Using HDP and HDF to Harness the Power of Social Media

This lab will walk through the collection and analysis of Twitter data leveraging the Hortonworks Data Platform 2.6 (HDP) and Hortonworks DataFlow 3.0 (HDF), combining both data in motion and data at rest. While the initial analytics are simple, this pattern enables the building blocks for more advanced use cases.

During this lab, participants will be able to perform Twitter live feed data ingestion based on some keywords, filtering, transformation, attribute extraction, and delivery to HDFS for data storage, as well as Apache Kafka using Apache NiFi.

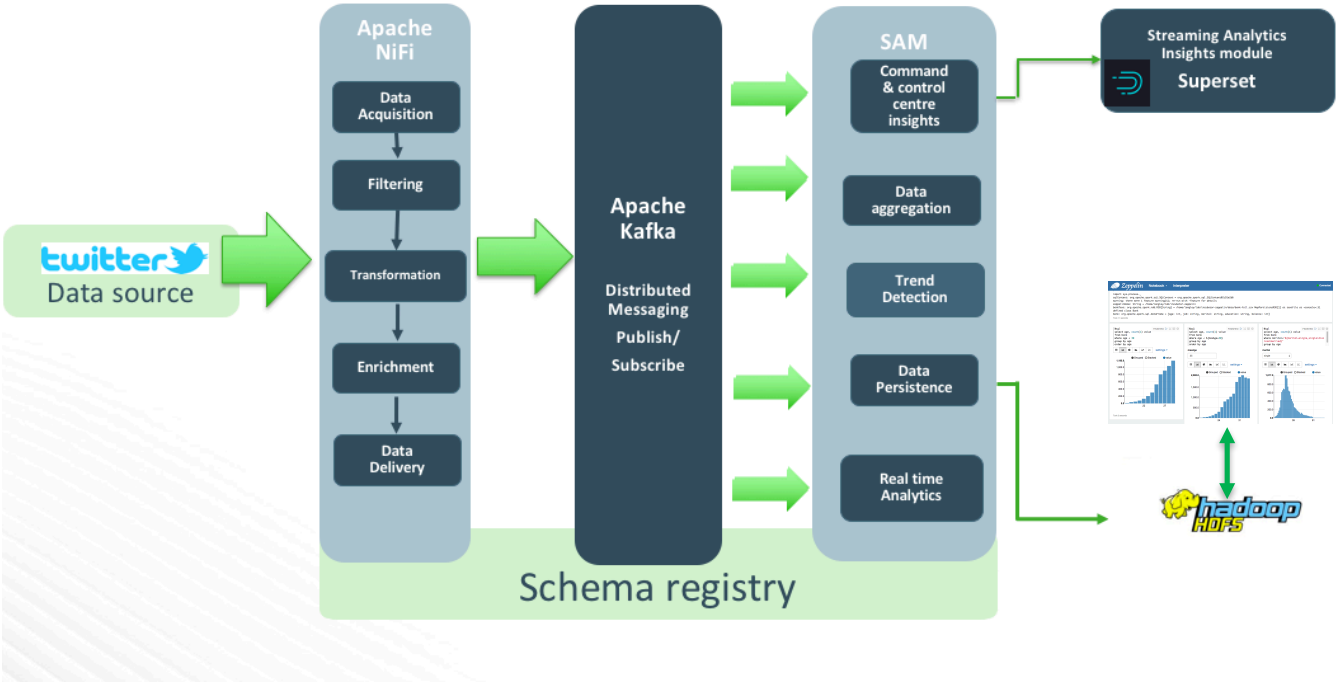
Participants will then be able to perform analytics on data at rest using Hive, SparkSQL as well as Apache Zeppelin for visualization, and data exploration.

Once the data is ingested into Kafka, participants will be able to develop quickly a Streaming Analytics Manager application to further analyze data and derive real-time insights from the data. Data will also be visualized in near real-time using Apache Druid and Superset.

Overview 3
Pre-requisites 4
Create your Twitter API keys 5
Create A Twitter Application 5
Verify your Hortonworks Sandbox web access 8
Using Apache NiFi for ingesting Twitter Data 9
Update the HDF schema registry 9
Download the Twitter dataflow template 11
Import the Template 11
Instantiate the Template 13
Configure the Twitter Grab Garden Hose Processor 14
Configure PublishKafkaRecord_0_10 processor 16
Configure the Kafka Brokers URI 17
Configure the Schema Registry and Enable all Controller Services 18
Review the remaining Processors 20
Start the Flow 21
Check the data on HDFS 22
Real-Time Data Analysis with Streaming Analytics Manager 24
Prepare the Environment 24
Define the Service Pool 25
Define a Development Environment 27
Develop a SAM Application 29
Read data from Kafka 31
Ingest data into Druid 32
Aggregate real-time Twitter data 34
Filter Data in real-time 35
Store results on HDFS 38
Configure the SAM Application 40
Run the SAM Application 41
Monitoring the Storm Application 42
Checking data on HDFS 43
Data Analysis with Hive 44
Create the Hive Table 45
Query Data using SQL on Hive 46
Data Visualization with Zeppelin 47
Real-Time Data Visualization with Apache Druid and Superset 51
Locate the Druid Twitter datasource 51
Create a visualization and a dashboard 53

Overview

Below is a conceptual diagram of the implementation of Twitter Analytics performed in this lab using Hortonworks Data Flow and Hortonworks Data Platform.



Pre-requisites

Where possible, the following pre-requisites should be completed prior to start of the lab. Please follow the instructions in the Lab prerequisites document. Prerequisites steps included in the documents are:

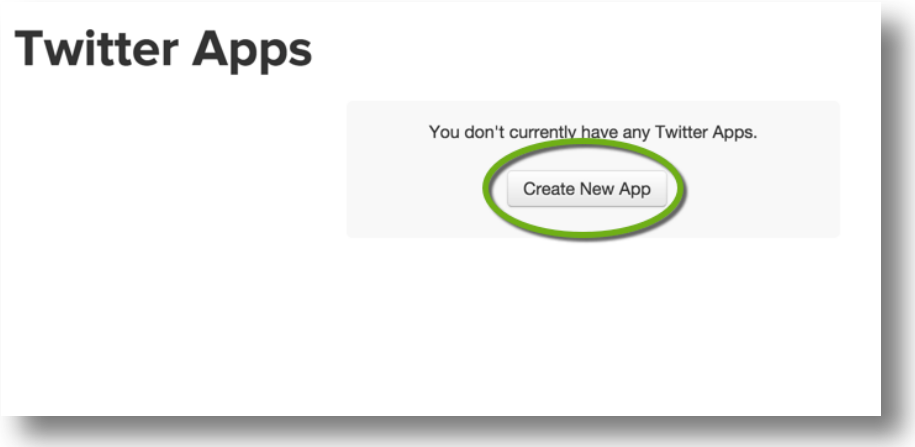
- **Create your Twitter API keys.**
- **Verify your Hortonworks Sandbox web access.**

Create your Twitter API keys

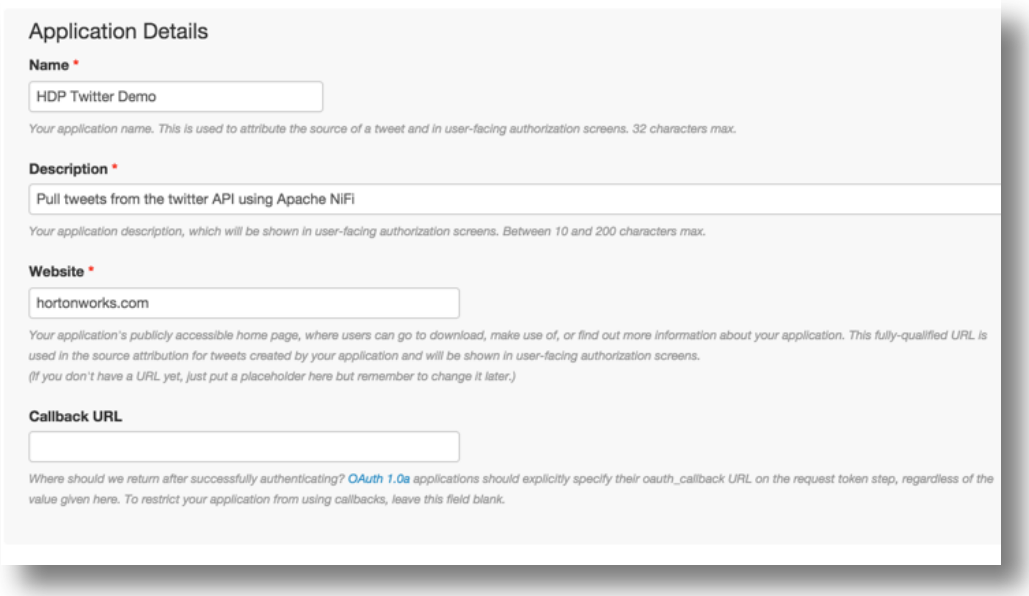
You'll need to register your own Twitter application. It's quite simple and only takes a few short steps

Create A Twitter Application

First head over to the Twitter Apps Website ([https://apps.twitter.com/)](https://apps.twitter.com/) and Sign In using your Twitter account (or make one if you don't have one yet!). Then click Create a New App.



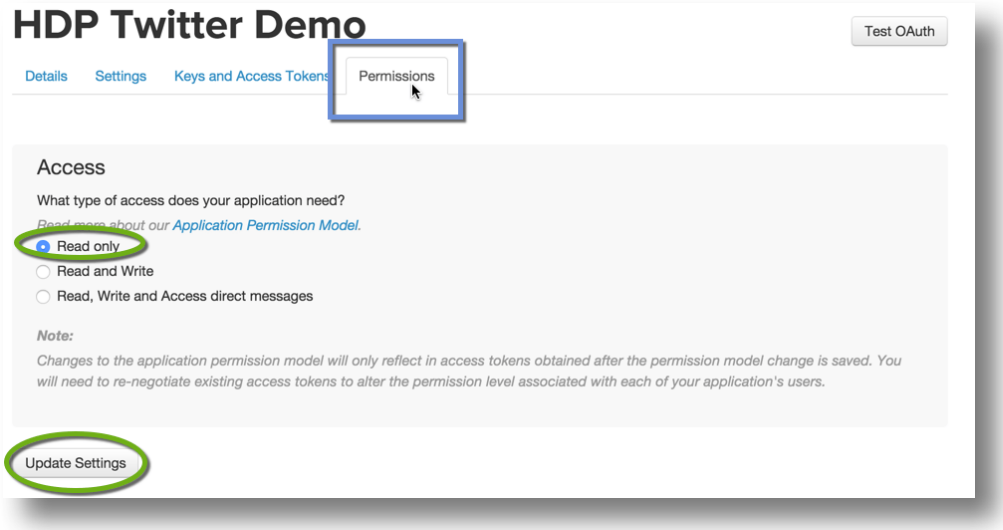
After you've clicked that you'll need to fill in some details about your application. Feel free to put whatever you want.



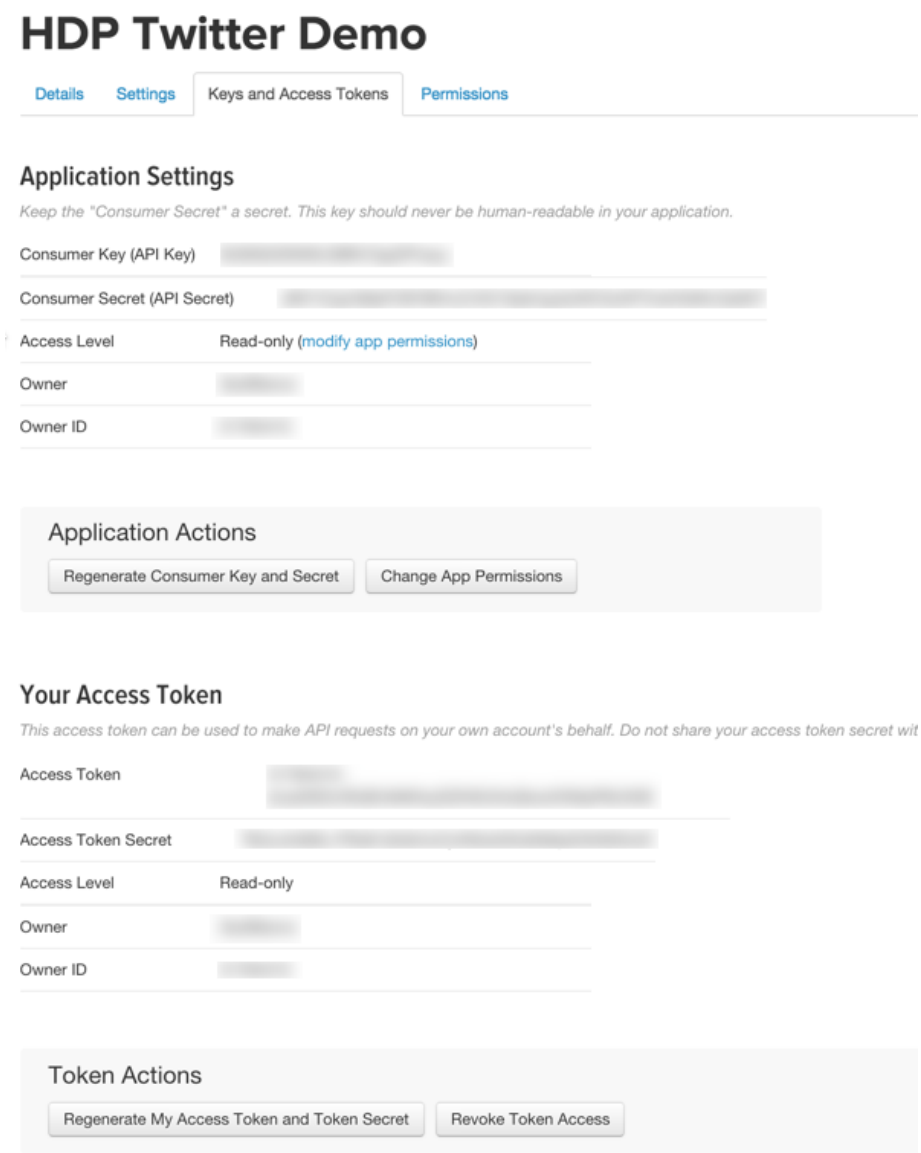
Then click **Create Your Twitter Application** at the bottom of the screen after reading the developer agreement.

Note that you might need to add your mobile phone to your Twitter account before creating your application

Once you've done that you should be greeted by a dashboard for your Twitter application. Head over to the permissions tab and select the **Read Only** Option and **Update** your application.



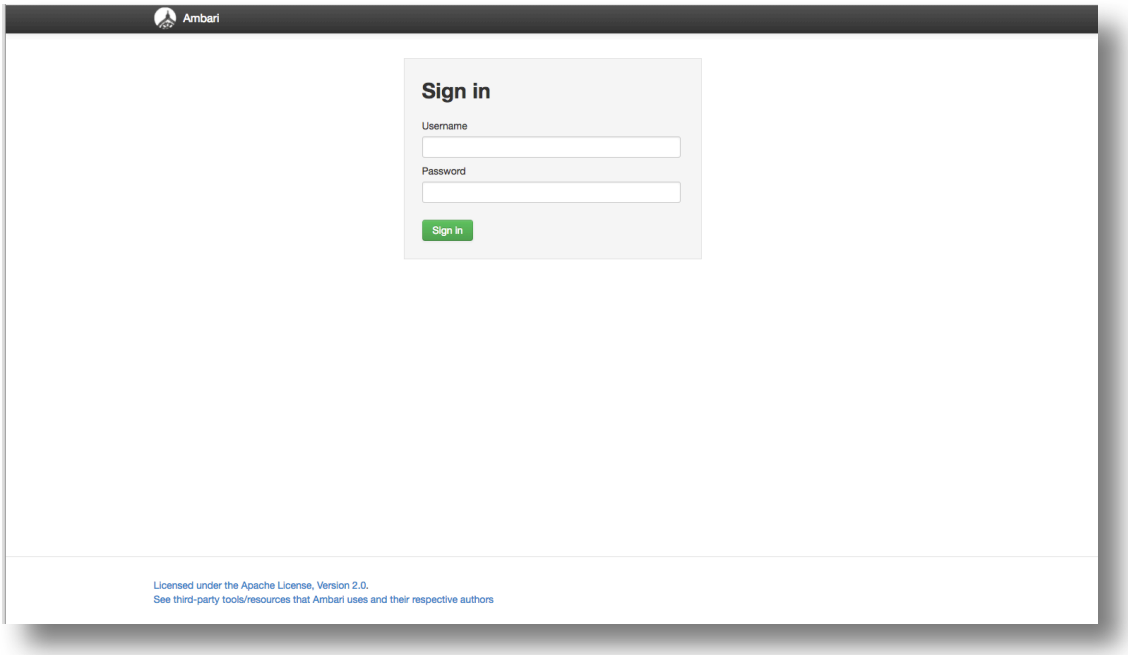
Finally you need to generate your OAuth key. You can do this by clicking **Test OAuth** on the top of the permissions page, or by heading to **Keys and Access Tokens** and then finding the option that allows you to generate your OAuth tokens. Your **keys** and **access tokens** should look similar to the following:



Please make note of your **Consumer Key**, **Consumer Secret**, **Access Token**, and Access Token Secret. You will need these to create the data flow in NiFi.

Verify your Hortonworks Sandbox web access

Once these prerequisites steps are done, please make sure your VM is started on Azure by accessing it with an Internet Browser at http://< sandbox-ip> :8080 .
You should expect the following web page :



If the VM is down, please contact the lab instructor to be provided with a new Hortonworks Sandbox.

Using Apache NiFi for ingesting Twitter Data

HDF (NiFi) has a powerful templating system to allow for easy sharing and consumption of data flows. The following section focuses on importing a prepared Twitter dataflow template, customize parameters according to your environment (Twitter credentials, Twitter keywords, IP addresses) and starts the collection of Tweets into HDFS and publishes them onto Kafka.

The prerequisite step is to update the Hortonworks Schema Registry.

Update the HDF schema registry

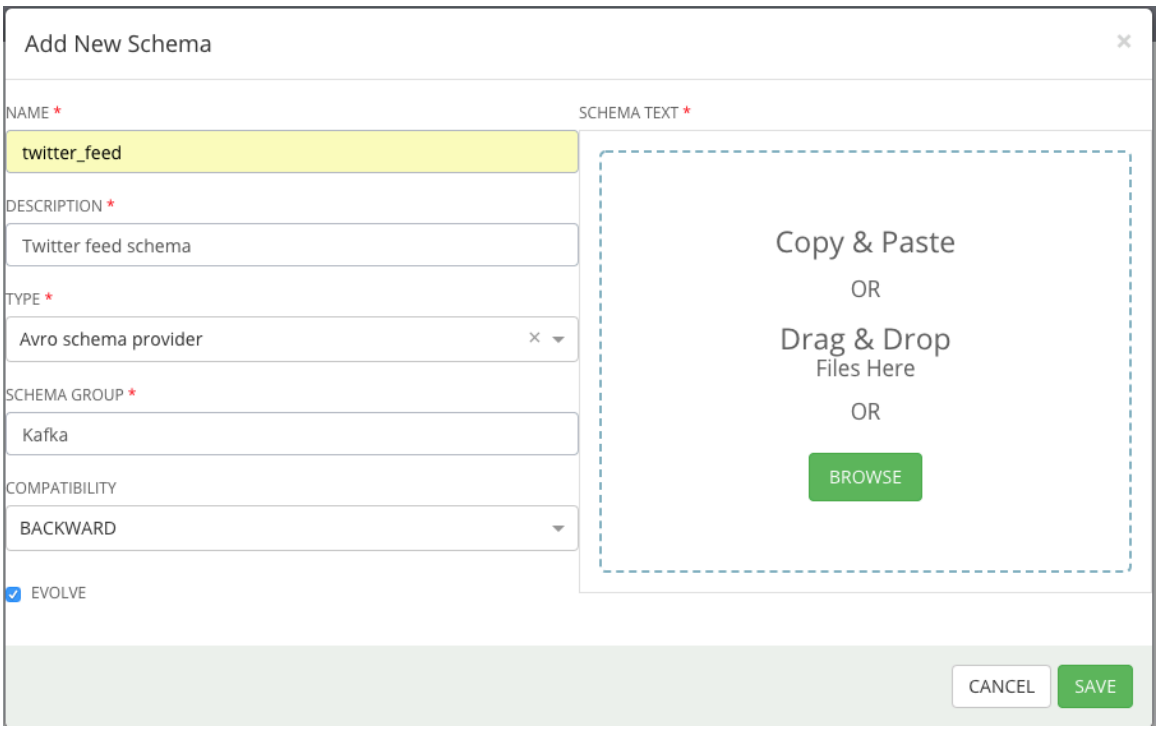
The Hortonworks Schema Registry is a shared repository of schemas that allows applications to flexibly interact with each other in order to save or retrieve schemas for the data they need to access. Having a common Schema Registry provides end to end data governance and introduce operational efficiency by providing reusable schema, defining relationships between schemas and enabling data providers and consumers to evolve at different speed.

For this lab, we will enter the schema of the tweeter feed once, and it will be used by various HDF components such as Apache NiFi, Apache Storm, Apache Kafka, and Streaming Analytics Manager by simply referring to its schema name without having to declare the schema for each different component.

