the Stream Builder canvas. Builder components on the canvas palette are the building blocks used to build stream applications. Now you are ready to start building the streaming application.

M	My Appli	ications Sample Application Edit and name the stream application		<u>*</u>
	Q 🕼	Last Change 0s ago Version: CURRENT	DEV O	Q
	SOURCE			1
æ	EVENT		/	
ø	HUBS	Processor, source, and sink palette contains builder components	Application a	
Ð	HDFS		deploymen configuratio	
	<b>%</b>			
ŗ	KAFKA PROCESSOR			
	Σ			
	AGGREGATE			
	х°			
	BRANCH			
	JOIN			
	PROJECTION			_
	<b>9</b> 00	Deploy button to deploy stream		
	RULE	applications to the streaming engine	Status: NOT RUNNING	1
	ENRICH	sureaming engine		
	ention ett			

# Creating and Configuring the Kafka Source Stream

The first step in building a stream application is to drag builder components to the canvas. As described in the *Hortonworks DataFlow Overview*, Stream Builder offers four types of builder components: sources, processors, sinks, and custom components.

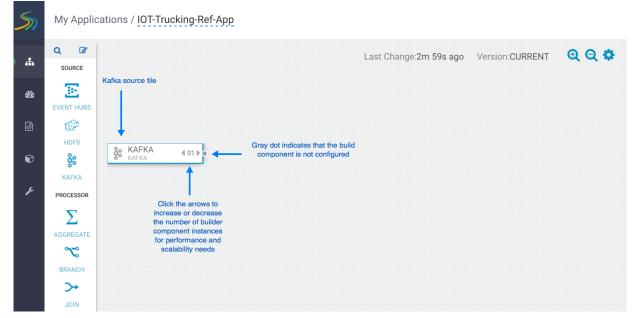
## About this task

Every stream application must start with a source.

Complete the following instructions to start building a stream application. Use these steps to implement Requirement 4 of the use case.

# Procedure

1. Drag the Kafka builder component onto the canvas, creating a Kafka tile:



- 2. Set the number of run-time instances for your Kafka tile component by clicking the up arrow on the tile.
- **3.** Double-click on the tile to begin configuring Kafka. After you specify a Kafka topic name, SAM communicates with the Schema Registry and displays the schema:

TruckGeoEvent Kafka connection settings are populated by SAM, based on the Kafka service in Environment from the Service Pool					
REQUIRED OPTIONAL NOTES					
streamanalytics_AUTOCREATED -	Output				
SECURITY PROTOCOL *	eventSource* STRING				
PLAINTEXT	truckId* INTEGER				
	driverId*				
BOOTSTRAP SERVERS *	driverName*				
hdf-3-1-build4.field.hortonworks.com:6667,hdf-3-1-buil	routeld*				
KAFKA TOPIC *	route*				
truck_events_avro T	eventType*				
READER SCHEMA VERSION *	latitude*				
1	longitude*				
CONSUMER GROUP ID *	correlationId*				
ref-geo-101	geoAddress strang				
After you select a Kafka topic, SAM fetches the topic schema from Schema Registry	Cancel Ok				

### Results

Once you have configured your Kafka component correctly, the tile component displays a green dot.

# **Connecting Components**

To pass a stream of events from one component to the next, create a connection between the two components. In addition to defining data flow, connections allow you to pass a schema from one component to another.

### Procedure

1. Click the green dot to the right of your source component.



2. Drag your cursor to the component tile to which you want to connect.

# Example

The following example shows two connections: a connection from Kafka sink TruckGeoStream to the join processor, and a connection from the Kafka sink TruckSpeedStream to the same join processor.

5	My Applic	ations / IOT-Trucking-Ref-App		
	Q 🕜		Last Change:0s ago Version:CURRENT 👲 🤤	ð
· #	SOURCE	Two Kafka components are		Τ.
ങ	E	connected to the Join processor. Both streams and the respective schemas are passed to the Join		
	EVENT HUBS	processor.		
ф	4 <u>0</u> 3			
Ø	HDFS	8 TruckGeoEv ∉01 ►		
æ	KAFKA	JOIN		
-	PROCESSOR	<sup>8</sup> <sup>TruckSpeed</sup> ∢ 01 ▶ •		
	Σ			
	AGGREGATE			
	$\sim$	You can connect components by		
	BRANCH	clicking the green dot on a component tile and dragging to the		
	≫	dot on the component tile to which you want to connect		
	JOIN			

# **Joining Multiple Streams**

Joining multiple streams is an important SAM capability. You accomplish this by adding the Join processor to your stream application. You can join on multiple streams using one of two concepts available in the **Window Type** filed:

### About this task

- Processing Time Represents window based on when the event is processed by SAM
- Event Time Represents the window based on the timestamp of the event.

JOIN	N4LOO				×
CONFIGURATION NOTES					
Input	kafka_stream_104 👻	driverId -			Output
kafka_stream_104 🔹	JOIN TYPE*	SELECT STREAM*	SELECT FIELD*	WITH STREAM*	truckld* ⊪reger driverld*
eventTime* string eventSource*	INNER -	kafka_stream_105 -	driverId 👻	kafka_stream_104 🛛 👻	INTEGER driverName* STRING
eventsource* string truckld*	WINDOW TYPE*				routeld* INTEGER
INTEGER driverId*	Processing Time	Ŧ			route* string
INTEGER driverName* STRING	WINDOW INTERVAL*		SLIDING INTERVAL		eventType* <sup>STRING</sup> latitude*
routeld* INTEGER	3	Seconds -	3	Seconds -	bouble longitude*
route* string eventType*	OUTPUT FIELDS* 1 ALIAS FOR	OUTPUT FIELDS ARE MANDATOR	Ŷ	SELECT ALL	CorrelationId*
STRING	× 1 eventTime as eventTime	× 1 eventSor	urce as eventSource		geoAddress STRING
DOUBLE	× 1 truckld as truckld	× 1 driverld as	driverId		speed* INTEGER
correlationId*					Cancel Ok

JOIN					×
CONFIGURATION NOTES					
Input	SELECT STREAM*	SELECT FIELD WITH*			Output
kafka_stream_104 🔹	kafka_stream_104 🗸	driverId 👻			eventTime* string
eventTime* string	JOIN TYPE*	SELECT STREAM*	SELECT FIELD*	WITH STREAM*	eventSource* string truckld*
eventTimeLong*	INNER -	kafka_stream_105 v	driverId -	kafka_stream_104 v	INTEGER driverId*
eventSource*	WINDOW TYPE*		TIMESTAMP FIELDS*	LAG MILLISECONDS*	INTEGER driverName* STRING
truckld* INTEGER driverld* INTEGER	Event Time	Ŧ	eventTimeLong	0	routeld*
driverName* <sup>STRING</sup> routeld*	WINDOW INTERVAL*		kafka_stream_104		string eventType* string
INTEGER <b>FOUTE*</b> STRING	3	Seconds 👻	CORRELATIONID	Seconds	latitude* <sup>DOUBLE</sup> longitude*
eventType*	OUTPUT FIELDS* 19 ALIAS FOR	OUTPUT FIELDS ARE MANDATORY	<pre>kafka_stream_105</pre>	SELECT ALL	correlationId*
latitude* DOUBLE longitude* DOUBLE					Cancel Ok

For the purposes of this reference application, use processing time.

This section shows you how to configure a Join processor that joins the truck geo-event stream with the speed event stream, based on Requirement 5 of the use case.

# Procedure

- 1. Drag a Join processor onto your canvas and connect it to a source.
- 2. Double-click the Join tile to open the Configuration dialog.
- **3.** Configure the Join processors according to the example below.

#### Example

JOIN	Join stream_1 on fi	eld driverld			Wait 5 seconds for
CONFIGURATION NOTES	<b>↓</b>		Inner join with st		streams to catch up before the join occurs
Input	kafka_stream_1 🔍	driverId -	L		
kafka_stream_1 👻	JOIN TYPE	SELECT STREAM	SELECT FIELD	WITH STREAM	INTEGER driverld* INTEGER
eventTime* STRING eventSource*	INNER -	kafka_stream_2 👻	driverId 👻	kafka_stream_1 🗸 👻	driverName* STRING
eventSource* string truckld* INTEGER	WINDOW INTERVAL TYPE*			•	routeld*
driverId* INTEGER driverName*	WINDOW INTERVAL*		SLIDING INTERVAL		eventType* strang latitude*
routeld*	05	Seconds -	5	Seconds -	
route* STRING	OUTPUT FIELDS*			SELECT ALL	
eventType* string latitude* double	× eventTime × eventSour × latitude × longitude	ce × truckid × driverid × × correlationid × geoAddress	driverName × routeld × × speed	route × eventType × -	geoAddress straing speed* integer
longitude* DOUBLE correlationId* LONG				The output of the joins	Cancel Ok

# Filtering Events in a Stream using Rules

SAM provides powerful capabilities to filter events in the stream. It uses a Rules Engine, which translates rules into SQL queries that operate on the stream of data. The following steps demonstrate this, implementing Requirement 6 of the use case.

# Procedure

1. Drag the Rule processor to the canvas and connect it from the Join processors.



- 2. Double-click the Rule processor, click the Add new Rules button, and create a new rule.
- 3. Click OK to save the new rule.

### Example

ONFIGURATION NOTES		Click to add rules whi translated into SQL on th	•			
nput	+Add New Rules	and allows filtering of	stream	Output		
eventTime* STRING		events		eventTime*		
eventSource* STRING	Name	Condition	Actions	eventSource*		
truckId*	Violation Event	eventType <> 'Normal'	e 🖉	truckld*		
driverId*		<b>↑</b>		driverId*		
driverName* STRING		driverName* strang routeld* INTEGER route* strang				
routeld* INTEGER						
route* STRING						
eventType* string		equal to Normal, which represent a Violation Event	IS	eventType*		
latitude* DOUBLE				latitude*		
longitude*				longitude*		
correlationId*				correlationId*		

# **Using Aggregate Functions over Windows**

Windowing is the ability to split an unbounded stream of data into finite sets based on specified criteria such as time or count, so that you can perform aggregate functions (such as sum or average) on the bounded set of events. SAM exposes these capabilities using the Aggregate processor. The Aggregate processor supports two window types, tumbling and sliding windows. The creation of a window can be based on time or count. The following images show how to use the Aggregate processor to implement Requirement 8 of the use case.

### Procedure

1. Drag the Aggregate processor to the canvas and connect it to the Rule processor.

5	My Applic	cations / IOT-Trucking-Ref-App
	Q 07	Last Change:9s ago Version:CURRENT 🤨 🝳 🌼
r de la constante de la consta	SOURCE	
æ	E	
	EVENT HUBS	
Ś	(C)	
	HDFS	% TruckGeoEv ∢o1 ▶ •
Ŷ	<b>%</b>	
<i>p</i> €	KAFKA	JOIN 401
10	PROCESSOR	Image: Second
	Σ	processor onto the canvas and connect it to
	AGGREGATE	Violent Events Rule
	×.	
	BRANCH	
	>≁	
	JOIN	

2. Double-click on the Aggregate processor, and configure it to calculate the average speed of driver over a threeminute duration.

CONFIGURATION NOTES	The fields to group by	At the end of the windo is the new schema that output to the stream: average speed of every	will be the		×
Input	SELECT KEYS*	average speed of every	unver	Output	
truckld*	× driverId × driverName × route		× 👻	driverId*	
driverId*	WINDOW INTERVAL TYPE*			driverName* STRING	
driverName* STRING	Time		-	route* STRING	
routeld* INTEGER	WINDOW INTERVAL*			speed_AVG*	
route* STRING	3	Minutes			
eventType* string latitude*	SLIDING INTERVAL				
	3	Minutes			
correlationId*	TIMESTAMP FIELD				
geoAddress* string	processingTime × 👻				
speed* INTEGER	Output Fields				
				Cancel	Ok

# **Implementing Business Rules on the Stream**

This section shows you how to implement the business rule you created above to detect high speeding drivers. "High speed" is defined as greater than 80 miles per hour over a three-minute time window. This step partially implements Requirement 8 of the use case.

# Procedure

1. Drag the Rule processor onto the canvas and connect to it to the DriverAvgSpeed Aggregate processor:

5	My Applic	ations / IOT-Trucking-Ref-App
	Q 🕼	Last Change: 0s ago Version: CURRENT 🤤 🔍 🌣
Ð	EVENT HUBS	
Ø	100	
Ŷ	HDFS	
r	PROCESSOR	ge TruckSpeed ∢o1 ▶
	Σ	Add the Rule processor to the canvas and connect it to the
	AGGREGATE	DriverAvgSpeed Aggregate
	~	processor
	BRANCH	
	>≁	
	JOIN	

**2.** Configure the business rule as follows:

Add New Rule					×
RULE NAME*					
Speeding Driver					
DESCRIPTION*					
Driver who is speedir	ng excessively				
CREATE QUERY*	speed_AVG	× -	AN × –	80	× • +
QUERY PREVIEW:					
<pre>speed_AVG &gt; 80</pre>					
					Cancel Ok

### Results

The fully configured business rule should look similar to the following. Only high speed events continue on in the stream.

IsDriverSpeeding Only high speed								
CONFIGURATION NOTES								
Input	♣Add New Rules			Output				
driverId*				driverId*				
INTEGER driverName* STRING	Name	Condition	Actions	INTEGER driverName* STRING				
route* STRING	Speeding Driver	speed_AVG > 80	e 🖉	route* STRING				
speed_AVG* DOUBLE				speed_AVG* DOUBLE				
				Cancel Ok				

# **Transforming Data using a Projection Processor**

It is common to do transformations on the events in the stream. In our case, before we alert on the speeding driver, we want to convert the average speed we calculated in the aggregate processor into a integer from a double so it is easier to display in the alert. The projection processor allows you to do these transformations.

### Procedure

1. Drag the Projection processor onto the canvas and connect to it to the IsDriverSpeeding Rule processor:

Last Change:23s ago Version:CURRENT source version:CURRENT version:CURRENT source versio	
Source So	@ Q #
EVENT HUBS HDFS RULE * Starka *	
Image: Second state	
HDFS Be TruckGeoEv (01)	
PROCESSOR & TruckSpeed (01)	
Σ	
AGGREGATE	
BRANCH	

**2.** When you double-click on the projection processor, you see a number of out-of-the-box functions, however a Round function does not exist.

PROJECTION						×
CONFIGURATION	NOTES					
Input		PROJECTION FIELDS*				Output
driverId*		Select			$\overline{\mathbf{v}}$	
driverName* straing		FUNCTION	ARGUMENTS	FIELDS NAME		
route* string		Select × 🔺	Select	•	+	
speed_AVG*		UPPER				
		LOWER				
		INITCAP				
		SUBSTRING				
		CHAR_LENGTH				
		CONCAT				
						Cancel Ok

- **3.** Adding UDFs (User Defined Functions) is easy to do within SAM. Follow the below steps to add Round UDF function to SAM.
  - a. From the left-hand menu, click Configuration, then Application Resources.
  - **b.** Select the **UDF** tab and click the + sign to create the ROUND UDF. The jar for this UDF can be downloaded from here, located in the custom-udf folder. The simple java class used to implement this Round function using the SAM SDK can be found here. Unzip the downloaded artifact and use the jar called sam-custom-udf-0.0.5.jar. Configure the UDF with the following values:

Edit UDF	×
NAME *	
ROUND_AUTOCREATED	
DISPLAY NAME *	
ROUND_AUTOCREATED	
DESCRIPTION *	.r)
Rounds a double to integer using Math.Round	tr
TYPE *	0
FUNCTION	-
CLASSNAME *	tr
hortonworks.hdf.sam.custom.udf.math.Round	
UDF JAR *	У
Browse sam-custom-udf.jar	tr
Canc	el Ok o

c. After uploading the UDF, you should see the new Round UDF created.

ROUND	om Processor	UDF	Notifiers			•
Name	Description	Туре	Class Name	Argument Types	Return Type	Actions
ROUND	Rounds a double to integer	FUNCTION	hortonworks.hdf.sam.custom.udf.math.Round	DOUBLE	LONG	<ul> <li> <ul> <li></li></ul></li></ul>

- **4.** After creating the UDF, go back to your Application and double-click Projection Processor you added to the canvas. You will see ROUND\_AUTOCREATE in the FUNCTION drop down list.
- **5.** Configure the ROUND\_AUTOCREATE function as the following:

#### Results

Round									×
CONFIGURATION	NOTES								
Input		PROJECTION FIELDS*					SELECT ALL	Output	
driverId*	k-	× driverId × driverName	× route				× -	driverId*	
driverName*		FUNCTION	ARGUMENTS		FIELDS NAME			driverName* STRING	
route*		ROUND_AUTOCREATE	× speed_AVG	× -	speed_AVG_Round	+		route* STRING	
speed_AVG*								speed_AVG_Round*	
								Cancel	Ok

# Streaming Alerts to an Analytics Engine for Dashboarding

In addition to creating notification alerts, a common use case requirement is to send these alerts to a dashboard so they can be displayed and visualized. SAM offers this capability by allowing you to stream data into DRUID and then using Superset to create dashboards and visualizations.

## Procedure

1. Drag the Druid sink to the canvas and connect to it to the Round Projection.

ckGeoEv 4 01 ) • SIS_AU						
	4 01 ▶• →• Performed Provide August 200 Performance Provide August 200 Performance Per	4011	DriverAvgS «01»	->• sDriverSp RULE	€01 Determine the second s	401 ► S3-Speedin S3_AUTOCREATED
ckSpeed ∢01 )● sis_AU						

**2.** Stream these events into a Druid cube called alerts-speeding-drivers-cube by configuring the Druid processor like the following:

Alert-Speeding-Driver-	Cube
REQUIRED OPTIONAL N	OTES
Input driverId* INTEGER driverName* STRING speed_AVG* DOUBLE speed_AVG_Round* LONG	DATASOURCE NAME * alerts-speeding-drivers-cube ZOOKEEPER CONNECT STRING * secure-fenton-hdf1.field.hortonworks.com:2181,secure-f DIMENSIONS * * driverId * driverName * route * speed_AVG * speed_AVG_Round TIMESTAMP FIELD NAME * processingTime * WINDOW PERIOD * PT10M Cancel Ok

3. In the Creating Visualization Section, describe how to create dashboards for the alerts-speeding-drivers-cube.

# **Streaming Violation Events to an Analytics Engine for Descriptive Analytics**

All infraction events need to be available for descriptive analytic (dash-boarding, visualizations, etc.) by a business analyst. The analyst needs the ability to do analysis on the streaming data. The analytics engine in SAM is powered by Druid. The following steps show how to stream data into Druid, so that a business analyst can use the Stream Insight Superset module to generate descriptive analytics.

# Procedure

1. Drag the Druid processor to the canvas and connect it to the ViolationEvents Rule processor.

5	My Applio	cations / IOT-Trucking-Ref-App						
ф.	Q 🗭					Last Change:1s ago	Version:CURRENT	ତ୍ତ୍ର 🕈
æ	EVENT HUBS					<b>→•</b> <u>(</u>		
ⅆ	CP HDFS				••• €01 ▶ • → • 🔭 isDriverSp <01 ▶ • → • 🛋	Round 01 PROJECTION		
Ø	<b>%</b> Kafka	% TruckGeoEv ∢o1 ▶ • KAFKA JOIN	∢01 ▶ • → ● ⊕ BeventType	<b>4</b> 01▶●		L.	Dashboard 401 P	
r	PROCESSOR	8 TruckSpeed ∢01 ▶ ●			401▶			
	AGGREGATE							
	BRANCH							

2. Configure the Druid processor.

You can edit the ZooKeeper connect string in the advanced section of the Druid Service in Ambari, under the property druid.zk.service.host.

Violation-Events-Cu	be	The name of the insight data source/cube to which you × want to stream data.
REQUIRED OPTIONAL	NOTES	Business analysts use these data sources to query the data
Input	DATASOURCE N	AME *
eventTime* string	violation-eve	ents-cube
eventSource* string	ZOOKEEPER COI	NECT STRING *
truckld* INTEGER	secure-fento	on-hdf1.field.hortonworks.com:2181,secure-f
driverld* INTEGER driverName* STRING	DIMENSIONS *	× eventSource × truckId
routeld* INTEGER	× eventTime × driverId	× driverName × routeId × route
route* STRING	× eventType	× latitude × longitude × -
eventType* string	× correlation	d × geoAddress × speed
latitude*	TIMESTAMP FIE	LD NAME *
longitude*	processingT	ime 👻
correlationId*	WINDOW PERIO	) *
		Cancel Ok

3. Configure the Aggregator Info settings, under the OPTIONAL menu

Violation-Events-Cub	e	×
REQUIRED OPTIONAL	NOTES	
Input eventTime* string eventSource* string truckId* integer driverId* integer driverName* string routeId* integer routeld* integer string eventType* string latitude* DOUBLE	DRUID PARTITIONS   1   PARTITION REPLICATION   1   Aggregator Info +   Aggregator Info (Count Aggregator )   NAME *   Cnt	
correlationId*	Add a Count Aggregator and give it a name Cancel C	lk

# Streaming Violation Events into a Data Lake and Operational Data Store

Another common requirement is to stream data into an operational data store like HBase to power real-time web applications as well as a data lake powered by HDFS for long term storage and batch ETL and analytics.

# Procedure

1. Drag the HBase sink to the canvas and connect it to the ViolationEvents Rule processor.

5	My Applic	ations / IOT-Trucking-Ref-App			
4	Q 🕼			Last C	Change:1s ago Version:CURRENT 🔍 Q 🌣
æ	EVENT HUBS				
đ	HDFS	a Turkovo		Projection	4 01 > •
ک عر	<b>%</b> Kafka	% TruckGeoEV ∢01 ► KAFKA	∢01 ▶ • → ● ♥ ⊕ EventType	<ol> <li>♦ 01 &gt; ● ● ● Violation</li> <li>♦ 01 &gt;</li> </ol>	→ Dashboard · 4 01 ►
-		8 TruckSpeed € 01 ►		HBASE 4 01 M	
	AGGREGATE				

**2.** Configure the Hbase Sink using the following parameters.

Operational-Store-Vio	olation-Events	×
	NOTES	
Input eventTime*	WRITE TO WAL?	
STRING eventSource* STRING truckId*	eventTime	•
INTEGER driverId* INTEGER driverName*		
STRING routeld* INTEGER route* STRING		
eventType* STRING latitude* DOUBLE		
longitude* DOUBLE correlationId* LONG		
	Cancel	Ok

Operational-Store-Violation-Events					
REQUIRED OPTIONAL N	OTES				
Input eventTime* string eventSource* string truckId* INTEGER driverId* INTEGER driverName* string routeId* INTEGER string eventType*	CLUSTER NAME *   streamanalytics   HBASE TABLE *   default:violation_events   COLUMN FAMILY *   events   BATCH SIZE *   100				
string latitude* DOUBLE longitude* DOUBLE correlationId* LONG	Cancel Ok				

3. Drag the HDFS sink to the canvas and connect it to the ViolationEvents Rule processor.

5	My Applic	ations / IOT-Trucking-Ref-App							
	Q 🕼					Last C	hange:0s ago	Version:CURRENT	ତ୍ତ୍ତ୍
æ	EVENT HUBS						<b>↓</b> •	NOTIFICATION 4 01 >	
6	HDFS	sa TruckGanEv		► ∑ DriverAvgS. AGGREGATE		401 PROJECTION	<b>₫ 01 ┣ ●</b>		
ي بر	KAFKA PROCESSOR	80 TruckGeoEv ∢01 ► KARKA See TruckSpeed ∢01 ►	∢ 01 ▶ • → ● ● EventType	<ul> <li>€ 01 ▶ ••• → Violation</li> </ul>	4 01 ▶		<b>↓</b> ••	Dashboard 401	
		So KARKA		Derationa	… ∢01▶				
	BRANCH			HBASE	€01.▶				
	PMML								

**4.** Configure HDFS as below.

Make sure you have permission to write into the directory you have configured for HDFS path.

Data-Lake-HDFS	×
REQUIRED OPTIONAL NO	OTES
Input eventTime* string eventSource* string truckId* integer driverId* integer driverName* string routeId* integer string	PATH * /apps/trucking-app  FLUSH COUNT * 1000  ROTATION POLICY Time Based Rotation  ROTATION INTERVAL MULTIPLIER *
eventType* STRING latitude* DOUBLE longitude* DOUBLE correlationId* LONG	3 ROTATION INTERVAL UNIT * MINUTES OUTPUT FIELDS * Cancel Ok

# **Deploy a SAM Application**

# **Configure Deployment Settings**

Before deploying the application, you must configure deployment settings such as JVM size, number of ackers, and number of workers. Because this topology uses a number of joins and windows, you should increase the JVM heap size for the workers.

