Inhoud

[Getting started with Kafka 2](#_Toc474140170)

[Install the VM 2](#_Toc474140171)

[Getting to know the VM 3](#_Toc474140172)

[Optional: Set Networking in VM 6](#_Toc474140173)

[Getting started with Kafka 8](#_Toc474140174)

[Create a topic 8](#_Toc474140175)

[Produce a message 9](#_Toc474140176)

[Consume a message 12](#_Toc474140177)

[Availability 13](#_Toc474140178)

[Bonus 14](#_Toc474140179)

[Questions 14](#_Toc474140180)

[Oracle Stream Analytics (minimal sample) 15](#_Toc474140181)

[Create a connection 15](#_Toc474140182)

[Create a Stream 16](#_Toc474140183)

[Create an exploration 17](#_Toc474140184)

[Confirm OSA receives messages 18](#_Toc474140185)

[OSA with Twitter and Service Bus 18](#_Toc474140186)

[Obtain requires data from Twitter 18](#_Toc474140187)

[OSA Configuration 20](#_Toc474140188)

[Create a Service Bus process 31](#_Toc474140189)

[References 36](#_Toc474140190)

# Getting started with Kafka

## Install the VM

Running the VM requires 2Gb of RAM and approximately 7Gb of free HD space

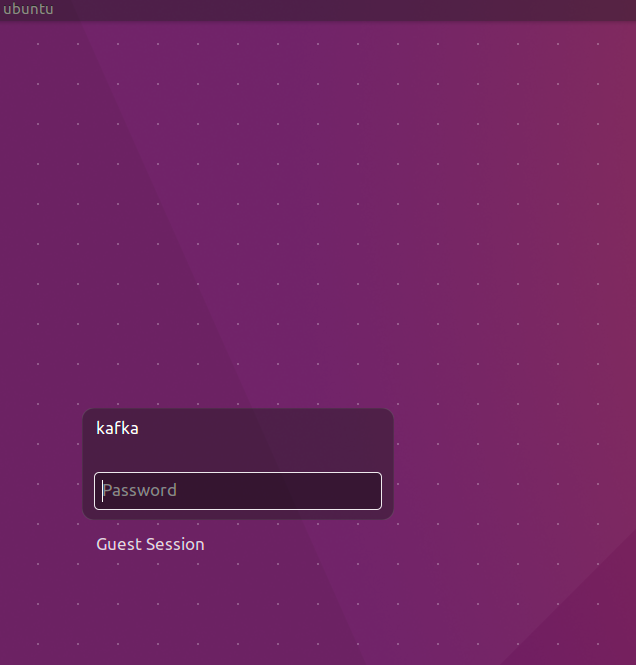
Download VirtualBox from <https://www.virtualbox.org>.

Start VirtualBox

Import the VM image by going to File, Import Virtual Appliance and browse to the image file. The VM requires 2Gb of RAM to run.

Start the VM

The VM will boot to a graphical login screen



The VM has two users: root and kafka. Both have password Welcome01.

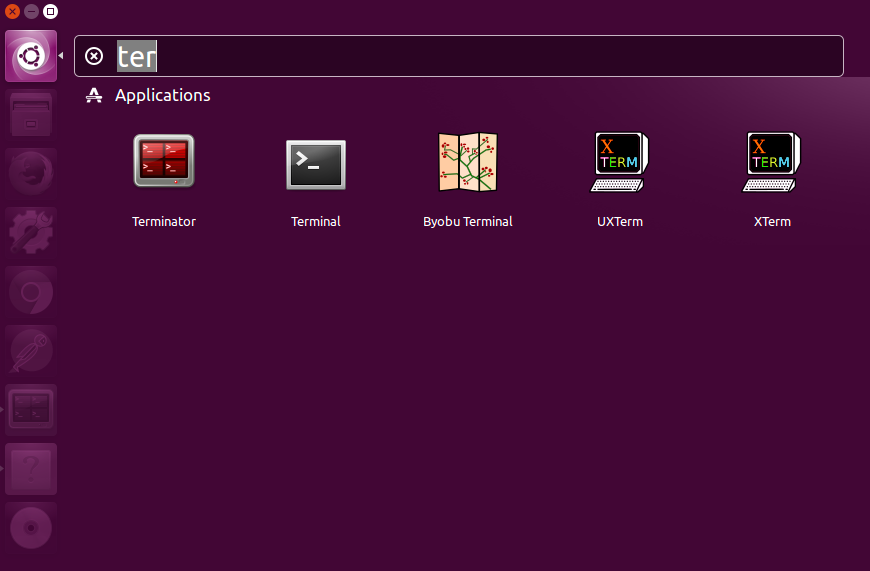
The VM contains:

* Ubuntu 16.04 LTS
* Oracle JDK 8
* Eclipse Neon.2
* Node 7.4
* Zookeeper
* Firefox
* Chrome
* Postman
* confluent-platform-oss-2.11
* kafka-manager 1.3.2.1. When started runs on <http://localhost:9000>
* kafkatool 1.0.1

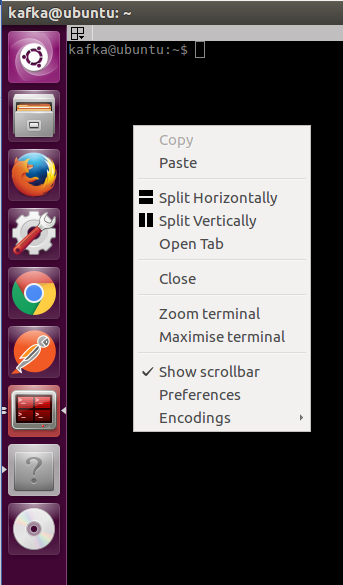
## Getting to know the VM

Login to the VM: kafka/Welcome01

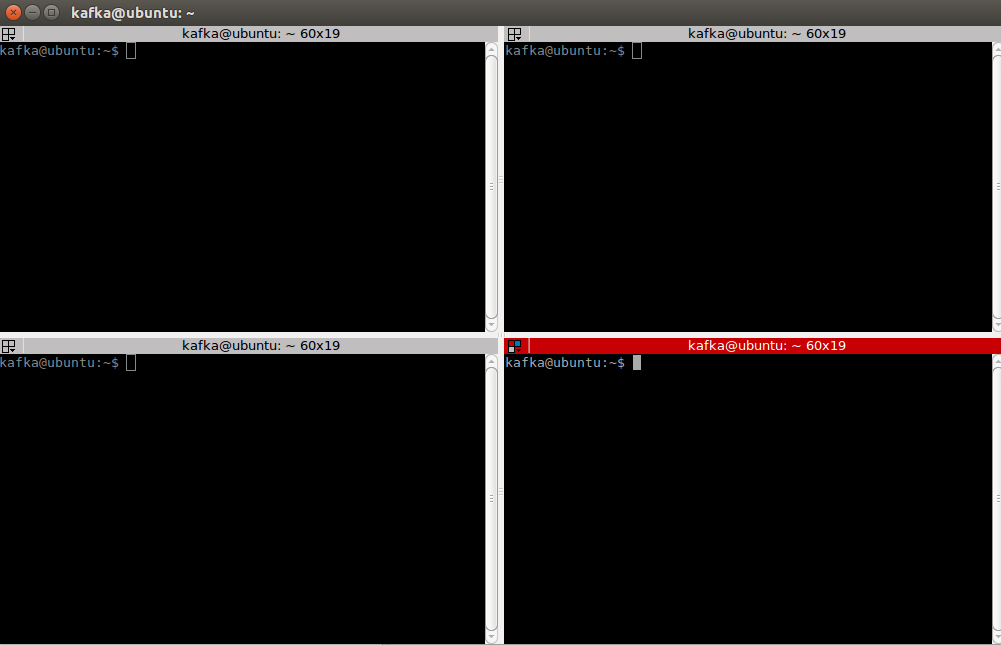
Start Terminator from the Unity menu



Split the screen horizontally – as shown in the next figure - and both screens vertically by right clicking in the window.



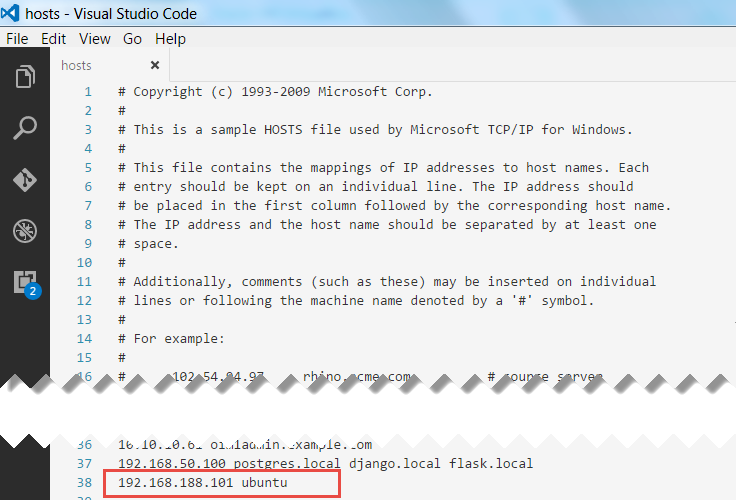
You will end up with four terminal windows.