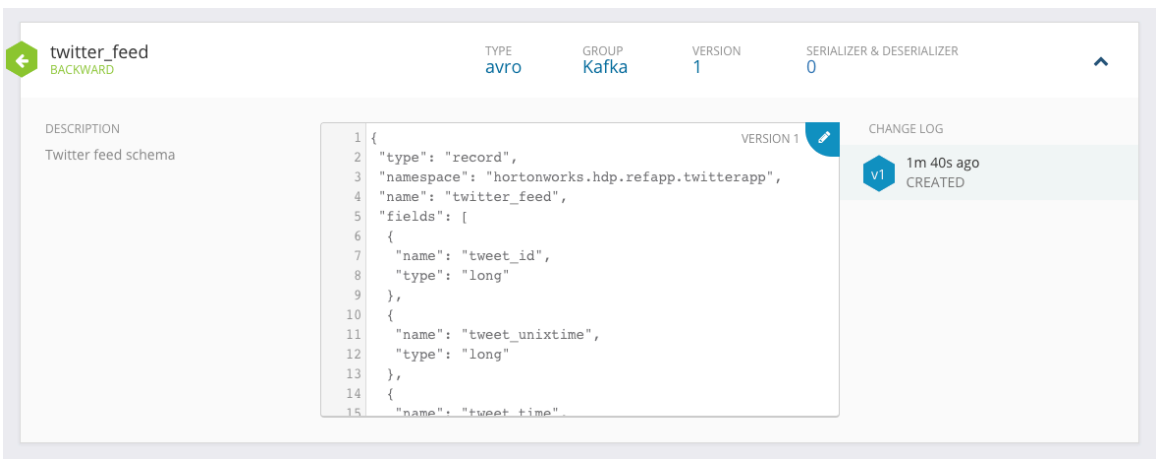
Navigate to the Schema Registry UI at *http://<public ip>:7788*

Click on the green hexagon with the '+' sign:

- For the schema name, please enter: **`twitter_feed`**
  It is important that no typos are done in the schema name, as this name is being used in the rest of the applications (NiFi, Kafka, Storm, Streaming Analytics Manager) to point to this schema.
- For the schema description, please enter: **Twitter feed schema**
- Leave the rest of the properties to their default value, as shown in the screen capture below.
- For the schema text, the schema is in avro format (json). For the current lab, you can copy and paste it from the actual schema available on github at the following link:
  https://github.com/oascofare/Datavore2018/blob/master/twitter_feed.json



Once done, you can click on SAVE. You should know see the schema appear in the registry if you click on the expand icon:



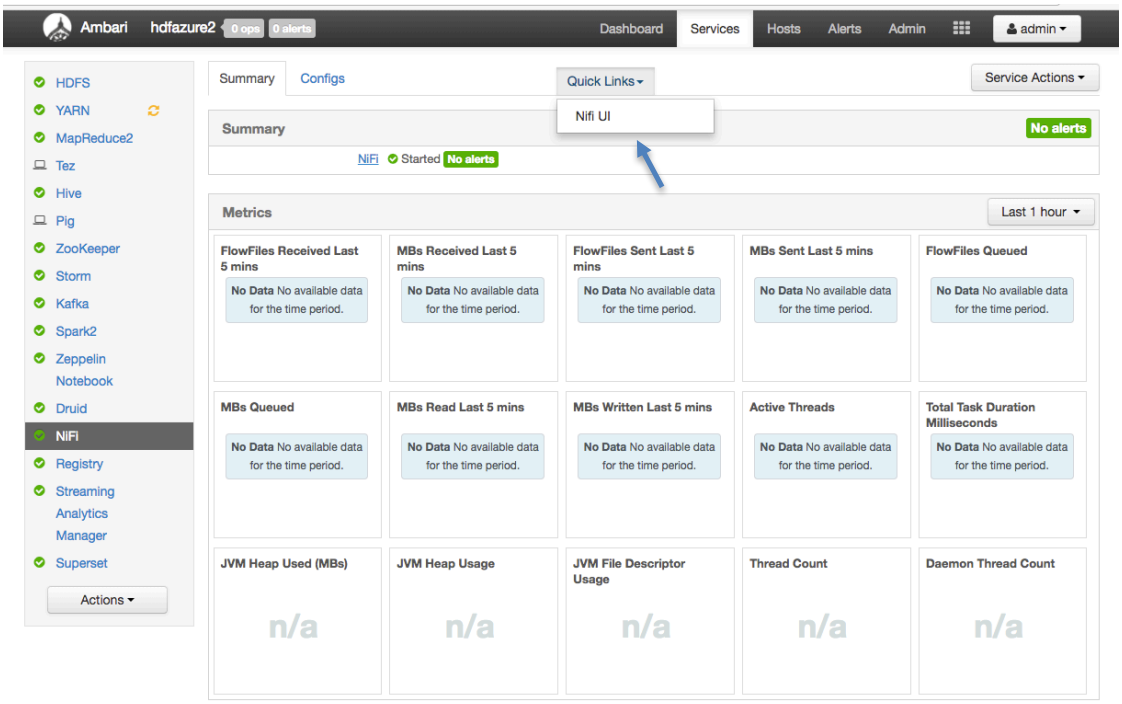
Download the Twitter dataflow template

The Twitter dataflow template should be downloaded to the workstation accessing NiFi at the following link.

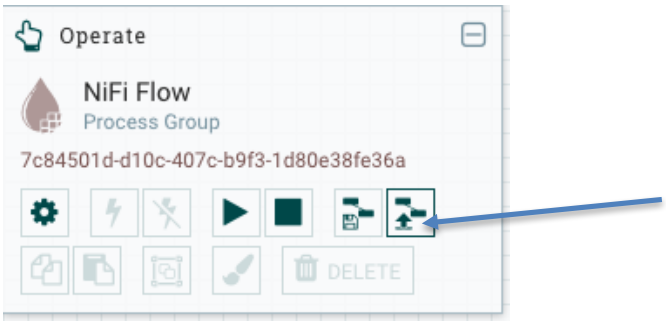
https://github.com/oascofare/Datavore2018/raw/master/DatavoreTwitterApplication.xml

Import the Template

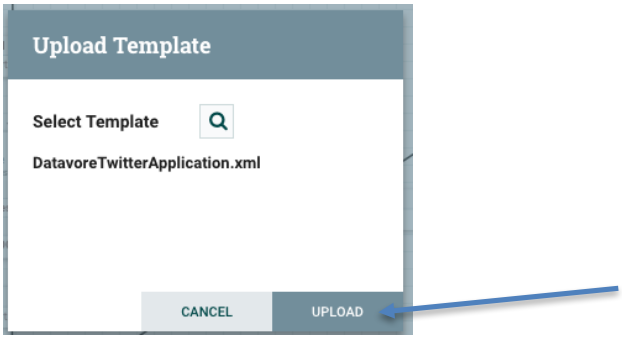
Click on Nifi Service Quick link or navigate to the NiFi web interface at *http://<public ip>: 9999/nifi/* via a web browser.



Import the template by selecting the Import Template Icon found in the NiFi Flow Operate tile on the left-hand side of the canvas.

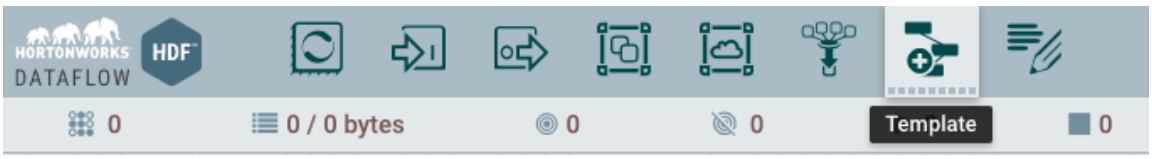


Click the Browse icon ( ) and navigate to where the DatavoreTwitterApplication.xml file was downloaded. Click on Upload.

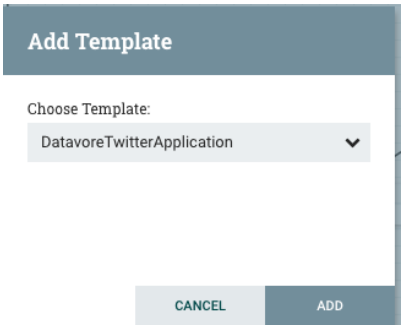


Instantiate the Template

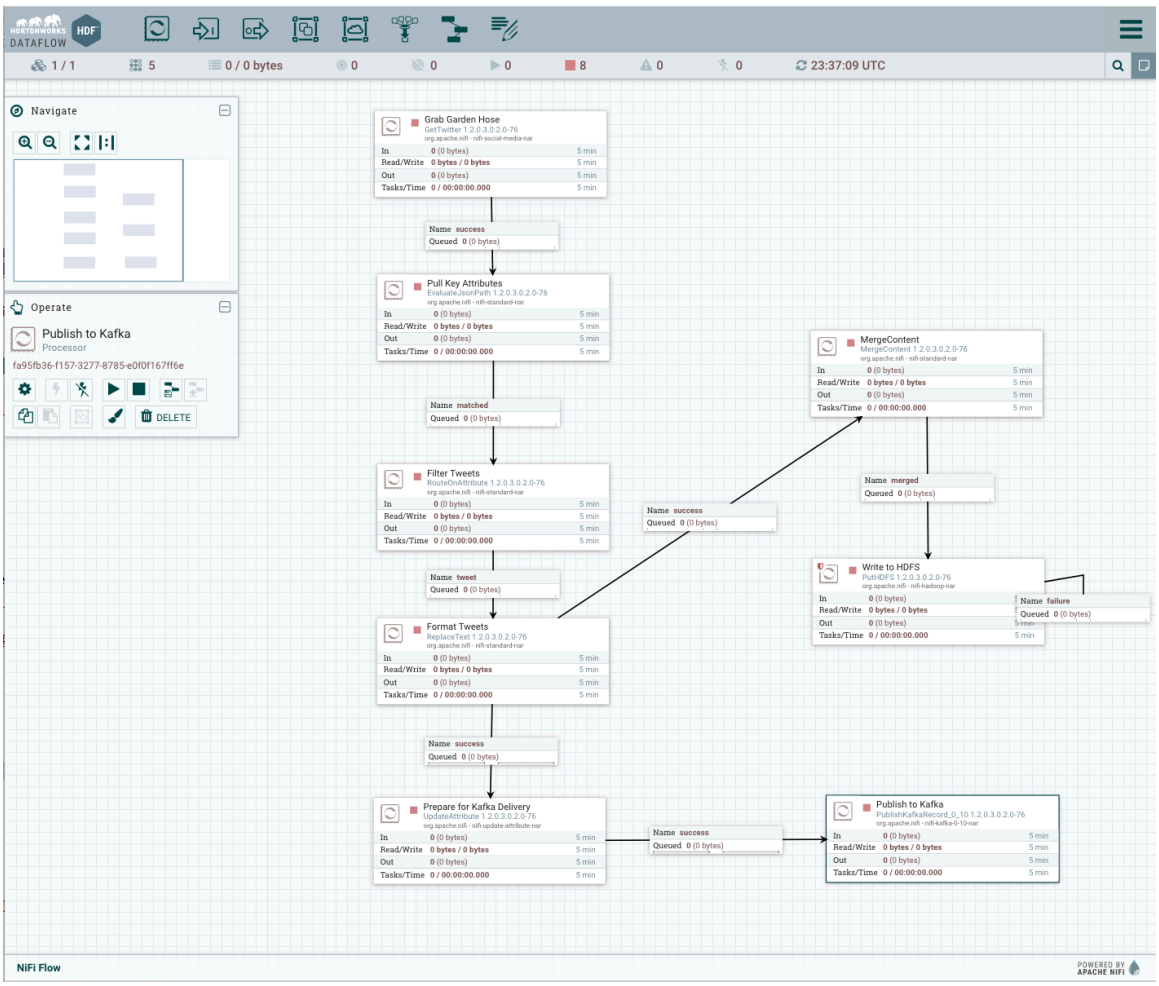
Drag and drop the template artifact onto the canvas.



Select *DatavoreTwitterApplication* and click *Add*

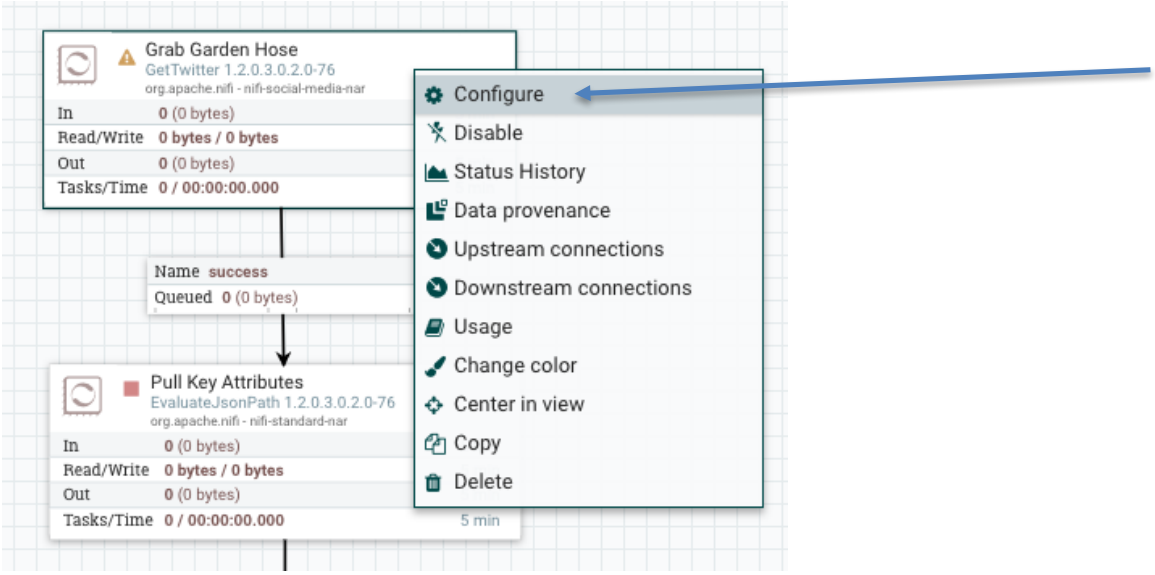


This will import the Data Flow onto your NiFi canvas.



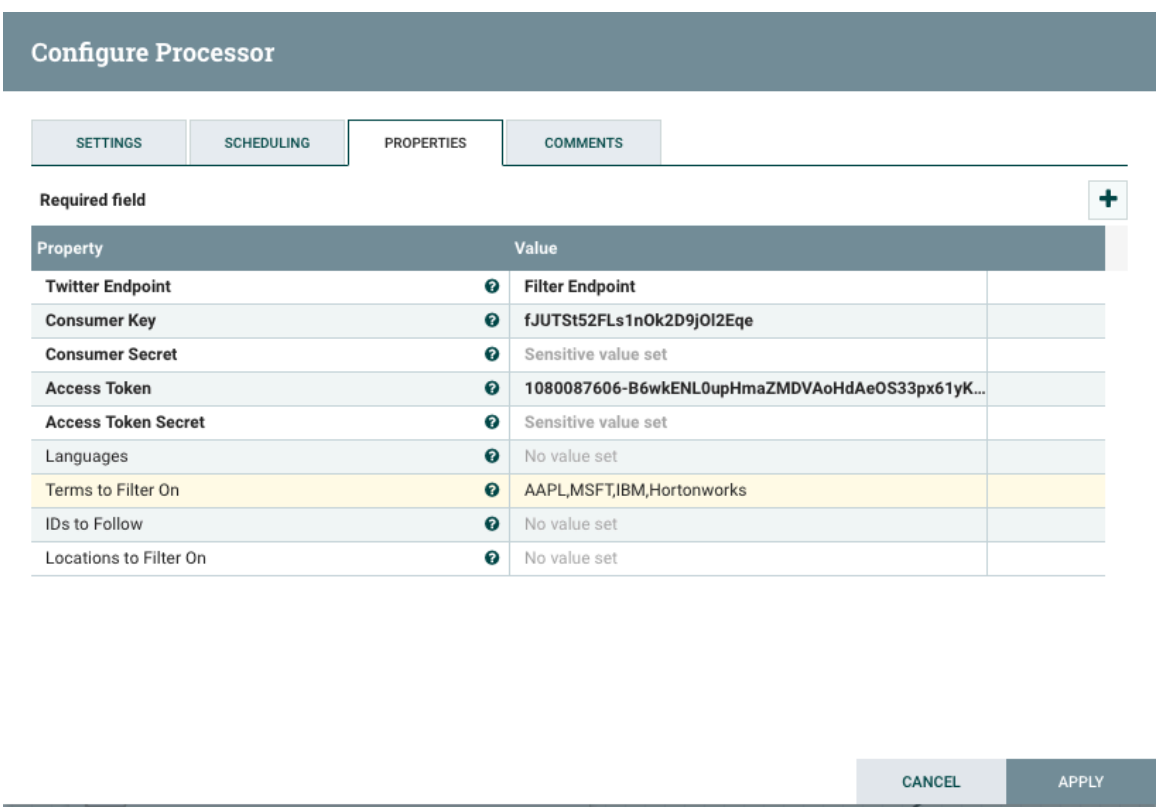
Configure the Twitter Grab Garden Hose Processor

Next, configure the Twitter Grab Garden Hose Processer with the Twitter API information. Right click the processor and select Configure. Then, select the PROPERTIES tab.



Fill in the following mandatory properties (in bold) matching to your Twitter application:

- Consumer Key
- Consumer Secret
- Access Token
- Access Token Secret

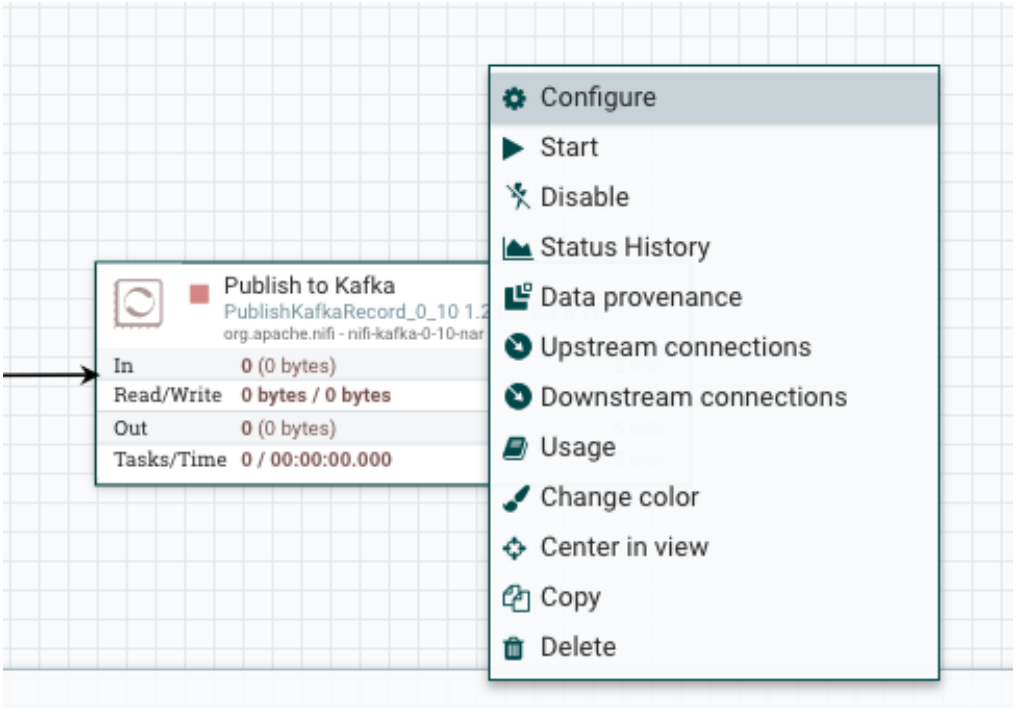


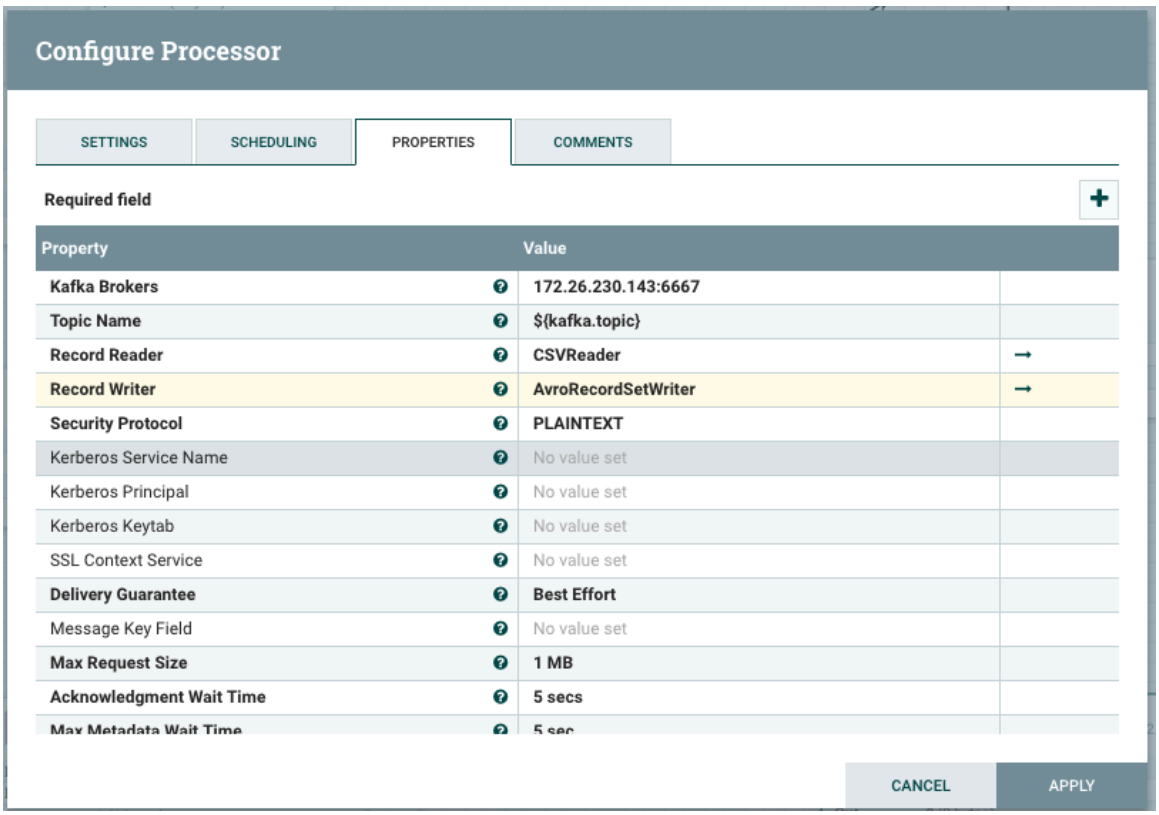
Configure PublishKafkaRecord_0_10 processor

We need to customize IP addresses in this processor for the Kafka broker, and the Schema Registry URI. Once done, Controller Services for Record Reader (CSVReader), Record Writer (AvroRecordSetWriter) and the Schema Registry (HortonworksSchemaRegistry) need to be enabled.

Controller Services are shared services that can be used by Processors, Reporting Tasks, and other Controller Services.

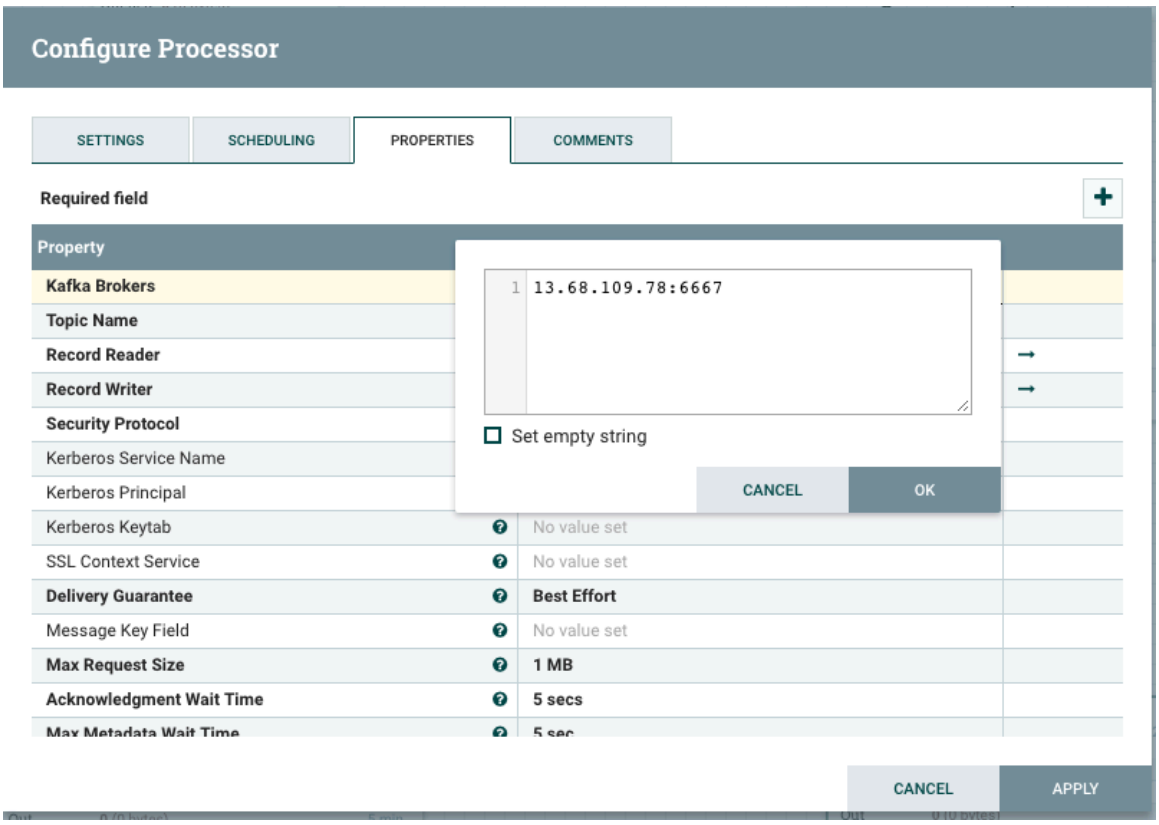
Right click on the PublishKafkaRecord_0_10 processor and select Configure.