# Procedure

- 1. Click the gear icon at the top right corner of the canvas to display the Application Configuration dialog.
- 2. Increase Number of Workers to 5.
- 3. Set Topology Worker JVM Options to -Xmx3072m.

# Example

Application Configuration	×
GENERAL ADVANCED	
NUMBER OF WORKERS	
3	
NUMBER OF ACKERS	
1	
TOPOLOGY MESSAGE TIMEOUT (SECONDS)	
40	
TOPOLOGY WORKER JVM OPTIONS	
-Xmx3072m	
Cancel	)k

# **Deploy the App**

After you have configure the application's deployment settings, click the **Deploy** button at the lower right of the canvas.

Application Configuration	×
GENERAL ADVANCED	
NUMBER OF WORKERS	
3	
NUMBER OF ACKERS	
1	
TOPOLOGY MESSAGE TIMEOUT (SECONDS)	
40	
TOPOLOGY WORKER JVM OPTIONS	
-Xmx3072m	
Cancel	

During the deployment process, Streaming Analytics Manager completes the following tasks:

- 1. Construct the configurations for the different big data services used in the stream app.
- 2. Create a deployable jar of the streaming app.
- **3.** Upload and deploy the app jar to streaming engine server.

As SAM works through these tasks, it displays a progress bar.



**Building Application Jars** 

The stream application is deployed to a Storm cluster based on the Storm Service defined in the Environment associated with the application.

After the application has been deployed successfully, SAM notifies you and updates the button to red to indicate it is deployed. Click the red button to kill/undeploy the app.

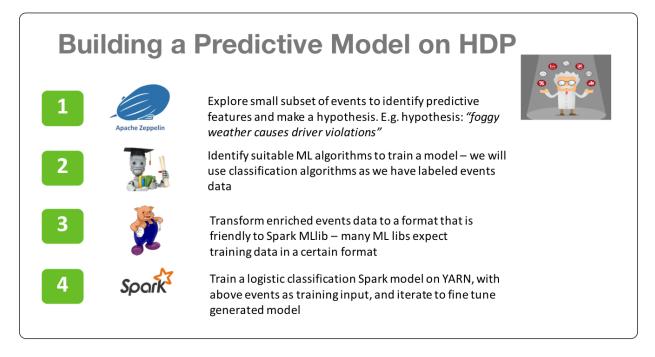
5	My Applica	ations / IOT-Trucking-Ref-App
ф	Q 🗭	Last Change:23s ago Version:CURRENT 🤨 📿 🌣
æ	EVENT HUBS	NOTFICATION 4 OI >
Ø	CP HDFS	DriverAvgS 401 ► • • • • • • • • • • • • • • • • • •
¢ بر	<b>&amp;</b> KAFKA	20 TruckSpeet (01) ● 30 Dashboard (01) ● 30 Dashboard (01) ● 30 Dashboard (01) ● 30 Dashboard (01) ●
		39 Martin Corporationa (01)
	AGGREGATE	Data-Lake 401 b
	→ JOIN	
	PMML	Status
»		

# **Advanced: Performing Predictive Analytics on the Stream using SAM**

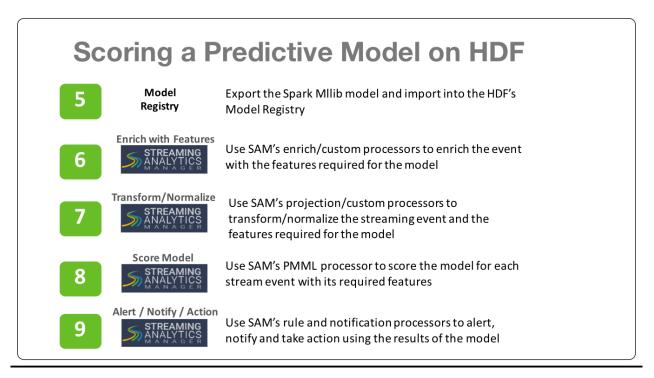
Requirement 10 of this use case states the following:

Execute a logistical regression Spark ML model on the events in the stream to predict if a driver is going to commit a violation. If violation is predicted, then alert on it.

HDP, the Hortonworks data at rest platform provides a powerful set of tools for data engineers and scientists to build powerful analytics with data processing engines like Spark Streaming, Hive, and Pig. The following diagram illustrates a typical analytics life cycle in HDP.



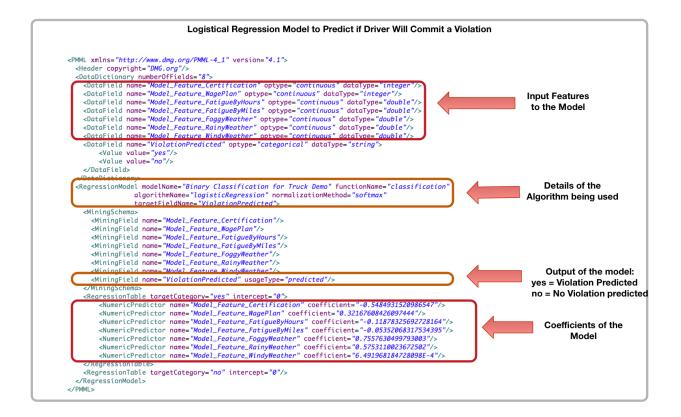
Once the model has been trained and optimized, you can create insights by scoring the model in real-time as events are coming in. The next set of steps in the life cycle score the model in real-time using HDF components.



In the next few sections we will walk through how to do steps 5 through 9 in SAM.

## **Logistical Regression Model**

In steps 1-4 with HDP, we were able to build a logistical regression model. The model was then exported into PMML. The following diagram illustrates the features, coefficients, and output of the model.



# Export the Model into SAM's Model Registry

SAM provides a registry where you can store PMML models. To get started with predictive analytics, upload this logistical regression model.

### Procedure

- 1. Download this PMML model and save it locally with an .xml extension.
- 2. Select the Model Registry menu item.
- 3. Click the + icon.
- **4.** Give your PMML model a name.
- 5. From Upload PMML File, select the PMML file you just downloaded.

×
_
)k

#### 6. Click Ok.

#### Results

The model is saved in the Model Registry.

5	Model Regis	try		
#			Search by name	<b>4</b>
æ		Model Name	PMML File Name	Actions
Ø		DriverViolationPredictionModel	$\label{eq:constraint} DriverViolationLogisticalRegessionPredictionModel-pmml.xml$	â
Ŷ				
ŗ				

## **Enrichment and Normalization of Model Features**

Now that the model has been added to the model registry, you can use it in the streaming application by the PMML processor. Before the model can be executed, you must enrich and normalize the streaming events with the features required by the model. As the above diagram illustrates, there are seven features in the model. None of these features come as part of the stream from the two sensors. So, based on the driverId and the latitude and longitude location, enrich the streaming event with these features and then normalize it required by the model. The table below describe each feature, enrichment store, and the normalization required.

Feature	Description	Enrichment Store	Normalization
Model_Feature_Certification	Identifies if the driver is certified or not	HBase/Phoenix table called drivers	"yes" # normalize to 1 "no" # normalize to 0
Model_Feature_WagePlan	Identifies if the driver is on an hourly or by miles wage plan	HBase/Phoenix table called drivers	"Hourly" # normalize to 1 "Miles" # normalize to 0
Model_Feature_Fatigue ByHours	The total number of hours driven by the driver in the last week	HBase/Phoenix table called timesheet	Scale by 100 to improve algorithm performance (e.g., hours/100)

Model_Feature_Fatigue ByMiles	The total number of miles driven by the driver in the last week	HBase/Phoenix table called timesheet	Scale by 1000 to improve algorithm performance (e.g.,miles/1000)
Model_Feature_Foggy Weather	Determines if for the given time and location, if the conditions are foggy	API to WeatherService	if (foggy) # normalize to 1 else 0
Model_Feature_Rainy Weather	Determines if for the given time and location, if the conditions are rainy	API to WeatherService	if (raining) -> normalize to 1 else 0
Model_Feature_Windy Weather	Determines if for the given time and location, if the conditions are windy	API to WeatherService	if (windy) # normalize to 1 else 0

# **Upload Custom Processors and UDFs for Enrichment and Normalization**

To perform the above enrichment and normalization, upload the custom UDFs and processors you downloaded in the previous section.

### Upload Custom UDFs

Steps for Uploading the Timestamp\_Long UDF

- 1. From the left-hand menu, click **Configuration**, then **Application Resources**.
- 2. Select the UDF tab and click the + sign to create the TIMESTAMP\_LONG UDF. This UDF will convert a string date time to a Timestamp long. The simple class for this UDF using the SAM SDK can be found here.

The jar for this UDF is located in SAM\_EXTENSIONS/custom-udf.

**3.** Configure the UDF with the following values:

Edit UDF	×
NAME *	
TIMESTAMP_LONG_AUTOCREATED	
DISPLAY NAME *	
TIMESTAMP_LONG_AUTOCREATED	
DESCRIPTION *	
Converts a string date time to a Timestamp Long	
TYPE*	
FUNCTION	~
CLASSNAME *	
hortonworks.hdf.sam.custom.udf.time.ConvertToTimestampLong	
UDF JAR *	
🗁 Browse sam-custom-udf.jar	
	Cancel Ok

Steps for Configuring the Get\_Week UDF

1. Select the UDF tab and click the + sign to create the GET\_WEEK UDF.

The jar for this UDF is located in SAM\_EXTENSIONS/custom-udf. This UDF will convert a timestamp string into the week of the year which is required for an enrichment query. The simple class for this UDF using the SAM SDK can be found here.

**2.** Configure the UDF with the following values:

Edit UDF	×
NAME *	
GET_WEEK_AUTOCREATED	
DISPLAY NAME *	
GET_WEEK_AUTOCREATED	
DESCRIPTION *	
For a given date time string, the functions returns the week of the date/time	
TYPE*	
FUNCTION	*
CLASSNAME *	
hortonworks.hdf.sam.custom.udf.time.GetWeek	
UDF JAR *	
🕞 Browse sam-custom-udf.jar	
Cancel	Ok

### **Upload Custom Processors**

Steps for Uploading the ENRICH-PHOENIX Custom Processor

- 1. From the left-hand menu, click Configuration, then Application Resources.
- 2. Select Custom Processor and click the + sign to create the ENRICH-PHOENIX processor.

Configure the processor with the following values. This processor can be used to enriched streams with data from Phoenix based on a user supplied SQL statement. The java class for this processor using the SAM SDK can be found here.

BIGGRIPPINA CLISIMULA CLISIMU	NAME*	ENRICH-PHOEM	NIX_AUTOCREATED				_	
UPLOAD JAR* C Brown ann-outdom-processor-jar-with-dependencies.jar CONFIG FIELS Field Name VI Name optional Type Value Tootip Notice According Field According Field Field Name VI Name optional Type Value Tootip Cookeeper server ut in the format of \$FQDN_ZK_HOST_\$ZK_PORT in the entit of the output field names to store new enriched values in the entit of the output field names to store new enriched values in the format of \$FQDN_ZK_HOST_\$ZK_PORT in the entit of the output field names to store new enriched values in the format of \$FQDN_ZK_HOST_\$ZK_PORT in the entit of the output field names to store new enriched values in the format of \$FQDN_ZK_HOST_\$ZK_PORT in the entit of the output field names to store new enriched values in the format of \$FQDN_ZK_HOST_\$ZK_PORT in the entit of the output field names to store new enriched values in the format of \$FQDN_ZK_HOST_\$ZK_PORT in the entit of the output field names to store new enriched values in the format of \$FQDN_ZK_HOST_\$ZK_PORT in the entit of the output field names to store new enriched values in the fields is accoreCluater in the format of \$FQDN_ZK_HOST_\$ZK_PORT in the entit of the output field names to store new enriched values is accoreCluater in the format of \$FQDN_ZK_HOST_\$ZK_PORT in the entit of the teres Cluater in the output field names to store new enriched values is accoreCluater in the format of \$FQDN_ZK_HOST_\$ZK_PORT in the entit of the teres Cluater in the output field names to store new enriched values is accorecluater in the format of \$FQDN_ZK_HOST_\$ZK_PORT in the entit of the teres Cluater in the output field names to store new enriched values is accorecluater in the format of \$FQDN_ZK_HOST_\$ZK_PORT in the output field names to store new enriched values is accorecluater in the format of \$FQDN_ZK_HOST_\$ZK_PORT in the output field names to store new enriched values is accorecluater in the format of \$FQDN_ZK_HOST_\$ZK_PORT in the output field names to store new enriched values is accorecluater in the format of \$FQDN_ZK_HOST_\$ZK_PORT in the output field n	DESCRIPTION*	Enriches the inj	out schema with data from P	hoenix base	ed on us	er supplied		
CONTO FIELDS CONTO FIELDS CONTOURS AND	CLASSNAME*	hortonworks.hd	lf.sam.custom.processor.en	rich.phoeni>	.Phoeni	ixEnrichmen		
Field Name       Ul Name       optional       Type       Defailt       Tookip       Actions         ZkServerUvl       Phoenix Zookeeper Conn       false       strin       Q       Cookeeper server url in the format of SFQDN,ZK,HOST,SZK,PORT       C         enrich-MentSQL       Enrichment SQL       false       strin       Q       SQL to execute for the enriched values       C         enrich-BdOutput       Enrichment Output Fields       false       gtrin       Q       The output field names to store new enriched values       C       C         secure Cluster       false       gtrin       ge       Check if connecting to a secure HBase/Phoenix Cluster       C       C         rincipal       Kerberos Client Principal       true       gtrin       Q       Retheros Key Tab File       C       C         rincipal       Kerberos Key Tab File       true       gtrin       Q       Retheros Key Tab File location on each of the worker nodes for the equi Principal configured       C       C         rincipal       Kerberos Key Tab File       true       gtrin       Q       Retheros Key Tab File location on each of the worker nodes for the equi Principal configured       C         rincipal       Kerberos Key Tab File       true       gtrin       Q       Kerberos Key Tab File location on each of the worker	UPLOAD JAR*	🗁 Browse 🛛 📢	am-custom-processor-jar	-with-depe	ndenci	es.jar		
Field Name       Ul Name       Option       Type       Value       Tooltip       Actions         RkServerUrl       Phoenix Zookeeper Conn       false       g <sup>1</sup> Zookeeper server url in the format of \$FQDN_ZK_HOST;SZK_PORT       g <sup>1</sup> enrichmentSQL       Enrichment SQL       false       g <sup>1</sup> Q       SQL to execute for the enriched values       g <sup>1</sup> enrichedOutput       Enrichment Output Fields       false       strin       g <sup>1</sup> The output field names to store new enriched values       g <sup>1</sup> secureCluster       Secure Cluster       false       b <sup>1</sup> false       b <sup>1</sup> the principal uses to connect to secure HBase/Phoenix Cluster. Requi       g <sup>1</sup> kerberosClientPincipal       Kreberos Key Tab File       true       g <sup>1</sup> the principal uses to connect to secure HBase/Phoenix Cluster. Requi       g <sup>1</sup> kerberosKeyTab       Kerberos Key Tab File       true       g <sup>1</sup> the principal uses to connect to secure HBase/Phoenix Cluster. Requi       g <sup>1</sup> NPUT SCHEMA •       1       Image: Secure Cluster       g <sup>1</sup> the principal uses to connect to secure HBase/Phoenix Cluster. Requi       g <sup>1</sup> NPUT SCHEMA •       1       Image: Secure Cluster       g <sup>1</sup> the principal use to configure       g <sup>1</sup> g <sup>1</sup> NPU	CONFIG FIELDS	Add Config Fields						
Image: Solution of the solutio		Field Name	UI Name	Optional	Туре		Tooltip	Actions
Image: Secure Cluster       G       G       The output field names to store new enriched values       G         Secure Cluster       Secure Cluster       false       bool       false       Check if connecting to a secure HBase/Phoenix Cluster       G         Kerberos Client Principal       true       strin       g       G       The principal uses to connect to secure HBase/Phoenix Cluster. Reque       G         Kerberos KeyTab       Kerberos Key Tab File       true       strin       G       Kerberos Key Tab File       false       strin       G       Kerberos Key Tab File       G       Kerberos Key Tab File       false       strin       G       Kerberos Key Tab File       G       Kerberos Key Tab File       false       Strin       G       Kerberos Key Tab File       G       Kerberos Key Tab File       false       Strin       G       Kerberos Key Tab File       G       Kerberos Key Tab File       false       false       Kerberos Key Tab File       false       fals		zkServerUrl		false			Zookeeper server url in the format of \$FQDN_ZK_HOST:\$ZK_PORT	
Fields       Image: Source Cluster       false       bool       false       Check if connecting to a secure HBase/Phoenix Cluster       false         kerberos ClientP       Kerberos Client Principal       true       strin       g       The principal uses to connect to secure HBase/Phoenix Cluster. Requi       false         kerberos KeyTab       Kerberos KeyTab File       true       strin       g       Image: Source Cluster       false       Check if connecting to a secure HBase/Phoenix Cluster. Requi       false         INPUT SCHEMA ©       Kerberos KeyTab File       true       strin       g       Image: Source Cluster       false       Check if connecting to a secure HBase/Phoenix Cluster. Requi       false         INPUT SCHEMA ©       Kerberos Key Tab File       true       strin       g       Image: Source Cluster       false       Check if connecting to a secure HBase/Phoenix Cluster. Requi       false         INPUT SCHEMA ©       Kerberos Key Tab File       true       strin       strin       strin       false       false       cherberos Key Tab File       false		enrichmentSQL	Enrichment SQL	false			SQL to execute for the enriched values	
Import Scheme I			Enrichment Output Fields	false			The output field names to store new enriched values	
INPUT SCHEMA O		secureCluster	Secure Cluster	false		false	Check if connecting to a secure HBase/Phoenix Cluster	
INPUT SCHEMA I O			Kerberos Client Principal	true				
			Kerberos Key Tab File	true			Kerberos Key Tab File location on each of the worker nodes for thee pri ncipal configured	
	OUTPUT SCHEMA							
				С	ancel	Save		

#### ENRICH-PHOENIX Configuration Values

- Streaming Engine Storm
- Name ENRICH-PHOENIX
- Description Enriches the input schema with data from Phoenix based on user supplied SQL
- ClassName hortonworks.hdf.sam.custom.processor.enrich.phoenix.PhoenixEnrichmentSecureProcessor
- Upload Jar The jar for this custom processor can be found under SAM\_EXTENSIONS/custom-processor/ sam-custom-processor-jar-with-dependencies.jar

Click the Add Config Fields button and the following three configuration fields:

• Add a config field called **zkServerUrl** with the following values:

- a. Field Name zkServerUrl
- b. UI Name Phoenix ZooKeeper Connection URL
- c. Optional false
- d. Type string
- e. ToolTip ZooKeeper server URL in the format of \$FQDN\_ZK\_HOST:\$ZK\_PORT
- Add a config field called **enrichmentSQL** with the following values:
  - a. Field Name enrichmentSQL
  - b. UI Name Enrichment SQL
  - c. Optional false
  - **d. Type** string
  - e. ToolTip SQL to execute for the enriched values
- Add a config field called **enrichedOutputFields** with the following values:
  - a. Field Name enrichedOutputFields
  - b. UI Name Enrichment Output Fields
  - c. Optional false
  - d. Type string
  - e. ToolTip The output field names to store new enriched values
- Add a config field called **secureCluster** with the following values:
  - a. Field Name secureCluster
  - b. UI Name Secure Cluster
  - **c. Optional** false
  - d. Type boolean
  - e. ToolTip Check if connecting to a secure HBase/Phoenix Cluster
- Add a config field called kerberosClientPrincipal with the following values:
  - a. Field Name kerberosClientPrincipal
  - b. UI Name Kerberos Client Principal
  - c. Optional true
  - d. Type string
  - e. ToolTip The principal uses to connect to secure HBase/Phoenix Cluster. Required if secureCluster is checked.
- Add a config field called **kerberosKeyTabFile** with the following values:
  - a. Field Name kerberosKeyTabFile
  - **b.** UI Name Kerberos Key Tab File
  - c. Optional true
  - **d. Type** string
  - e. ToolTip Kerberos Key Tab File location on each of the worker nodes for the configured principal

Steps for Uploading the ENRICH-WEATHER Custom Processor

- 1. Select **Custom Processor** and click the + sign to create the ENRICH-WEATHER processor. This processor can be used to enrich streams with weather data based on time and lat/long location. The java class for this processor using the SAM SDK can be found here.
- **2.** Configure the processor with the following values.

Configuration / /	Application Resources							
	Custom Processor UDF	Notifiers						
	STREAMING ENGINE*	STORM						
	NAME*	ENRICH-WEATHER_AUTOC	REATED					
	DESCRIPTION*	Enrichment with normalize	d weather data required for the model					
	CLASSNAME*	hortonworks.hdf.sam.cust	om.processor.enrich.weather.WeatherEnric	chmentProcesso				
	UPLOAD JAR*	Browse CustomProc	essor.jar					
	CONFIG FIELDS	Add Config Fields						
		Field Name	UI Name	Optional	Туре	Default Value	Tooltip	Actions
		weatherServiceURL	Weather Web Service URL	false	string	http://weather.com/api?lat=\${latitude}&lng=\${longitude}	The URL to the Weather Web Service	/ 0
	Revit Schema <b>O</b>	1 [ 2 { 3 'name': 'dr. 4 'type': 'lr. 5 'optional': 6 }, 7 { 8 'name': 'la 9 'type': 'Do 10 'optional': 1 }, 12 { 13 'type': 'Do 15 'optional': 15 'optional': 16 'type': 'Do 17 'type': 'Do 18 'type': 'Do 18 'type': 'Do 18 'type': 'Do 18 'type': 'Do 19 'type': 'Do 10 'type:'type': 'Do 10 'type:'type': 'Do 10 'type:'type': 'Do 10 'type:'type': 'Do 10 'type:'type:'type:'type': 'Do 10 'type:'type:'type:'type:'type:'type': 'Do 10 'type:'ty	RGER', false ;iude', /RLE', false ngitude', /RLE',			✓ I CLEAR		
	OUTPUT ISOBAA 0	<pre>4 "type": "DOI 5 "optional": 6 }, 7 { 8 "name": "Noo 9 "type": "DOI 10 "optional": 11 }, 12 {</pre>	<pre>false lel_Feature_RainyWeather", stE*, false lel_Feature_WindyWeather", stE*,</pre>			↓ CLEAR		

**ENRICH-WEATHER Configuration Values** 

- Streaming Engine Storm
- Name ENRICH-WEATHER
- **Description** Enrichment with normalized weather data for a geo location
- ClassName hortonworks.hdf.sam.custom.processor.enrich.weather.WeatherEnrichmentProcessor
- Upload Jar The jar for this custom processor can be found under SAM\_EXTENSIONS/custom-processor/samcustom-processor.jar

Click the **Add Config Fields** button and a configuration field with the following values:

- **Field Name** weatherServiceURL
- **UI Name** Weather Web Service URL
- **Optional** false
- **Type** string
- **Tooltip** The URL to the Weather Web Service

Input and Output Schema for ENRICH-WEATHER

- Copy this input schema and paste into the INPUT SCHEMA text area box
- Copy this output schema and paste into the OUTPUT SCHEMA text area box

Steps for Uploading the NORMALIZE-MODEL-FEATURES Custom Processor

- **1.** Select the **Custom Processor** tab and click the + sign to create the NORMALIZE-MODEL-FEATURES processor. This processor normalizes the enriched fields to a format that the model is expecting.
- **2.** Configure the processor with the following values:

Configuration / Application Re	sources								
STREAMING EM	NGINE*	TORM							
NAME*	NO	ORMALIZE-MODEL-FEATURES	DELAY_AUTOCREATED						
DESCRIPTION*	No	iormalize the features of the mo	odel before passing it to model with option to cause laten	у					
CLASSNAME*	hor	ortonworks.hdf.sam.custom.pr	ocessor.enrich.driver.predictivemodel.FeatureNormalizati	onWithDelayProcessor					
UPLOAD JAR*	B	Browse CustomProcessor	jər						
CONFIG FIELDS	S Add	dd Config Fields							
		Field Name	UI Name		Optional	Туре	Default Value	Tooltip	Actions
	dela	ayTimeOutSecs	Timeout Delay for Monitoring Use Case (Seconds)		true	number	D	timeout delay in seconds	/ 0
1840T 5046A	• •	•				CLEAR			
OUTFUT SCHB		<pre>4 "type": "DOUBLE' 5 "optional": fal: 6 }, 7 { 8 "name": "Model_] 9 "type": "DOUBLE' 10 "optional": fal: 11 }, 12 {</pre>	ee Peature_RainyWeather", 'ee Peature_MindyWeather",		21	CLEAR			

NORMALIZE-MODEL-FEATURES Configuration Values

- Streaming Engine Storm
- Name NORMALIZE-MODEL-FEATURES
- Description Normalize the features of the model before passing it to model
- ClassName -

hortonworks.hdf.sam.custom.processor.enrich.driver.predictivemodel.FeatureNormalizationProcessor

• Upload Jar – The jar for this custom processor can be found under SAM\_EXTENSIONS/custom-processor/samcustom-processor.jar

Input and Output Schema for NORMALIZE-MODEL-FEATURES

• Copy this output schema and paste into the OUTPUT SCHEMA text area box

Result

You have uploaded three custom processors required to do enrichment of the stream and normalization of the enriched values to feed into the model.