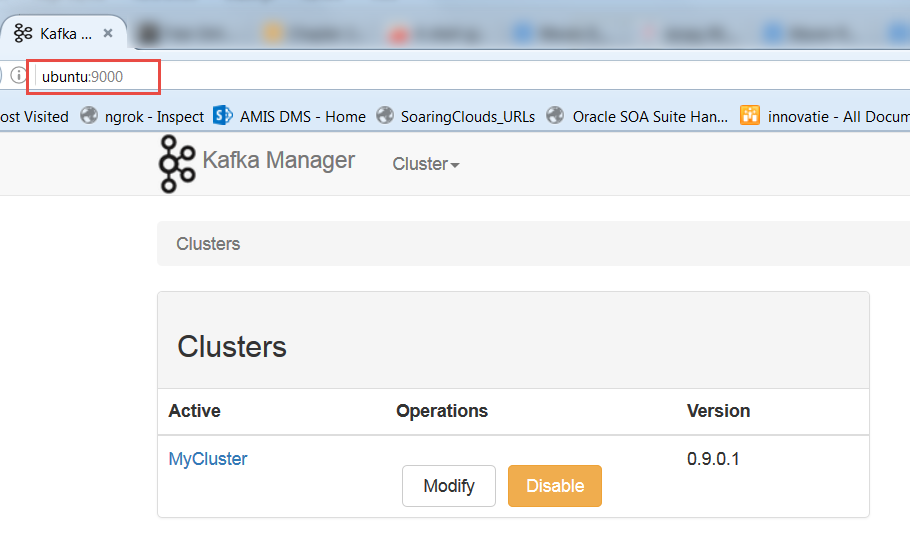
# Optional: Set Networking in VM

If and when you want to access the Kafka instance running inside the VM from your laptop, you need to ensure that you enable network access to the VM. The instructions in this article (AMIS Technology Blog: <https://technology.amis.nl/2017/01/29/network-access-to-ubuntu-virtual-box-vm-from-host-laptop/> ) show what you have to configure in order to be able to access the VM on its own IP address. Note that for this first series of labs, you do not need this. So you can skip this step for now, and optionally return to it later.

After arranging access to the VM, you may want to define a logical host name for the Virtual Machine – by adding an entry to the hosts file: C:\Windows\System32\drivers\etc\hosts, as shows below:



Note: ubuntu is the host name of the VM; to play nice with Zookeeper it seems convenient to use that same name as the host name in our hosts file.



# Getting started with Kafka

In the first terminal window start a Kafka broker:

kafka-server-start /etc/kafka/server.properties

Start in the second terminal window start the kafka manager:

kafka-manager

Start kafkatool in the third terminal window:

kafkatool

Minimize the tool. You will use it at a later time.

Continue in the fourth terminal window

### Create a topic

Create a topic:

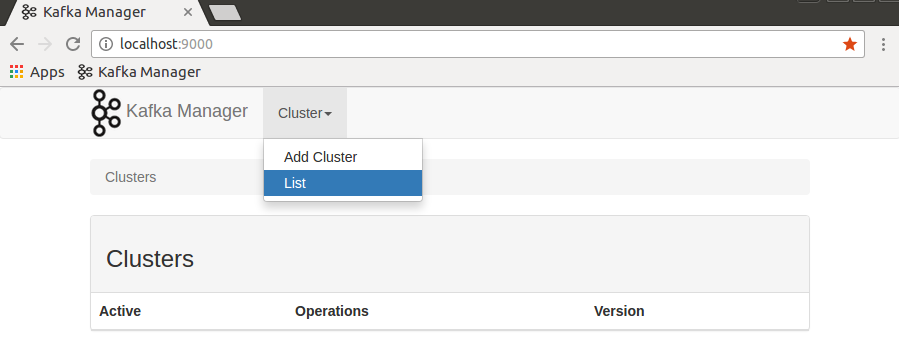
kafka-topics --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic test

Confirm the topic is created:

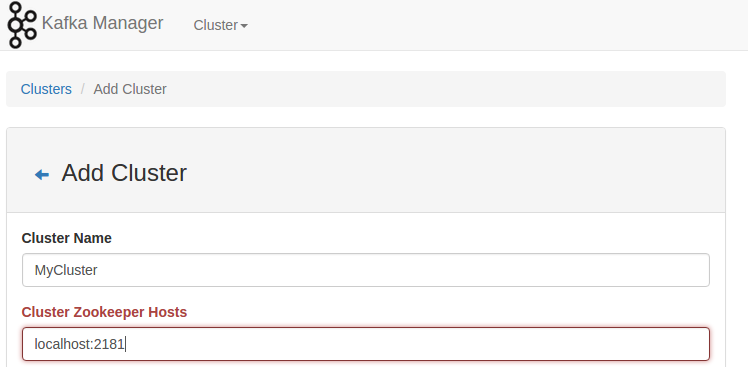
kafka-topics --list --zookeeper localhost:2181

Open a browser and go to <http://localhost:9000> and confirm the topic is created using a webinterface.