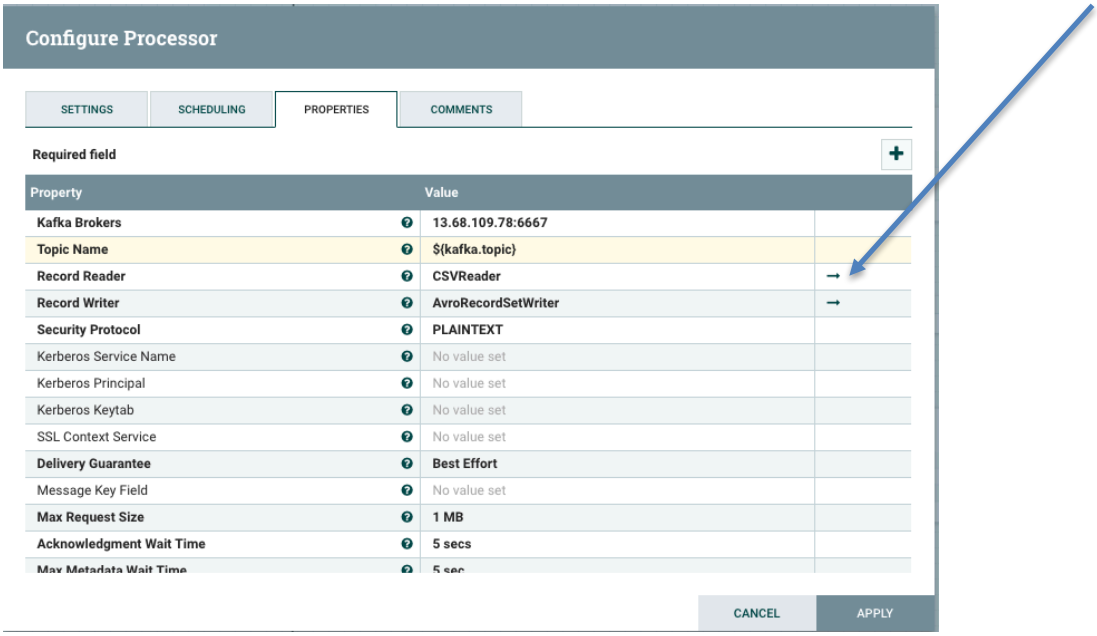
Configure the Kafka Brokers URI

Update the Kafka Brokers IP address with the public IP address of your VM in the URI:

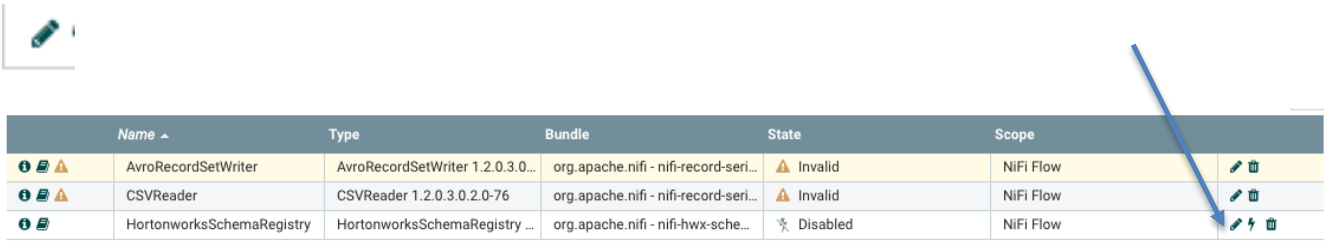


Configure the Schema Registry and Enable all Controller Services

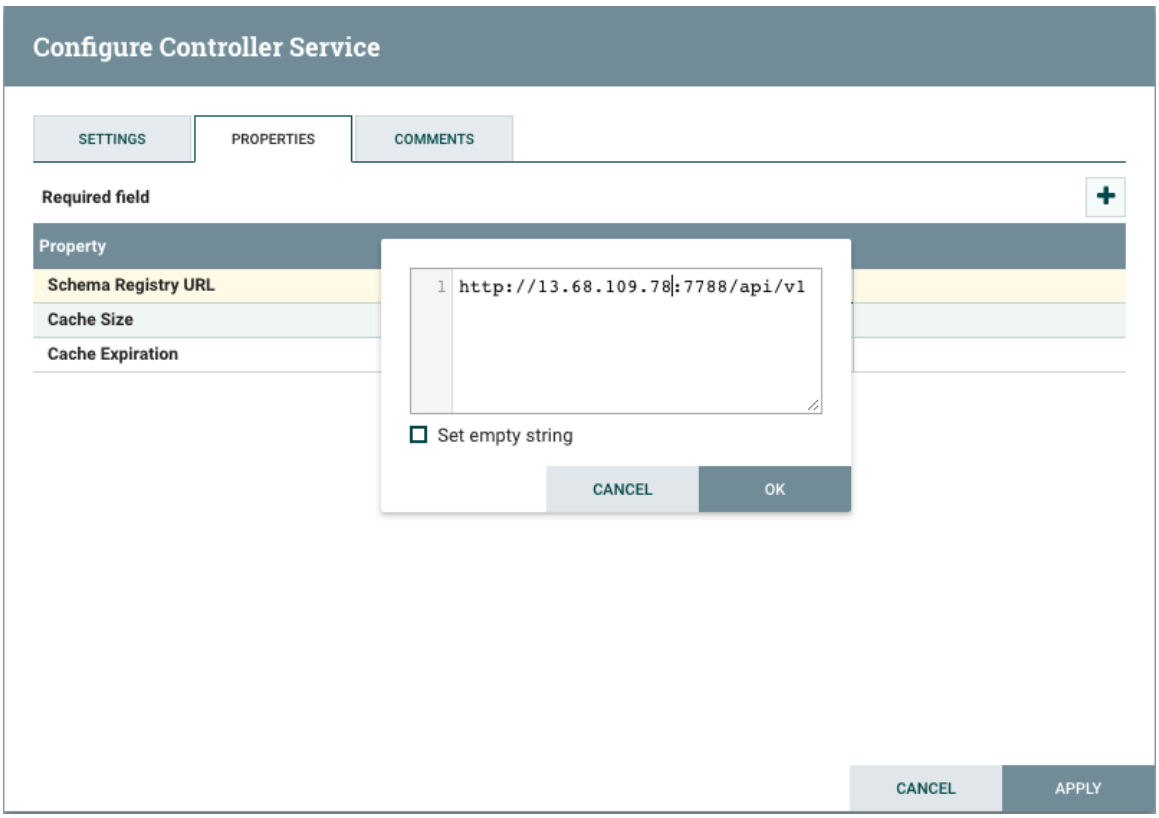
Click on the arrow next to Record Reader:



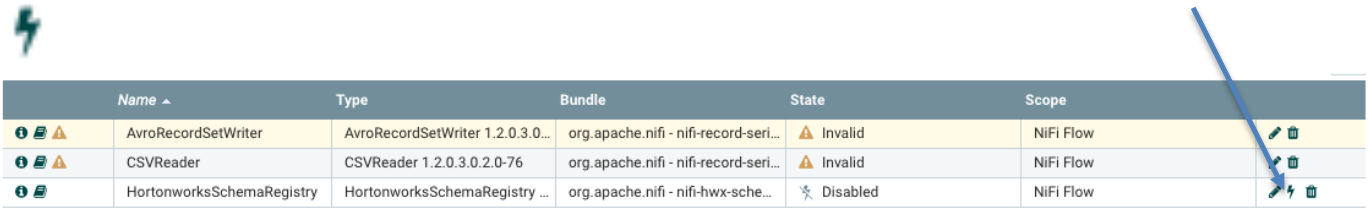
On the new screen, click on Edit for the HortonworksSchemaRegistry - pencil icon:



In the PROPERTIES tab, click on Schema Registry URL and customize the URL with the public IP address of your VM:

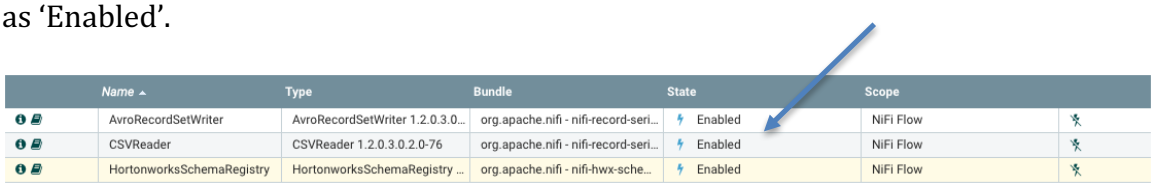


Now, you can click on the flash icon next to HortonworksSchemaRegistry to enable the controller service:



In the 'Enable Controller Services' window, for the scope, select Service and referencing components, then click 'ENABLE':

Close the window. Now, you should see that all the Controller Services State appear as 'Enabled'.



Name	Type	Bundle	State	Scope
AvroRecordSetWriter	AvroRecordSetWriter 1.2.0.3.0...	org.apache.nifi - nifi-record-seri...	Enabled	NiFi Flow
CSVReader	CSVReader 1.2.0.3.0.2.0-76	org.apache.nifi - nifi-record-seri...	Enabled	NiFi Flow
HortonworksSchemaRegistry	HortonworksSchemaRegistry ...	org.apache.nifi - nifi-hwx-sche...	Enabled	NiFi Flow

You can close the window by clicking on the 'X' on the top right corner.

Review the remaining Processors

Review the other processors and modify properties as needed:

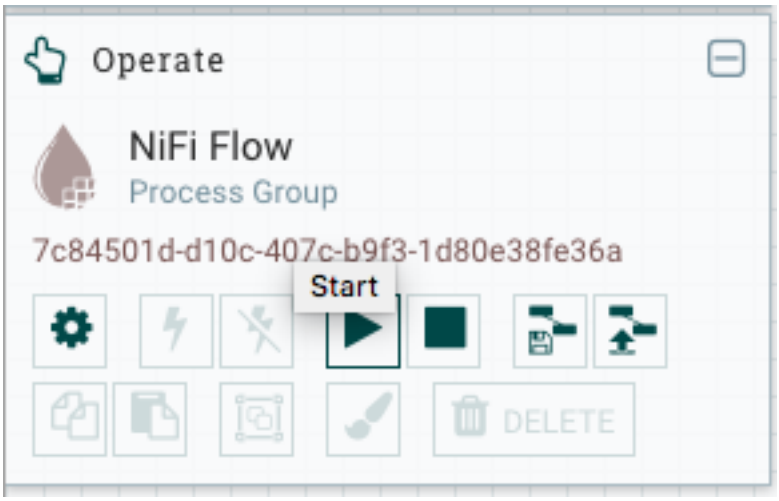
- **EvaluateJsonPath**: Pulls out attributes of tweets
- **RouteonAttribute**: Ensures only tweets with non-empty messages are processed
- **ReplaceText**: Formats each tweet as pipe (|) delimited line entry e.g. tweet_id|tweet_unixtime|tweet_stringtime}|tweet_timezone|tweet_screenname|tweet_username|tweet_followers|tweet_hashtags

- **MergeContent**: Merges tweets into a single file (either 20 tweets or 120s, whichever comes first) to avoid having a large number of small files in HDFS. These values can be configured.
- **PutHDFS**: writes tweets to HDFS under /tmp/tweets_staging
- **UpdateAttribute**: Add additional attributes to the flow files, which are the schema name pointing to the schema name in the schema registry, as well as the kafka topic name where the tweets are being written.

Start the Flow

If the processors are correctly configured, all processors should have a red stop symbol in the upper left (except green for PublishKafkaRecord, as this particular processor was started as a referencing component of the controller services). If a yellow caution symbol is present, this indicates required configuration is missing.

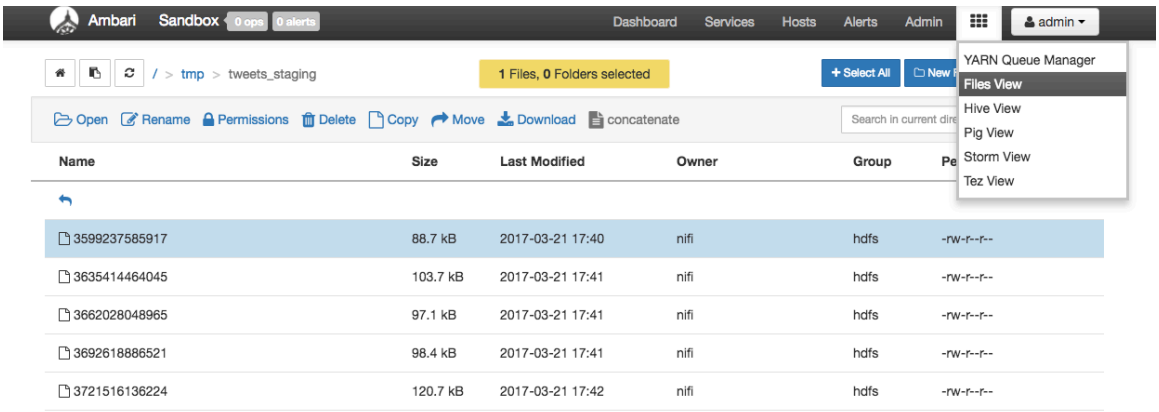
Start the flow by selecting an empty area of the canvas and clicking the start icon.



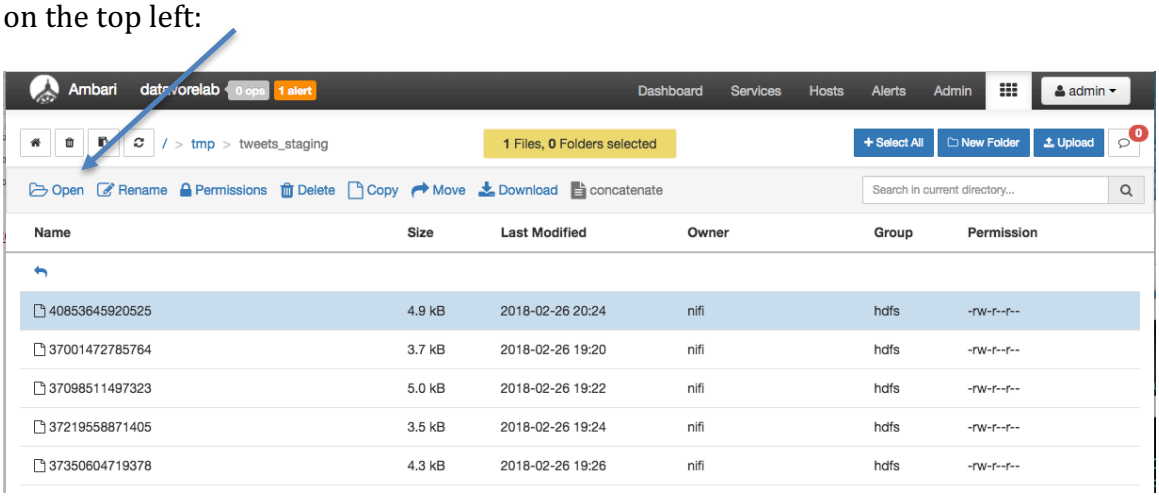
Check the data on HDFS

Now that the Twitter dataflow is running, the matching tweets will be persisted to HDFS and Kafka. Validate that the tweets are successfully written to HDFS for now.

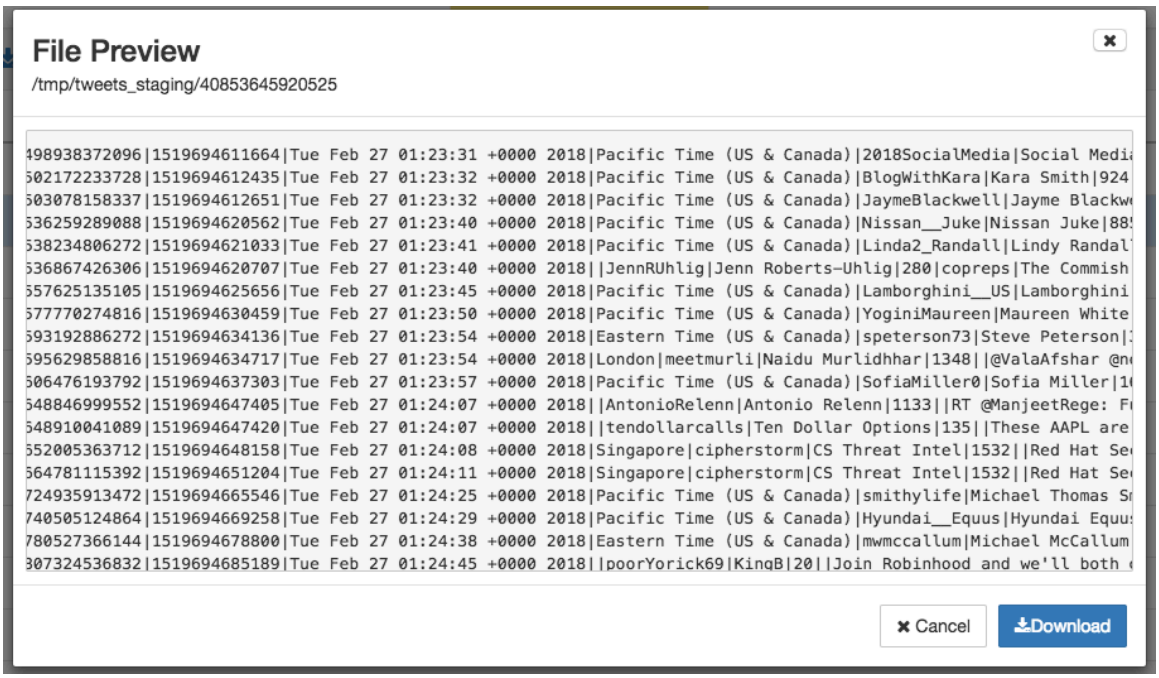
Tweets appear under /tmp/tweets_staging dir in HDFS. You can see this via Files view in Ambari:



You can open the file to visualize its content. Select one of the file, and click on Open on the top left:



You can now preview the contents of the file. If needed, you can download the file on your local workstation for further review of its contents:


File Preview
/tmp/tweets_staging/40853645920525
498938372096|1519694611664|Tue Feb 27 01:23:31 +0000 2018|Pacific Time (US & Canada)|2018SocialMedia|Social Media
502172233728|1519694612435|Tue Feb 27 01:23:32 +0000 2018|Pacific Time (US & Canada)|BlogWithKara|Kara Smith|924
503078158337|1519694612651|Tue Feb 27 01:23:32 +0000 2018|Pacific Time (US & Canada)|JaymeBlackwell|Jayme Blackw
536259289088|1519694620562|Tue Feb 27 01:23:40 +0000 2018|Pacific Time (US & Canada)|Nissan__Juke|Nissan Juke|88
538234806272|1519694621033|Tue Feb 27 01:23:41 +0000 2018|Pacific Time (US & Canada)|Linda2_Randall|Lindy Randal
536867426306|1519694620707|Tue Feb 27 01:23:40 +0000 2018||JennRUhlig|Jenn Roberts-Uhlig|280|copreps|The Commish
557625135105|1519694625656|Tue Feb 27 01:23:45 +0000 2018|Pacific Time (US & Canada)|Lamborghini__US|Lamborghini
577770274816|1519694630459|Tue Feb 27 01:23:50 +0000 2018|Pacific Time (US & Canada)|YoginiMaureen|Maureen White
593192886272|1519694634136|Tue Feb 27 01:23:54 +0000 2018|Eastern Time (US & Canada)|speterson73|Steve Peterson|
595629858816|1519694634717|Tue Feb 27 01:23:54 +0000 2018|London|meetmurli|Naidu Murlidhhar|1348||@ValaAfshar @n
506476193792|1519694637303|Tue Feb 27 01:23:57 +0000 2018|Pacific Time (US & Canada)|SofiaMiller0|Sofia Miller|1
548846999552|1519694647405|Tue Feb 27 01:24:07 +0000 2018||AntonioRelenn|Antonio Relenn|1133||RT @ManjeetRege: F
548910041089|1519694647420|Tue Feb 27 01:24:07 +0000 2018||tendollarcalls|Ten Dollar Options|135||These AAPL are
552005363712|1519694648158|Tue Feb 27 01:24:08 +0000 2018|Singapore|cipherstorm|CS Threat Intel|1532||Red Hat Se
564781115392|1519694651204|Tue Feb 27 01:24:11 +0000 2018|Singapore|cipherstorm|CS Threat Intel|1532||Red Hat Se
724935913472|1519694665546|Tue Feb 27 01:24:25 +0000 2018|Pacific Time (US & Canada)|smithylife|Michael Thomas S
740505124864|1519694669258|Tue Feb 27 01:24:29 +0000 2018|Pacific Time (US & Canada)|Hyundai__Equus|Hyundai Equu
780527366144|1519694678800|Tue Feb 27 01:24:38 +0000 2018|Eastern Time (US & Canada)|mwmccallum|Michael McCallum
807324536832|1519694685189|Tue Feb 27 01:24:45 +0000 2018||poorYorick69|KingB|20||Join Robinhood and we'll both
Cancel
Download

Real-Time Data Analysis with Streaming Analytics Manager

Hortonworks Streaming Analytics Manager is an open source tool used to design, develop, deploy and manage streaming analytics applications using a drag drop visualize paradigm. Now users can build streaming analytics applications that do event correlation, context enrichment, complex pattern matching, analytical aggregations and create alerts/notifications when insights are discovered.