Search by name	٩		
Name	Description	Jar File Name	Actions
ENRICH-PHOENIX	Enriches the input schema with data from Phoenix based on user supplied SQL	sam-custom-processor-0.0.5-jar-with-dependencies.jar	× ±
ENRICH-WEATHER	Enrichment with normalized weather data for a geo location	sam-custom-processor-0.0.5.jar	× ±
NORMALIZE-MODEL-FEATURES	Normalize the features of the model before passing it to model	sam-custom-processor-0.0.5a.jar	e 🖉

If you go back to the Stream Builder, you will see three new custom processors on the palette.



# Scoring the Model in the Stream using a Streaming Split Join Pattern

Now that you have created the enrichment store, loaded the enrichment data, and uploaded the custom UDFs and processors to SAM, build the stream flow to score the model in real-time. In this case, you want to predict violations for events that are not blatant infractions.

### Procedure

- **1.** Click into the Trucking IOT application you built.
- 2. Double-click the Event Type rule processor to display the Add New Rule dialog.
- **3.** Configure the new rule with the following values:

Add New Rule				2
RULE NAME*				
Non Violation Eve	ents			
DESCRIPTION*				
Events that are no	ot violations that we want	t to do predictions on		
CREATE QUERY*	eventType	× 👻 EQUALS	× 💌 'Normal'	× -
QUERY PREVIEW:				
eventType = 'N	lormal'			
				Cancel Ok

### Results

Your new rule is added to the Event Type processor.

ONFIGURATION NOT	ES			
nput	+Add New Rules			Output
eventTime*				eventTime*
eventSource*	Name	Condition	Actions	eventSource*
truckId*	ViolationEvents	eventType <> 'Normal'	e 🖉	truckId* INTEGER
driverId*	Non Violation Events	eventType = 'Normal'	e 🖉	driverId*
driverName* STRING				driverName* STRING
routeld* INTEGER				routeld*
route* STRING				route* STRING
eventType* STRING				eventType* string
latitude*				latitude*
longitude*				longitude*
correlationId*				correlationId*

## **Streaming Split Join Pattern**

About This Task

Your objective is to perform three enrichments:

- Retrieve a driver's certification and wage plan from the driver's table.
- Retrieve the driver's hours and miles logged from the timesheet table.
- Query weather information for a specific time and location.

To do this, use the split join pattern to split the stream into three, perform the enrichment in parallel, and then re-join the three streams.

Steps for Creating a Split Join Key

1. Create a new split key in the stream which allows you to join in a common field when you join the three stream.

To do this, drag the projection processor to the canvas and create a connection from the EventType rule processor to this projection processor.

When configuring the connection, select the Non Violation Events Rule which tells SAM to only send non-violation events to this project processor.

Event	Type-PROJECTION	×
STREAM	ID*	
rule_tr	ansform_stream_3	•
FIELDS		
1	t	
2	{	
3	"name": "eventTime",	
4	"type": "STRING",	
5	"optional": false	
6	},	
7	{	
8	"name": "eventSource",	
9	"type": "STRING",	
10	"optional": false	
11	},	
12	{	
13	"name": "truckId",	
14	"type": "INTEGER", "optional": false	
15	"optional": talse	
RULES*		
Non V	iolation Events	•
GROUPIN	G*	
SHUF	E	•
	Cancel	k

**2.** Configure the projection processor to create the split join key called splitJoinValue using the custom UDF you uploaded earlier called "TIMESTAMP\_LONG".

You will also do a transformation which calculates the week based on the event time which is required for one of the enrichments downstream. Configure the processor with the following parameters:

ONFIGURATION NOTES					
nput	PROJECTION FIELDS*				Output
eventTime* string eventSource*	× eventTime × eventSourd × latitude × longitude	ce × truckld × driverlo × correlationId × geoAdd		ute × eventType	driverName* STRING routeId*
STRING truckld* INTEGER	FUNCTION	ARGUMENTS	FIELDS NAME		INTEGER route* STRING
driverId*	TIMESTAMP_LONG $\times$ $\neg$	× eventTime	✓ splitJoinValue	+	eventType* STRING
driverName* STRING	GET_WEEK × 👻	× eventTime	week	+ 🛍	latitude*
routeld* INTEGER					longitude*
route* STRING					correlationId*
eventType* string					geoAddress* STRING
latitude*					speed*
longitude*					splitJoinValue*
correlationId*					week*

Steps for Splitting the Stream into Three to Perform Enrichments in Parallel

1. With the split join key created, you can split the stream into three to perform the enrichments in parallel.

To do the first split to enrichment the wage and certification status of driver, drag the "ENRICH-PHOENIX" processor to the canvas and connect it from the Split project processor.

- 2. Configure the enrich processor with the following parameters:
  - a. ENRICHMENT SQL: select certified, wage\_plan from drivers where driverid=\${driverId}
  - b. ENRICHMENT OUTPUT FIELDS: driverCertification, driverWagePlan
  - c. SECURE CLUSTER: false
  - d. OUTPUT FIELDS: Click Select All.
  - e. NEW OUTPUT FIELDS: Add new output fields for the two enriched values: driverCertification and driverWagePlan.

After this processor executes, the output schema will have two fields populated called driverCertification and driverWagePlan.

×

	KE	ERBEROS KEY TAB FILE				Quitaut
nput	C					Output
eventTime* STRING						eventTime* STRING
eventTimeLong*	OL	JTPUT FIELDS			SELECT ALL	eventSource* STRING
eventSource* STRING		× eventTime × eventSo				truckId*
truckld*		× driverName × routele × longitude × correlatio			atitude 🗸 👻	driverId*
driverId*		× splitJoinValue × wee				driverName* STRING
driverName* string						routeld*
routeld*	NE	EW OUTPUT FIELDS +				route* STRING
route* STRING		Field Name	Туре	Optional	Actions	eventType* STRING
eventType* string		driverCertification	STRING	false	/ ±	latitude*
latitude* DOUBLE		driverWagePlan	STRING	false	/ <b>û</b>	longitude*
longitude*		anvenvagen lan	011110	Tuise		correlationId*

**3.** Create the second stream to enrich the drivers hours and miles logged in last week by dragging another "ENRICH-PHOENIX" processor to the canvas and connecting it from the Split projection processor.

5	My Applic	ations / IOT-Trucking-Ref-App						
4	Q 🕼					Last Change:0s ago	Version:CURRENT	ତ୍ତ୍ ପ୍ 🌣
æ	EVENT HUBS					<b> </b> *•	NOTIFICATI ∉ 01 ▶ NOTIFICATION	
đ	COP HDFS			► DriverAvgS	401 ► → ● Se isDriverSp < 01 ► ● ■	Round PROJECTION 01		
Ŷ	<b>%</b> KAFKA	KAFKA     KAFKA     JOIN     JOIN	∢01 » • → • 💝 EventType	<ul> <li>€ 01 ▶ ••• → Violation</li> </ul>	<b>4</b> 01 <b>▶</b>	L.	Dashboard 401	
ŗ	PROCESSOR	& TruckSpeed (01)	TIOLE					
	AGGREGATE			HBASE	401			
	BRANCH			Data-Lake	401▶			
				. 0.15	ENRICH-HR ENRICH-PHOENIX 401 •			
	PMML				COLOR ENRICH-PHOENIX			
	•<							
	PROJECTION							

- 4. Configure the enrich processor with the following parameters:
  - **a.** ENRICHEMNT SQL: select hours\_logged, miles\_logged from timesheet where driverid= \${driverId} and week=\${week}
  - b. ENRICHMENT OUTPUT FIELDS: driverFatigueByHours, driverFatigueByMiles
  - c. SECURE CLUSTER: false
  - **d.** OUTPUT FIELDS: Select the splitJoinValue field.
  - e. NEW OUTPUT FIELDS: Add new output fields for the two enriched values driverFatigueByHours and driverFatigueByMiles.

 $\times$ 

ENRICH-Timesheet

CONFIGURATION	NOTES					
Input						Output
eventTime* string						splitJoinValue*
eventTimeLong*		KERBEROS KEY TAB FILE	driverFatigueByHours* straing			
eventSource*						driverFatigueByMiles* STRING
truckld*		OUTPUT FIELDS				
driverId* INTEGER		× splitJoinValue				
driverName* STRING		NEW OUTPUT FIELDS				
routeld*						
route* string		Field Name	Туре	Optional	Actions	
eventType* string		driverFatigueByHours	STRING	false	P 🖞	
latitude*		driverFatigueByMiles	STRING	false	/ Î	
longitude*						
						Cancel Ok

After this processor executes, the output schema will have two fields populated called driverFatigueByHours and driverFatigueByMiles.

**5.** Create the third stream to do weather enrichment by dragging the custom processor you uploaded called "ENRICH-WEATHER" processor to the canvas and connect it from the Split project processor.

5	My Applic	ations / IOT-Trucking-Ref-App						
4	Q 🕼				Las	t Change: <b>0s ago</b> Ve	rsion:CURRENT	ତ୍ତ୍ର 🕈
B	EVENT HUBS						OTIFICATI 401	
ൾ	60 HDFS			DriverAvgS	€01 P • → • ** isDriverSp €01 P • ■ Round Project	d ction 401 b		
Ŷ	<b>&amp;</b> KAFKA		∢ 01 » • → • 🗞 EventType	401 ▶ • • • → Violation DRUID	401▶	<b>↓•</b> ∋ }	Dashboard 4 01 D	
r	PROCESSOR	KAFKA € 01 €						
	AGGREGATE			HBASE	≪ 01 ▶			
	BRANCH			Data-Lake	401 1			
	>+				ENRICH-HR ENRICH-PHOENIX      OI			
	JOIN			PROJECTION	401 ▶ ■ ■ ■ ENRICH-TIM			
	PMML				ENRICH-WEATHER			
	PROJECTION							
	20							

- **6.** Configure the weather process with the following parameters (currently the weather processor is just a stub that generates random normalized weather info).
  - a. WEATHER WEB SERVICE URL: http://weather.com/api?lat=\${latitude}&lng=\${longitude}
  - **b.** INPUT SCHEMA MAPPINGS: Leave defaults
  - c. OUTPUT FIELDS: Select the splitJoinValue and the three model enriched features

ENRICH-WEATHER				3
CONFIGURATION NOT	ES			
Input	WEATHER WEB SERVIC	E URL *		Output
eventTime*	http://weather.com	n/api?lat=\${latitude}&lng=\${long	gitude}	splitJoinValue*
eventTimeLong*	INPUT SCHEMA MAPP	ING		Model_Feature_FoggyWeather
eventSource* string	driverId	driverId	× -	Model_Feature_RainyWeather
truckId*	latitude	latitude	× -	Model_Feature_WindyWeather
driverId*	to a straight			
driverName* string	longitude	longitude	× •	
routeld*	OUTPUT FIELDS*		SELECT ALL	
route* STRING	× splitJoinValue	× Model_Feature_FoggyWeather	× •	
eventType* straing	× Model_Feature_Ra	ainyWeather × Model_Feature_W	indyWeather	
latitude*				
longitude*				
				Cancel Ok

After this processor executes, the output schema will have three fields populated called Model\_Feature\_FoggyWeather, Model\_Feature\_RainyWeather, and Model\_Feature\_WindyWeather.

Steps for Rejoining the Three Enriched Streams

1. Now that you have done the enrichment in parallel by splitting the stream into three, you can now join the three streams by dragging the join processor to the canvas and connecting the join from the three streams.

S My	plications / IOT-Trucking-Ref-App
A sou	Last Change: Us ago Version: CURRENT 🔍 🔍 🖓
	es
	► DriverAvgS 401 → ● ● isDriverSp 401 → ● ● Kale Rate 401 → ■ Avg Reader
e ș	
F PROCE	% TruckSpeed
AGGRE	Te Departiona (01)
~	igentiate for the state of th
BRAI	► Standard (1) •
oc Billio	BOLTERION     ADD     ADD
PM	PRI ENDICHAVEA
PROJE	
e e e e e e e e e e e e e e e e e e e	

Configure the join processor like the following where you use the joinSplitValue to join all three streams.
 For the Output field, click SELECT ALL to select all the fields across the three streams.

JOIN-ENRICHMENTS					×
CONFIGURATION NOTES					
Input	SELECT STREAM*	SELECT FIELD WITH*			Output eventTime*
splitJoinValue*	JOIN TYPE*	SELECT STREAM*	SELECT FIELD*	WITH STREAM*	eventSource* string truckld*
Model_Feature_FoggyWeather* DOUBLE Model_Feature_RainyWeather*	INNER -	custom_processor_stre=	splitJoinValue	custom_processor_stre.	INTEGER driverld* INTEGER driverName*
Model_Feature_WindyWeather*		custom_processor_strem	splitJoinValue ~	custom_processor_stre	string routeld*
	WINDOW TYPE* Processing Time				route* strang eventType* strang
	WINDOW INTERVAL*		SLIDING INTERVAL		Iatitude*
	4	Seconds 👻	4	Seconds 👻	longitude*
		OUTDUT FIELDO ADE MAMO ATODA	,	SELECT ΔΗ	correlationId*

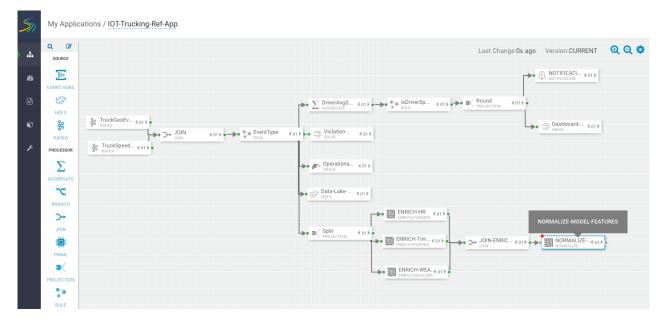
**3.** Now that you have joined three enriched streams, normalize the data into the format that the model expects by dragging the "NORMALIZE-MODEL-FEATURES" custom processor that you added to the canvas.

For the output fields, select all the fields and leave the mapping as defaults.

ONFIGURATION NO	DTES	
eventTime* strang eventSource* strang truckld* intreger driverId* intreger driverName* strang routeld* intreger routed* strang eventType* strang Doube	TIMEOUT DELAY FOR MONITORING USE CASE (SECONDS)	Output eventTime* strang eventSource* strang truckId* INTEGER driverId* INTEGER driverName* strang routeId* INTEGER routed* strang eventType* strang latitude* DOUBLE
longitude* DOUBLE correlationId*		longitude* DOUBLE correlationId*

### Result

Your flow looks similar to the following.

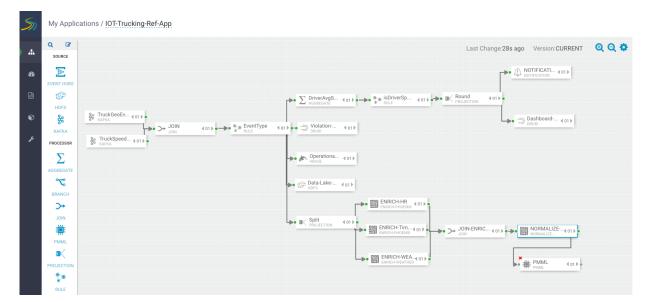


## Score the Model Using the PMML Processor and Alert

Now you are ready to score the logistical regression model.

#### Procedure

1. Drag the PMML processor to the canvas and connect it to the Normalize processor.



2. Configure the PMML processor like the following by selecting the DriverViolationPredictionModel that you uploaded earlier to the **Model Registry**.

After this processor executes, a new field called **ViolationPredicted** is added to stream for the result of the prediction. In output fields, select all the contextual fields you want to pass on including the model value result.

Predict		×
CONFIGURATION NOTES		
Input eventTime* string eventSource* string truckld* integer driverld* integer driverName* string routeld* integer routeld*	MODEL NAME* DriverViolationPredictionModel  OUTPUT FIELDS* SELECT ALL  * eventTime * eventSource * truckId * driverId  * driverName * routeId * route * latitude * longitude *  geoAddress * speed	Output ViolationPredicted* strang eventTime* strang eventSource* strang truckId* INTEGER driverName* strang routeId* INTEGER
eventType* sTRING latitude* DOUBLE correlationId* LONG		route* strand latitude* DOUBLE longitude* DOUBLE geoAddress* strand Cancel Ok

**3.** Determine if the model predicted if the driver will commit a violation by dragging a rule processor to the canvas and configuring a rule like the following:

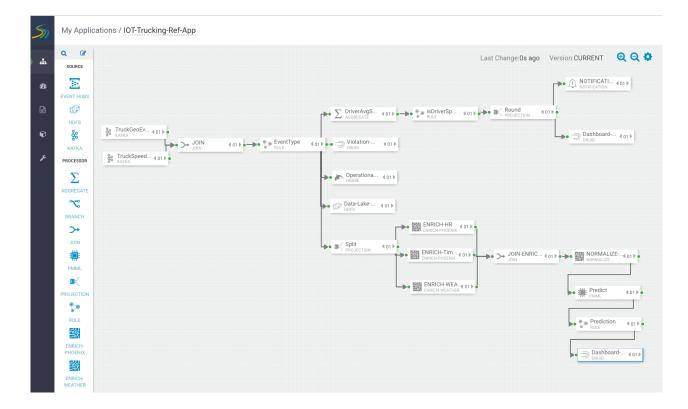
Edit Rule				×
RULE NAME*				
Violation Predict	ed			
DESCRIPTION*				
model returned a	prediction			h
CREATE QUERY*	ViolationPredicted	× 👻 EQUALS	× 👻 'yes'	× 👻 🕇
QUERY PREVIEW:				
ViolationPredi	cted = 'yes'			
				Cancel Ok

- **4.** If a violation is predicted, send it to a Druid to display on a dashboard. Drag the Druid processor to canvas and configure.
- 5. Stream the events into a cube called **alerts-violation-predictions-cube**.

Dashboard-Predictions	s ×
REQUIRED OPTIONAL N	OTES
Input	DATASOURCE NAME *
ViolationPredicted*	alerts-violation-predictions-cube
eventTime* string	ZOOKEEPER CONNECT STRING *
eventSource* string	secure-sam-hdf2.field.hortonworks.com:2181,secure-sar
truckld* INTEGER	DIMENSIONS *
driverId*	× ViolationPredicted × eventTime
driverName* string	× eventSource × truckId × driverId
routeld*	× driverName × routeld × route × -
route* STRING	× eventType × latitude × longitude × correlationId × geoAddress × speed
eventType* string	
latitude*	TIMESTAMP FIELD NAME *
longitude*	processingTime -
	Cancel Ok

# Results

The final flow looks like the following:



# **Creating Visualizations Using Superset**

A business analyst can create a wide array of visualizations to gather insights on streaming data. The platform supports over 30+ visualizations the business analyst can create. For visualization examples, see the Gallery of Superset Visualizations.

The general process for creating and viewing visualizations is as follows:

- 1. Whenever you add new data sources to Druid via a Stream App, perform the Refresh Druid Metadata action on the **Superset** menu.
- 2. Using the Superset Stream Insight UI, create one or more "slices". A slice is one business visualization associated with a data source (for example, Druid cube).
- 3. Using the Dashboard menu, add the slices to your dashboard and organize their layout.



When a SAM app streams data to a new cube using the Druid processor, it will take about 30 minutes for the cube to appear in Superset. This is because Superset has to wait for the first segment to be created in Druid. After the cube appears, users can analyze the streaming data immediately as it is streaming in.

### **Creating Insight Slices**

Note:

The following steps demonstrate a typical flow for creating a slice:

#### Procedure

- 1. Choose Slices on the Menu.
- 2. Click + to create a new Slice.
- 3. Select the Druid Data Source that you want to use for the new visualization:

rse	t of Securi	ity 👻 🥕 Manage	e 🗸 🐻 Sources 🗸 🕍	Slices 🚯 Dashboards 👗 SQL Lab	~	0
ci	lick on a druid li	nk to create a Slice				
Li	ist Druid Datas	lource				
S	earch 🛩					
I	Actions	-			Record	Count: 2
		Data Source I	Cluster	Changed By	Changed On I	Time Offset
	Q 2 #	Alerts-High- Speed-Cube-V2	Streaming Analytics Manager - Stream Insight	<a href='/superset/profile/admin/"&gt;George Vetticaden</a 	2017-02-07 15:50:59.807995	0

**4.** Select a Chart Type from the menu.

The following example creates a "Sunburst" visualization of rolling up multiple dimensions like route, eventType, and driver info..

Configure the chart and click Execute Query

♦ Query     Save as       Datasource & Chart Type	Violation Details Brea	kdown ☆œ	0.38 sec 9
	Lane Departure	Des Moines to Chicago	7.58%
[druid-ambari].[violation-events-cube] *	Lane Departure	Des moines to Unicago	7.58%
Sunburst *			
Time 🕐			
Time Granularity Origin ®  1 hour *  Since ® Until 7 days ago * Now *  Hierarchy ®  X route X eventType  Primary Metric ®	7	7.58% of tota 32.3% of parent m1: 220	al
COUNT(*) *			
Secondary Metric ®			
COUNT(*) *			
Row limit			
5000 *			
Filters @			
+ Add filter			

5. Another visualization could be integration with MapBox Here we are mapping where violations are occurring the most based on the lat/long location of the event

Query O Save as		
	Route Violations Map 👙	아 Bijson Bicsv Quer
Datasource & Chart Type	Rochester	
[druid-ambari].[violation-events-cube] *	Sigur Falls	Toron
feren europhinemen erenn enrel		
Mapbox *	Lansing	roit
	IOWA Cedar Rapids	
Time 🕲	Des Moines 7 12 Chicago	
ine ()	North Platte	PE
Time Granularity  Origin  Origin		оню
1 hour * *	gillinois Indiana Colu	
ince 🏵 Until		
7 days ago + now +	United States Toppka 16 Cincinnation Cincinnation	WEST
	VANCAE	VIRGINIA
	MISSOURI 4 20 Evanville Frankfort	
ongitude 1 Latitude 1	Spring 6.1 KENTUCKY	X July
longitude v latitude v	11 Carbonile	City.
60 v		
tow limit	OKLAHOMA TU TENNESSEE Minoville	NOR
5000 *	Ito Oktahoma City ARKANSAS 11 Memohis	Charlotte CAROL
iroup by ®		AM
× latitude × longitude	Atlanta	SOUTH
ive render @	ick Tuscalcopa	CAROLINA
_	Aver the second test of the second se	
	Abitene Datlas MISSISSIPPI ALABAMA	
/iewport		
Pefault longitude ®		
-90.1994		
efault latitude ® 38.627		
Zoom ®		

6. To save the slice, specify a name and name and click Save.

E Managan M Stormer M He Slicer - Darbhoarde - 3 Col Lab M	
Save a Slice	×
Save as Driver Violations Break	
Do not add to a dashboard	
Add slice to existing dashboard     *	
Add to new dashboard [dashboard name]	
Save Save & go to dashboard Cance	н

# **Adding Insight Slices to a Dashboard**

After you create slices, you can organize them into a dashboards:

#### Procedure

0

- 1. Click the **Dashboard** menu item.
- **2.** Click + to create a new Dashboard.
- 3. Configure the dashboard: specify a name and the slices to include in the Dashboard.

Add Dash	board
Title	Trucking IOT Dashboard
Slug	trucking-iot-dashboard
	To get a readable URL for your dashboard
Slices	* Total Violations in Last Hour
	* Top Violation Drivers
	# Driver Violations Breakdown
	K Direction Infraction Details
	Routes with Infractions
Owners	M George Vetticaden
	Owners is a list of users who can alter the dashboard.
Position JSON	Position JSON
	This json object describes the positioning of the widgets in the dashboard. It is dynamically generated when adjusting the widgets size
	positions by using drag & drop in the dashboard view
CSS	CSS
	The css for individual dashboards can be altered here, or in the dashboard view where changes are immediately visible
JSON Metadata	JSON Metadata
	This JSON object is generated dynamically when clicking the save or overwrite button in the dashboard view. It is exposed here for reference and for power users who may want to alter specific parameters.