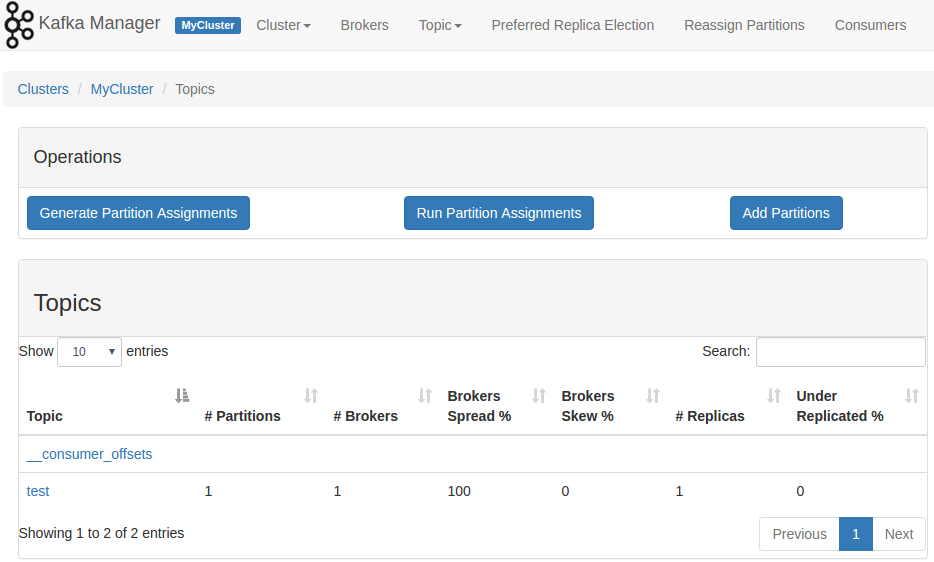
First create a cluster



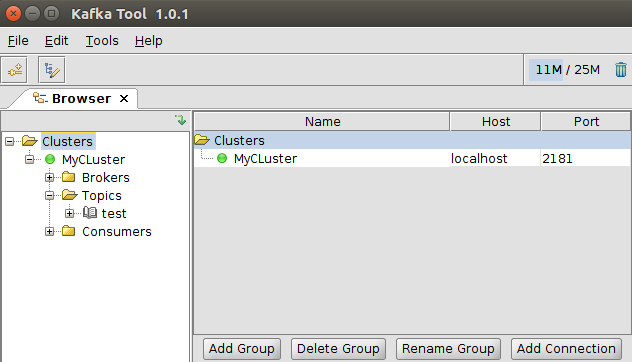
Name the cluster MyCluster and enter localhost:2181 as Zookeeper cluster. Scroll down and click save



Go to the cluster view and determine the topic has been created



Open kafkatool and confirm the topic is created



### Produce a message

Focus on the fourth terminal window again and type:

kafka-console-producer --broker-list localhost:9092 --topic test

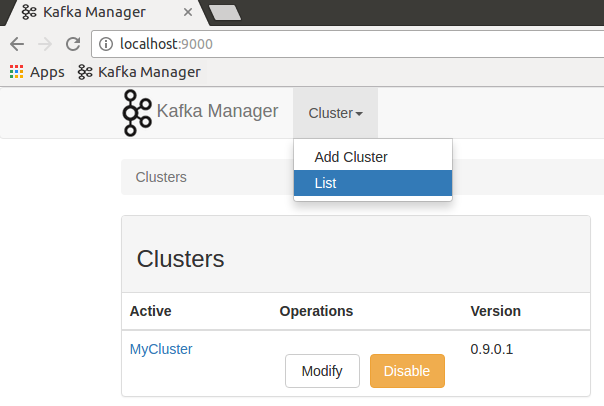
Press enter. No feedback appears; the cursor sits blinking and waiting for your input.