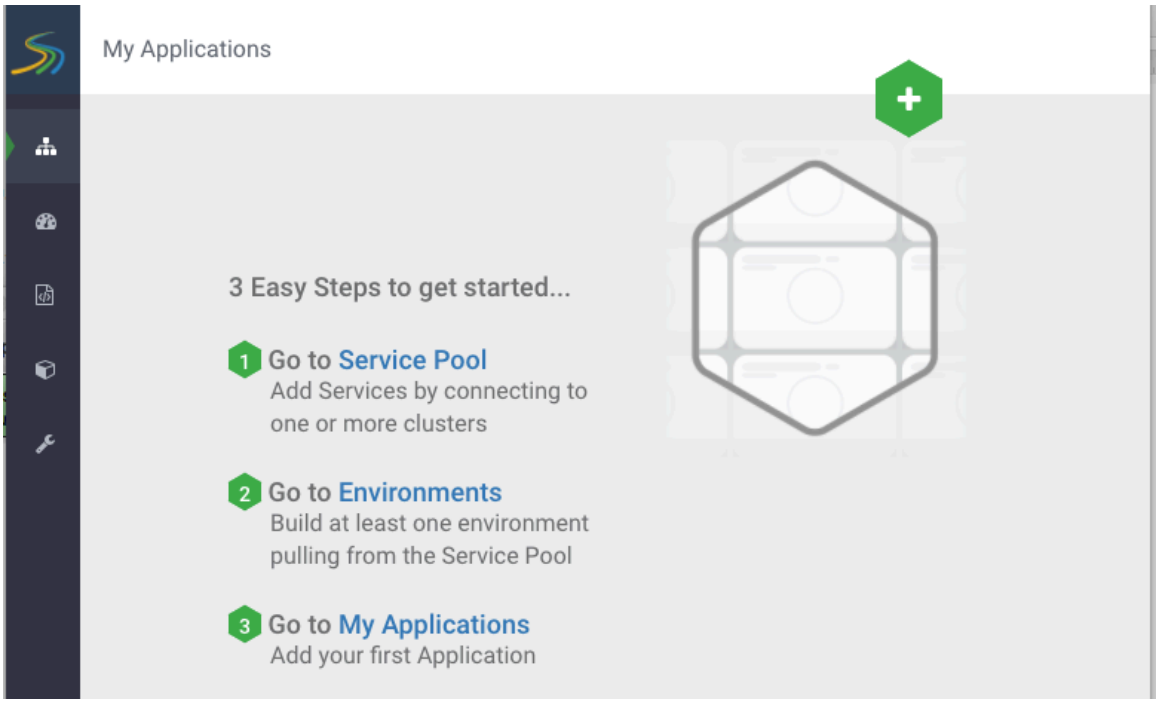
Prepare the Environment

Before using Streaming Analytics Manager, a couple of administrative steps needs to be performed from an Operations perspective to enable a development environment.

Doing so within Streaming Analytics Manager greatly accelerate application development, as developers do not need to worry about Hadoop cluster configuration files & topologies. This integration is taken care of by the tool itself by interacting with the target clusters part of the service pool and retrieving all required configuration through the Ambari REST API.

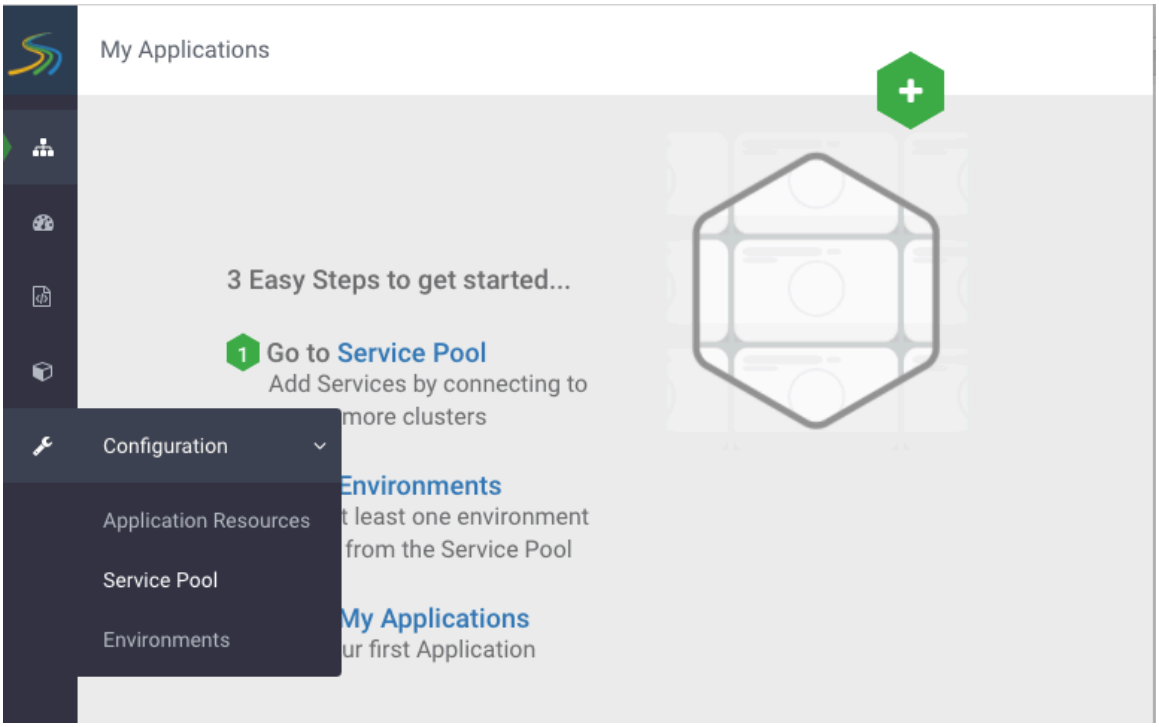
To access SAM, go to http://<vm-public-ip>:7777

The following screen will appear:



Define the Service Pool

As described in the welcome screen, we first need to define a Service Pool. Click on the tool icon on the left-hand side tool bar and select Service Pool in the menu:

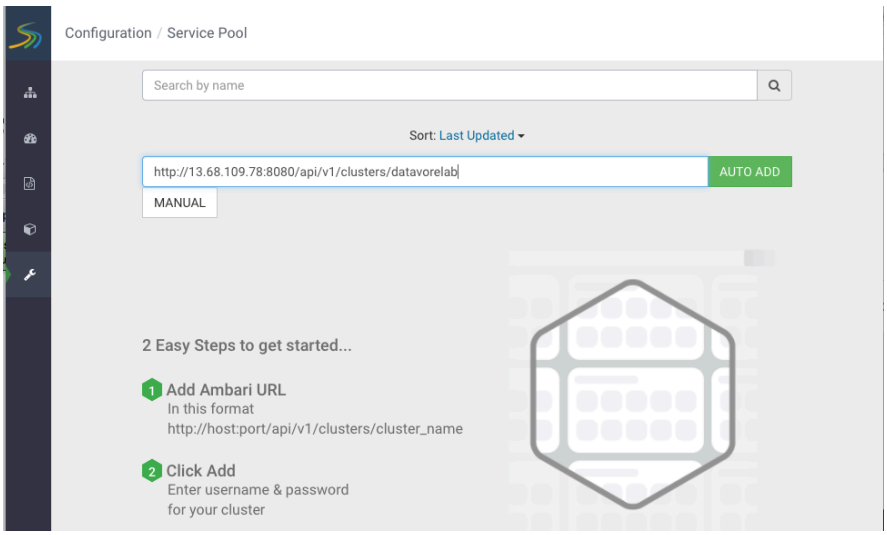


Update the Ambari URL corresponding to your cluster by replacing the place holders in the URL with the following values:

- Ambari_host: Public IP of your VM.
- Port: 8080
- CLUSTER_NAME: datavorelab << particiant VM number >>

For example: `http://13.68.109.78:8080/api/v1/clusters/`<< particiant VM number >>

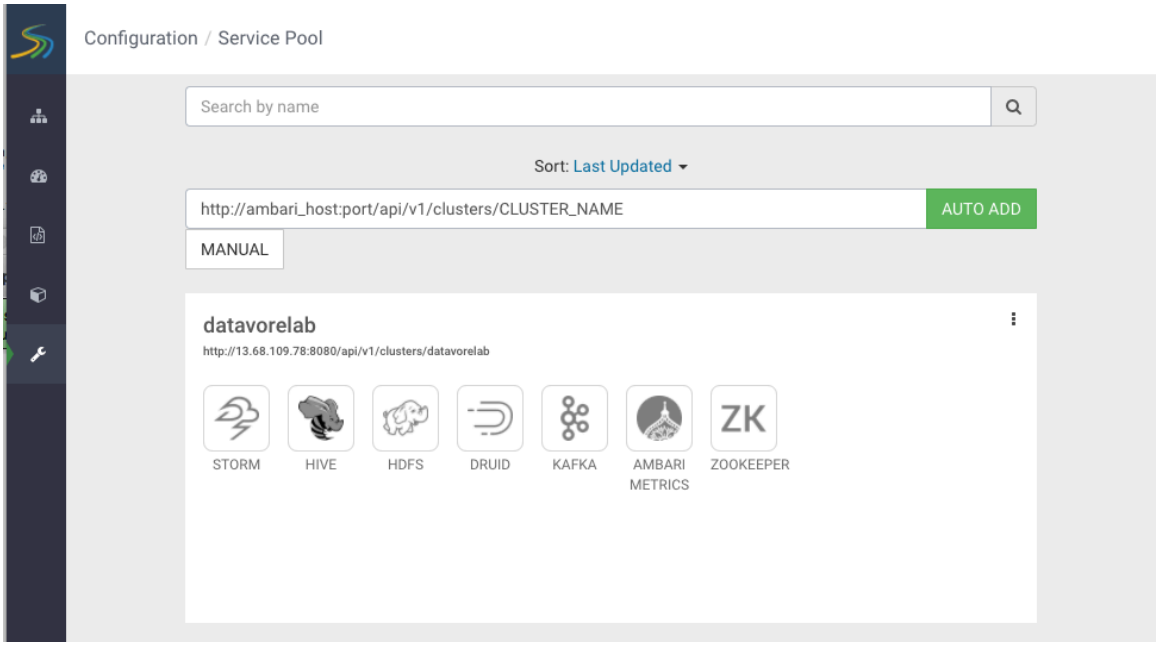
Then, click on the AUTO-ADD button.



You will be prompted for the USERNAME/PASSWORD for your Ambari console. Enter the following values:

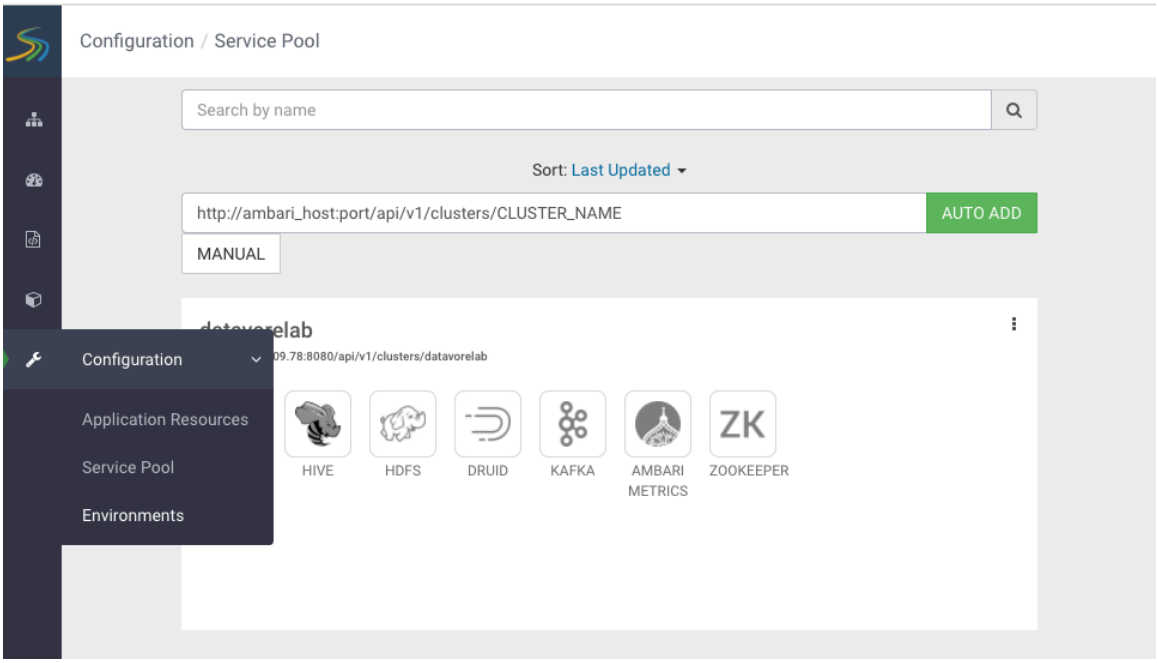
- USERNAME: admin
- PASSWORD: admin

It may take a few minutes to add the cluster as SAM interacts with the target cluster and retrieves all required configurations through the Ambari REST API. Once the cluster has been added successfully, you will see it appear as a service pool:

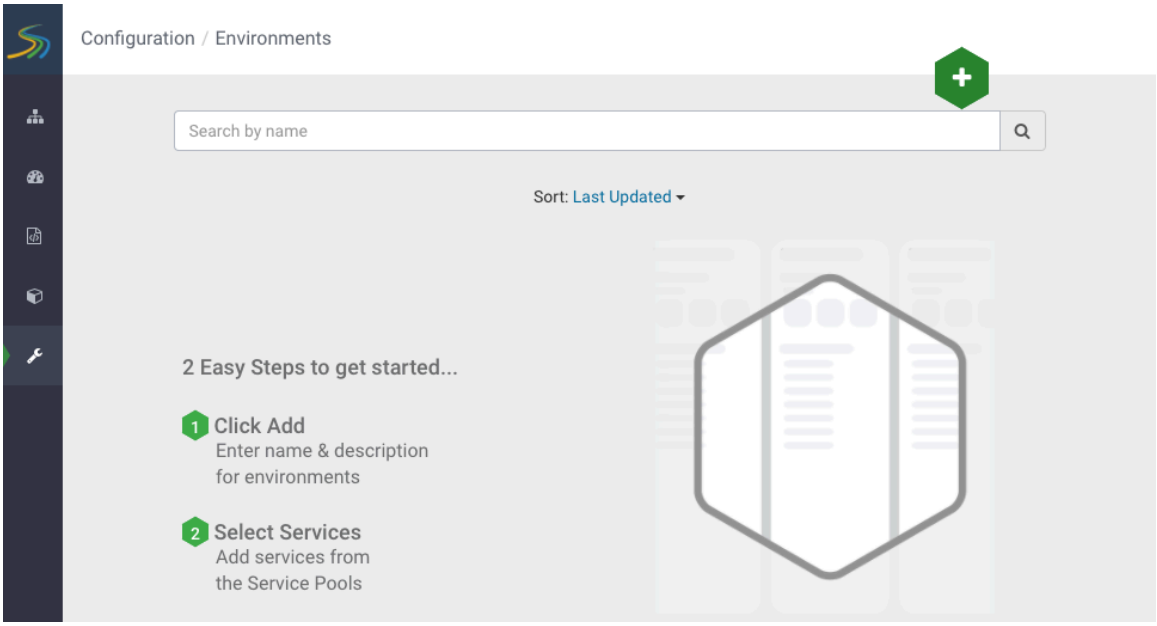


Define a Development Environment

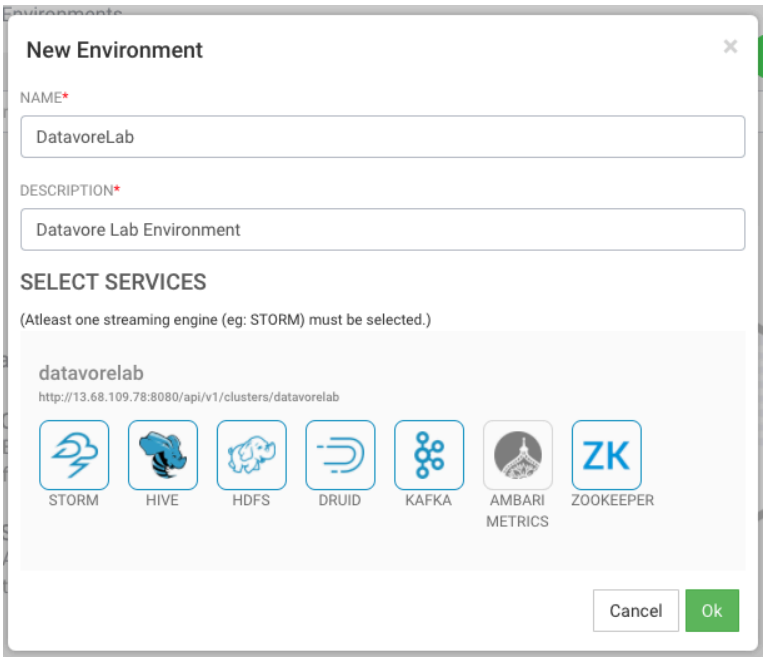
A new development environment needs to be defined now based on the available Service Pool. Click on the tool icon on the left-hand side toolbar and select Environments:



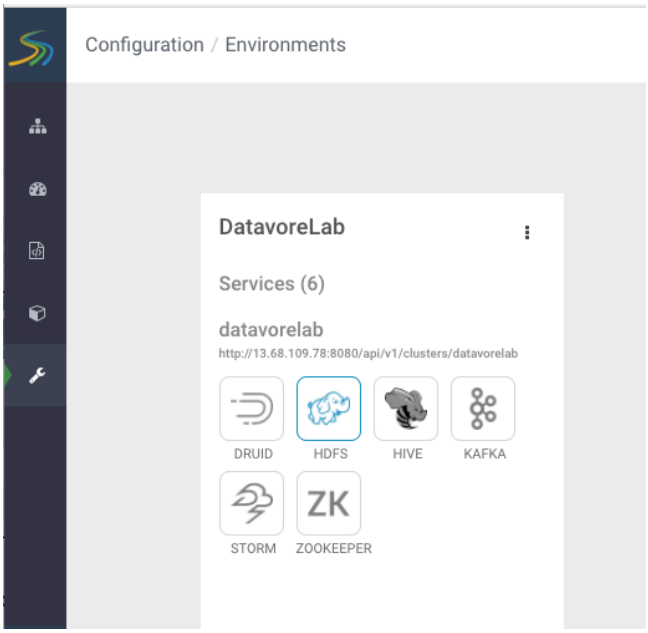
In the new screen, please click Add (Green hexagon with the '+' sign on the top right):



Give the name **DatavoreLab** << particiant VM number >> to the new environment. For Description, you can for example use "**Datavore Lab Environment**". For services, you can select all services (except Ambari Metrics). Selected services will be highlighted in blue:



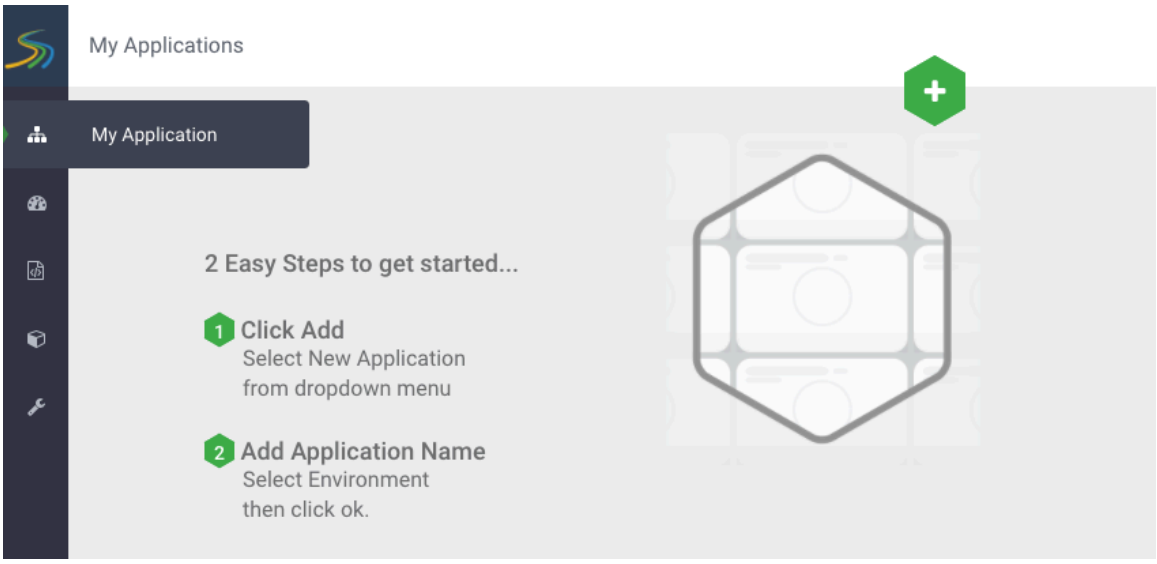
Then, click OK. The new environment will now appear as a new tile as follows:



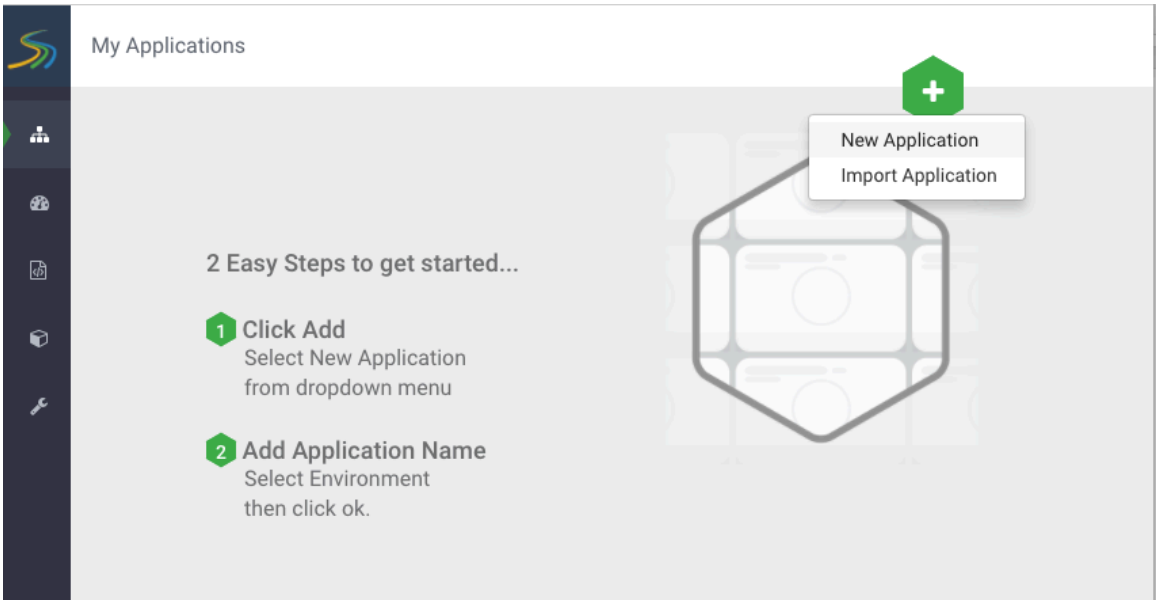
At this point, we are ready to develop a new SAM (Streaming Analytics Manager) Application.