4. Arrange the slices on the dashboard as desired, and then click Save.

#### Dashboards for the Trucking IOT App

The IOT Trucking application that we implemented using the Stream Builder streams violation events, alerts, and predictions into three cubes:

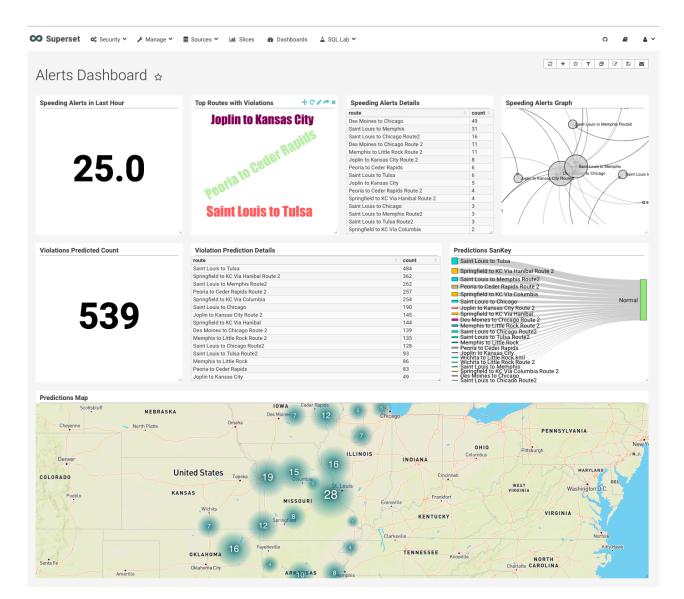
- violation-events-cube
- alerts-speeding-drivers-cube
- alerts-violation-predictions-cube

Based on the powerful visualizations that SuperSet offers, you can create the following powerful dashboards in minutes.

IoT Dashboard

ucking IOT Dashboard 🌣				
ations in Last Hour		Top Routes with Violations		
	<b>22.0</b>	Neeking to be a second a secon	Saint Louis to T DUIS to Chicago Rou to KC Via Hanibal Route 2 Saint Louis to Chicago <sub>Saint</sub>	ulsa Ute2 1005 <sup>10 TUE8</sup> Route2 1005 <sup>10 TUE8</sup> Route2
ation Details Breakdown		Route Details		
		eventType	o route	<ul> <li>count</li> </ul>
		Overspeed	Des Moines to Chicago	211
		Unsafe tail distance	Des Moines to Chicago	182
		Unsafe following distance Lane Departure	Des Moines to Chicago Des Moines to Chicago	172
		Lane Departure Unsafe tail distance	Saint Louis to Memphis	168
		Lane Departure	Saint Louis to Memphis	109
		Unsafe following distance	Saint Louis to Memphis	108
		Overspeed	Saint Louis to Memphis	94.0
		Overspeed	Saint Louis to Chicago Route2	67.0
		Unsafe tail distance	Saint Louis to Chicago Route2	66.0
		Unsafe following distance Unsafe tail distance	Saint Louis to Chicago Route2 Des Moines to Chicago Route 2	65.0 52.0
		Lane Departure	Des Moines to Chicago Route 2 Des Moines to Chicago Route 2	51.0
		Unsafe following distance	Des Moines to Chicago Route 2	48.0
		Unsafe tail distance	Memphis to Little Rock Route 2	47.0
		Overspeed	Des Moines to Chicago Route 2	47.0
		Lane Departure	Saint Louis to Chicago Route2	45.0
		Unsafe following distance Lane Departure	Memphis to Little Rock Route 2 Memphis to Little Rock Route 2	44.0
		Lane Departure Overspeed	Memphis to Little Rock Route 2 Memphis to Little Rock Route 2	41.0 39.0
		Unsafe following distance	Peoria to Ceder Rapids	27.0
1 Orwens Darver Orend Junktin COLDRADD	NEERASKA Dew North Patte Oniona United States KANSAS	ILLINOIS INDIANA Illinois Indiana Missouri 29 Energia Franker	OHIO Caunica M M VEST VEST VICINIA VICINIA	Codik, Rail
Dargos Dargos	Weite Sp OKLAHOMA B Fryntiwild Oklahoma City	ARKANSAS 10-min	Norths Martin Charlyse CARDLINA	
Eventh Infractions Over Time Constructions Over Time Constructions Time Constructio	Amarillo Oktandria City	Curponie Curponie ARKANSAS 10	Montale Martine Charlope CARDLINA Med Rapi. @Sant Lawis to China. @Sant Lawis to	Memph 🖷 Saint Louis to Memph
Serie Fe Ser	Arruphile Dislations City Displin to Kansas Cit. Unplin to Kansas Cit. Memphis to Little	Curionia ARKANSAS 10	Montale Martine Charlope CARDLINA Med Rapi. @Sant Lawis to China. @Sant Lawis to	Memph 🔮 Sant Louis to Memph

Alerts Dashboard



# SAM Test Mode

In a typical SDLC (Software development lifecycle), you want to test the streaming analytics app locally before deploying the SAM app to a cluster. SAM's "Test Mode" allows you to test the app locally using test data for the sources. SAM's Test Mode allows you to do the following:

- Create a Named Test Case
- Mock out the sources of the app and configure test data for each test source. SAM validates the test data using the configured Schema in the Schema Registry for each source
- Execute the Test Case and visually see how the data looks like at each component/processor in the app as flows across your application.
- Download the output of the test which represents the state of the data at each processor and sink.

In the following sections, we will walk through creating Test Cases for different test scenarios for the reference app. If you ran the test utility, these 4 test cases will already be created for you.

## Four Test Cases using SAM's Test Mode

### Test Case 1: Testing Normal Event with No Violation Prediction

The Assertions of this test case are the following:

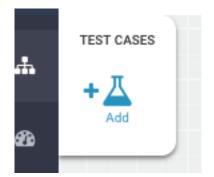
- Assertion 1: Validate test data for geo steam and speed stream that are non violations
- Assertion 2: Validate the Join of data between geo stream and speed stream
- Assertion 3: Validate that the filter "EventType" detects that this is a "Non Violation Event"
- Assertion 4: Validate the joined event gets split into three events by the "Split" projection.
- Assertion 5: Validate that the three enrichments are applied: weather enrichments, timesheet enrichment and HR enrichment.
- Assertion 6: Validate the three enrichment streams are joined into a single stream.
- Assertion 7: Validate that data after normalization for the model
- Assertion 8: Validate the output of the Prediction model is that no violation is predicted
- Assertion 9: Validate the filter "Prediction" detects that it is non violation.

Follow the below steps to create this test case for the reference app in Edit Mode. (If you ran the test utility, these 4 test cases will already be created for you.)

1. Click "TEST" Mode



2. Click Add Test Case



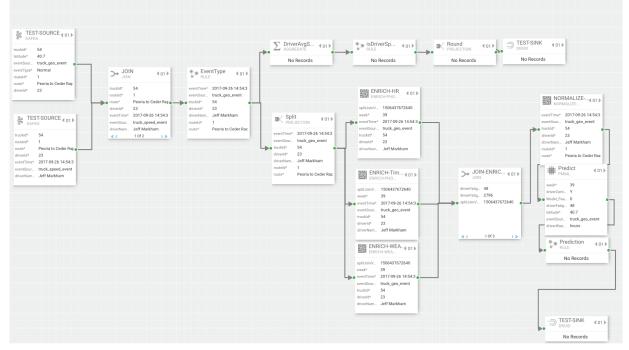
**3.** Provide Test Case details. Provide a name for test case, test data for TruckGeoEvent and test data for TruckSpeedEvent.

IAME*				
Test-Normal-Event-No-Vi	olation-Predicti	on		
Sources	1 [ 2	{		Output
<ul> <li>Test-TruckSpeed</li> <li>Kafka</li> <li>Test-TruckGeoE</li> <li>Kafka</li> </ul>	3 4 5 6 7 8 9 10 11 12 13 14 15 	"eventTime":"20 "eventSource":" "truckId":54, "driverId":23, "driverName":"J "routeId":1,	<pre>a to Ceder Rapids Route 2" prmal",</pre>	eventTime* STRING eventSource* STRING truckld* INTEGER driverVame* STRING routeld* INTEGER route* STRING eventType* STRING latitude* OOREE
	1 SLEEP TIME*		times	longitude* DOUBLE correlationId*

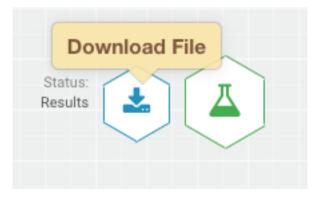
4. Execute the Test Case.



5. You should see the result of the test case as the following.



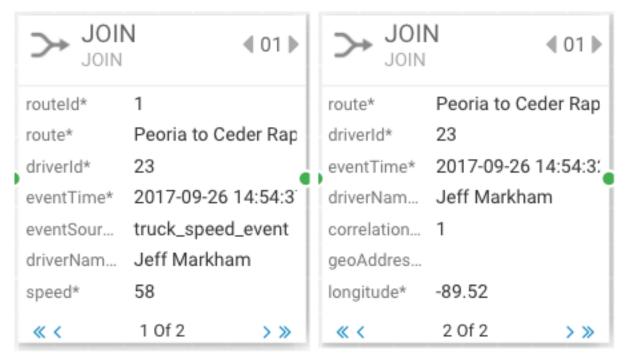
6. Download the test case results.



#### **Analyzing Test Case 1 Results**

The key to reading the test case results is to keep in mind that when you look at the results of the component, you are viewing the input into that component.

• Assertion 1 is to Validate test data for geo steam and speed stream that are non violations. For this assertion, you would look at the downstream component after the sources. So in this case, it would be the Join component. Use the paging features to see the inputs to the join processor.



• Assertion 2 is to validate the Join of data between geo stream and speed stream. For this assertion, you would look at the downstream component after the Join. So in this case, it would be the EventType component. Note that you see speed and geo information.

e Ever	ntType ◀ 01 ▶
route*	Peoria to Ceder Rap
eventType*	Normal
latitude*	40.7
longitude*	-89.52
correlation	1
geoAddres	
speed*	58

• Assertion 3 is to validate that the filter "EventType" detects that this is a "Non Violation Event". View the Split Component.

	C Split	ECTION				
••	eventTime*	2017-09-26 14:54:3				
	eventSour	truck_geo_event				
	truckId*	54 🖕				
	driverId*	23				
	driverNam	Jeff Markham				
	routeld*	1				
	route*	Peoria to Ceder Rap				

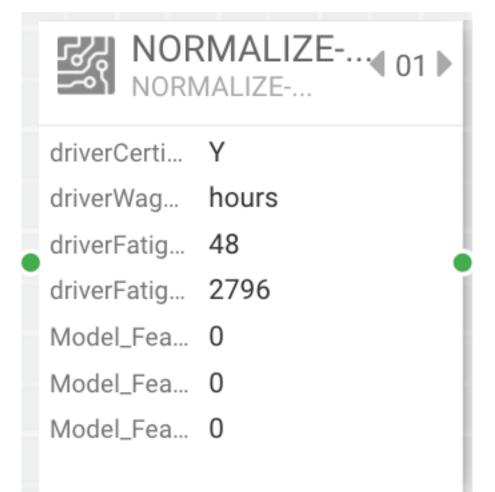
• Assertion 4 is to Validate test data for geo steam and speed stream that are non violations. View the JOIN-ENRICHMENT component

	CH-PHO ◀ 01 ►		RICH-Tim	ENRICH-WEA 01 ENRICH-WEA		
splitJoinV	1506437672640	splitJoinV	1506437672640	splitJoinV	1506437672640	
week*	39	week*	39	week*	39	
eventTime*	2017-09-26 14:54:3:	eventTime*	2017-09-26 14:54:3:	eventTime*	2017-09-26 14:54:3.	
eventSour	truck_geo_event	eventSour	truck_geo_event	eventSour	truck_geo_event	
truckId*	54	truckId*	54	truckId*	54	
driverId*	23	driverId*	23	driverId*	23	
driverNam	Jeff Markham	driverNam	Jeff Markham	driverNam	Jeff Markham	

• Assertion 5 is to validate that the three enrichments are applied: weather enrichments, timesheet enrichment and HR enrichment. Use the paging features to page through the three enrichment outputs.

→ JOIN-ENRIC 4 01 ►			→ JOIN-ENRIC ↓ 01 ►				→ JOIN-ENRIC ∢ 01 ►			
driverFatig driverFatig			Model_Fea Model_Fea				truckId* week*	54 39		
· · · ·	1506437672	640	Model_Fea splitJoinV		72640	11	driverCerti. latitude*	Y 40.7	•	
							eventSour driverWag		event	
« <	1 Of 3	>	« <	2 Of 3	>		eventTvpe*		> »	

• Assertion 6 is to validate the three enrichment streams are joined into a single stream. View the NORMALIZE component.



• Assertion 7 is to validate that data after normalization for the model. View the Predict component.

•

ĺ	Prec		01 ▶
	week*	39	
	driverCerti	Y	
ļ	Model_Fea	0	
	driverFatig	48	- 1
	latitude*	40.7	- 1
	eventSour	truck_geo_eve	nt
1	driverWaa	hours	- 1

Assertion 8 is to validate that the output of the Prediction model is that no violation is predicted. View the Prediction component.

		1
truckld* eventTime latitude* eventSour	54 1506455684451 40.7 truck_geo_event	•
	WiolationP truckld* eventTime latitude* eventSour eventType*	Image: RULE         ViolationP       no         truckId*       54         eventTime       1506455684451         latitude*       40.7         eventSour       truck_geo_event         eventType*       Normal

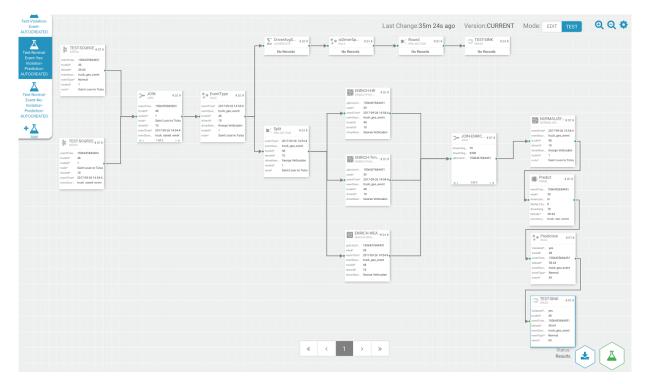
#### Test Case 2: Testing Normal Event with Yes Violation Prediction

In this test, we are validating all the same assertions as previous test but in this test case the violation prediction model should return true and be . Similar to above, Create a test named "Test-Normal-Event-Yes-Violation-Prediction", use the following test data for TruckGeoEvent and use the following test data for TruckSpeedEvent.

#### **Analyzing Test Case 2 Results**

1. Analyzing the Test Case Results.

The output of the test case should be the following:



To validate the PMML processor returns a violation prediction and sent to the sink, view the Prediction and Druid component.

ST-SINK
yes 48 1506455684451 38.64 truck_geo_event Normal 60
ſ

#### **Test Case 3: Testing Violation Event**

The Assertions of this test case are the following:

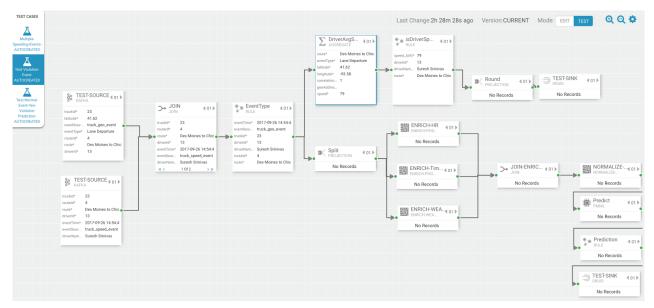
- Assertion 1: Validate test data for geo steam and speed stream that are "violation" events
- Assertion 2: Validate the Join of data between geo stream and speed stream
- Assertion 3: Validate that the filter "EventType" detects that this is a "Violation Event"

- Assertion 4: Validate that the inputs to the aggregate speed processor. There should only be 1 in the window
- Assertion 5: Validate the result of the DriverAvgSpeed aggregate process is average speed of 79 since there only 1 event
- Assertion 6: Validate the isDriverSpeeding rule recognized it was not speeding since the speed wasn't greater than 80. The event should stop.

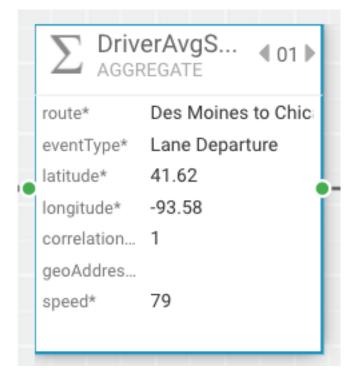
Create a test named "Test-Violation-Event", use the following test data for TruckGeoEvent and use the following test data for TruckSpeedEvent.

#### Analyzing Test Case 3 Results

The output of the test case should look something like the following:



• Assertion 3 is to validate that the filter "EventType" detects that this is a "Violation Event" and Assertion 4 is to validate that the inputs to the aggregate speed processor should be 1 event within the window. View the DriverAvSpeed component to validate these assertions:



• Assertion 5 is to validate the result of the DriverAvgSpeed aggregate process is average speed of 79 since there only 1 event. View the isDriverSpeeding component:

	● isDri ● RULE	iverSp ∢ 01 ▶
	speed_AVG*	79
	driverId*	13
•	driverNam	Suresh Srinivas 🛛 💧
	route*	Des Moines to Chic

• Assertion 6 is to validate the isDriverSpeeding rule recognized it was not speeding since the speed wasn't greater than 80. The event should stop. See the downstream components after isDriverSpeeding.

Round PROJECTION	101		€01
No Records		No Records	_

#### Test Case 4: Testing Multiple-Speeding-Events

The assertions of this test case are the following:

- Assertion 1: Validate that there are two geo events both of which are violations (Overspeed, Excessive Breaking) in source. Validate there are two speeding events both of which are speeding (96, 83)
- Assertion 2: Validate the Join of data between geo stream and speed streams
- Assertion 3: Validate that the filter "EventType" detects that this is a "Violation Event"
- Assertion 4: Validate the inputs of the window should be two events (geo/speed 1 with speed of 83, geo/speed 2 with speed of 96)
- Assertion 5: Validate the result of the DriverAvgSpeed aggregate processor should be one event that represents the average of 83 and 96...89.5
- Assertion 6: Validate the isDriverSpeeding rule recognizes it as speeding event (89.5) since it is greater than 80 and continue that event to custom round UDF
- Assertion 7: Validate the output of the round UDF event should change the speed from 89.5 to 90.0 and that is the final event that goes to the sink.

Create a test named "Test-Multiple-Speeding-Events", use the following test data for TruckGeoEvent and use the following test data for TruckSpeedEvent.

#### **Analyzing Test Case 4 Results**

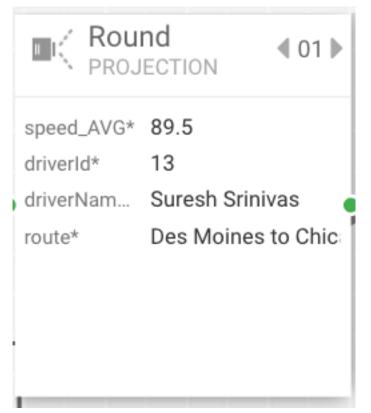
The output of the test case should look something like the following:

TEST CASES				Last Change:2h 46m 4	2s ago Version:CURRENT Mode	EDIT TEST QQ
Multiple- Speeding-Events-			DriverAvgS 4 of 1	● ● isDriverSp 4 01 ▶		
AUTOCREATED Test-Violation- Event- AUTOCREATED			eventTima* 2017/09-26 14:54:5 eventSour: truck_gee_event eventSour: truck_gee_event divertation: Suresh Srinivas divertation: Suresh Srinivas	s speed_AV0* 89.5 driverid* 13 driverid* 13 driverid*Suresh Srinivas route* Des Moines to Chic	■ Round 401 > TES PROJECTION 401 > ⇒ TES	ST-SINK 401 P
Test-Normal- Event-Yes- Violation-	truckid* 23		EventType €01 ►		speed_AVG* 89.5 speed_AV. drivertd* 13 driverid* drivertam Suresh Srinivas drivertam	13
Prediction- AUTOCREATED	IstRude* 41.62 eventSour truck_geo_event eventType* Excessive Breaking routeId* 4		ne* 2017-09-26 14:54:5 urtruck_geo_event 23	ENRICH-PHO 401	route* Des Moines to Chic route*	
	route* Des Moines to Chic driverId* 13 « < 10f2 > >>	driverid* 13 driverid* eventTime* 2017-09-26.14:54:5 eventSou… truck_speed_event croutel* driverNam Suresh Srinivas route*	M. Suresh Srinivas 4 Des Moines to Chic	•		
	& TEST-SOURCE 4 01 ▶	≪ < 10f4 > ≫ ≪ <	1 0f 4 >>> No Records	No Records	→ JOIN·ENRIC ∢ 01 ► JOIN No Records	NORMALIZE
	truckid* 23 routeid* 4 route* Des Moines to Chic			ENRICH-WEA 01		Predict 4 c
	driverid* 13 eventTime* 2017-09-26 14:54:5 eventSour truck_speed_event driverNam Suresh Srinivas			No Records		No Records
	« < 10f2 > >>					No Records

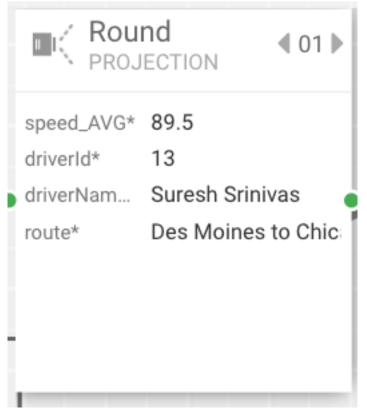
• Assertion 4 is to validate the inputs of the window should be two events (geo/speed 1 with speed of 83, geo/speed 2 with speed of 96). View the two events in the Join processor (use the paging feature to see the events)

∑ DriverAvgS ∢ 01 ► AGGREGATE		∑ DriverAvgS ∢ 01 ►		
eventType*	Lane Departure	eventType* Lane Departure	1	
latitude*	41.62	latitude* 41.62	1	
longitude*	-93.58	longitude* -93.58	I.	
correlation	1	correlation 1	Į	
geoAddres		geoAddres	I	
speed*	83	speed* 96	I	
eventTime	1506455684451	eventTime 1506455684451	Į	
« <	1 Of 2 > >>	≪ < 2 Of 4 > ≫		

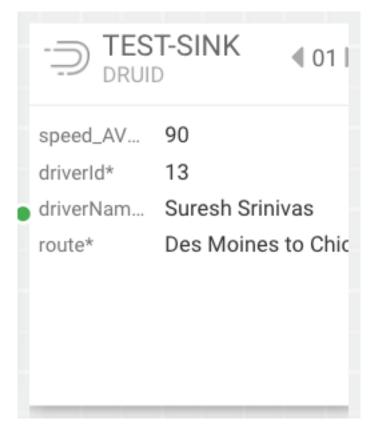
• Assertion 5 is to validate the result of the DriverAvgSpeed aggregate processor should be one event that represents the average of 83 and 96...89.5. View the Round Projection processor.



Assertion 6 is to validate the isDriverSpeeding rule recognizes it as speeding event (89.5) since it is greater than 80 and continue that event to custom round UDF. View the Round Projection processor



• Assertion 7 is to validate the output of the round UDF event should change the speed from 89.5 to 90.0 and that is the final event that goes to the sink. View the Druid Test Sink component.



#### **Running SAM Test Cases as Junit Tests in CI Pipelines**

Using SAM's Test Mode provides a quick and effective way to test your applications locally visualizing the output within each component of the app without deploying to a cluster. Since all of SAM's capabilities is backed by REST APIs, you can execute the these SAM Test Cases as part of your Junit Tests. This provides the power of using Junit assertions to validate the results of the test and incorporating them in automated tests as part of your continuous integration and delivery pipelines.