Develop a SAM Application

We can now develop a new SAM (Streaming Application Manager) application. Click on the application icon on the left-hand side toolbar, and select 'My Application':

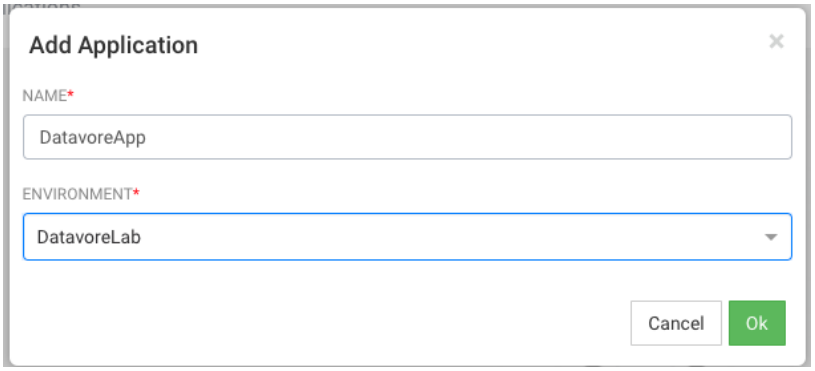


Click on Add (Green hexagon on the top righ with the '+' sign), and select 'New Application':



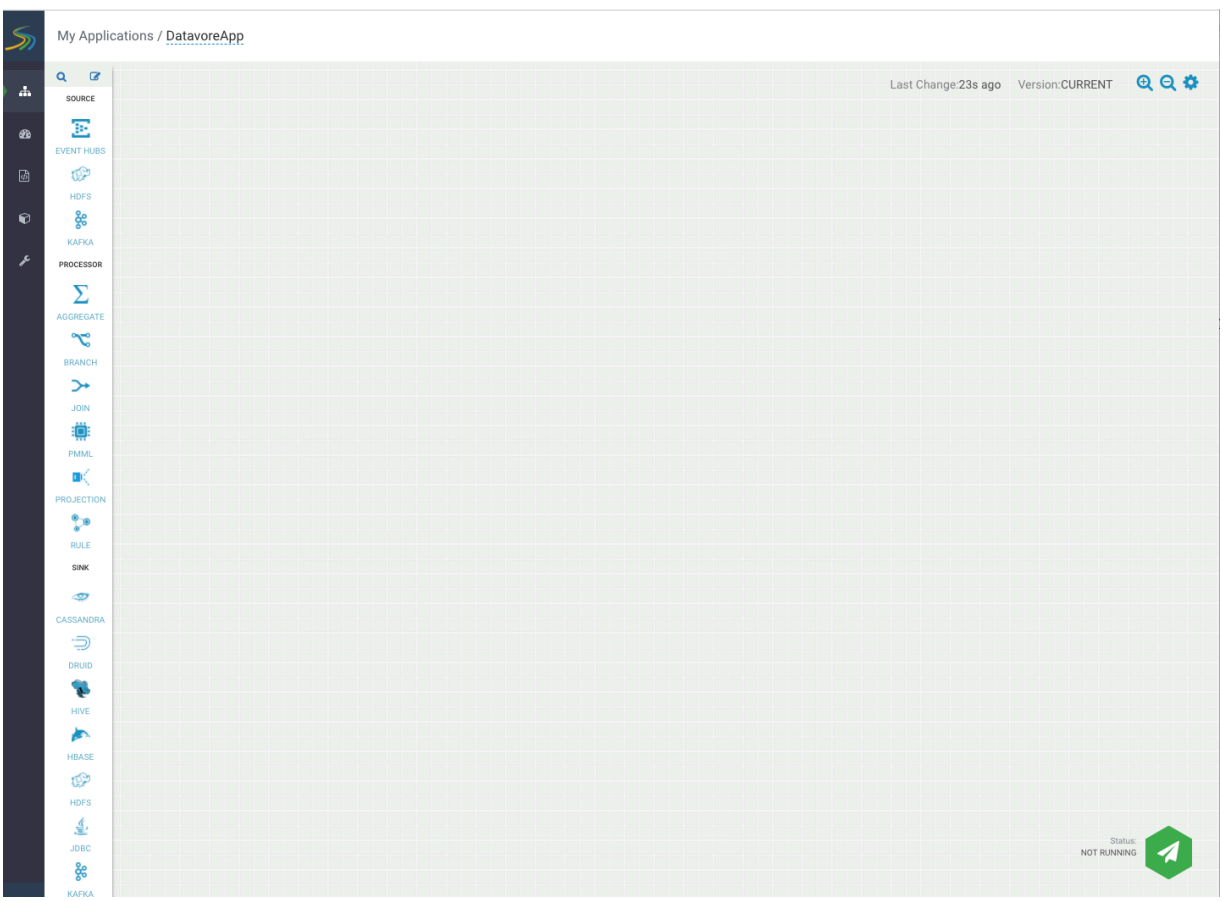
- Enter the following Application NAME: 'DatavoreApp'. It's important that there are no spaces in your application name, as this could potentially cause some issues with Storm.

- For the environment, please select 'DatavoreLab << particiant VM number >>' that we just created previously.



Click OK.

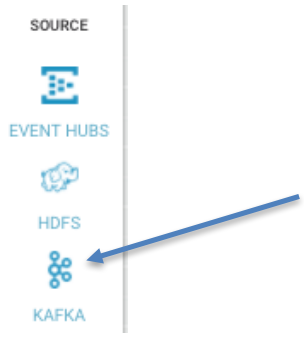
You will now have an empty canvas. We are ready to develop our first SAM application.



Read data from Kafka

We first want to read data from a source. As a source, we are going to use the Kafka topic on which we wrote with Apache NiFi on the previous section.

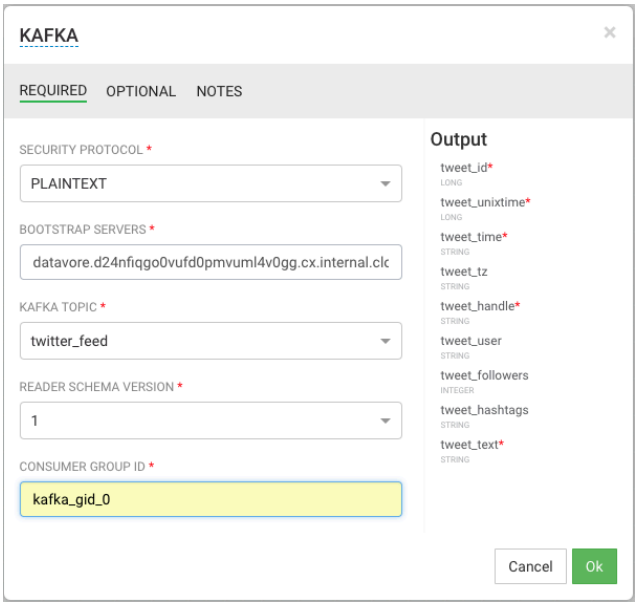
From the various operators available on the processor menu, please select Kafka from **SOURCE**, then drag and drop it onto the canvas:



Double-click on the Kafka operator on the canvas, and enter the following values:

- CLUSTER NAME: **datavorelab**
- SECURITY PROTOCOL: **PLAINTEXT**
- BOOTSTRAP SERVERS: Leave default value
- KAFKA_TOPIC: **twitter_feed**
- READER SCHEMA VERSION: **1**
- CONSUMER GROUP ID: **kafka_gid_0**

Click OK.

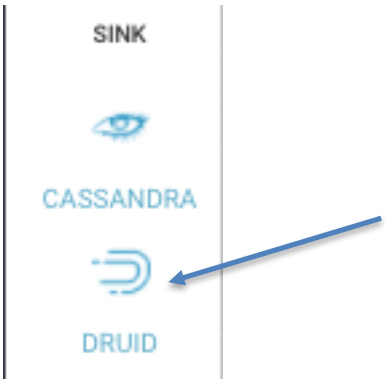


You will notice that the schema will appear on the output. The schema is retrieved from the schema registry.

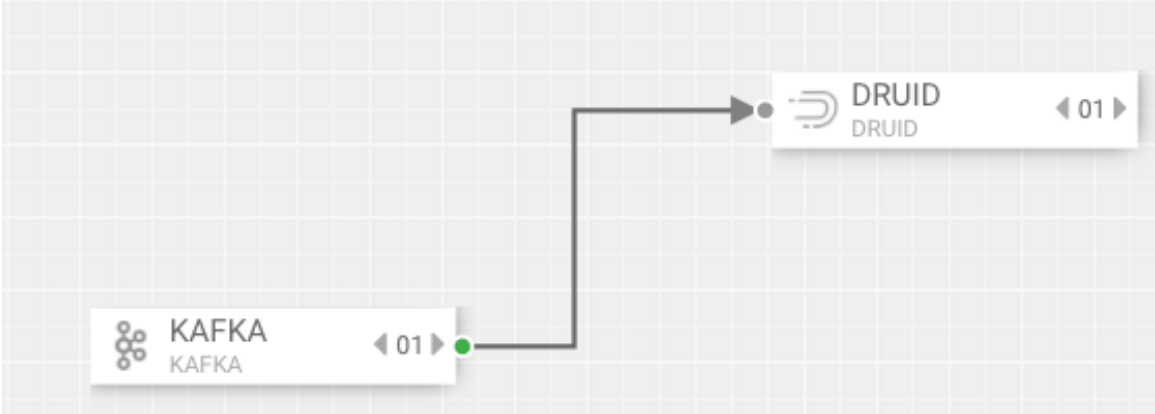
Ingest data into Druid

Druid is a column-oriented, open-source, distributed data store written in Java. **Druid** is designed to quickly ingest massive quantities of event data and provide low-latency queries on top of the data. For real-time dashboards, we want to write data to Apache Druid and visualize the data later with a real-time dashboard implemented with Superset.

Select the Druid processor from SINK, then drag and drop it on the canvas:



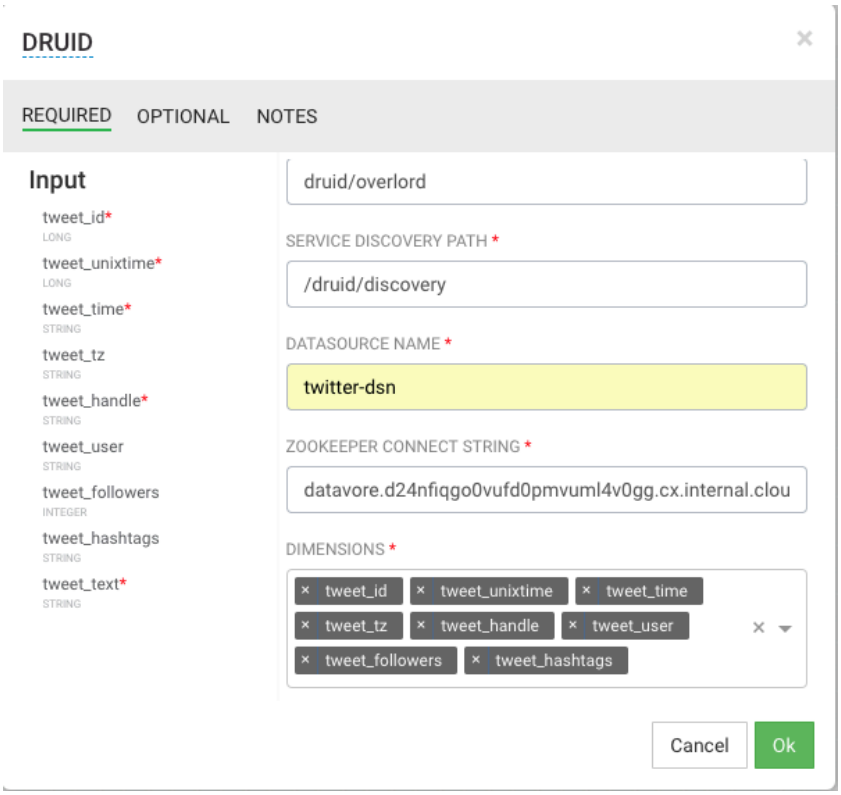
At this point, link the 'KAFKA' and 'DRUID' operators by clicking on the green circle on the right-hand side of the 'KAFKA' operator and bringing the arrow to the grey circle on the 'DRUID' operator:



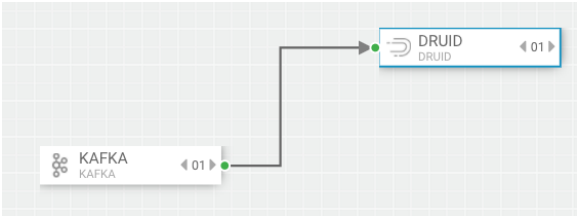
Double-click on the 'DRUID' operator and enter the following values:

- DATASOURCE NAME: **twitter-dsn**
- DIMENSIONS: Add all the available dimensions from the drop-down list.

- TIMESTAMP FIELD NAME: **processingTime**.
- WINDOW PERIOD: **PT5M**
- INDEX RETRY PERIOD: **PT5M**
- SEGMENT GRANULARITY: **MINUTE**
- QUERY GRANULARITY: **MINUTE**



Click OK. At this stage, the SAM application should look like the following:

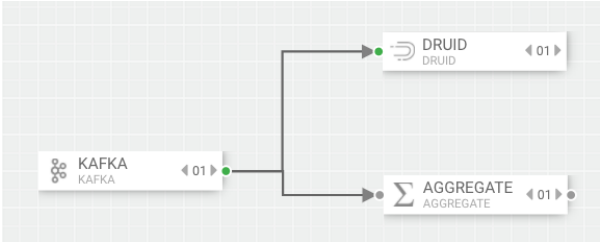


Aggregate real-time Twitter data

For real-time analytics, we want to aggregate the twitter feed every 1 minute based on the number of tweets per hashtag to detect trending hashtags.

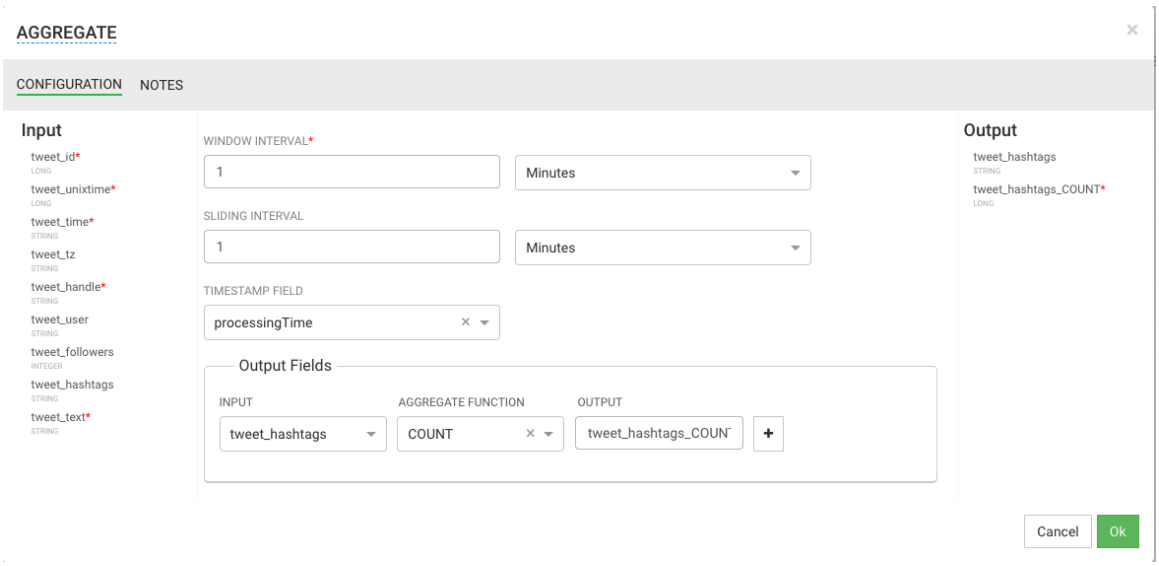
Drag and Drop the 'AGGREGATE' processor to the canvas:

Link it to the 'KAFKA' processor:

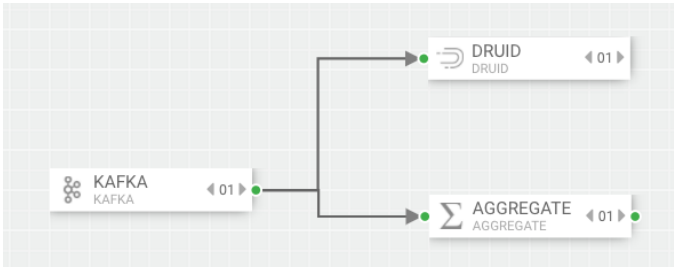


Double-click on 'AGGREGATE' and enter the following values:

- SELECT KEYS: **tweet_hashtags**
- WINDOW INTERVAL TYPE: **Time**
- WINDOW INTERVAL: **1 Minutes**
- SLIDING INTERVAL: **1 Minutes**
- TIMESTAMP FIELD: **ProcessingTime**
- Output Fields:
    - INPUT: **tweet_hashtags**
    - AGGREGATE FUNCTION: **COUNT**
    - OUTPUT: **tweet_hashtags_COUNT**.



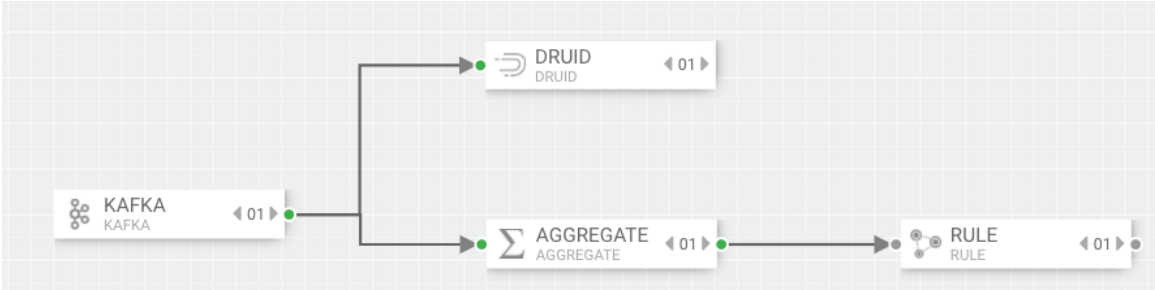
Click OK. At this stage, the SAM application should look like the following:



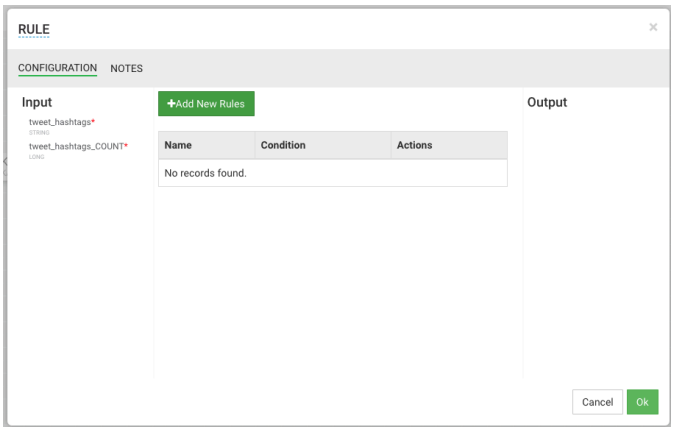
Filter Data in real-time

At this point, we want to filter only the trending hashtags. For the purpose of this lab, we will define the trending hashtags by hashtags appearing more than 3 times per minute. Click on the 'RULE' icon and drag it on the canvas:

Link the 'AGGREGATE' processor with the newly added 'RULE' icon.

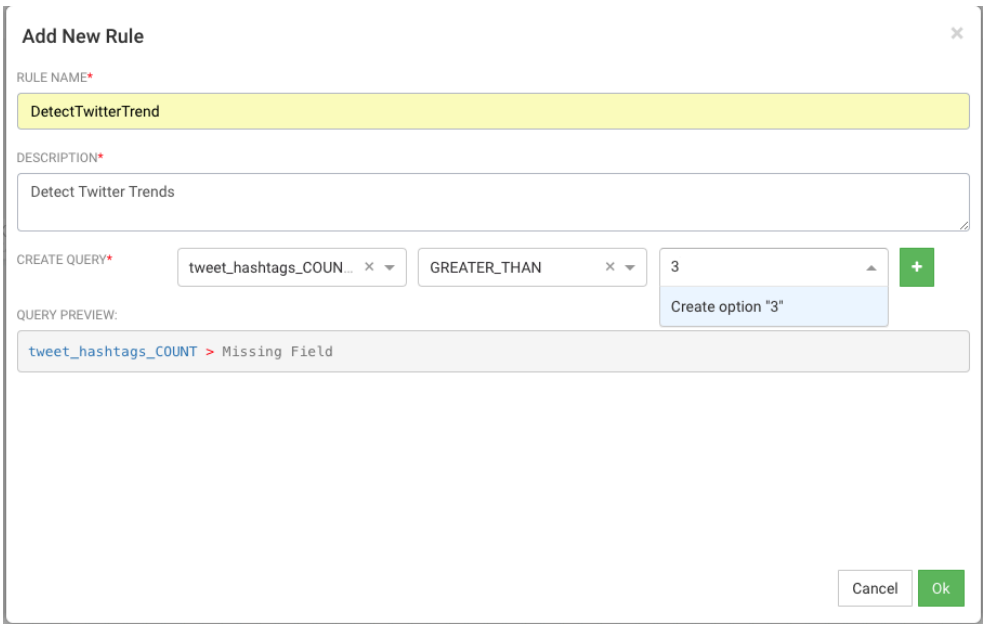


Double-click on the 'RULE' icon.

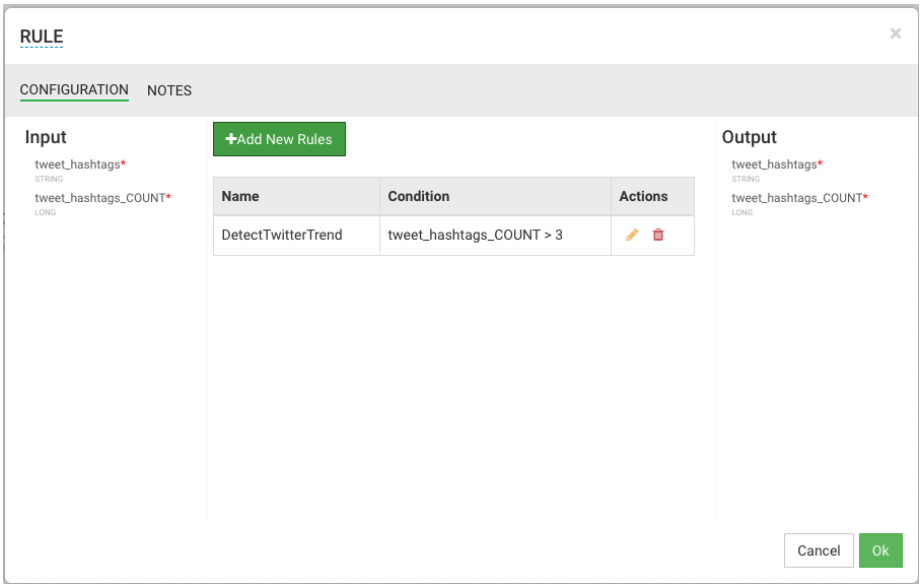


Click on '+Add New Rules' green button on top and enter the following values:

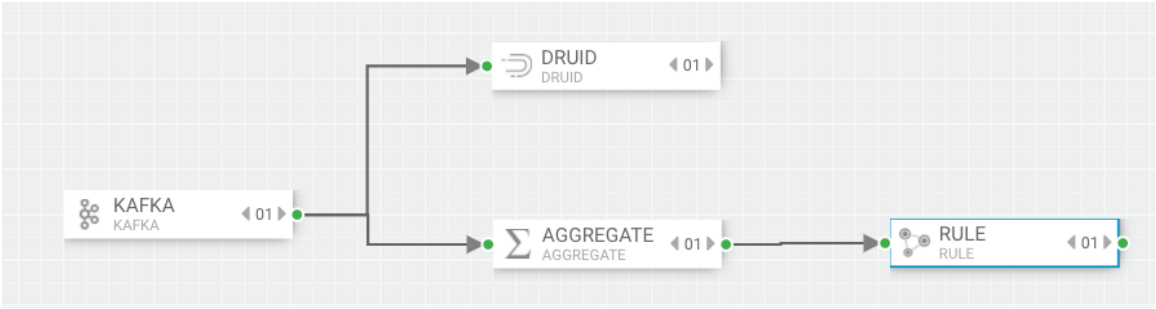
- RULE NAME: **DetectTwitterTrend**
- DESCRIPTION: **Detect Twitter Trends**
- CREATE QUERY:
    - **Tweet_hashtags_COUNT**
    - **GREATER THAN**
    - **3** (Type 3 in the window. It will offer you the option to Create option "3". Select that option.)