Examples of incorporating SAM Test Cases as part of unit tests can be found in the following artifacts:

- Trucking Ref App Git Hub Project
- TruckingRefAppAdvancedApp Junit Test Case

The Junit Test case above uses the SAM SDK project to setup a self contained Junit test that executes the SAM test cases and validates the result. The test case performs the following on setup of the this Junit Test Case:

- Create SAM Service Pool
- Create SAM Environment
- Import the Trucking Ref App

Then the following 4 test cases are executed:

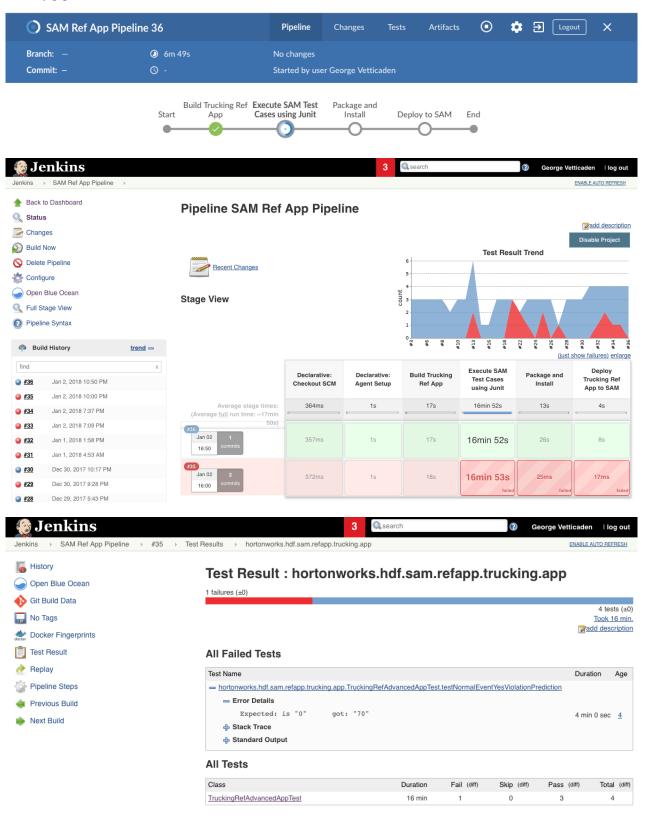
- testNormalEventNoViolationPrediction
- testNormalEventYesViolationPrediction
- testViolationTruckingEvents
- testMultipleSpeedingEvents

Each of these test cases will do the following:

- Create the SAM Test Case
- Setup test data for each of the sources for each test case.
- Execute the SAM Test Case using SAM Test Mode and wait for test to complete.
- Download the results of the test case.

• Validate the results of the Test Case.

SAM Test Mode execution via Junit Tests allows you to integrate these tests as part of your continuous integration / delivery pipeline.



😥 Jenkins	3 🔍 search 🕜	George Vetticader	n ∣log out
Jenkins	#35	ENABLE	AUTO REFRESH
📕 History 🍚 Open Blue Ocean	Test Result : TruckingRefAdvancedAppTest		
🕑 Git Build Data	1 failures (±0)		4 to at a ( )
No Tags			4 tests (± Took 16 m
Docker Fingerprints		d	dd descripti
Test Result	All Tests		
heplay 🗞	Test name	Duration	Status
Pipeline Steps	testMultipleSpeedingEvents	4 min 0 sec	Passed
🗽 Pipeline Steps	testMultipleSpeedingEvents testNormalEventNoViolationPrediction	4 min 0 sec 4 min 0 sec	
			Passed

# **Creating Custom Sources and Sinks**

Throughout the getting started doc with the trucking reference application, we have showcased the powerful extensibility features of SAM including:

- Uploading custom UDFs
- Uploading custom processors

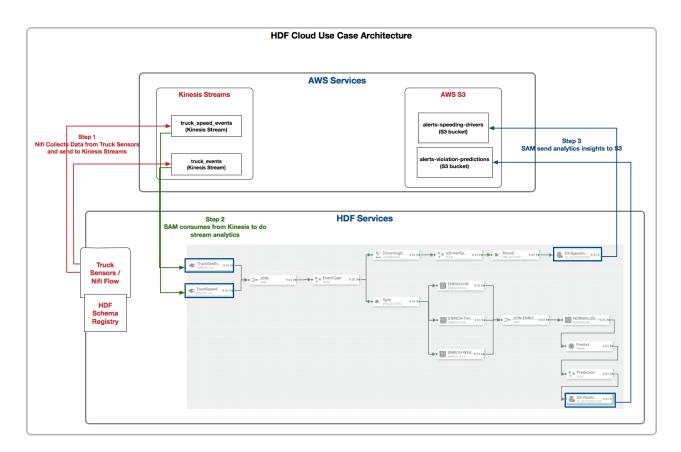
In this section, we walk through how to register custom sources and sinks in SAM integrated with Schema Registry.

## **Cloud Use Case: Integration with AWS Kinesis and S3**

To showcase registering custom sources and sink, lets modify our Trucking Reference Application Use Case Requirements with the following:

- The Trucking company wants to deploy the Trucking Application on AWS
- The Trucking company wants to streams the sensor data into AWs Kinesis instead of Apache Kafka.
- The trucking company wants to use SAM for streaming analytics.
- The insights generated by SAM should be stored into AWS S3 instead of Druid.

The below diagram illustrates this Cloud architecture:



## **Registering a Custom Source in SAM for AWS Kinesis**

To register any custom source in SAM, there are three artifacts you need:

- 1. Artifact 1: Code for the custom source using the underlying streaming engine. Since SAM today supports Storm as the Streaming engine, you can refer to the following artifacts for the custom source:
  - Git Project for Storm Kinesis
  - AWS Storm Kinesis Spout
- **2.** Artifact 2: Code for mapping the SAM configs to the custom source/spout. Refer to the following artifacts for this mapping code:
  - Git Project for SAM Storm Kinesis Mapping
  - SAM Kinesis Flux Mapping Class
- 3. Artifact 3: Flux mapping file to map the SAM config to the Kinesis Spout. Refer to the following artifacts
  - SAM Kinesis Flux Mapping Config

More Details on implementing a custom source and registering with SAM can be found here: https://github.com/ hortonworks/streamline/tree/master/examples/sources

To register the custom Kinesis Source in SAM using the above three artifacts, perform the following steps:

- 1. Download the Sam-Custom-Extensions.zip to the host where SAM is installed (if you haven't done it in a past step)
- 2. Unzip the contents. We will call the unzipped folder \$SAM\_EXTENSIONS
- **3.** Switch to user streamline:

sudo su streamline

4. Install Artifact 1 (the custom source code) on host's local maven repo

```
cd $SAM_EXTENSIONS/custom-source/kinesis/
mvn install:install-file -Dfile=storm-kinesis-1.1.0.5.jar \
-DgroupId=org.apache.storm \
-DartifactId=storm-kinesis \
-Dversion=1.1.0.5 \
-Dpackaging=jar
```

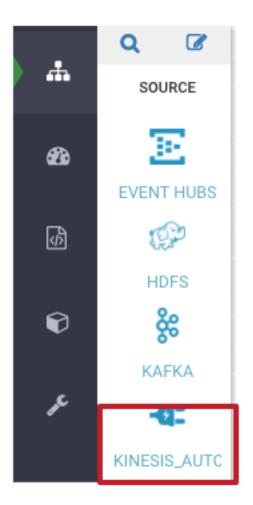
5. Register the custom source via SAM REST call. Replace SAM\_HOST and SAM\_PORT.

```
curl -sS -X POST -i -F \
topologyComponentBundle=@config/kinesis-source-topology-component.json -F
\
bundleJar=@sam-custom-source-kinesis.jar \
http://SAM_HOST:SAM_PORT/api/v1/catalog/streams/componentbundles/SOURCE
```

6. If the registration was successful, you should see a message like the following by the REST response:

```
HTTP/1.1 201 Created
Date: Wed, 03 Jan 2018 20:26:22 GMT
Content-Type: application/json
Content-Length: 4569
```

7. On the SAM Application Canvas Palette, you should now see KINESIS source.



8. Dragging the kinesis source onto the canvas and double clicking it, you should see the following kinesis dialog. The dialog properties comes from the topologyComponentBundle flux config you used to register the custom source.

KINESIS 🗸 🗙	×
REQUIRED OPTIONAL NOTES	
AWS ACCESS KEY ID * AWS ACCESS KEY SECRET * AWS REGION * US_WEST_2 * KINESIS STREAM * Truck_events_avro READER SCHEMA VERSION * 1 *	Output eventTime straws eventTimeLong Lows eventSource straws truckdd wrease driverId wrease driverName straws driverName straws straws couted straws straws couted straws broute straws coute straws
	Cancel Ok

## **Registering a Custom Sink in SAM for AWS S3**

To register any custom sink in SAM, there are three artifacts you need:

- 1. Artifact 1: Code for the custom sink using the underlying streaming engine. Since SAM today supports Storm as the Streaming engine, you can refer to the following artifacts for the custom sink:
  - Git Project for Storm S3
  - Storm S3 Sink
- **2.** Artifact 2: Code for mapping the SAM configs to the custom/spout. Refer to the following artifacts for this mapping code:
  - Git Project for SAM Storm S3 Mapping
  - SAM S3 Flux Mapping Class
- 3. Artifact 3: Flux mapping file to map the SAM config to the S3 Sink. Refer to the following artifacts
  - SAM S3 Flux Mapping Config

To register the custom S3 Sink in SAM using the above three artifacts, perform the following steps:

- 1. Download the Sam-Custom-Extensions.zip to the host where SAM is installed (if you haven't done it in a past step).
- 2. Unzip the contents. We will call the unzipped folder \$SAM\_EXTENSIONS.

3. Switch to user streamline.

sudo su streamline

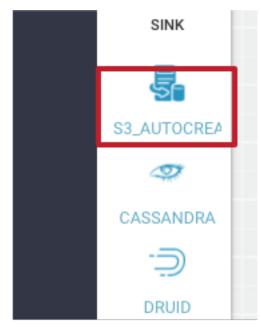
4. Install Artifact 1 (the custom sink/bolt code) on host's local maven repo.

```
cd $SAM_EXTENSIONS/custom-sink/s3
mvn install:install-file \
    -Dfile=storm-s3-0.0.1-SNAPSHOT.jar \
    -DgroupId=hortonworks.storm.aws \
    -DartifactId=storm-s3 \
    -Dversion=0.0.1-SNAPSHOT \
    -Dpackaging=jar
```

5. Register the custom sink via SAM REST call. Replace SAM\_HOST and SAM\_PORT.

```
curl -sS -X POST -i -F \
topologyComponentBundle=@config/s3-sink-topology-component.json -F \
bundleJar=@sam-custom-sink-s3.jar \
http://SAM_HOST:SAM_PORT/api/v1/catalog/streams/componentbundles/SINK
```

6. On the SAP App Canvas Palette, you should now see S3 sink.



7. Dragging the S3 sink onto the canvas and double clicking it, you should see the following s3 dialog. The dialog properties comes from the topologyComponentBundle flux config you used to register the custom sink.

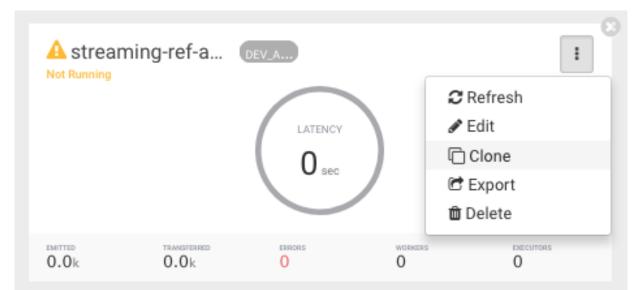
\$3	×
REQUIRED OPTIONAL	NOTES
Input eventTime* straws eventSource* straws truckid* attean driverid* attean driverid* attean driverid* attean truckid* attean driverid* attean truckid* attean driverid* attean truckid* attean	AWS ACCESS KEY ID •
	Cancel Ok

# Implementing the SAM App with Kinesis Source and S3 Sink

Now that we have registered teh custom Kinesis and S3 sources and sink, we can now build the streaming application in SAM to implement the cloud use case requirements.

#### Procedure

**1.** Clone the trucking reference application.



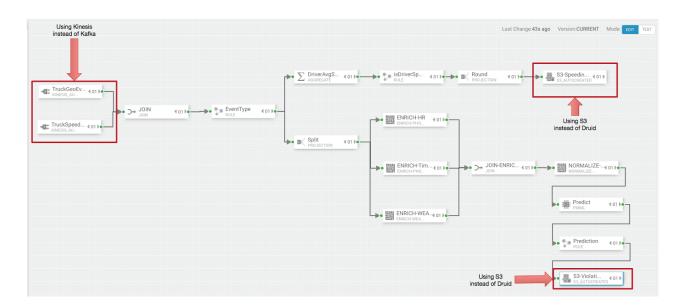
- 2. Rename the clone app to streaming-ref-app-advanced-cloud
- 3. Delete the Kafka sources and druid sinks from the SAM App
- 4. Add Kinesis sources for the deleted the Kafka Topics. Make sure to create the create the kinesis streams in AWS with the same names as the schemas you defined SAM's SR. You you will need to reevaluate the config for other components that are marked as yellow.

uckGeoEvent	× TruckSpeedEvents	
QUIRED OPTIONAL NOTES	REQUIRED OPTIONAL NOTES	
SREGION*  SweetTa SweetTa SweetTa Stress Stress Stress Stress Stress ADD TERATOR TYPE* ATEST WEMA REGISTRY URL* Ittp://hdf-3-1-build3.field.hortonworks.com:7788/api/v	Long* KINESIS STREAM * KINESIS STREAM * Ce* KINESIS STREAM * KINESIS STREA	g*

- **5.** Add S3 sink for the deleted druid sinks. Make sure to create the S3 buckets in AWS. If you can't connect to the S3 from the Round projection, try deleting the Round projection, adding it back in and then connecting it to the S3.
- 6. Remove any HDFS or HBase Sinks that you have in the app.

#### Results

The SAM App should look like the following:



# **Stream Operations**

The Stream Operation view provides management of the stream applications, including the following:

- Application life cycle management: start, stop, edit, delete
- Application performance metrics
- Troubleshooting, debugging
- Exporting and importing applications

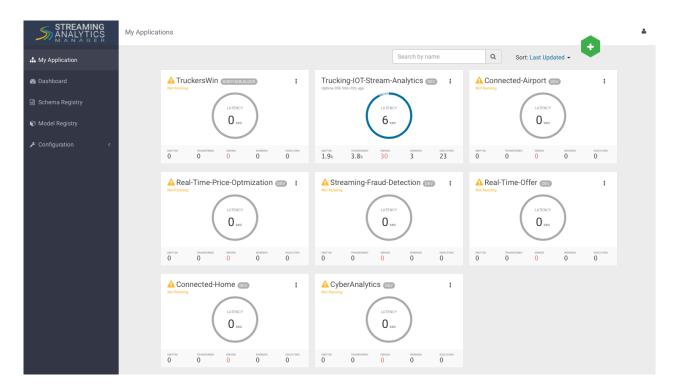
# **My Applications View**

Once a stream application has been deployed, the Stream Operations displays operational views of the application.

One of these views is called My Application dashboard.

To access the application dashboard in SAM, click **My Application** tab (the hierarchy icon). The dashboard displays all applications built using Streaming Analytics Manager.

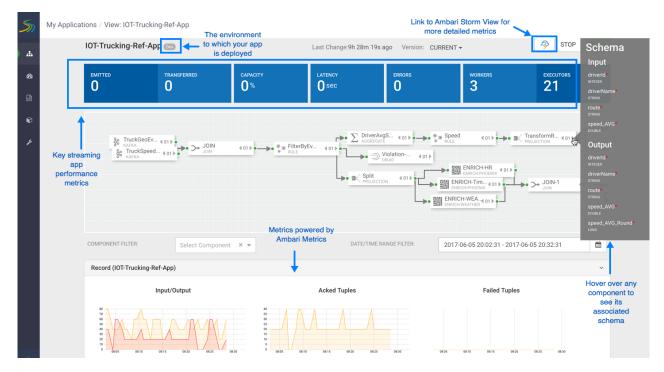
Each stream application is represented by an application tile. Hovering over the application tile displays status, metrics, and actions you can perform on the stream application.



### **Application Performance Monitoring**

To view application performance metrics (APM) for the application, click the application name on the application tile.

The following diagram describes elements of the APM view.

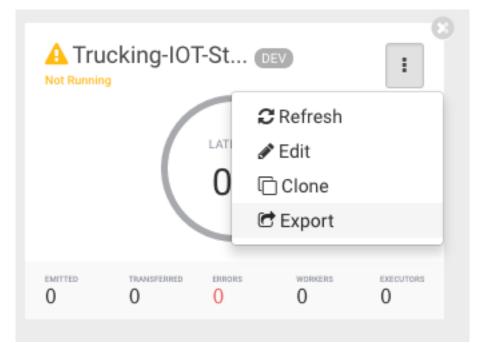


## **Exporting and Importing Stream Applications**

Service pool and environment abstractions combined with import and export capabilities allow you to move a stream application from one environment to another. This task provides instructions about importing a stream application that was exported in JSON format.

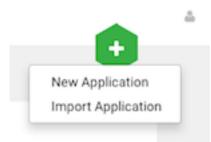
#### About this task

To export a stream application, click the Export icon on the **My Application** dashboard. This downloads a JSON file that represents your streaming application.



#### Procedure

1. Click on the + icon in My Applications View and select import application:



**2.** Select the JSON file that you want to import, provide a unique name for the application, and specify which environment to use.

Import Stream	×
SELECT JSON FILE *	
Choose File Trucking-IOT-Streaming-Analtyics.json	
TOPOLOGY NAME	
Trucking-IOT-Streaming-Analtics-App-Import	
ENVIRONMENT *	
Dev	~
	Cancel Ok

# **Troubleshooting and Debugging a Stream Application**

Once we have deployed the streaming app, common actions performed by users such as DevOps, Developers, and Operations teams are the following:

- Monitoring the Application and troubleshooting and identifying performance issues
- Troubleshooting an application through Log Search
- Troubleshooting an application through Sampling

SAM makes performing these tasks easier by using the same visual approach as users have when developing the application. We will walk through these common use cases in the below sections.

## **Monitoring SAM Apps and Identifying Performance Issues**

After deploying SAM and running the test generator for about 30 mins, your Storm Operation Mode of the app renders important metrics within each component on the canvas like below.

All Components Log: None Sampling: 0% -	Mode OVERVIEW METRICS SAMPLE	🧇 💿 10 minutes 🗸 🕘 🖉
See Overview Metrics Directly on the SAM App within each Component	Definition Angle. (a)     Construction Section 1.	Q Q
Autor A Dirac Auss Bindie Geweiner Falat Auss Sith 242m 0 2.5 More TouckSpeed_ (4) More TouckSpeed_ (4)		(01)         ************************************
	Control Prince France Acade     Control Prince France Acade     Control Prince France Acade     Control Prince Prince Prince     Control Prince Prince     Control Prince     Contr	Dashboard 401 )     Social States States Falet Adult     Takes States Falet Adult     Takes States Falet Adult     Takes States Falet Adult     Takes States States     Takes States     Sampling Int

You can click on **Show Metrics** to get more details on the metrics and drill down on individual metrics. Note the detailed level metrics for **All Components**, **TruckGeoEvent Kafka** source, and **Dashboard-Predictions** Druid Sink.

All Components -	Emitted 🎽 94k -0.0k	Acked ≌ 99k -0.0m	Latency <b>7</b> 34.0sec +15.4s	Failed ec <b>0</b> 0	Workers 3	Executors 19	Hide Metrics 🗸
Input/Output							
Acked Tuples							
Failed Tuples							
Queue							
Latency	- T						

All Components -	Emitted 🎽 94k -0.0k	Acked 🎽 99k -0.0m	Latency 🛪 34.0sec +	15.4sec	Failed 00	Workers 3	Executors 19	Hide Metrics 🗸
Input/Output								
Acked Tuples								
Failed Tuples								
Queue								
Latency	-							

Dashboar	d-Predictions -	Acked ᢂ 350.0 -160.0 cutors	Process Latency 9.4ms -508.7ms	Failed 0 0 ide Metrics ❤
Input/Output				
Acked Tuples				
Failed Tuples				
Queue		 		
Process Latency				
Execute Latency				

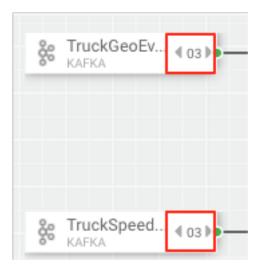
Key metrics include the following:

Metric Name	Description
Execute Latency	The average time it takes an event to be processed by a given component
Process Latency	The average time it takes an event to be acked. Bolts that join, aggregate or batch may not Ack a tuple until a number of other Tuples have been received
Complete Latency	How much time an event from source takes to be fully processed and acked by the topology. This metrics is only available for sources (e.g.: Kafka Source)
Emitted	The number of events emitted for the given time period. For example, for a Kafka Source, it is the number of events consumed for the given time period
Acked	The number of events acked for the given time period. For example, for a Kafka Source, it is the number of events consumed and then acked.

#### **Identifying Throughput Bottlenecks**

Looking through the metrics the Source and Sink metrics, we want to increase the throughput such that we emit/ consume more events from the Kafka Topic and send more events to Druid sink over time. We make some changes to the app to increase throughput.

Increase the parallelism of TruckGeoEvent (kafka topic: truck\_events\_avro) and TruckSpeedEvent (kafka topic: truck\_speed\_events\_avro) from 1 to 3. Note that each of these kafka topics have three partitions.



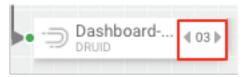
Increase the parallelism of the Join from 1 to 3. Since the join is grouped by driverId, we can configure the connection to use fields grouping to send all events with driverId to the same instance of the Join.

KAFKA	Configure each connection to do a gro by driverld so that a events with the sam driverld go to the san instance of the Join	ll e ne	
		FIELDS	-
		SELECT FIELDS*	
	-	× driverId	× -
KAFKA			

Increase the parallelism of the DriverAvgSpeed aggregate window from 1 to 3. Since the window groups by driverId, driverName and route, we can configure the connection to use fields grouping to send all events with those field values to the same instance of the window.

	- DriverAvgS 4 03 P-		
		GROUPING*	
		FIELDS	•
		SELECT FIELDS*	
→• 💬 EventType 🔹 👍 –	-	× driverId × driverName × route	× Ŧ

Increase the parallelism of the Dashboard-Predictions Druid sink from 1 to 3 so we can have multiple JVM instances of Druid writing to the cube.



After making these changes, we re-deploy the app using SAM and run the data generator for about 15 minutes and view seeing the following metrics.

SAM's overview and detailed metrics makes it very easy to verify if the performance changes we made had the desired effect.

Al Components Log None Sampling D+ Al Components Log None Sampling D+	My Applications / View: streaming-ref-app-advanced_AUTOCREATED			
Control of the con	All Components Log: None Sampling: 0% -		Mode: OVERVIEW METRICS SAMPLE	30 minutes •
The control for the control fo	S. Tratflerfy	Emitted Process Execute Failed Acked  Envitted Process Execute Failed Acked  Latency L	ifed Acked T70.0 1.7k 0.1m 0.0m 0 160.0 1.3k 3.6t 1.3k 3.5t 1.3	e volumente l'actual de la constanti de la con
2 12.1 13.0 6 67.	Lotted (copyeter field Alast 35: 22.8 → 0 7.4 k         Surgetion field Alast 35: 22.8 → 0 7.4 k         Surgetion field Alast 36: 22.8 → 0 7.4 k         Surgetion field Alast 37: 22.8 → 0 7.4 k         Surgetion field Alast 38: 22.8 → 0 7.4 k <td>BC Split Comparison from the split of the split</td> <td>abit Abits         •</td> <td>Process Deoute Failed Acked *** Emitted Process Deoute Failed Acked Emitted Process Decute Failed Acked Latency Latenc</td>	BC Split Comparison from the split of the split	abit Abits         •	Process Deoute Failed Acked *** Emitted Process Deoute Failed Acked Emitted Process Decute Failed Acked Latency Latenc
All Components - Emitted 71 Acked 71 Latency 24 Failed Workers Executors Show Metrics A	All Components . Emitted 계 Acked 계 Latency 뇌 Failed Worker	The Description of the Descripti	Failed Acked	<ul> <li>⇒ guide</li> <li>⇒ guide</li></ul>

#### Throughput Improvements for the Kafka Source

The below is the before and after metrics for the TruckGeoEvent Kafka Sink:

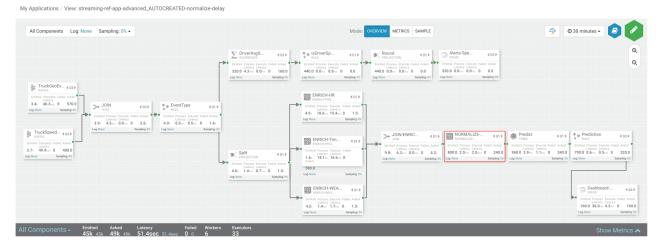
BEFORE			After
ge     TruckGeoEv     4 01 }       Fritted Complete Failed Acked     Latercy       5.1s     24.2cec     0       Log: None     Sampling: U%	from 1 to substantial throughput w events consu	g parallelism 3, we see l increase in vith respect to med (emitted) acked	See TruckGeoEV     € 0.3 I/       Emitted Complete Failed Acked       15k       23.6 sec     0       7.4k       Sampling (7k)
TruckGeoEvent - Emitted M Acked M Complete Latency 77 Failed Workers Executors Katha 5.1k -1.1k 2.5k -500.0 24.2sec +2.2sec 0 0 3 19 Hide	e Metrics 🗸	Kafka	GeoEvent - Emitted <b>31</b> Acted <b>31</b> Complete Littery <b>31</b> Failed Workers Executors <b>15k</b> +3k 7.4k +1.0k 23.65eC +1.8sec 0 0 6 33 Hide Metrics ↓
Acked Tuples		Acked Tuples	
Failed Tuples Queue		Failed Tuples	
Complete Latency		Queue Complete Lat	
		Ľ.	