Click OK.



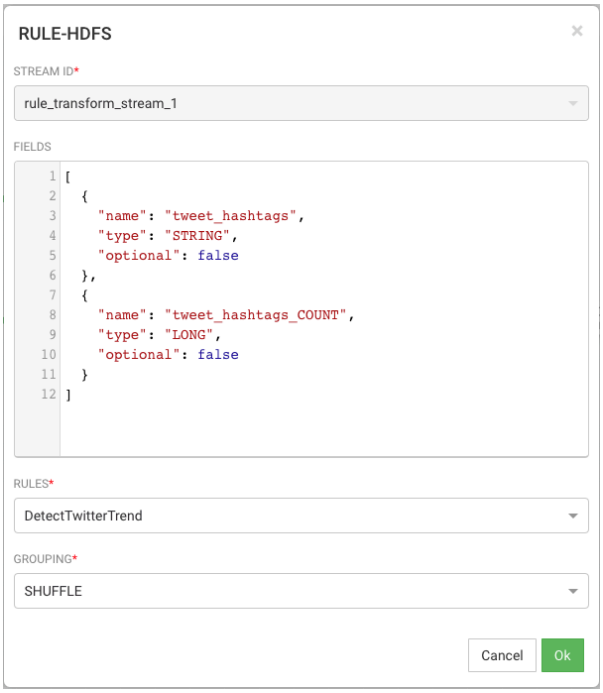
Click OK again. At this stage, the SAM application should look like the following:



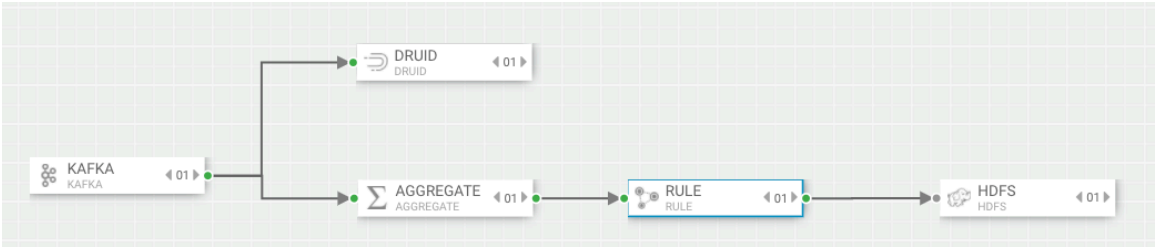
Store results on HDFS

For real-time control and command operations, we could for example use the Notification processor to send an email once an abnormal trend has been detected. For the lab purpose, we simply store them on HDFS. Select HDFS from the **SINK** Operators and drag it to the canvas:

Link the 'HDFS' processor with the previously added 'RULE' processor. A new window will appear. Accept all default values and click OK:

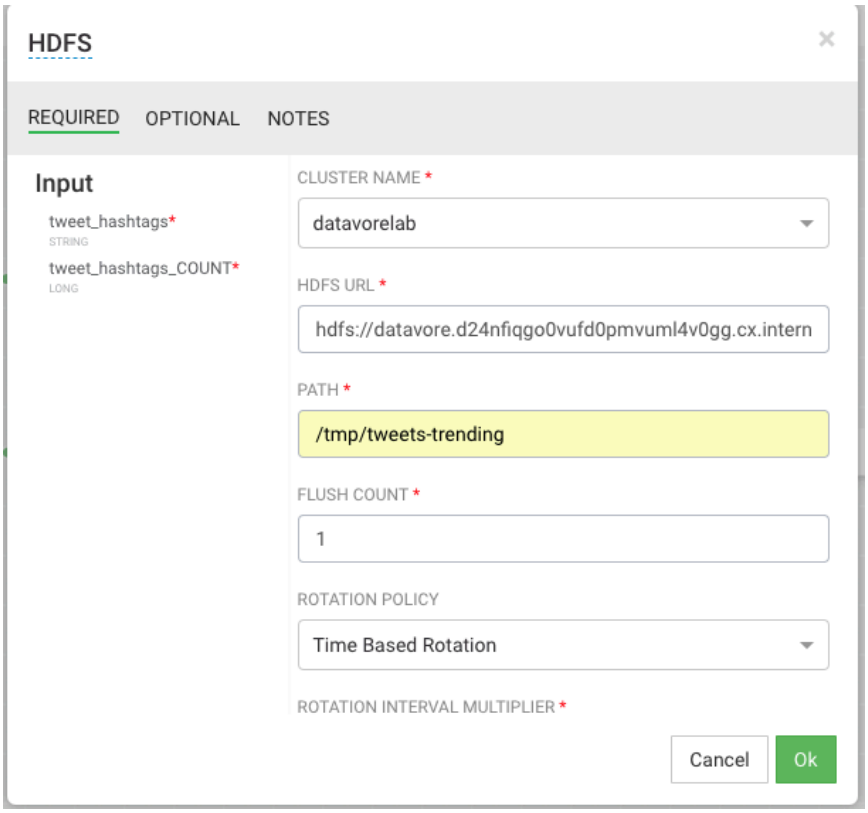


The canvas will now show the following application:



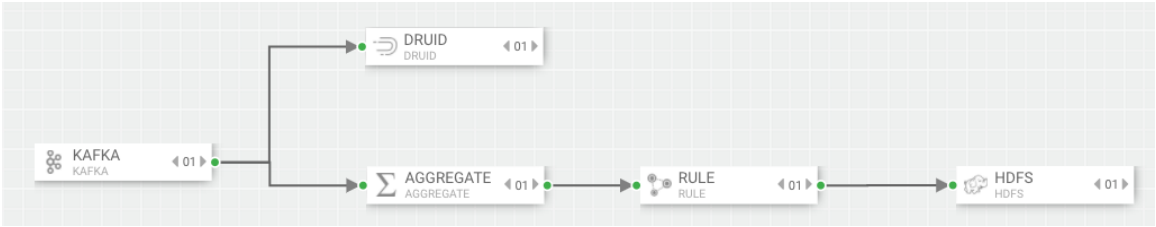
Double-click on HDFS and enter the following values:

- PATH: **/tmp/tweets-trending**
- FLUSH COUNT: **1**
- ROTATION POLICY: **Time Based Rotation**
- ROTATION INTERVAL MULTIPLIER: **1**
- ROTATION INTERVAL UNIT: **MINUTES**
- OUTPUT FIELDS:
    - **Tweet_hashtags**
    - **Tweet_hashtag_COUNT**



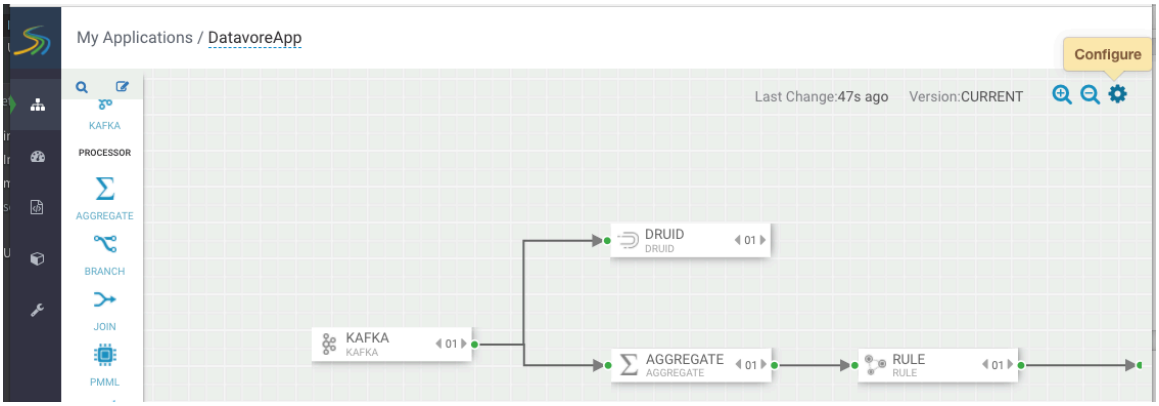
Click OK.

At this stage, the SAM application is now completed and should look like this:



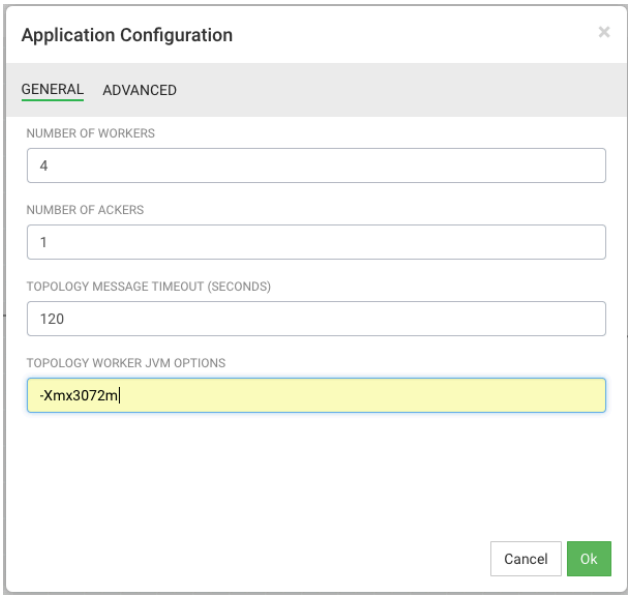
Configure the SAM Application

Click on the Configure icon on the top right of the canvas:



Enter the following values:

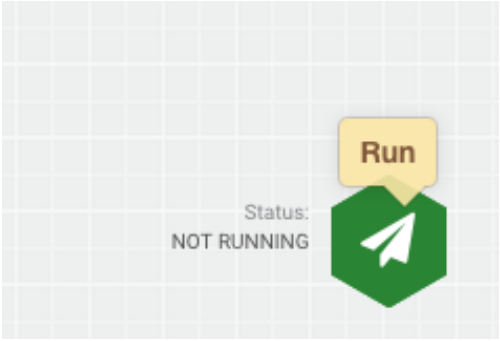
- NUMBER OF WORKERS: **4**
- NUMBER OF ACKERS: **1**
- TOPOLOGY MESSAGE TIMEOUT (SECONDS): **120**
- TOPOLOGY WORKER JVM OPTIONS: **-Xmx3072m**



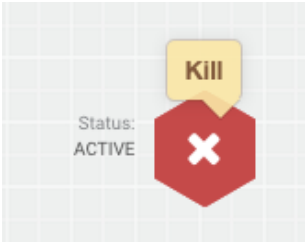
Click OK.

Run the SAM Application

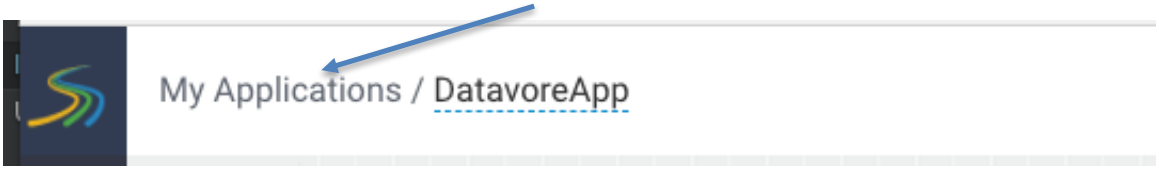
On the bottom right of the canvas, click on Run icon:



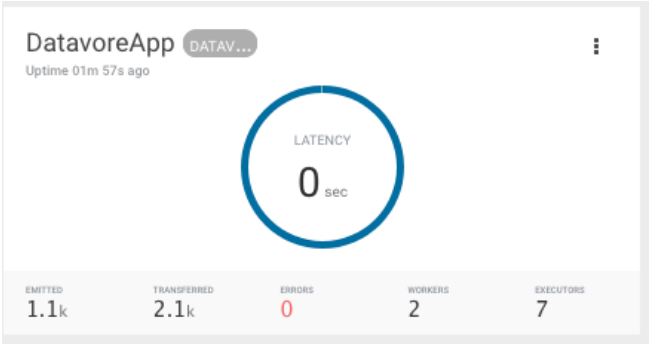
Click OK on the window asking you to confirm the configuration. Give a few minutes for SAM to compile the application into a Storm topology and deploy this topology. You should get a notification that the application has been deployed successfully. The icon will now change to the following state. **Do NOT click on Kill**:



On the top left of your web browser window, click on 'My Applications' to get back to the main Application screen:

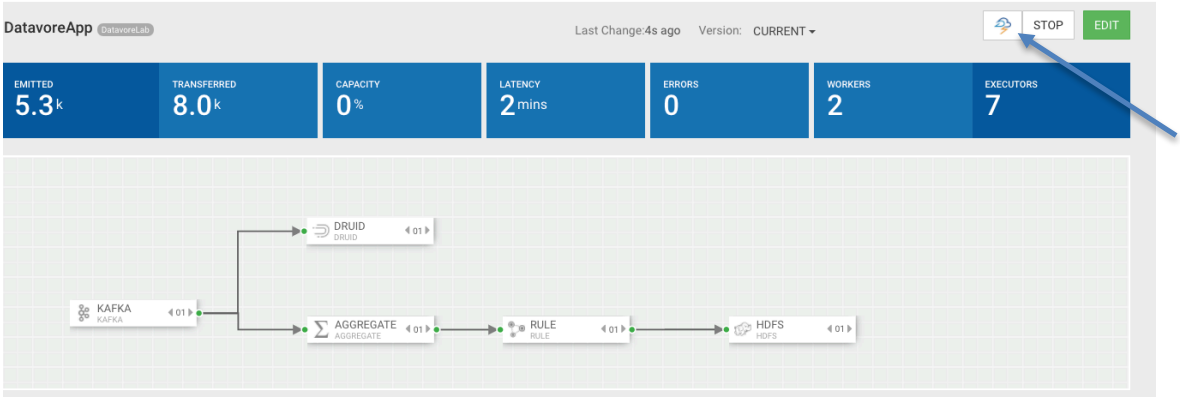


You will now see your application running and processing tuples read from Kafka. In the below picture, we can see that 1.1K tweets have been emitted:



Monitoring the Storm Application

When clicking deploy, SAM will compile the application into a Storm topology and will deploy it. We can check the Storm application topology and monitor it from the Storm UI. Double-click on the DatavoreApp tile from the previous screen and click on the Storm icon:



This will open a new tab in your web browser with the Storm UI. You can explore various metrics available. One metrics of interest are the Spouts and the Bolts metrics.

For the Spouts, we have a single Kafka Spout in this SAM application:

Spouts (All time)

Search:

Id	Executors	Tasks	Emitted	Transferred	Complete latency (ms)	Acked	Failed	Error Host	Error Port	Last error	Error Time
1-KAFKA	1	1	4600	9200	116081.430	3000	1000				

Showing 1 to 1 of 1 entries

The above shows us that 4600 tweets have been read from Kafka and 9200 have been transferred. This is because we have 2 branches where it is delivered from Kafka, one being the Druid sink the other being the Aggregate branch.

For Bolts, we see statistics about all the Bolts which are part of the SAM application. These bolts correspond to the different SAM operators. Identify which SAM operators they correspond to.

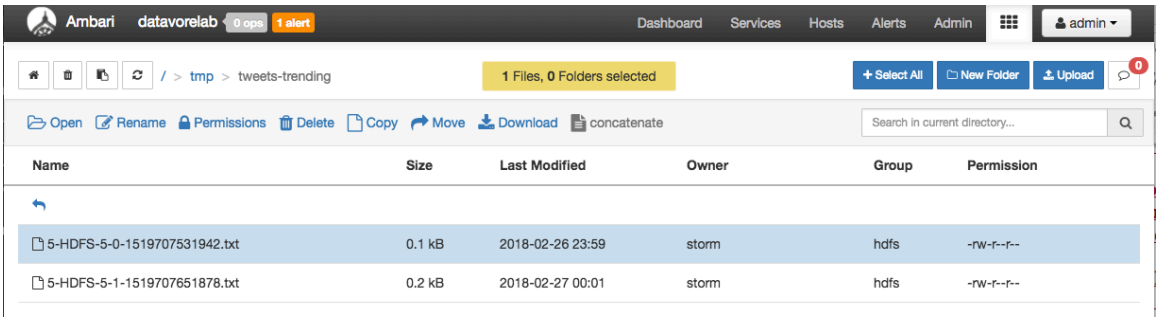
Bolts (All time)

Search:

Id	Executors	Tasks	Emitted	Transferred	Capacity (last 10m)	Execute latency (ms)	Executed	Process latency (ms)	Acked	Failed	Error Host	Error Port	Last error	Error Time
2-DRUID	1	1	2200	0	0.080	8.082	4860	67391.195	4000	0				
3.1-WindowedRulesProcessor	1	1	500	500	0.000	0.012	4840	117104.820	4000	0				
4-RULE	1	1	80	80	0.002	1.407	540	2.667	540	0				
5-HDFS	1	1	0	0	0.004	21.600	100	9.067	100	0				

Checking data on HDFS

As we can see above in the Storm application, 100 tuples have been written onto HDFS. We can check on the HDFS path /tmp/tweets-trending the aggregates that are being generated in real-time through the Ambari File View:



We can check the content of the file by opening a file preview or downloading it (Click on Open on the top left):

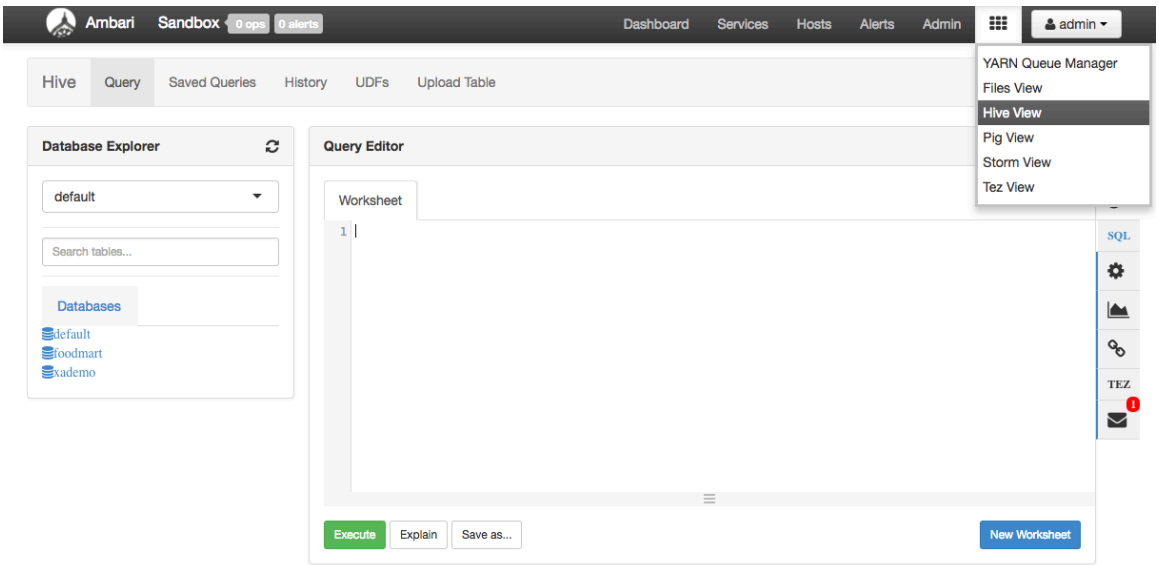
We can see the trending hashtags above.

As it takes approximately 30 minutes to initialize a Druid Data Source, we can analyze the Data at rest on Hive before proceeding to the real-time visualization for the data on Druid.

Data Analysis with Hive

After persisting the tweets in HDFS, the opportunity to perform analytics against the data is nearly unlimited. Leveraging Hive, SQL can be used to analyze the Tweets. The following sections will create the Hive table and run some basic queries against the tweets.