The below is the before and after metrics for the Dashboard-Predictions Druid Sink:

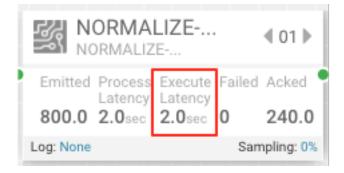
BEFORE	After
	By increasing parallelism from 1 to 3, we see substantial increase in throughput of the events written to Druid
Dashboard-Predictions - Emitted M Acked M Process Latency M Fail Dead 730.0 290.0 350.0 160.0 9.4ms -308.7ms 1.4ms 0 Workers Decedern Hide I Input/Output	ailed D Dashboard-Predictions - Emited 7 Acked 7 Process Latency 2 Execute Latency 7 Failed pred 2.6k +40.0k 1.3k +22.00 Weters Executer 6 33 Hilde Metrics ~ Hilde Metrics ~
Failed Tuples	Failed Tuples
Cueue Process Latency Execute Latency Execute Latency	Oceve Process Latency Execute Latency

## **Identifying Processor Performance Bottlenecks**

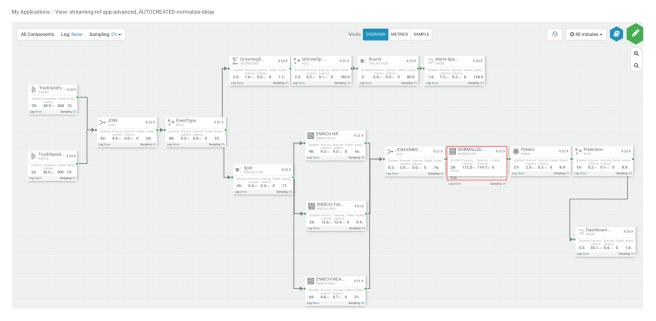
In this scenario, we identify a custom processor that has high latency. After running the data simulator for 30 mins, we view the Overview Metrics of the topology.



Scanning over the metrics, we see that the NORMALIZE-MODEL-FEATURES custom processor has high execute latency of 2 seconds. This means that over the last 30 minutes the average time an event spends in this component is 2 seconds.



After making changes to the custom processor to address the latench, we re-deploy the app via SAM and run the data generator for about 15 minutes and view seeing the following metrics.



SAM's overview and detailed metrics makes it very easy to verify if the performance changes we made had the desired effect.

#### **Latency Improvements**

The below is the before and after metrics for the NORMALIZE-MODEL-FEATURES custom processor.

BEFORE			AFTER
NORMALIZE 4 01 NORMALIZE Emitted Process Execute Failed Acked Latency Latency 2.0sec 0 240.0 Log: None Sampling: 0%	processor, we se decrease subs correlates to incr	ing the custom e execute latency stantially which eased throughput cked increases)	NORMALIZE     Image: 01 million       Emitted Process     Executor       28%     112.5ms       112.5ms     114.7ms       9.2x
NORMALIZE-MODEL-FEATURES-DELAY - Emitted Acked P Custom Custom C		Custom	9.2.2. DEL-FEATURES-DELAY. 286 vote 9 2A vote 112.5ms 1992 5ms Events Latercy 3J 114.7ms 1888 3ms 0 0 6 33 Hide Metrics ↓
Input/Output		Input/Output	
Acked Tuples		Acked Tuples	
Failed Tuples		Failed Tuples	
Queue		Queue	
Process Latency		Process Latency	
Execute Latency		Execute Latency	

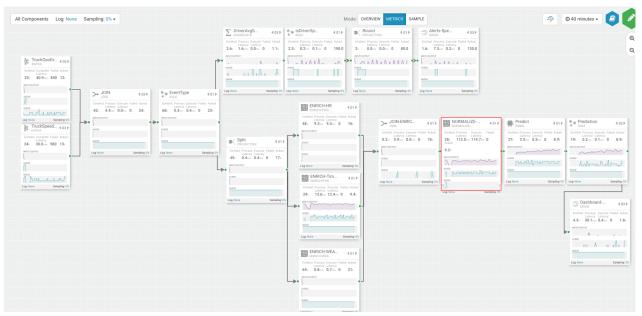
In the metric details view, the graphs provides an easy way to compare metrics before and after the code change.

	Custom	ODEL-FEATURES-DELAY •	Emitted   Acked   F 28k +26k  9.2k +8.4k Execute Latency   Faile 114.7ms -1888.3ms  0  0	
The time before code change was done:	Input/Output			
1/23/18 12:31:30				
	Acked 140			
	<ul> <li>1/23/2018</li> <li>○ 12:31:30 PM</li> <li>Failed</li> <li>0</li> </ul>			
Events being	€ 1/23/2018 © 12:31:30 PM -Wait 1024			
queued up	Process Latency			
	<ul> <li></li></ul>			
	Execute Latency			
2 sec Execute	<ul> <li></li></ul>			
		-MODEL-FEATURES-DELAY	<b>Z8K</b> +26k <b>9.2K</b> +8.4k	Process Latency ¥ 112.5ms -1892.5ms led Workers Executors 0 6 33 Hide Metrics ❤
The time after the app was re-deployed with updated		-MODEL-FEATURES-DELAY	● 28k +26k 9.2k +8.4k Execute Latency ■ Fai	112.5ms     -1892.5ms       led     Workers     Executors       0     6     33
The time after the app was re-deployed with updated custom processor: 1/23/2018 12:38	Custom	☐ 1/29/2018 O 12:38:00 PM	● 28k +26k 9.2k +8.4k Execute Latency ■ Fai	112.5ms     -1892.5ms       led     Workers     Executors       0     6     33
re-deployed with updated custom processor:	Custom	1/23/2018	● 28k +26k 9.2k +8.4k Execute Latency ■ Fai	112.5ms     -1892.5ms       led     Workers     Executors       0     6     33
re-deployed with updated custom processor:	Custom	<ul> <li>☐ 1/23/2018</li> <li>O 12:38:00 PM</li> <li>Output 130</li> <li>Output 410</li> <li>☐ 1/23/2018</li> <li>O 12:38:00 PM</li> </ul>	● 28k +26k 9.2k +8.4k Execute Latency ■ Fai	112.5ms     -1892.5ms       led     Workers     Executors       0     6     33
re-deployed with updated custom processor:	Custom	<ul> <li>☐ 1/23/2018</li> <li>O 12:38:00 PM</li> <li>Output 110</li> <li>Output 410</li> <li>☐ 1/23/2018</li> <li>O 12:38:00 PM</li> <li>Acked 130</li> </ul>	● 28k +26k 9.2k +8.4k Execute Latency ■ Fai	112.5ms     -1892.5ms       led     Workers     Executors       0     6     33
re-deployed with updated custom processor:	Custom Input/Output Acked Tuples Failed Tuples	<ul> <li>☐ 1/23/2018</li> <li>O 12:38:00 PM</li> <li>Output 130</li> <li>Output 410</li> <li>☐ 1/23/2018</li> <li>O 12:38:00 PM</li> </ul>	● 28k +26k 9.2k +8.4k Execute Latency ■ Fai	112.5ms     -1892.5ms       led     Workers     Executors       0     6     33
re-deployed with updated custom processor:	Custom	<ul> <li> <sup>1</sup>/23/2018         <sup>0</sup> 12:38:00 PM         <sup>1</sup>/100 12:38:00 PM         <sup>1</sup>/23/2018         <sup>0</sup> 12:38:00 PM         <sup>2</sup>/38:00 PM         <sup>1</sup>/23/2018         <sup>0</sup>/2:38:00 PM         <sup>1</sup>/23/2018         <sup>1</sup>/23/2018</li></ul>	● 28k +26k 9.2k +8.4k Execute Latency ■ Fai	112.5ms     -1892.5ms       led     Workers     Executors       0     6     33
re-deployed with updated custom processor: 1/23/2018 12:38	Custom	<ul> <li>☐ 1/23/2018</li> <li>○ 12:38:00 PM</li> <li>○ Output 130</li> <li>○ Output 410</li> <li>☐ 1/23/2018</li> <li>○ 12:38:00 PM</li> <li>Failed</li> <li>○</li> <li>□</li> </ul>	● 28k +26k 9.2k +8.4k Execute Latency ■ Fai	112.5ms     -1892.5ms       led     Workers     Executors       0     6     33
re-deployed with updated custom processor: 1/23/2018 12:38 Event queue dropped from	Custom Input/Output Acked Tuples Failed Tuples	<ul> <li>☐ 1/23/2018</li> <li>O 12:38:00 PM</li> <li>Output 130</li> <li>Output 410</li> <li>☐ 1/23/2018</li> <li>O 12:38:00 PM</li> <li>Failed</li> <li>O 12:38:00 PM</li> <li>Failed</li> <li>O 12:38:00 PM</li> <li>Wait</li> <li>☐ 1/23/2018</li> <li>O 12:38:00 PM</li> </ul>	● 28k +26k 9.2k +8.4k Execute Latency ■ Fai	112.5ms     -1892.5ms       led     Workers     Executors       0     6     33
re-deployed with updated custom processor: 1/23/2018 12:38 Event queue dropped from	Custom Input/Output Acked Tuples Failed Tuples Queue Process Latency	<ul> <li>☐ 1/23/2018</li> <li>○ 12.38.00 PM</li> <li>Input 130</li> <li>Output 410</li> <li>☐ 1/23/2018</li> <li>○ 12.38.00 PM</li> <li>Acked 130</li> <li>☐ 1/23/2018</li> <li>○ 12.38.00 PM</li> <li>Failed 0</li> <li>☐ 1/23/2018</li> <li>○ 12.38.00 PM</li> <li>✓ 12.38.00 PM</li> <li>✓ 12.38.00 PM</li> <li>✓ Wait 1</li> </ul>	● 28k +26k 9.2k +8.4k Execute Latency ■ Fai	112.5ms     -1892.5ms       led     Workers     Executors       0     6     33
re-deployed with updated custom processor: 1/23/2018 12:38 Event queue dropped from	Custom	<ul> <li>☐ 1/23/2018</li> <li>O 12.38:00 PM</li> <li>Input 130</li> <li>Output 410</li> <li>(1/23/2018)</li> <li>O 12.38:00 PM</li> <li>Acked 130</li> <li>(1/23/2018)</li> <li>O 12.38:00 PM</li> <li>Failed 0</li> <li>(1/23/2018)</li> <li>O 12.38:00 PM</li> <li>Failed 0</li> <li>(1/23/2018)</li> <li>O 12.38:00 PM</li> <li>(1/23/2018)</li> </ul>	● 28k +26k 9.2k +8.4k Execute Latency ■ Fai	112.5ms     -1892.5ms       led     Workers     Executors       0     6     33

You can also select the Metrics tab to validate the performance improvement.

.

My Applications / View: streaming-ref-app-advanced\_AUTOCREATED-normalize-delay



If you zoom in on the NORMALIZE-MODEL-FEATURES component, you will see that after the code change is made, throughput increases and the wait drops to 0.

	NORMALIZE ◀ 01 ►
	Emitted Process Execute Failed Latency Latency 28k 112.5ms 114.7ms 0
	Acked 9.2k
The time when app was re-reployed	QUEUE
with changes	

#### **Debugging an Application through Distributed Log Search**

In a distributed system, searching for logs on different hosts for different components can be extremely tedious and time consuming. With SAM, all the application logs are indexed via the Ambari Log Search Server via Solr. SAM makes it easy to drill into and search for logs for specific components directly from the DAG view. Follow the below steps to use distributed log search:

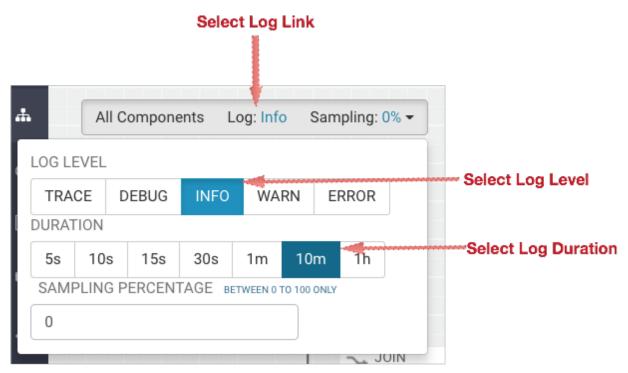
#### Procedure

- **1.** To enable Log Search in SAM, perform the following actions in Ambari.
  - a. In Ambari, select the Log Search service and select 'Log Search UI' from Quick Links.
  - **b.** Select the filter icon on the top right menu.
  - **c.** For the storm\_worker component, configure the filter like the following and click Save.

#### Log Feeder Log Levels Filter

Components	Override	SATAL	VARN	DEBUG	
storm_worker					

2. In SAM, you can dynamically change the logging level. For example, in SAM view mode of an application, click on the Log link, select the log level and the duration you want that log level.



3. Then click on the component you want to search logs for and under Actions select Logs.

				ct the o nt to s	-		-	J										
	→ JOIN-ENRIC	∉01 ▶	Real No.	IORMAI ORMALIZ	, LIZE	4	01 🕨	:	Prec				€01	8_0	Predicti RULE	on		€03
<b>-</b> *•	Emitted Process Execute Latency Latency 52k 5.9sec 0.0ms		<b>22</b> k	Process Latency 1.4ms	Latency	0 7			La 0.	tency	Execute Latency <b>0.2</b> ms	0	d Acked 7.3k		0.0ms	Latency	0	T Acked •
			MPLING P		AGE BE		-	-	ero.		Actions View Lo	3		Lug. In			34	
									Cli	ck \	View	Log	S					

4. This brings you to the Log Search page where you can search by component (s), log level(s) and search for strings using wildcard notation.

My Applications / View: streaming-ref-app-advanced\_AUTOCREATED-normalize-delay / Log Search

COMPONENT				1	LOG LEVEL	
× NORMAL	IZE-MODEL-F	EATURES-DELAY	× 🔻		Select Log Level	•
SEARCH						
Search						Ø 3 hours - Q
Date/Time	Log Level	Component Name	Log Message			4
3 hours ago	INFO	NORMALIZE- MODEL- FEATURES-DELAY	Preparing bolt 52-NORMALIZE-	/10	DEL-FEATURES-DELAY:(31)	
3 hours ago	INFO	NORMALIZE- MODEL- FEATURES-DELAY	Initialzing FeatureNormalization	pı	rocessor	
3 hours ago	INFO	NORMALIZE- MODEL- FEATURES-DELAY	Configured Delay timeout is (ne	w):	2	
3 hours ago	INFO	NORMALIZE- MODEL- FEATURES-DELAY	Finished Initialzing FeatureNorn	nal	ization processor	
3 hours ago	INFO	NORMALIZE- MODEL- FEATURES-DELAY	Prepared bolt 52-NORMALIZE-N	101	DEL-FEATURES-DELAY:(31)	
3 hours ago	INFO	NORMALIZE- MODEL- FEATURES-DELAY	entTime':"2018-01-23 18:11:11. emiec","routeld":6,"route":"Memp onld":1,"geoAddress":"No Addre n":"Y","driverWagePlan":"hours"," ather":0.0,"ModeLFeature_Rainy 9);"auxiliaryFieldsAndValues":(), 1-4666-a4e4-e046ab3bb2f8","7f	17 hi: ss driv We he ee 4ae	event: StreamlineEvent{"dataSourceld": "multiple s 9","eventSource": "truck_geo_event","truckld":84,"dri s to Little Rock", "event"Type": "Normal", "latitude":35. Available", "speed": 67,"splitJoinValue": 1516731071 verFatigueByHours": "51","driverFatigueByMiles": 12,"e ather ": 0.0,"ModeL_Feature_WindyWeather": 1.0,"ev ader": ("sourceComponentName": "JOIN-ENRICHMI d3d0-6b40-4e68-ac3a-cec94e040a9b"),"parentIds" e3-ba99-6180405d7806","318ffe99-00a5-4bf4-936 5", "sourceStream": "default").	verld":15,"driverName":"Joe I 19,"longitude":-90.04,"correla 179,"week":4,"driverCertifical 01","Model_Feature_FoggyV antTimeLong":15167310711 NTS","roottds":["4a149dff-5 ["688aaa81-2375-4f3c-af86
3 hours ago	INFO	NORMALIZE- MODEL- FEATURES-DELAY	Normalized Feautres are: {Mode ature_Certification=1, Model_Fe		Feature_FatigueByHours=0.51, Model_Feature_Fat ure_WagePlan=0}	igueByMiles=2.701, Model_I

## **Debugging an Application through Sampling**

For troubleshooting, a convenient tool is to turn on sampling for a component or for the entire topology based on a sample percentage. Sampling allows you to log the emitted values of a component in the SAM App.

#### Procedure

- 1. To enable Log Search in SAM, perform the following actions in Ambari.
  - a. In Ambari, select the Log Search service and select 'Log Search UI' from Quick Links.
  - **b.** Select the filter icon on the top right menu.
  - c. For the storm\_worker\_event component, configure the filter like the following and click Save.

#### Log Feeder Log Levels Filter

Components	Override	S FATAL	VARN WARN	DEBUG	
storm_worker_event					

2. In SAM view mode of the App, click on the component you want to turn on sampling for and enter a sampling percentage.

	KAFKA	4 03 ▶			
	Emitted Complete Failed Latency 17k 33.6sec 0 Log: None Sam				
SAM	PLING PERCENTAGE	ETWEEN 0 TO 100 C	Disable	Actions View Logs	
10		٢	Disable		

**3.** Click the 'SAMPLE' Tab .



4. Use the Sample Search UI to search for different events that were logged.

SELECT COM	PONENT :			C	DATE / TIME :				
× TruckGe	oEvent		× •		2018-01-23 14:54:08 - 2018-01-23 15:24:08	@ 30 minu	ites 🕶		
SEARCH BY K	EY:			S	SEARCH BY ID :				
Search by	Key Values, Hea	ders, Aux Key Values	Q		Search by Event Id, Root Id, Parent Id		Q		
Date/Time	Component	Key Values				1	<u>1</u> -0-		
8 minutes ago	TruckGeoEvent				2473616, eventSource=truck_geo_event, truckId=14, driverId=13, d ture, latitude=34.8, longitude=-92.09, correlationId=1, geoAddress=				
8 minutes ago	TruckGeoEvent	1	ventTime=2018-01-23 21:21:20.486, eventTimeLong=1516742480486, eventSource=truck_geo_event, truckId=106, driverId=11, driverName=Jamie Enges outeId=12, route=Springfield to KC Via Hanibal, eventType=Normal, latitude=39.78, longitude=-93.13, correlationId=1, geoAddress=No Address Available)*						
8 minutes ago	TruckGeoEvent		"(eventTime=2018-01-23 21:21:30.056, eventTimeLong=1516742490056, eventSource=truck_geo_event, truckId=56, driverId=10, driverName=George Vettii en, routeId=0, route=Peoria to Ceder Rapids Route 2, eventType=Normal, latitude=42.23, longitude=-91.78, correlationId=1, geoAddress=No Address Availat e)"						
8 minutes ago	TruckGeoEvent		~		2491546, eventSource=truck_geo_event, truckId=101, driverId=21, al, latitude=34.78, longitude=-92.31, correlationId=1, geoAddress=N				
7 minutes ago	TruckGeoEvent				2502586, eventSource=truck_geo_event, truckId=104, driverId=14, al, latitude=37.31, longitude=-94.31, correlationId=1, geoAddress=1				
7 minutes ago	TruckGeoEvent				2505086, eventSource=truck_geo_event, truckId=38, driverId=26, d de=38.43, longitude=-90.35, correlationId=1, geoAddress=No Addre		oorn, rou		
7 minutes ago	TruckGeoEvent				2508166, eventSource=truck_geo_event, truckId=64, driverId=28, d ude=37.66, longitude=-94.3, correlationId=1, geoAddress=No Addre		Aube, r		
7 minutes ago	TruckGeoEvent	<pre>{eventTime=2018-01-23 21:21:57.636, eventTimeLong=1516742517636, eventSource=truck_geo_event, truckId=92, driverId=22, driverName=Chris Harris, rou eld=7, route=Saint Louis to Chicago, eventType=Normal, latitude=38.65, longitude=-90.2, correlationId=1, geoAddress=No Address Available)*</pre>							
7 minutes ago	TruckGeoEvent	"(eventTime=2018-01-23 21:21:58.666, eventTimeLong=1516742518666, eventSource=truck_geo_event, truckId=17, driverId=29, driverName=Mark Lochbihler, routeId=10, route=Springfield to KC Via Hanibal Route 2, eventType=Normal, latitude=39.71, longitude=-92.07, correlationId=1, geoAddress=No Address Availa ble)"							

# **Spark Streaming**

Information on how to create a Streaming Analytics Application with Spark Streaming is under development and will be available after the HDF 3.2.0 release is complete.

# **Running the Stream Simulator**

Now that you have developed and deployed the NiFi Flow Application and the Stream Analytics Application, you can run a data simulator that generates truck geo events and sensor events for the apps to process.

#### About this task

To generate the raw truck events serialized into Avro objects using the Schema registry and publish them into the raw Kafka topics, do the following:

#### Procedure

- **1.** Download the Data-Loader.
- **2.** Unzip the Data Loader file and copy it to the cluster. Lets call the directory to which you unzipped the file as \$DATA\_LOADER\_HOME.

**3.** Change into the Data Loader directory:

cd \$DATA\_LOADER\_HOME

**4.** Untar the route.tar.gz file:

tar -zxvf \$DATA\_LOADER\_HOME/routes.tar.gz

- 5. Open the file startTruckGenerators.sh and make the following changes:
  - a. Modify the kafkaBrokers value based on your cluster
  - **b.** If your cluster is not secure, set the value of SECURE\_MODE to NONSECURE and set JAAS\_CONFIG to an empty space
  - **c.** Set the value of ROUTES\_LOCATION to the location where you untar the routes in step 4 and then (e.g: \$DATA\_LOADER\_HOME/routes/midwest)
- 6. Run the simulator/data generator:

```
./startTruckGenerators.sh
```

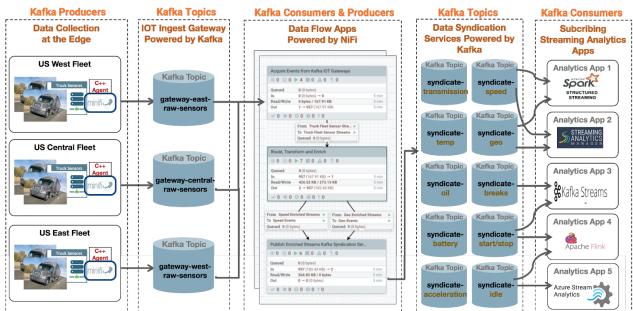
#### Results

You should see events being sent to the gateway kafka topics. NiFi should consume the events, enrich them, and then push them into the syndicate topics. SAM, Spark Streaming and other apps should consume from these syndicate topics

# Managing Kafka with Streams Messaging Manager

# **SMM Overview**

A key part of this streaming application has been the use of Kafka which powers the IOT Gateway and the Syndication Services. In this architecture, Kafka is everywhere.



# Kafka is Everywhere. Critical Component of Streaming Architectures

As a result, it becomes critical to be able to monitor and understand what is going on in the cluster to cure the Kafka blindness. To accomplish this, we will use Hortonworks Streams Messaging Manager to monitor the Kafka components of this reference application.

# **Installing DataPlane Streams Messaging Manager**

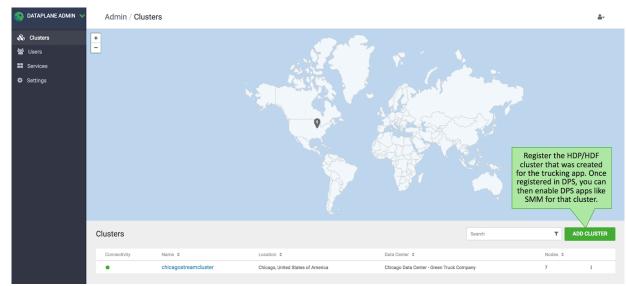
Follow the *SMM Installation* documentation to install SMM which requires DataPlane Service (DPS) platform as well as certain prerequisites required on the HDP/HDF cluster where Kafka is running.

# **Enabling Reference Application Cluster for SMM**

After installing DPS with the SMM application, you need to register the cluster you have created for the trucking reference application. The below steps walks you through this registration process.

#### Procedure

1. Log into DataPlane that was installed and click Add Cluster.



2. Provide Ambari endpoint and details for the HDP/HDF cluster that is being added to DataPlane.

🌍 DATAPLANE ADMIN 🗸	Admin / Clusters / Add	<u>Å</u>
🖌 💫 Clusters	dd Your Cluster	
Users	Please add Ambari managed cluster to DataPlane. DataPlane will auto-discover cluster information such as the cluster name, active services & important metrics running.	
	Ambari and Cluster Services behind Knox Gateway 0	
Services	Check this option if Ambari and your cluster services are accessed via a Knox Gateway.	
Settings	Validate the SSL certificate and only allow trusted connections 0	
	Check this option to let DataPlane validate SSL certificates if your cluster is SSL er Ambari URL O	
	http://dep.connected.dp0.feld.hortonworke.com/8080 Provide Ambari	
	endpoint for cluster	
	Cluster Name	
	IP Address 172 28 25 46	
	Services DP will discover the	
	HIVE O HORS O ATLAS O ZOOKEEPER O KARKA	
	VAIIServices	
	0 Datalake	
	This cluster has been identified as a Datalake since it has data management services, ATLAS running.	
	Cluster Location* Geographical location of the cluster	
	Orlando, Florida, United States of America Step 3	
	Data Center* Enter details of the	
	Data Center of the cluster cluster cluster being added	
	Criando Data Center - Green Truck Company	K
	Tags Tags for the cluster	
	Description	in the
DataPlane Version 1.2	Description of the cluster	RYP

3. After the cluster has been added, go to the cluster details page and enable the SMM application.

orlandostreamcluster									
LOCATION Orlando, United States of America	DATA CENTER Orlando Data Center - Green Truck Company	CLUSTER VERSI HDP-3.0	ON	NODES 16/16		TAGS		REGISTERED AT Thu Aug 02 2018	REGISTERED BY gvetticaden
INFORMATION							I	Cluster Services	
Description								ATLAS	0.7.0.3
Description			Services					HDFS	3.1.0
Connectivity	Security	/ Туре	Do you want to	enable Streams	Messaging Manager	on cluster orlandostr	amcluster?	HIVE	3.0.0.3
Reachable	KERBEF	IOS				CANCEL	ок	KAFKA	1.0.0.3
								RANGER	1.0.0.3
DataNodes	NameN	ode Heap Size	12.49%		HDFS Disk Space	- 25.71%		<ul> <li>STREAMS MESSAGIN MANAGER</li> </ul>	IG 1.0.0
	125 MB	/1004 MB			61 GB/237 GB			ZOOKEEPER	3.4.9.3
NodeManagers	Resource	eManager Heap Siz	ze		ResourceManager	e			
• 3 • 0	57 MB/		6.21%	:	2 days		1		
	o, 110)				Click	Enable and			
						n enabling of			
SERVICES					the	SMM App	Show All Services		
	can access and manage your cluster. B	elow are the services	that are available in [	DataPlane. Click c	on Enable against a se	pr associating this	cluster to the		
corresponding service.					```				

**4.** Once the SMM app is enabled, you should see the SMM Icon from the app picker. Click on the SMM App to start monitoring the Kafka brokers in the cluster you registered.

<b>S</b>	Admir	n / Cluster	rs / Det	ails								<b>.</b>
			ər									C
DataF	Plane	Streams		DATA CENTER Orlando Data Center - Green Truck Company	CLUSTER VERSION HDP-3.0	NODES 16/16		TAGS		GISTERED AT u Aug 02 2018	REGISTERED BY gvetticaden	
Adi	Descriptie Connecti Reach DataNode 3 3 0	vity able es 0			d, user smm app elect it NameNode Heap Size 125 MB/1004 MB ResourceManager Heap Size		No of Services 26 HDFS Disk Space 61 GB/237 GB ResourceManager Up	25.71% time		Cluster Services  ATLAS HDFS HIVE HIVE KAFKA RANGER STREAMS MESSAGII XANGGER ZOOKEEPER	0.7.0.3.0 3.1.0 1.0.0.3.0 1.0.0.3.0 1.0.0.3.0 NG 1.0.0 3.4.9.3.0	
	SERVICE				57 MB/911 MB		2 days	Show	/ All Services			
		corresponding s	service.	an access and manage your of	uster. Below are the services that are ava	ilable in DataPlane. Clici	k on Enable against a serv	ice for associating this cluster	to the			
V 1.2												

**5.** In the SMM App, every HDP/HDF cluster you enabled with SMM shows up in the cluster dropdown. Hence, a single SMM App can monitor multiple clusters. Select the cluster you want to monitor.

WESSAGING	Overview								Clu	ster: <u>orlandostreamcluster</u> -	4
Overview	Producers 71	-	Brokers 5			enabled	P/HDF cluster y with SMM show s cluster selection	vs	Consum 2	e chicagostreamcluster	Clea
	TOPICS (27) BROKERS (5)					box. F	Pick a cluster to monitor it			ت 30 mir	nutes
	Producers (71)	NAME		DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS			Consumer Groups (26)	,
	ACTIVE (71) PASSIVE (0) ALL	syndicat	e-transmission	DATA IN 641KB	data out <b>10KB</b>	MESSAGES IN	CONSUMER GROUPS	<u>†</u> ©Q ≡	~	ACTIVE (5) PASSIVE (21)	
sumer Groups	MESSAGES minifi-eu-i1 37k	T		04160	TURD	JK	0			nifi-truck-sensors-east	L
	ninifi-eu-i1 37k load-optimizer-apps 35k	aundioa	te-speed-event-j	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	<u>6</u> @Q ,≡		nifi-truck-sensors-west	
	geo-critical-event-collec 25k	Syndica	te-speed-event-j	513KB	OB	1.8k	0	छ⊌⊄≣	×	nifi-truck-sensors-cent	
	minifi-eu-i2 19k									ranger_entities_consu	
	supply-chain-apps 12k	syndicat	te-speed-event-a	DATA IN 317KB	data out 41MB	MESSAGES IN 1.8k	CONSUMER GROUPS	<mark>∲</mark> @Q ≡	~	atlas	
	minifi-eu-i4 9.5k										
	geo-critical-event-collec 9.5k	syndica	te-oil	DATA IN	data out <b>OB</b>	MESSAGES IN 3.5k	CONSUMER GROUPS	60Q 🗐	~		
	predictive-apps 8.9k			/5/KB	UB	3.5K	0				
	geo-critical-event-collec 7.6k			DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS				
	minifi-eu-i5 7.6k	syndica	te-geo-event-json	695KB	OB	1.8k	0	<u>∲</u> @Q ≡	~		
	geo-critical-event-collec 6.9k										
	fuel-apps 6.9k	syndicat	te-geo-event-avro	DATA IN 397KB	DATA OUT 39GB	MESSAGES IN 1.8k	CONSUMER GROUPS	<u>©</u> @Q ≡	~		
	minifi-eu-i6 6.4k										
	geo-critical-event-collec 6.3k	syndica	te-battery	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	<u>†</u> 600,≣	~		
	audit-apps 5.9k geo-critical-event-collec 5.9k	- Official		703KB	OB	3.2k	0	<b>~</b> €Q			
	minifi-eu-i7 5.5k			DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS				
	geo-critical-event-collec 5.4k	syndicar	te-all-geo-critical	26MB	OB	0.1m	0	<u>\$</u> @Q <b>≡</b>	~		
	geo-critical-event-collec 5.4k										

# Monitoring Kafka with SMM

SMM helps address the operational, management and monitoring needs of Kafka for two distinct teams: the Platform Operations and the DevOps / AppDev teams. Each of these teams have a different lens through which they monitor Kafka and hence have different needs for monitoring Kafka.

#### **SMM Platform Operations Persona**

A Platform Operations user is less concerned about the individual performance for a given consumer and/or producer application but rather more focused on the Kafka cluster holistically and the infrastructure that it runs on. Some specific needs, requirements, and questions from a Platform Operator may include the following:

Platform Ops Use Case	Description
Use Case 1	I would like a single platform to monitor all the Kafka clusters within my organization. I want to be able to quickly switch from one Kafka cluster to another.
Use Case 2	I would like to get quick current snapshot of my cluster: number of producers, number of brokers, number of topics, number of consumers.
Use Case 3	Across the entire cluster, which producers are generating the most data right now?
Use Case 4	Across the entire cluster, which of my consumer groups and consumer instances are falling behind with respect to reading from a topic or partition?
Use Case 5	I would like to see a snapshot view of all the Kafka brokers in my cluster with information including the hosts on which the broker is running, throughput in, messages in, number of partitions, and number of replicas.
Use Case 6	Are any of my brokers running hot? Which broker has the highest throughput in and out rates?
Use Case 7	Which topic partitions are on a given Kafka broker?
Use Case 8	Are there any skewed partitions for a broker? What is the throughput in and out for a given partition on that broker?
Use Case 9	For a given broker, topic, or partition, which producers are sending data to it, and which consumer groups are consuming from it.
Use Case 10	View detailed level metrics of a broker across time to see trends and patterns.
Use Case 11	Whats are host metrics on the host where my broker is running? What are the other services running on my broker host?

Let's walk through how SMM can answer these questions for a Platform Operations user:

1. Select the Kafka Cluster you want to monitor with SMM. This takes you to the main dashboard view for that cluster selected. This view gives you a powerful snapshot of the workings of the cluster displaying: total number of active and inactive producers and consumers, all topics with summary metrics. This view also provides the ability to filter on four key Kafka entities: producers, brokers, topics, and consumer groups. This addresses

Use Case 1 which is the ability for SMM to manage multiple Kafka clusters in the organization and Use Case

STREAMS MESSAGING	Overview							Kafka cluster andostreamcl selected	user	Cluster: orlandostreamcluster -	· ≜·
🚳 Overview	Produce 59	rs	•	Brokers 5		-	Topics 27	÷	Consi	umer Groups 26	Clear
Brokers	TOPICS (27) BROKERS	(5)		5						20 D 30 mi	
Topics		/								5 30 mi	nutes •
	Producers (59)		NAME		DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS		Consumer Groups (26)	
Producers	ACTIVE (59) PASSIVE (0)	ALL	syndica	ate-transmission	DATA IN 647KB	DATA OUT 10KB	MESSAGES IN <b>3k</b>	CONSUMER GROUPS	<mark>⊚</mark> @Q≣ ∨	ACTIVE (18) PASSIVE (8)	ALL
	minifi-eu-i1	ESSAGES 39k								route-micro-service	LAG 97k
	load-optimizer-apps	36k	syndica	ate-speed-event-j	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	<u>6</u> 90	load-optimizer-micro-se	
	minifi-eu-i2	20k	• oynaida	ate opeca crent j	523KB	0B	1.8k	0		fuel-micro-service	2.5k
	minifi-eu-i4	9.8k								supply-chain-micro-serv	
	geo-critical-event-collec		syndica	ate-speed-event	DATA IN 319KB	DATA OUT 41MB	MESSAGES IN 1.8k	CONSUMER GROUPS	<mark>⊚</mark> @Q≣ ∨	predictive-micro-service	
	geo-critical-event-collec									energy-micro-service	984
	geo-critical-event-collec	7.1k	syndica	ate-oil	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	<u>©</u> @Q≣ ∨	audit-micro-service	812
	geo-critical-event-collec	6.5k			779KB	0B	3.6k	0	<b>••</b> ••	compliance-micro-servi	698
	minifi-eu-i6	6.5k								adjudication-micro-servi	. 599
	audit-apps	бk	syndica	ate-geo-event-json	DATA IN 703KB	DATA OUT <b>OB</b>	MESSAGES IN 1.8k	CONSUMER GROUPS	<u> ©</u> Q⊡ ∨	approval-micro-service	542
	geo-critical-event-collec	6k								flink-analytics-geo-event	224
	geo-critical-event-collec	5.6k	syndica	ate-geo-event-avro	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	<u>©</u> @Q≣ ∨	kafka-streams-analytics	224
	minifi-eu-i7	5.5k			400KB	39GB	1.8k	8		spark-streaming-analyti	224
	compliance-apps	5.2k								nifi-truck-sensors-east	4
	minifi-eu-i8	4.8k	syndica	ate-battery	DATA IN 706KB	DATA OUT <b>OB</b>	MESSAGES IN 3.2k	CONSUMER GROUPS	<mark>⊚</mark> @Q≣ ∨	nifi-truck-sensors-west	2
	geo-critical-event-collec	4.8k								nifi-truck-sensors-central	1
	geo-critical-event-collec	4.6k	syndica	ate-all-geo-critica	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	<u>ö</u> @Q≣ ∨	ranger_entities_consum	. 1
	adjudication-apps	4.5k		3	16MB	0B	74k	0		atlas	0
	minifi-eu-i9	4.3k									
	approval-apps	4k	supply-	chain	DATA IN 299MB	DATA OUT 530KB	MESSAGES IN 1.4m	CONSUMER GROUPS	<mark>≬</mark> @Q≣ ∨		

2. From the Producers panel on the left hand side of the screen, select the Messages header to sort on messages sent by all the producers in the system. We see that the Kafka producer called minifieui1 is the most active producer, sending 39K messages in the last 30 minutes. This addresses Use Case

STREAMS MESSAGING	Overview									Cluster: orlandostreamo	cluster 👻 🛔
n Overview	on me	essages	ages to sort sent by all the last 30	Brokers 5		-	Topics 27	•	Con	sumer Groups 26	- Clea
Brokers	TOPICS (27)	mir									0 30 minutes
Topics			NAME		DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS			(2.4)
Producers	Producers (59)		syndicate-tra	nsmission	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	<u> 690</u> ×	Consumer Group	. ,
	ACTIVE (59) PASSIVE (0)	SSAGES	Syndicate tra	13111331011	647KB	10KB	3k	0		ACTIVE (18) PASSI	VE (8) ALL
<ul> <li>Consumer Groups</li> </ul>	minifi-eu-i1	39k								route-micro-servic	
	load-optimizer-apps	36k	syndicate-spe	eed-event-j	DATA IN 523KB	DATA OUT <b>OB</b>	MESSAGES IN 1.8k	CONSUMER GROUPS	<u>†</u> ©Q≣ ∨	load-optimizer-mid	ro-se 5.1k
	ninifi-eu-i2	20k	T		OLOND	00	non			fuel-micro-service	2.5k
A Kafka producer called minfi-eu-i1 is the most		9.8k			DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS		supply-chain-micro	>-serv 1.7k
active producer sending	geo-critical-event-collec	9.8k	syndicate-spe	eed-event	319KB	41MB	1.8k	3	<u>†</u> ©Q≡ ∨	predictive-micro-s	ervice 1.3k
39K messages in the las 30 mins	t geo-critical-event-collec	7.7k								energy-micro-serv	ice 984
SUTIIIIS	geo-critical-event-collec	7.1k	syndicate-oil		DATA IN 779KB	DATA OUT	MESSAGES IN 3.6k	CONSUMER GROUPS	<b>©Q</b> ⊡ ∨	audit-micro-servic	e 812
	geo-critical-event-collec	6.5k			77980	00	3.0K	0		compliance-micro	·servi 698
	minifi-eu-i6	6.5k			DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS		adjudication-micro	-servi 599
	audit-apps	6k	syndicate-geo	o-event-json	703KB	OB	1.8k	0	<u>†</u> ©Q≡ ∨	approval-micro-se	rvice 542
	geo-critical-event-collec	6k								flink-analytics-geo	event 224
	geo-critical-event-collec	5.6k	syndicate-geo	o-event-avro	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	<u>⊚</u> Q⊒ ∨	kafka-streams-ana	lytics 224
	minifi-eu-i7	5.5k			400KB	39GB	1.8k	8		spark-streaming-a	nalyti 224
	compliance-apps	5.2k			DATA IN	DATA OUT	N5004050 IN			nifi-truck-sensors-	east 4
	minifi-eu-i8	4.8k	syndicate-bat	ttery	DATA IN 706KB	0B	MESSAGES IN 3.2k	CONSUMER GROUPS	<u>†</u> @Q≡ ∨	nifi-truck-sensors-	west 2
	geo-critical-event-collec	4.8k								nifi-truck-sensors-	central 1
	geo-critical-event-collec	4.6k	syndicate-all-	geo-critica	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	<u>ö</u> @Q≣ ∨	ranger_entities_co	nsum 1
	adjudication-apps	4.5k		5	16MB	0B	74k	0	Yest .	atlas	0
	minifi-eu-i9	4.3k									
	approval-apps	4k	supply-chain		DATA IN 299MB	DATA OUT 530KB	MESSAGES IN 1.4m	CONSUMER GROUPS	<mark>∲</mark> @Q≣ ∨		
«	minifi-eu-i10	3.9k									

**3.** On the Consumer Groups panel on the right hand side, select the column LAG to sort on consumer group lag. This lag is defined as the summation of all consumer instances lag in the consumer group. This addresses

Use Case 4 because we can easily see that we have a consumer group called micro-alert-service that has a lag of 97K over the last 30 minutes which is significantly more than any other consumer group in the

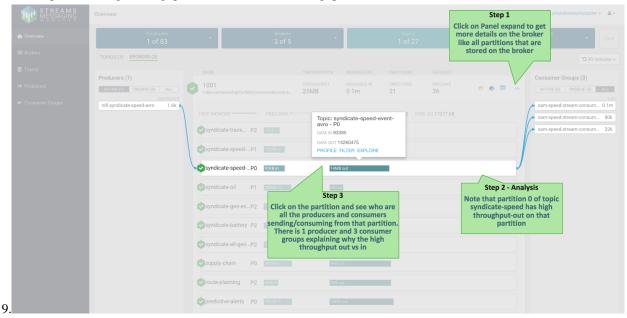
MESSAGING	Overview								Cluster: orlandostreamo	cluster 👻 🛔 👻	
d Overview	Produc		• Broker 5	rs ,	-	Topic: <b>27</b>	s 🔹	Cons	umer Groups 26	consume	LAG to sort on r lag across all rs in the last 30
Brokers	TOPICS (27) BROKERS	; (5)							-		mins
Topics			NAME	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS				
Producers	Producers (59)			DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS		Consumer Group		
C Producers	ACTIVE (59) PASSIVE (0)		syndicate-transmission	647KB	10KB	3k	0	<u>©</u> @Q≣ ∨	ACTIVE (18) PASSI	V	
	minifi-eu-i1	essages• 39k							route-micro-servic	e 97k	Consumer group name
	load-optimizer-apps	36k	syndicate-speed-event-j	523KB	DATA OUT OB	MESSAGES IN 1.8k	CONSUMER GROUPS	<u> </u>	load-optimizer-mic	cro-se 5.1k	route-micro-service h
	minifi-eu-i2	20k		JZJKD	00	1.0K	0		fuel-micro-service	2.5k	significantly more la (97K) than any anoth
	minifi-eu-i4	9.8k		DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS		supply-chain-micro	o-serv 1.7k	consumer in the clust
	geo-critical-event-collec	9.8k	syndicate-speed-event	319KB	41MB	1.8k	3	¢@Q≣ ∨	predictive-micro-s	ervice 1.3k	
	geo-critical-event-collec	7.7k							energy-micro-servi	ice 984	
	geo-critical-event-collec		syndicate-oil	DATA IN 779KB	DATA OUT OB	MESSAGES IN 3.6k	CONSUMER GROUPS	¢@Q≣ ∨	audit-micro-service	e 812	
	geo-critical-event-collec	6.5k		779KB	OB	3.0K	0		compliance-micro-	-servi 698	
	minifi-eu-i6	6.5k		DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS		adjudication-micro	o-servi 599	
	audit-apps	6k	syndicate-geo-event-json	703KB	OB	1.8k	0	<b>©</b> @Q≡ ∨	approval-micro-set	rvice 542	
	geo-critical-event-collec	6k							flink-analytics-geo	-event 224	
	geo-critical-event-collec	5.6k	syndicate-geo-event-avro	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	<u>6</u> @Q≣ ∨	kafka-streams-ana	alytics 224	
	minifi-eu-i7	5.5k		400KB	39GB	1.8k	8		spark-streaming-a	nalyti 224	
	compliance-apps	5.2k							nifi-truck-sensors-	east 4	
	minifi-eu-i8	4.8k	syndicate-battery	706KB	DATA OUT OB	MESSAGES IN 3.2k	CONSUMER GROUPS	¢@Q≣ ∨	nifi-truck-sensors-	west 2	
	geo-critical-event-collec	4.8k							nifi-truck-sensors-	central 1	
	geo-critical-event-collec	4.6k	syndicate-all-geo-critica	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	<u>ö</u> @Q≣ ∨	ranger_entities_co	nsum, 1	
	adjudication-apps	4.5k	- Synanoute un geo-ontiod.	16MB	0B	74k	0	<u>~</u> ₩ч= *	atlas	0	
	minifi-eu-i9	4.3k									
	approval-apps	4k	supply-chain	299MB	DATA OUT 530KB	MESSAGES IN 1.4m	CONSUMER GROUPS	<mark>©</mark> @Q≣ ∨			
	minifi-eu-i10	3.9k									

4. Click the Brokers tab to see a broker centric view of the dashboard. You can view important metrics for each broker in the cluster including: hosts the broker is running on, throughput in, messages in, number of partitions, and number of replicas. Click the THROUGHPUT column to sort by throughput across all brokers. You can easily see that broker 1001 has the highest rate of data in over the last 30 minutes: 80K messages totaling 17MB. This addresses Use Cases 5 and

Click on t	itep 1 he Brokers tab proker centric			tep 2 hroughput to					(	Cluster: orlandostream	ncluster	- <u> </u>
	he Dashboard		sort on da	ta in across a rokers		Topics 27	÷		Cons	umer Groups 26	•	
Brokers	TOPICS (27) BROKERS (5)										<b>'D</b> 30 mi	inutes
Topics	Producers (59)		NAME	THROUGHPUT	MESSAGES IN	PARTITIONS	REPLICAS			Consumer Grou	aa (26)	
Producers	ACTIVE (59) PASSIVE (0)		1001 c-dps-connected-dp13.field.hortonwor	THROUGHPUT	MESSAGES IN 80k	partitions 21	REPLICAS	🍅 🔕 📼	~	ACTIVE (18) PASS		ALL
<ul> <li>Consumer Group</li> </ul>	Analysis	AGES 391	7							route-micro-servi	се	LAG 97k
Brok	ker 1001 has the highest		1002	THROUGHPUT	MESSAGES IN	PARTITIONS	REPLICAS	🧔 🔕 🥸	~	load-optimizer-m	icro-se	. 5.1k
rate o	of data in over the last 30	20k	c-dps-connected-dp12.field.hortonwor	TOMB	7 DK	10	34			fuel-micro-servic	e	2.5k
mins	s. 80K messages totaling 17MB	.8k	1005	THROUGHPUT	MESSAGES IN	PARTITIONS	REPLICAS			supply-chain-mic	ro-serv	. 1.7
	271110	.8k	c-dps-connected-dp11.field.hortonwor	14MB	63k	15	31	🧔 🗞 🧔	~	predictive-micro-	service	1.3k
	geo-critical-event-collec	7.7k								energy-micro-ser	vice	984
	geo-critical-event-collec	7.1k	1003 c-dps-connected-dp14.field.hortonwor	THROUGHPUT 10MB	MESSAGES IN	PARTITIONS	REPLICAS 30	🧔 🔕 🥸	~	audit-micro-servi	ce	812
	geo-critical-event-collec	6.5k	c-aps-connected-ap14.neid.nortonwor	TONID	721	10	50			compliance-micr	o-servi	698
	minifi-eu-i6	6.5k	1004	THROUGHPUT	MESSAGES IN	PARTITIONS	REPLICAS			adjudication-mic	ro-servi	. 599
	audit-apps	6k	c-dps-connected-dp15.field.hortonwor	7MB	33k	14	30	🧔 🗞 🧔	~	approval-micro-s	ervice	542
	geo-critical-event-collec	бk								flink-analytics-ge	o-event	224
	geo-critical-event-collec	5.6k								kafka-streams-ar	nalytics	. 224
	minifi-eu-i7	5.5k								spark-streaming-	analyti	224
	compliance-apps	5.2k								nifi-truck-sensors	s-east	4
	minifi-eu-i8	4.8k								nifi-truck-sensors	s-west	2
	geo-critical-event-collec	4.8k								nifi-truck-sensors	-central	1
	geo-critical-event-collec	4.6k								ranger_entities_c	onsum	. 1
	adjudication-apps	4.5k								atlas		0
	minifi-eu-i9	4.3k										
	approval-apps	4k										
*	minifi-eu-i10	3.9k										