You can access Hive via the Hive view available on Ambari:

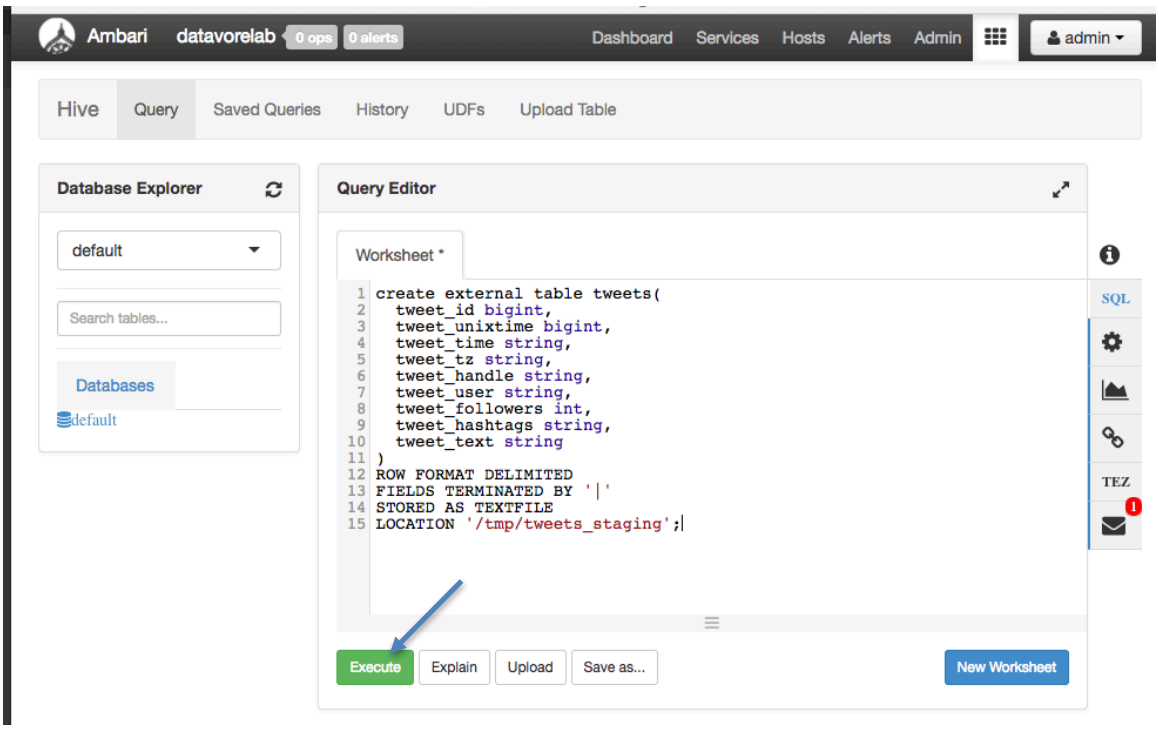


Create the Hive Table

Create the hive table to allow querying the tweets by pasting query available at the following link onto the worksheet:

https://github.com/oascofare/Datavore2018/blob/master/tweets.sql

Click on Execute button:

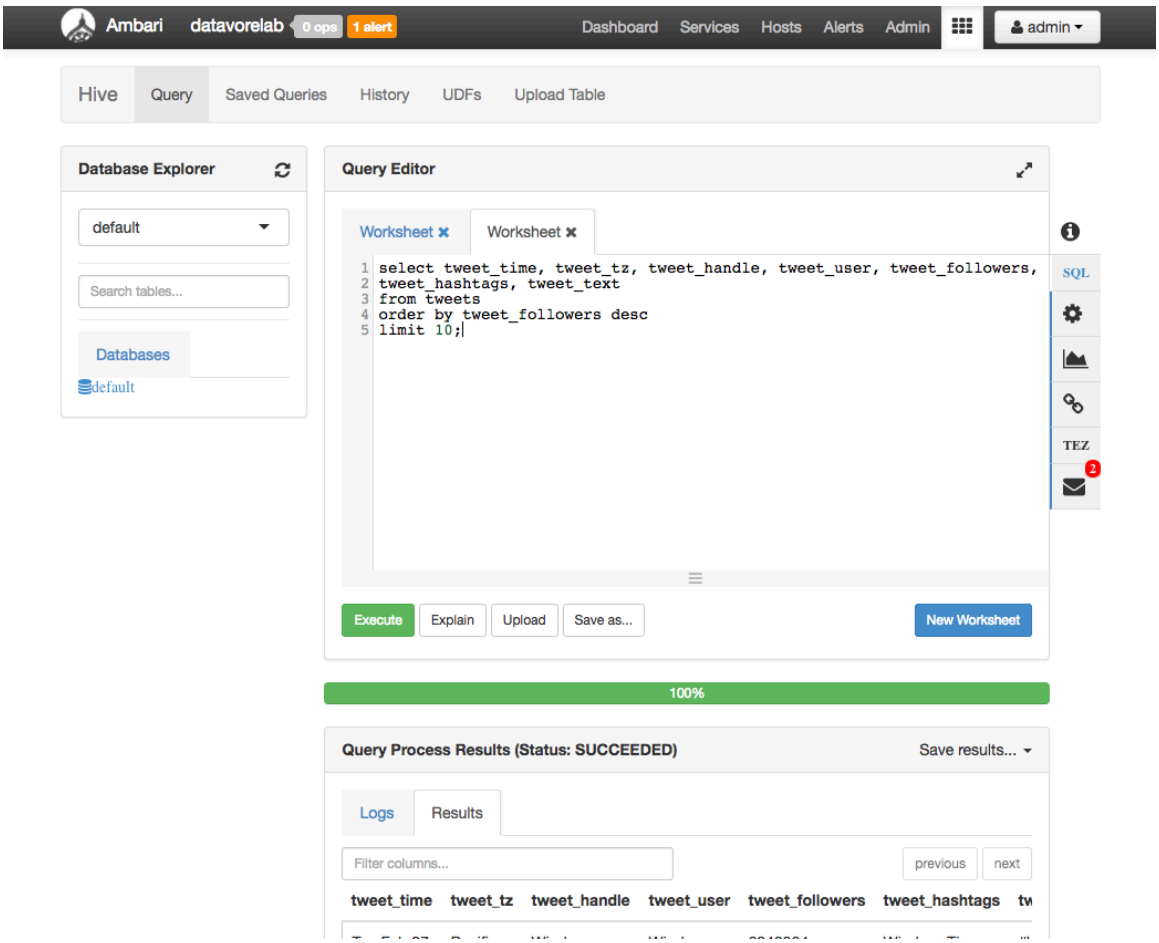


Query Data using SQL on Hive

Click on New Worksheet (blue button on the right-hand side of the screen). Copy and paste the query found at the following link onto the new worksheet:

https://github.com/oascofare/Datavore2018/blob/master/top_tweets.sql

This query will find the top 10 influential tweets (assuming it is based on the number of followers). Click on Execute.

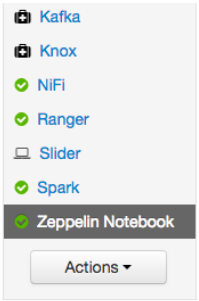


Data Visualization with Zeppelin

In this section, we are going to visualize the twitter data using Zeppelin. SparkSQL will be used to perform data analysis.

All the queries in this section are available on the following link:
https://github.com/oascofare/Datavore2018/blob/master/zeppelin-queries.sql

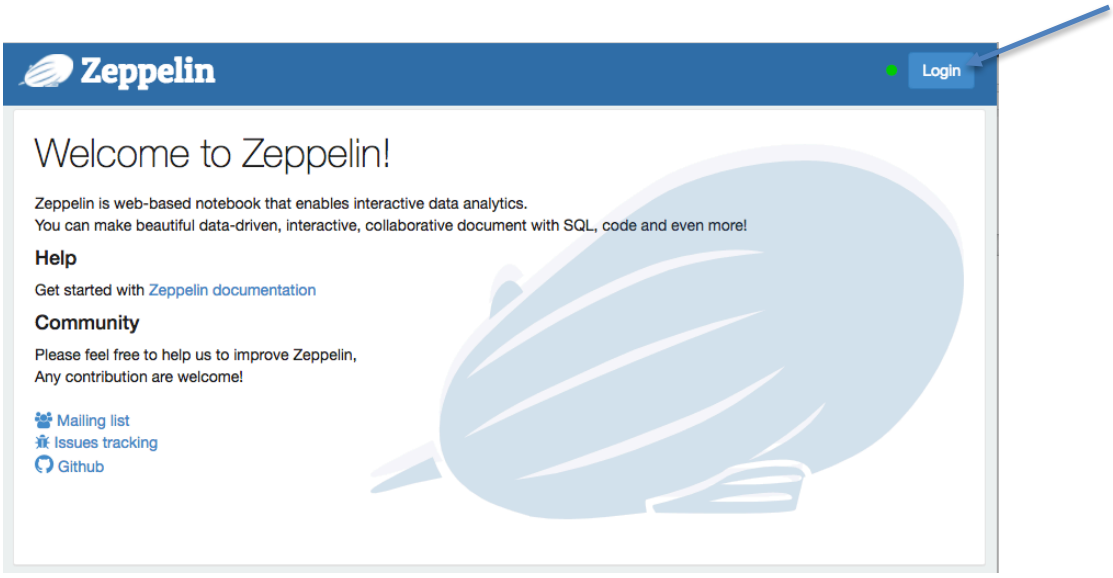
Make sure the Zeppelin service is started in Ambari:



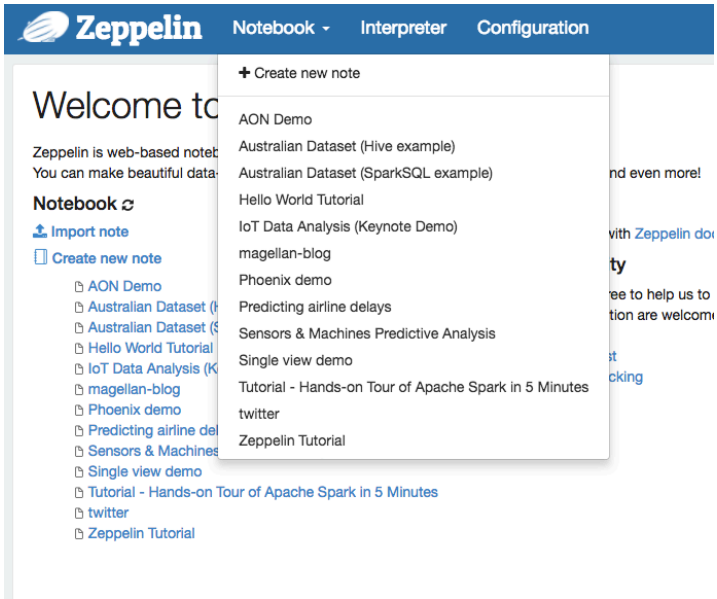
Navigate to Zeppelin using the following URL: http://<public-ip>:9995/#/

Click on Login on the top right-hand side of the screen and use the following credentials:

Username: admin
Password: admin



Use the Notebook drop down menu to create a new note called "Twitter Analysis"



After creating the note, open it up to the blank Notebook screen and type the following command. `%sql` is the SparkSQL interpreter.

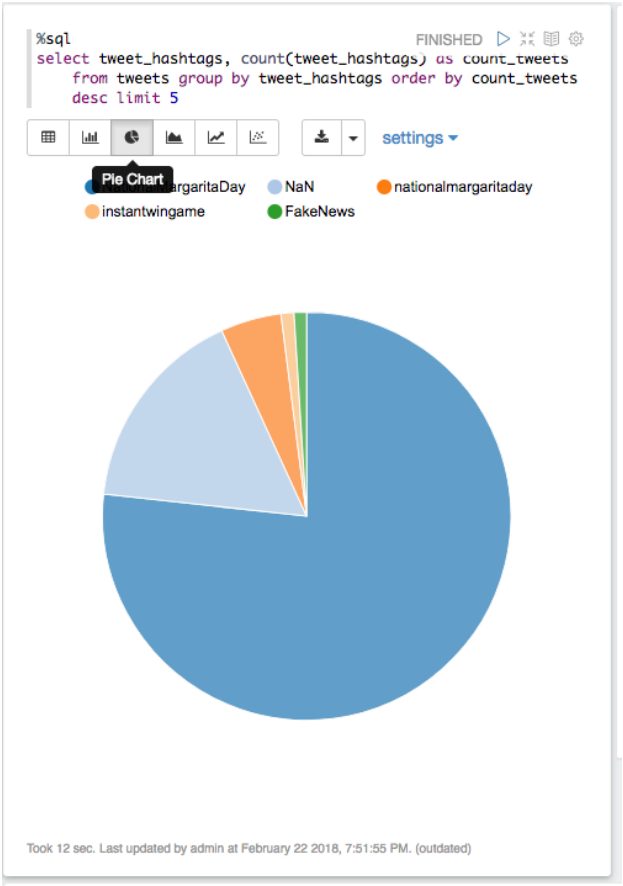
```
%sql
--Top 10 tweets
select tweet_handle, tweet_user, tweet_followers, tweet_hashtags,
tweet_text from tweets order by tweet_followers desc limit 10
```

The query above is similar to the query submitted from Hive view and will give the top 10 influential tweets (based on the number of followers). In this case, Hortonworks is leveraging the Spark processing engine, through the SparkSQL API to perform analytics on the data.

On a new paragraph, the query below will count the top 5 trending hashtags (based on the number of tweets)

```
%sql
--Top 5 trending hashtags
select tweet_hashtags, count(tweet_hashtags) as count_tweets from tweets
group by tweet_hashtags order by count_tweets desc limit 5
```

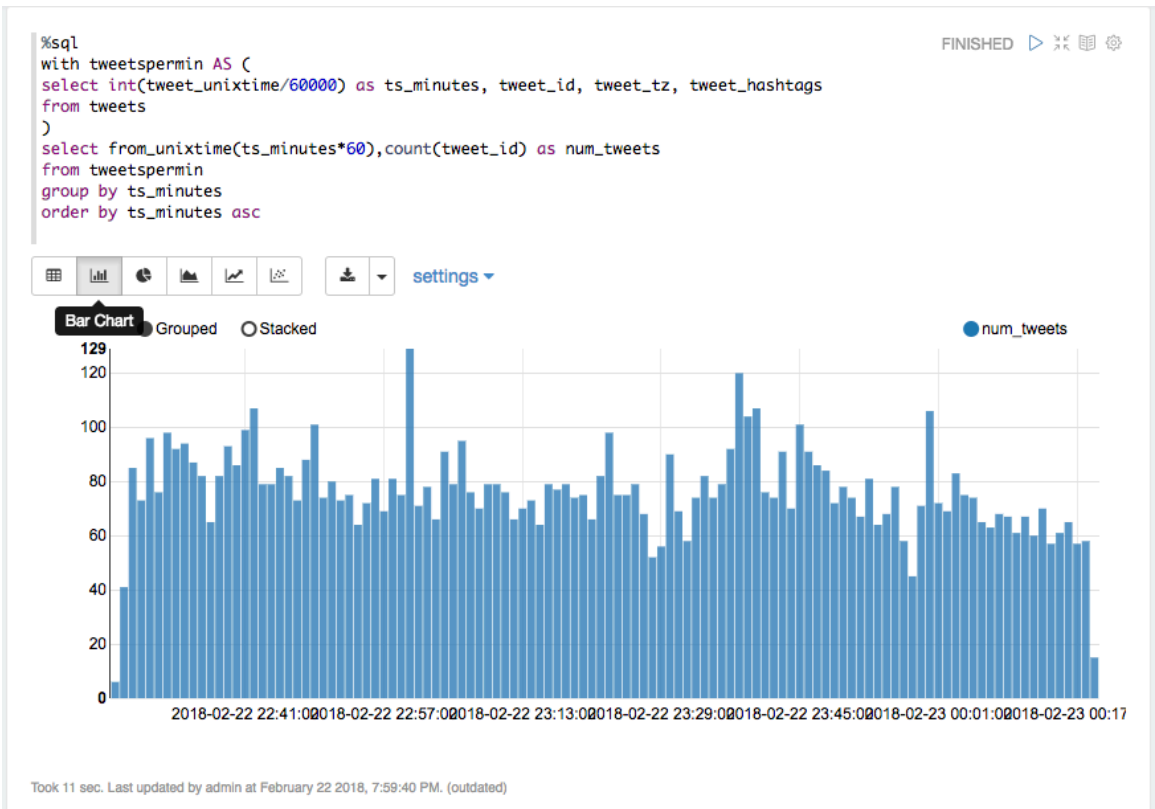
Click on the pie chart icon to visualize the data:



In a next paragraph, we can try to visualize the tweet throughput per minute for the time frame the data has been collected using the following query:

```
%sql
--Tweet Flow per minute
with tweetspermin AS (
select int(tweet_unixtime/60000) as ts_minutes, tweet_id, tweet_tz,
tweet_hashtags
from tweets
)
select from_unixtime(ts_minutes*60),count(tweet_id) as num_tweets
from tweetspermin
group by ts_minutes
order by ts_minutes asc
```

Click on the bar chart icon to visualize the data as a bar chart:



Time allowing, you can optionally add additional paragraphs to this Zeppelin notebook to further explore the data set.

Real-Time Data Visualization with Apache Druid and Superset

Superset is a Business Intelligence tool packaged with many features for designing, maintaining and enabling the storytelling of data through meaningful data visualizations for real-time data.

In this section, we will explore data visualization with Superset for the Twitter dataset, that has been ingested into Druid, through the SAM application.

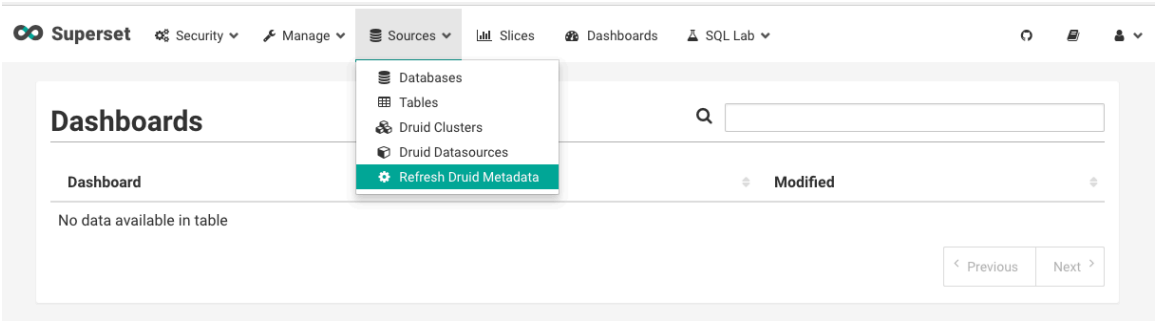
In order to access Superset, use the following URL:
http://<vm-public-ip>:9087/login

In order to login, use the following credentials:

- Username: **admin**
- Password: **admin**

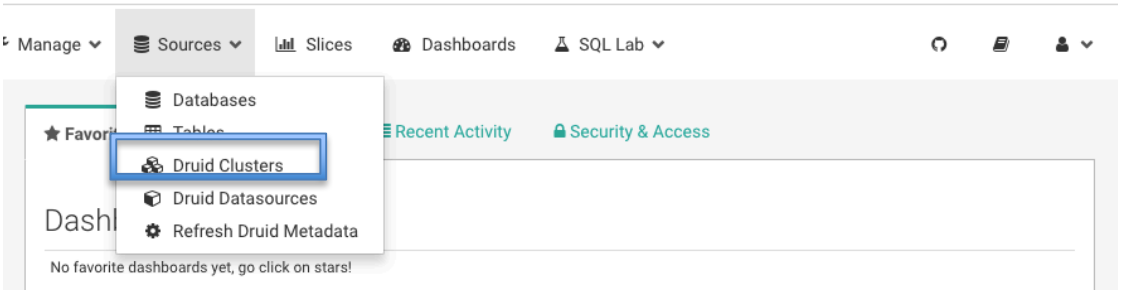
Locate the Druid Twitter datasource

Once on the main menu for Superset, please click on Sources from the menu and select 'Refresh Druid Metadata':

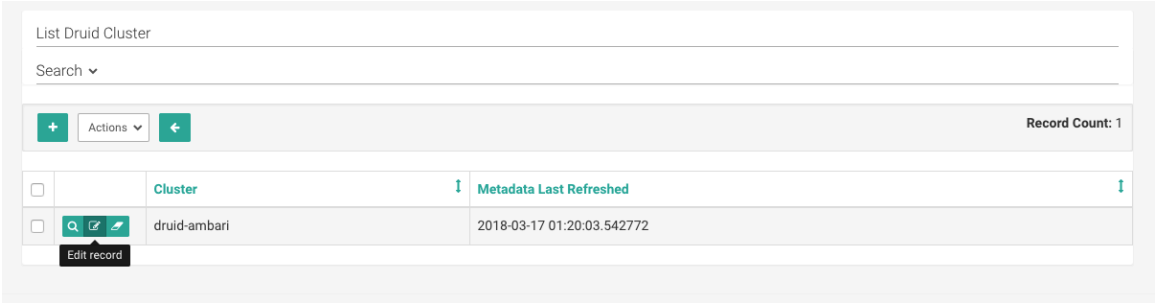


The twitter-dsn data source created by the SAM application should now appear. It takes around 30 minutes for the initialization of the new datasource under Druid:

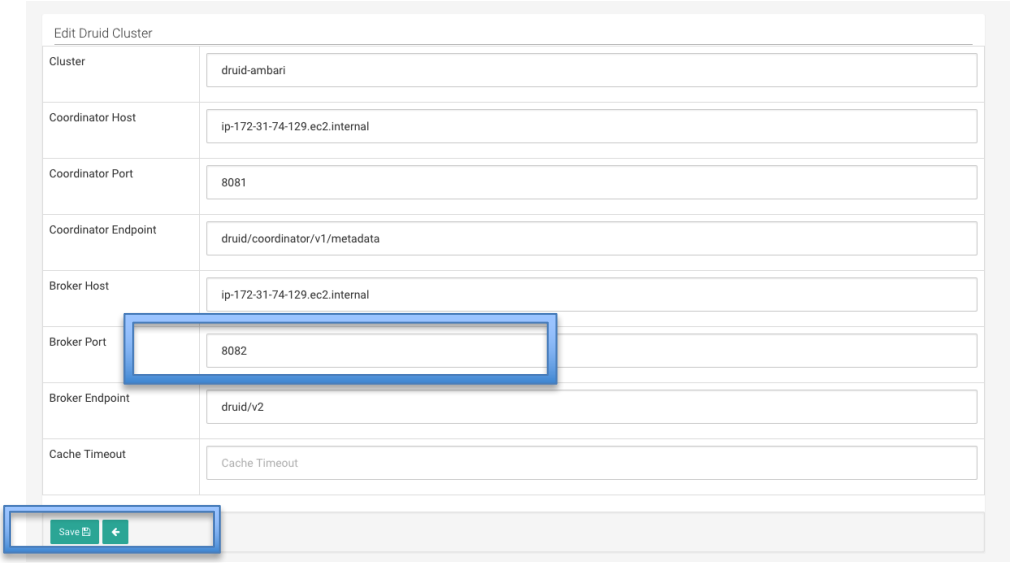
If you can't your data source or get an error when you click on the on twitter-dsn hyperlink ; go on the source menu on the top, then click on Druid Clusters.



Then click on "edit record"



Update the Broker Port to 8082 , then click on save .



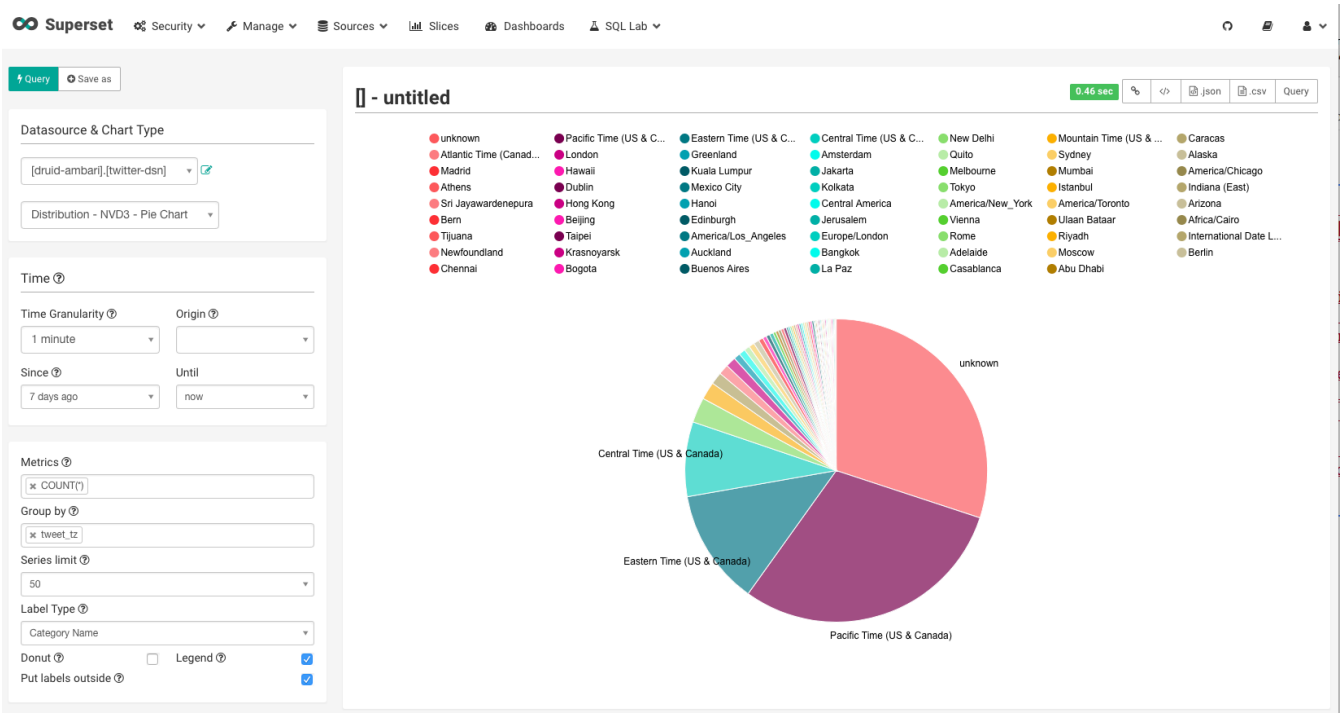
If you still can't see your data source you may want to go back to the previous step to monitor the Storm application and ensure that tuples are written onto Druid. Once confirmed, you should wait for more time before refreshing the Druid Metadata again.

Create a visualization and a dashboard

We can now generate at least one visualization to check the timezone region most active in terms of tweet counts. Click on twitter-dsn hyperlink. A new window will open (See below). Enter the following fields:

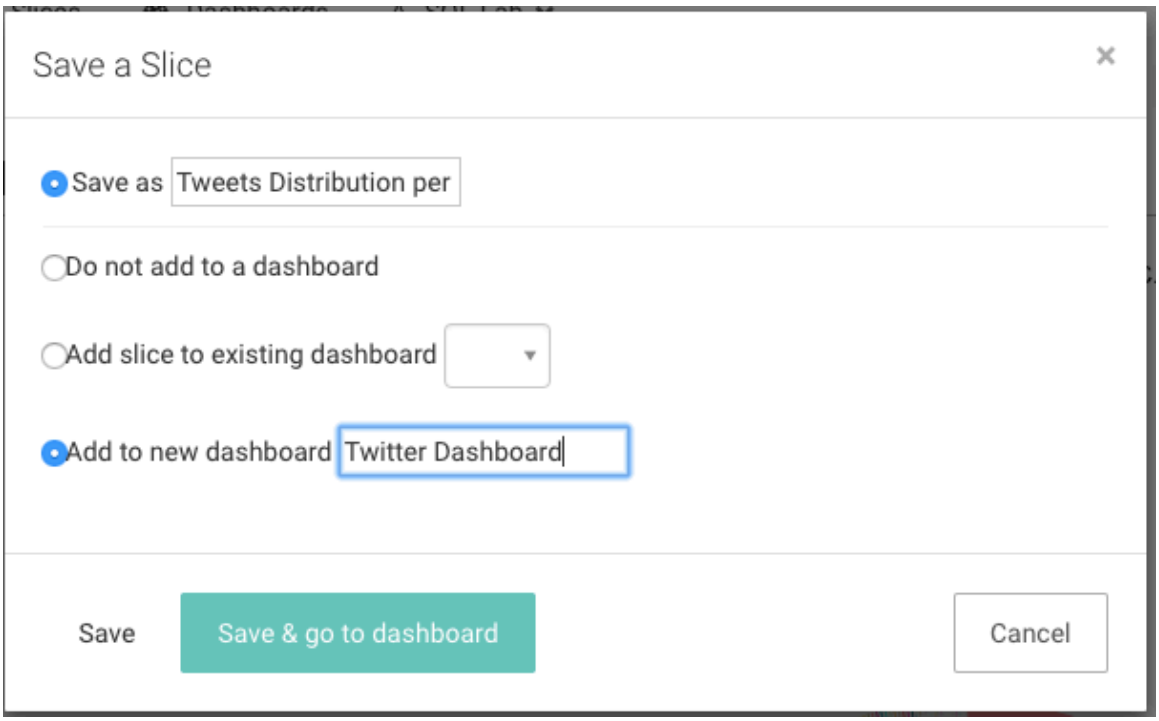
- Chart Type: **Distribution – NVD3 – Pie Chart**
- Time Granularity: **1 minute**
- Since: **7 days ago**
- Metrics: **COUNT(*)**
- Group by: **tweet_tz**

Once complete, click on Query (green button) on the top left of the screen. You should see the visualization appear.

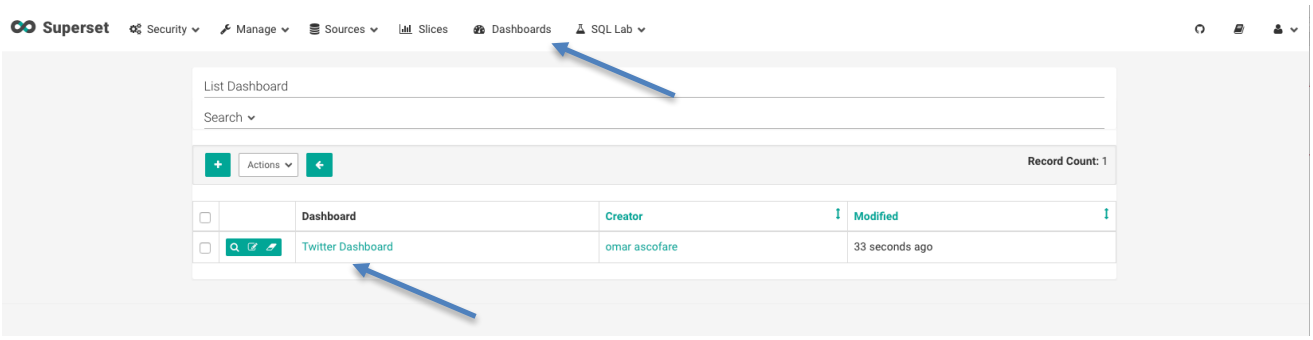


Click on Save as:

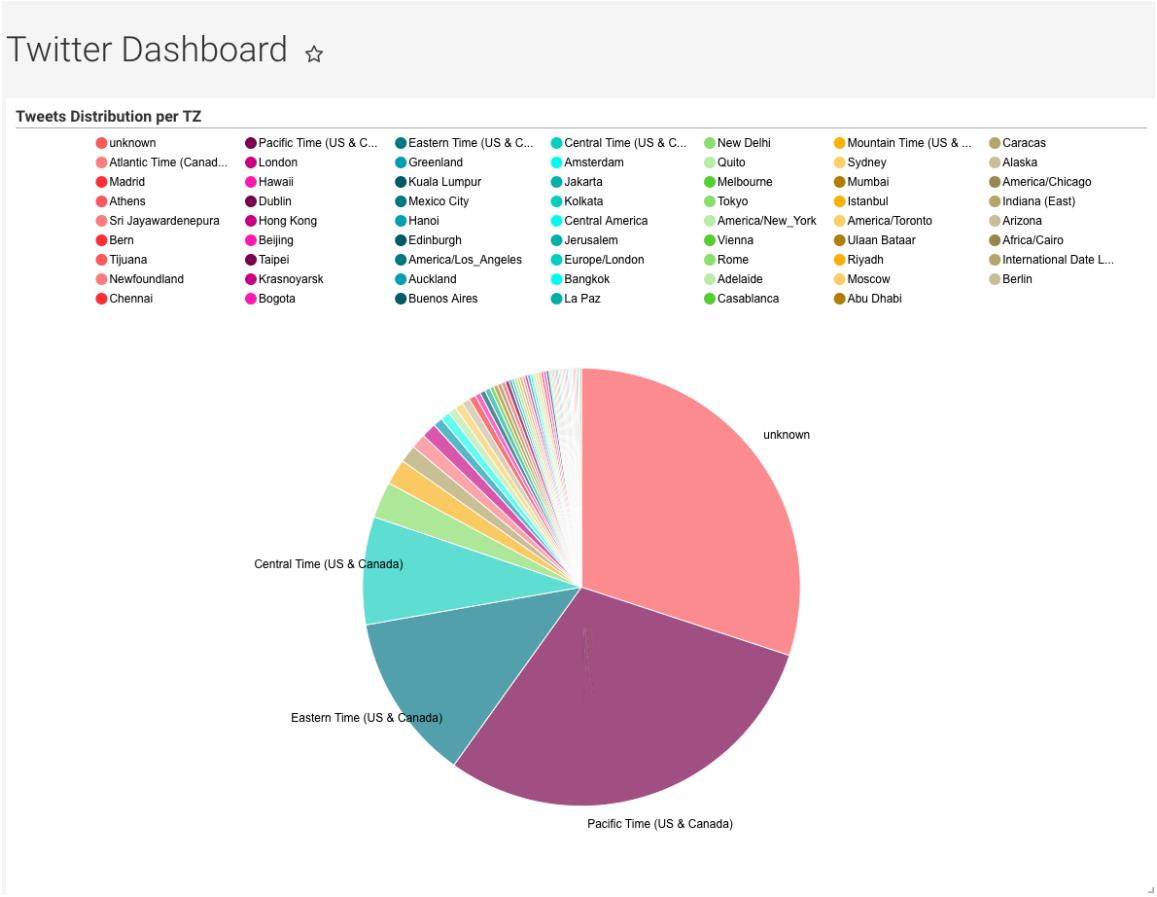
- Save as: **Tweets Distribution per TZ**
- Add to new dashboard: **Twitter Dashboard**



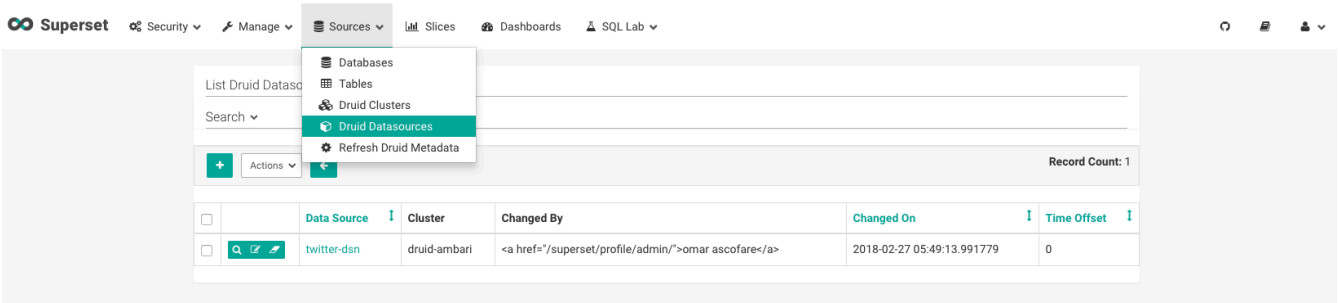
Click on Save. Then, click on Dashboards on the top menu, then click on Twitter Dashboard hyperlink:



You should now see your dashboard with a single visualization:



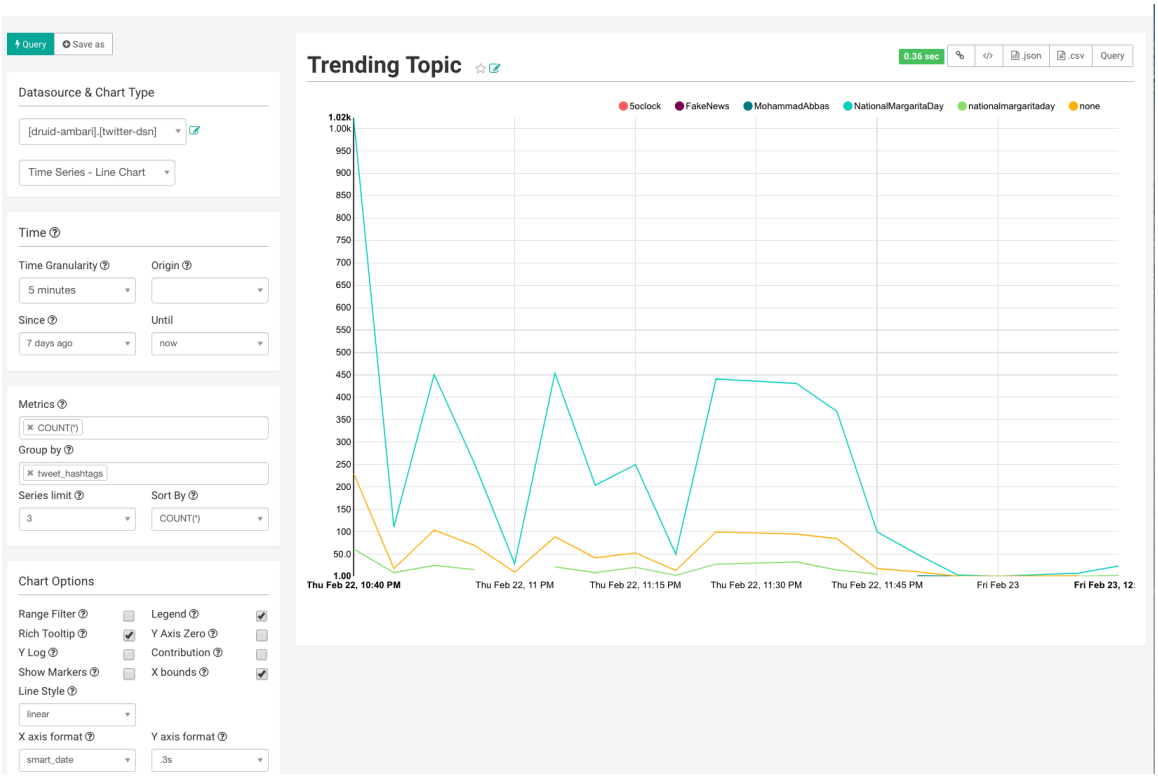
We can add another visualization to the dashboard. Select Sources from the top menu and click on the twitter-dsn hyperlink:



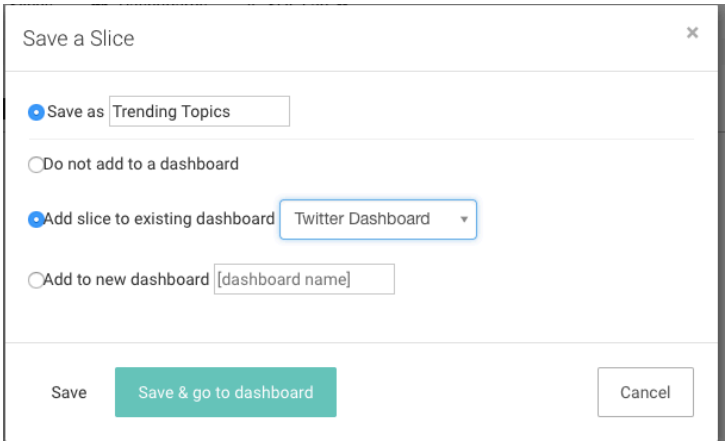
Enter the following fields:

- Chart Type: **Time-Series – Line Chart**
- Time Granularity: **5 minutes**
- Since: **7 days ago**
- Metrics: **COUNT(*)**
- Group by: **tweet_hashtags**
- Series limit: **3**
- Sort By: **COUNT(*)**

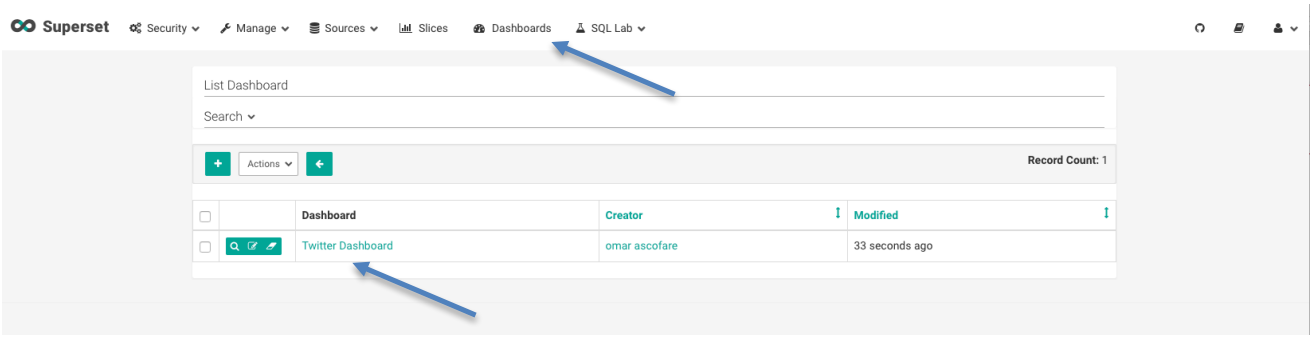
Once complete, click on Query (green button) on the top left of the screen. You should see the visualization appear.



You can now save this slide and add it as part of the Twitter Dashboard. Click the 'Save as' button:



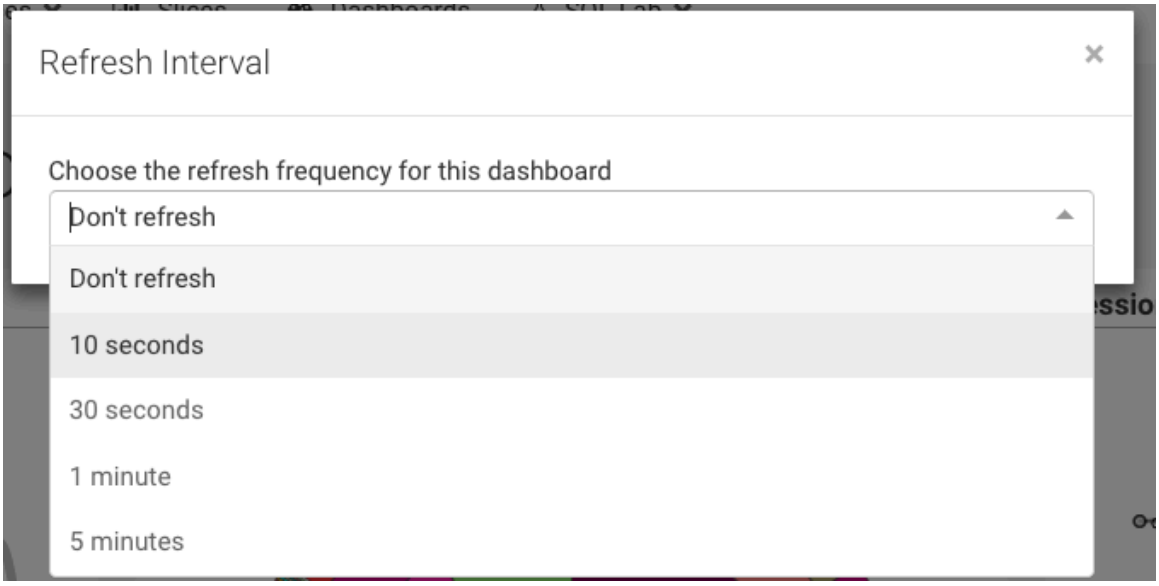
Click on Save. Then, click on Dashboards on the top menu, then click on Twitter Dashboard hyperlink:



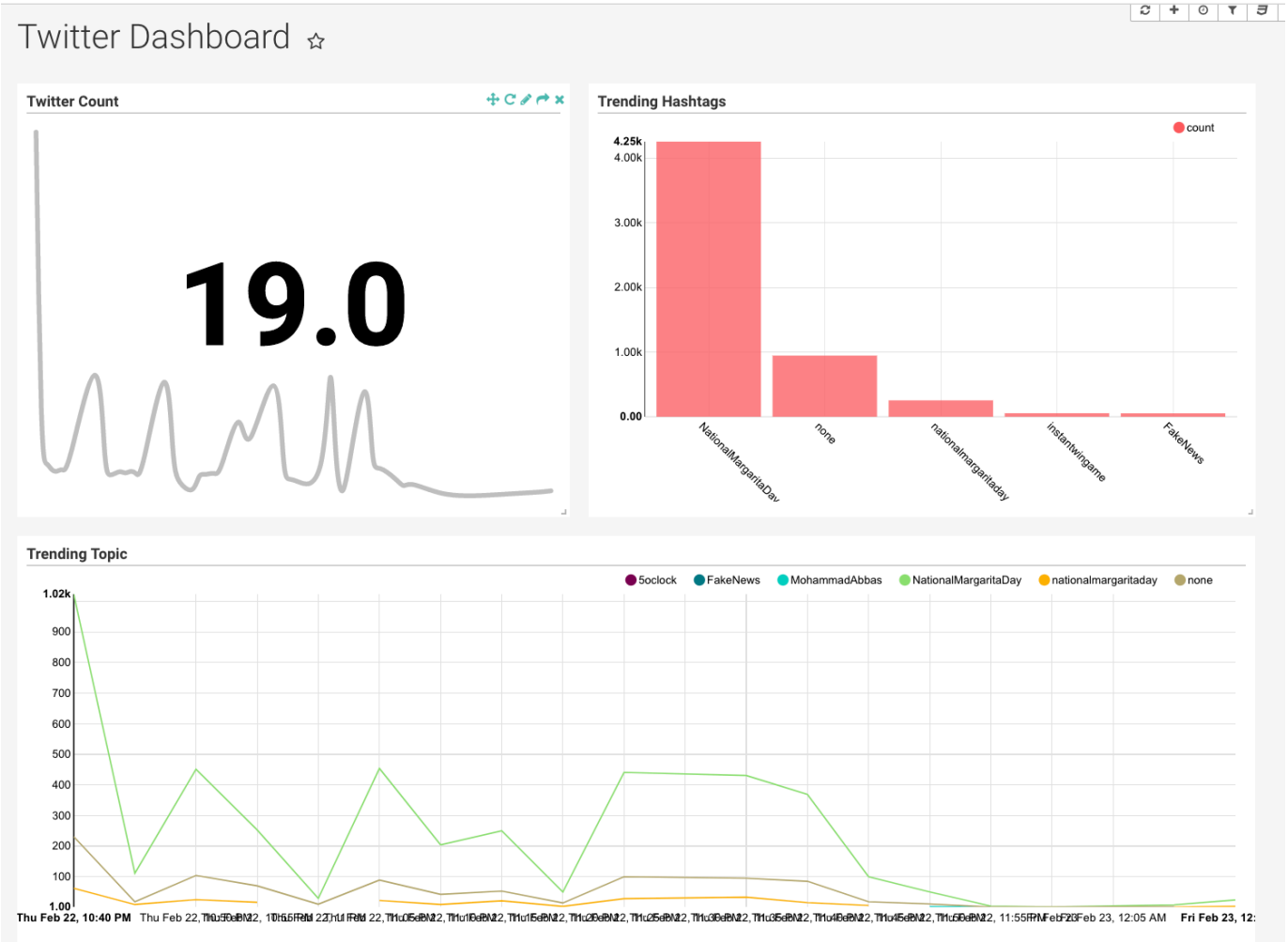
It may take a few minutes to execute the initial queries required for your visualization, and see the new visualization added on your dashboard.

Note that these queries can be refreshed automatically at a set interval to get a dynamic dashboard. This interval can be set if you click on the clock icon on the top right of the dashboard:

You can then optionally choose the refresh interval:



If time permits, please consider adding additional visualization such as tweet count, or a histogram showing the trending hashtags. Below are additional examples of visualizations:


Twitter Dashboard
Twitter Count
19.0
Trending Hashtags
count
4.25k
4.00k
3.00k
2.00k
1.00k
0.00
NationalMargaritaDay
none
nationalmargaritaday
instantwingame
FakeNews
Trending Topic
5oclock
FakeNews
MohammadAbbas
NationalMargaritaDay
nationalmargaritaday
none
1.02k
900
800
700
600
500
400
300
200
100
1.00
Thu Feb 22, 10:40 PM
Fri Feb 23, 12:05 AM