**5.** Click on the broker panel to expand it and see more details and metrics for that broker. The expanded panel shows all partitions for different topics that are stored on that broker. For each partition, we see the throughput in and out relative to other partitions. We can click on a given partition and see all the

consumers currently sending data to that broker, topic, or partition and all consumer groups consuming from that broker, topic, or partition. We can easily see that partition 0 of topic syndicate-speed on broker 1001 has considerably higher throughput out than any of the other partitions on that broker. By viewing how data flows in and out of that partition, we see that the partition has 1 producer but 3 consumer groups which explains the high throughput out relative to the throughput in. This addresses Use Case 7, 8 and



**6.** Click the Grafana icon on the broker panel to see more detailed metrics within Grafana for that broker. This addresses Use Case

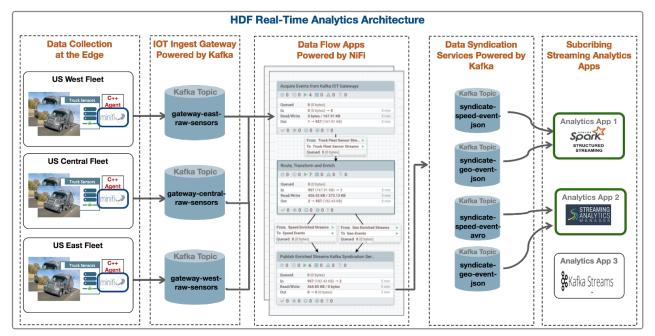
NAME	THROUGHPUT -	MECCACEC IN	PARTITIONS	REPLICAS			
1001	THROUGHPUT	MESSAGES IN	PARTITIONS	REPLICAS	0	Click on the Grafana icon on the	
c-dps-connected-dp13.field.hortonwor	17MB	80k	21	36		broker panel and a Grafana dashboard for that broker is	
1002	THROUGHPUT	MESSAGES IN	PARTITIONS	REPLICAS	o 🙍 🗖 🗸	displayed providing more broker	
c-dps-connected-dp12.field.hortonwor	16MB	75k	16	34		metrics graphed across time	
1005	THROUGHPUT	MESSAGES IN	PARTITIONS	REPLICAS	ö 🗶 🗖 🗸		
c-dps-connected-dp11.field.hortonwor	14MB	03K	🇔 📰	Kafka - Hosts .		• • • • • • • • • • • • • • • • • • •	
1003	THROUGHPUT	MESSAGES IN	PA kafka_broke	er - Cluster: -		show: top - 20 - aggregator: avg -	≡ Kafka Dasht
c-dps-connected-dp14.field.hortonwor	TOMB	425			Selected (1)	per broker level. Click on each row title to expand on demand to look at variou y Ardak. You my less my charge made to the dealford. If you want to catorine, make your own copy.	s metrics.
1004	THROUGHPUT	MESSAGES IN	PA		c-dps-connected-dp1 c-dps-connected-dp1 c-dps-connected-dp1	rtonworks.com BYTES / MESSAGES / PARTITIONS	
c-dps-connected-dp15.field.hortonwor	7 IMID	33K	16 GIB		Bytes In 6 c-dps-connected-dp1 c-dps-connected-dp1	rtonworks.com 1.0	Under Replicated Partitions
			14 GIB 12 GIB		c-dps-connected-dp1	23.0 Mil 0.5	
			9 GIB				
			7 GB 5 GB				
			2 GB	09:00 10:00	0 11:00 12:00 12:00	14.00 21.5 Mil 08:00 10:00 11:00 12:00 13:00 14:00 -1.0 08:00	
						PRODUCER & CONSUMER REQUESTS	
			11.0 Mi		Producer Reque	7.65 Mil	er Requests /s
			10.8 MI				
			10.5 MI				
			10.0 Mi				
			9.8 MI			7/79/9	
				8:30 08:00		12:00 12:30 13:00 13:30 14:00 08:30 08:30 08:30 08:30 10:30 10:30	
					Partition Count	Leader Count	MaxLag Replica
			65.010				
			es.005				
			65.995				
			65.990				
			65.935			20.000	
			65.935	08:00 10:00	11:00 12:00 13:00	33,990 -1.0 -1.0 400 09:00 10:00 11:00 12:00 13:00 14:00 09:00	10:00 11:00 12:00 13:00

7. Click the Ambari icon on the broker panel and view Broker Host detail metrics. This addresses Use Case

	NAME 1001 c-dps-connected-dp13.field.hortonwor	THROUGHPUT • THROUGHPUT 17MB	MESSAGES IN MESSAGES IN 80k	partitions partitions 21	REPL REPL 36	cas displayed metrics and	iew for that broker is d providing host level a view of other services ning on that host
9	1002 c-dps-connected-dp12.field.hortonwor	THROUGHPUT 16MB	MESSAGES IN 75k	partitions 16	8 8	A / Hosts / c-dps-connected-dp13.field.hortonworks.com / S	Summary orlandostr 🌣 🗊 🔺 🌖 🏢 🔳 🗴
	1005 c-dps-connected-dp11.field.hortonwor	THROUGHPUT 14MB	MESSAGES IN 63k	partitions 15	R	łost: c-dps-connected-dp13.field.hortonworks.com o summary configs alerts o versions logs	HOST
	1003 c-dps-connected-dp14.field.hortonwor	THROUGHPUT 10MB	MESSAGES IN 42k	PARTITIONS	R	Components	+ ADD Host Metrics LAST 1
	1004 c-dps-connected-dp15.field.hortonwor	THROUGHPUT 7MB	MESSAGES IN 33k	partitions 14	R	Kufka Bioker / Kafka Master     HST Agent / Smithers Size     Log Feeder / Log Seach Size     Metrics Monitor / Ambair Metrics     Korthers Client     Client	50%
						Summary	1
						Hostname: c-dps-connected dp13 field hortonworks.com IP Address: 172.26.254.59 Rate: (default-rack / 05: center 27 (086.54) Corres (CPU); 8(8) Disk: 63.408/179.9058 (7.93%, used) Memory: 14.5308 Load Arej: 0.02 Heartbeat: (ess than a minute ago	0.5 2.5 2.5 2.5 2.5 2.5 2.5 2.5 2

## **DevOps and Application Developer Persona**

Unlike a Platform Operations persona, the DevOps / AppDev persona is most interested in the entities (producers, topics, consumers) specific to their application. So let's assume we are on the DevOps team responsible for monitoring the Trucking Reference Application that is deployed in production, based on this architecture.



Some specific needs/requirements/questions as a DevOps member for the trucking ref app might be the following.

DevOps/App Dev Use Case	Description
Use Case 1	I want to quickly find all entities (producer, consumers, topics) associated with my application.

Use Case 2	For all the topics associated with my applications, I want to easily see important metrics such as throughput in/out, number of consumer groups, number of messages across a period of time.
Use Case 3	Which of my topics is being sent the most amount of data over a certain period of time? In other words, which regional/geo truck fleet is sending the most amount of data?
Use Case 4	For a given Kafka topic that is part of my application, which are all the connected producers sending data? In other words, which truck fleets are sending data to a gateway topic?
Use Case 5	Are there any topics who have producers but no consumer groups connected to it? In other words, are trucks sending data to a topic but no analytics or processing is being done?
Use Case 6	For a given topic, how many partitions are there? Where are the partitions located? How is data distributed across the partitions? Are there any partition skews?
Use Case 7	Which consumer groups are connected to a given topic and are actively consuming data from it?
Use Case 8	For a given topic, I want to be able to explore data in the topic searching using offsets and/or partition.
Use Case 9	I want to find Metadata and Lineage of the Topic across producers, consumers, and multiple Kafka hops.

Let's walk through how SMM can answer these questions for a DevOps/App Dev user.

**1.** Select the Topic filter and select all the IoT gateway topics by searching for all topics that start with gateway.

MESSAGING	Overview											Cluster: orlandostreamcluster	r* Å
20 Overview	Producer 83	5			okers 5	•		Topics 27			Cons	umer Groups	
Brokers	TOPICS (27) BROKERS (5)								Q			Filter to filter on diselect all the IOT	minutes
Topics				NAME	DATA IN -	DAT	gatev	v	ų	top		teway topics	minute
	Producers (83)				DATA IN	DAT	🔲 Na	me					
Producers	ACTIVE (83) PASSIVE (0)	ALL	V	syndicate-all-geo-critical-ev	e 26MB	OB	🔲 gat	leway-europe-raw-senso	rs 🔲	<u> 영</u> 및 📼	× 1	ACTIVE (18) PASSIVE (8)	ALL
Consumer Groups	route-apps	0.1m				_	🔲 gat	leway-west-raw-sensors				route-micro-service	0.2
	minifi-eu-i1	25k		route-planning	DATA IN 26MB	DAT 28	🗐 gat	leway-central-raw-senso	rs 📼	6 9 Q II	~	load-optimizer-micro-service	2 1
	load-optimizer-apps	23k			ZOIMB	28	🗉 gat	leway-east-raw-sensors				fuel-micro-service	
	geo-critical-event-collector-i1	23k			DATA IN	DAT	A OUT	MESSAGES IN	CONSUMER GROUPS			supply-chain-micro-service	
	minifi-eu-i2	12k		gateway-europe-raw-sensor	rs 18MB	0B		88k	0	👨 😡 Q 🕅	~	predictive-micro-service	
	geo-critical-event-collector-i2	11k										energy-micro-service	2
	geo-critical-event-collector-i3	7.7k		load-optimization	DATA IN 5MB	DAT. 28	A OUT	MESSAGES IN 23k	CONSUMER GROUPS	👩 😡 Q 🔳	~	audit-micro-service	1
	minifi-eu-i4	6.3k			SIVID	201	ND	ZOK	1			compliance-micro-service	1
	geo-critical-event-collector-i8	6.3k			DATA IN	DAT	A OUT	MESSAGES IN	CONSUMER GROUPS			adjudication-micro-service	1
	geo-critical-event-collector-i4	5.8k		fuel-logistics	1MB	28		6.5k	1	🙇 🛛 Q 🔳	~	approval-micro-service	1
	fuel-apps	5.3k										flink-analytics-geo-event	1
	geo-critical-event-collector-i10	5k		supply-chain	DATA IN 863KB	DAT. 28	A OUT	MESSAGES IN 4.3k	CONSUMER GROUPS	🧑 😡 Q 🔳	~	kafka-streams-analytics-geo.	. 1
	geo-critical-event-collector-i5	4.6k			003KD	201	ND	4.3K	1			spark-streaming-analytics-g.	. 5
	geo-critical-event-collector-i11	4.6k			DATA IN	DAT	A OUT	MESSAGES IN	CONSUMER GROUPS			nifi-truck-sensors-west	
	minifi-eu-i6	4.2k		audit-events	804KB	27		3.9k	1	👨 😡 Q 📼	ř	nifi-truck-sensors-east	
	geo-critical-event-collector-i12	4.2k										nifi-truck-sensors-central	
	audit-apps	3.9k		compliance	DATA IN 697KB	DAT. 291	A OUT	MESSAGES IN 3.4k	CONSUMER GROUPS	🧑 😡 Q 📼	~	ranger_entities_consumer	
	geo-critical-event-collector-i13	3.9k			03140	29	ND D	0.4K	1			atlas	
	minifi-eu-i3	3.8k			DATA IN	DAT	A OUT	MESSAGES IN	CONSUMER GROUPS				
<	geo-critical-event-collector-i6	3.8k		predictive-alerts	648KB	28		3.3k	1	🧔 😔 🥸	~		

2. When the filter is applied, SMM provides intelligent filtering by showing only the producers that are sending data to the 4 gateway topics and the consumer groups only consuming data from those topics. So, when the user selects the 4 gateway topics, SMM displays 34 of the 83 producers sending data to those topics and 3 of the 26 consumer groups consuming data from it. Key metrics for the selected topics are shown including

data-in and out, number of messages, number of consumer groups, etc. This addresses Use Cases 1 and

Intelligent Filte SMM automatically producers associated selected topics. 34 of producers have been	filters the d with the of the 84								Clu	Intelligent Filte SMM automatically consumers associate selected topics. 3 ( consumers have identified as consur from the 4 topics	filters the d with th of the 26 been ming data
as sending data to th	ne 4 topics	Producers 34 of 83	÷.	Brokers 5 of 5	-	Topics 4 of 2			Consumer ( 3 of 2		
selected	CS (4)	BROKERS (5)				User Action				30 minutes -	
Topics	Producers (	34)	NAME	DATA	IN - DAT 41	OT Gateway topi	cs have		C	onsumer Groups (3)	
		PASSIVE (0) ALL	gateway-europe	-raw-sensors 18N		been selecte	d ROUPS	🧑 🖌 Q 📼		ACTIVE (3) PASSIVE (0) ALL	
<ul> <li>Consumer Groups</li> </ul>	minifi-eu-i1	MESSAGES 25k	T							LAG nifi-truck-sensors-west 2	
	minifi-eu-i2	25k 12k	gateway-west-ra	DATA	IN DATA OUT	MESSAGES IN	CONSUMER GROUPS	5 Q Q 💷		nifi-truck-sensors-east 2	
	minifi-eu-i4	6.3k	gateway-west-n	aw-sensors 225	KB 225KB	1.1k	1	5 9 Q 📖		nifi-truck-sensors-central 1	
	minifi-eu-i6	4.2k								The second s	
	minifi-eu-i3	3.8k	gateway-central	-raw-sensors 146		MESSAGES IN 731	CONSUMER GROUPS	🧒 🖌 Q 📼	~		
	minifi-eu-i7	3.6k									
	minifi-eu-i8	3.1k	gateway-east-ra	DATA		MESSAGES IN	CONSUMER GROUPS	6 e q 💷			
	minifi-eu-i9	2.8k	gatemay cust to	111	KB 111KB	551	1	000	Ť		
	minifi-eu-i10	) 2.5k									
	minifi-eu-i5	2.3k									
	minifi-eu-i12	2 2.1k									
	minifi-eu-i13	3 1.9k									
	minifi-eu-i14	1.8k									
	minifi-eu-i15	5 1.7k									
	minifi-eu-i16	5 1.6k									
	minifi-eu-i17	7 1.5k									
	minifi-eu-i18	1.4k									
	minifi-eu-i19	1.3k									
	minifi-eu-i20	) 1.3k									
<	minifi-eu-i22	2 1.2k									

**3.** Click DATA IN to sort on data throughput-in across all topics. We see that gateway-europe-raw-sensor has significantly more data coming in than any other topic: 18 MB totaling 88K in the last 30 minutes. This addresses Use Case 3.

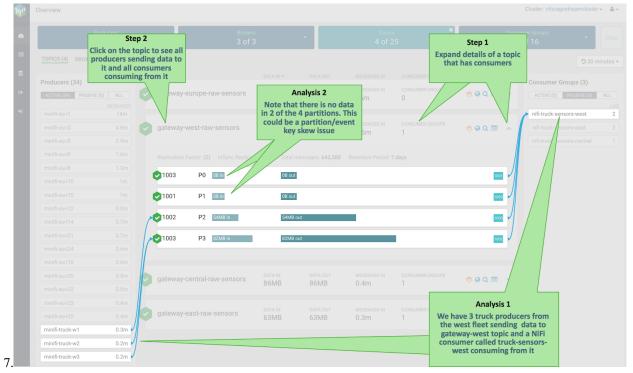
	S Overview			Step						Cluster: orlandostreamo	luster -
2 Overview		roducers 4 of 83	- Bro	Click on DAT/ on data-in opics in the l	across all	Topic 4 of 2				mer Groups of 26	- Ck
III Brokers	TOPICS (4) BROKERS	6 (5)									30 minute
Topics			NAME	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS				
	Producers (34)			DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS			Consumer Groups (3	:)
Producers	ACTIVE (34) PASSIVE		gateway-europe-raw-sensors	- 18MB	OB	88k	0	🧑 🥹 Q 🕅	~	ACTIVE (3) PASSIVE	
•) Consumer		MESSAGES								nifi-truck-sensors-wes	L t
	Analysis		gateway-west-raw-sensors	DATA IN	DATA OUT	MESSAGES IN	CONSUMER GROUPS	6 Q Q 🔳		nifi-truck-sensors-east	
Kafka	topic called gateway-er sensors has more data l	urope-	gateway-west-taw-sensors	225KB	225KB	1.1k	1	10 V U	Ť	nifi-truck-sensors-cent	
messa	to it than any other topi ages totaling 18 MB in t 30 mins	he last	gateway-central-raw-sensors	DATA IN 146KB	DATA OUT 146KB	MESSAGES IN 731	CONSUMER GROUPS	<mark>6</mark> Q Q 🔳	~		
	minifi-eu-i7	3.6k									
	minifi-eu-i8	3.1k	gateway-east-raw-sensors	DATA IN 111KB	DATA OUT 111KB	MESSAGES IN 551	CONSUMER GROUPS	🧑 😡 Q 🕅	~		
	minifi-eu-i9	2.8k									
	minifi-eu-i10	2.5k									
	minifi-eu-i5	2.3k									
	minifi-eu-i12	2.1k									
	minifi-eu-i13	1.9k									
	minifi-eu-i14	1.8k									
	minifi-eu-i15	1.7k									
		1.6k									
	minifi-eu-i16	1.0K									
	minifi-eu-i16 minifi-eu-i17	1.6K									
	minifi-eu-i17	1.5k									
	minifi-eu-i17 minifi-eu-i18	1.5k 1.4k									

4. Expand the topic panel for gateway-europe-raw-sensor to get more details for the topic, including like partition layout. Click the topic to see which producers are sending data to each partition of that topic. We see that all the producers are trucks from the EU fleet. Also note that all five partitions for that topic have 0B going out and we see no data flowing from the topic to any consumer group. This could be worth investigating to

He	Overview								С	Step 1
æ		Step 2 Click on the To who are all the	pic to see producers					-	Con	Expand topic panel to see more details of the topic that has high data-in rates
	TOPICS (4) BRO	sending data to	the topic						_	D 30 minutes -
8	Producers (34)				DATA IN 🔻					Consumer Groups (3)
•	ACTIVE (34) PASSIVE		gateway-e	urope-raw			MESSAGES IN 35m		<u>∜</u> @Q≣ ^	ACTIVE (0) PASSIVE (3) ALL
•	minifi-eu-i1	MESSAGES								
	minifi-eu-i3	4.8m							od: 7 days	nifi-truck-sensors-e2
	minifi-eu-i5	2.9m	1001	P0 2GB in	0B out				1002	nifi-truck-sensors-c 1
	minifi-eu-i9	1.6m	-						—7/	
	minifi-eu-i8	1.3m	1002	P1 658MB in	0B out					
	minifi-eu-i10	1m	1003	P2 3GB in	0B out				1001	
	minifi-eu-i15	1m 🦂 🖊								
	minifi-eu-i12	0.8m	21001	P3 2GB in	0B out			Analy	sis	
	minifi-eu-i14	0.7m	1002	P4 339MB in	0B out			te that for each no data going o		
	minifi-eu-i21	0.7m	1002	F-4 005/10/11	00.001		see	no data going t	o any consume	er
	minifi-eu-i24	0.6m					to	pups. This mean pic has lots of p	roducers, there	
	minifi-eu-i18	0.6m		vest-raw-se			MESS	is no consumers indicate a p		
	minifi-eu-i20	0.5m	guteway	10011011-00	136MB	136MB	0.6n	1		
	minifi-eu-i22	0.5m								
5.	minifi-eu-i23	0.4m	gateway-c	entral-raw-s	data in 86MB	DATA OUT 86MB	MESSAGES IN 0.4m	CONSUMER GROUPS	େତ୍ତ୍ର ସ≣ ∽	

identify why a topic with the most amount of producers has no consumers. This addresses Use Cases 4 and

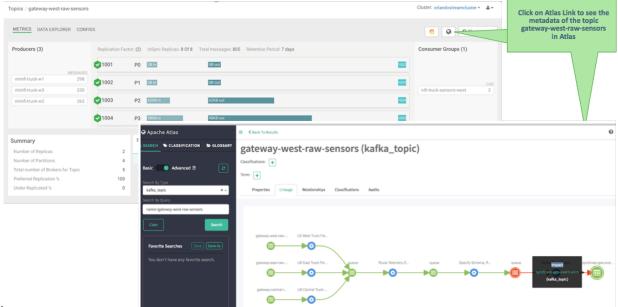
**5.** Click another topic called gateway-west-raw-sensors that has consumers. This topic has three producers sending data to it and a NiFi consumer consuming from it. We also see that two of the four partitions for this topic have no data in them, which indicates that there is partition skew issue. This addresses Use Cases 6 and



6. Click the explorer/magnifying glass icon to search for events in the selected Kafka topic. This addresses Use Case

Producers (34)		NAME	DATA IN -	DATA OUT	MESSAGES IN	CONSUMER GROUPS		Consumer Groups (3)	Click on the explorer icon to search for events in the Kafka Topic
ACTIVE (34) PASSP	/E (0) ALL	gateway-europe-raw-sensors		DATA OUT OB	MESSAGES IN CONSU 88k 0	CONSUMER GROUPS	<mark>6</mark> 90,⊞ ∨		спе катка торіс
minifi-eu-i1	MESSAGES 25k							eur west	2
minifi-eu-i2	12k	gateway-west-raw-sensors		DATA OUT 225KB	MESSAGES IN 1.1k	CONSUMER GROUPS	6 9 Q III 🔨	nifi-truck-sensors-east 2	
minifi-eu-i4	6.3k							nifi-truck-sensors-central	1
minifi-eu-i6	4.2k	Replication Factor: (2) Indyric Replicas: 8 of 8 Total mess							
minifi-eu-i3	3.8k								
minifi-eu-i7	3.6k							NESER	HALIZER: Keys: String - Values: String
minifi-eu-i8	3.1k	21002 P1 08m	08 out						Palace Palace only
minifi-eu-i9	2.8k	1003 P2 8848 in	88K8 out	1		FROM OFFSET			TO OFFS
minifi-eu-i10	2.5k	P2 000011	8049.031	Partition	2 •	275851	45976	91952 137928	183904 229880 <b>27586</b> 0
minifi-eu-i5	2.3k	21004 P3 137K8 in	137KB ou			0	43976	31325 131350	163904 22960 27560
minifi-eu-i12	2.1k			Offset 🕶	Timestamp	Value			
minifi-eu-i13	1.9k			275850	Tue, Jul 31 2018,	5:59:32 2018-07-31	22:59:32.679 15330779	972679 truck_geo_event 455 10 Geor	ge Vetticaden 10 Saint Louis to Tulsa Normal 37.8 -92.48 1
minifi-eu-i14	1.8k	gateway-central-raw-sensors	DATA IN 146KB	275851	Tue, Jul 31 2018,	5:59:32 2018-07-31	22:59:32.68 153307797	72680[truck_speed_event 455 10]Geo	rge Vetticaden 10 Saint Louis to Tulsa 72
minifi-eu-i15	1.7k	· · · · · · · · · · · · · · · · · · ·		275852	Tue, Jul 31 2018,	5:59:37 2018-07-31	22:59:37.63 153307797	77630[truck_geo_event]455[10[Georg	e Vetticaden 10 Saint Louis to Tulsa Normal 37.81 -92.31 1
minifi-eu-i16	1.6k		DATA IN	275853	Tue, Jul 31 2018,	5:59:37 2018-07-31	22:59:37.631115330779	977631 itruck speed eventi455110 Ge	orge Vetticaden 10 Saint Louis to Tulsa)68
minifi-eu-i17	1.5k	gateway-east-raw-sensors	111KB		Tue, Jul 31 2018,				ge Vetticaden 10 Saint Louis to Tulsa Normal 37.81 -92.08 1
minifi-eu-i18	1.4k								
minifi-eu-i19	1.3k				Tue, Jul 31 2018,				orge Vetticaden 10 Saint Louis to Tulsa 66
minifi-eu-i20	1.3k			275856	Tue, Jul 31 2018,	5:59:45 2018-07-31	22:59:45.76 153307798	85760 truck_geo_event 455 10 Georg	e Vetticaden 10 Saint Louis to Tulsa Normal 37.94 -91.99 1
minifi-eu-i22	1.2k			275857	Tue, Jul 31 2018,	5:59:45 2018-07-31	22:59:45.76 153307798	85760 truck_speed_event 455 10 Geo	rge Vetticaden 10 Saint Louis to Tulsa 73
				275858	Tue, Jul 31 2018,	F.F.C.F.C. 0010.07.03	22.50.50 (70) 522077	00660 truck and quanti 455110 Georg	ge Vetticaden 10 Saint Louis to Tulsa Normal 37.99 - 91.69 1

7. Click the Atlas icon to see the metadata and lineage of the Kafka topic in Atlas. This addresses Use Case



9.