Type a message and press enter:

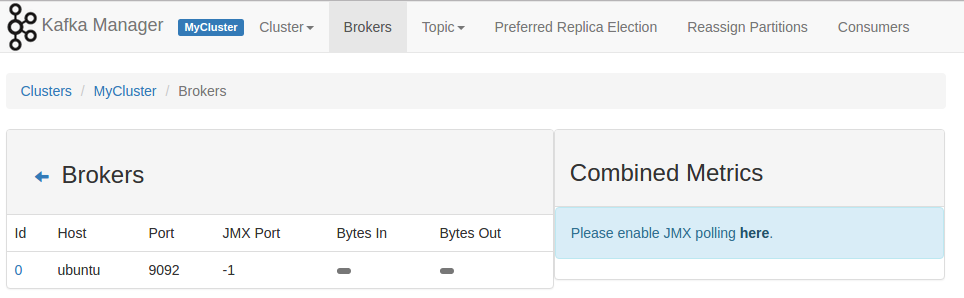
Hello World!

Check the message is created using kafka-manager and kafkatool.

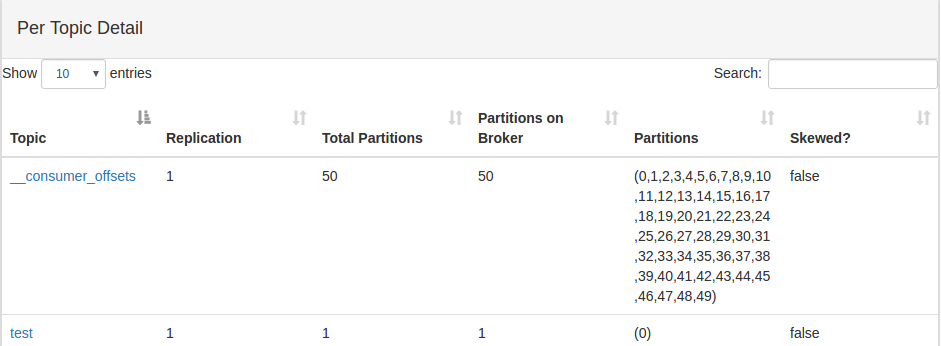
In the kafka-manager, go to Cluster, List and click the cluster you previously created



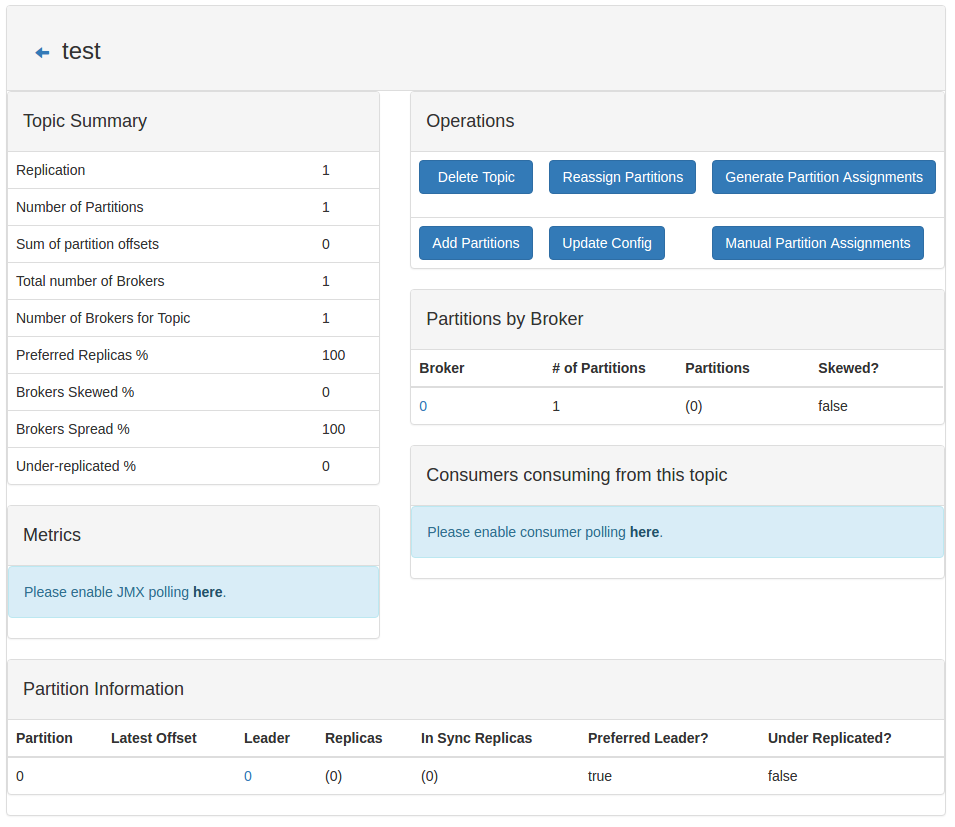
Click brokers, click 0



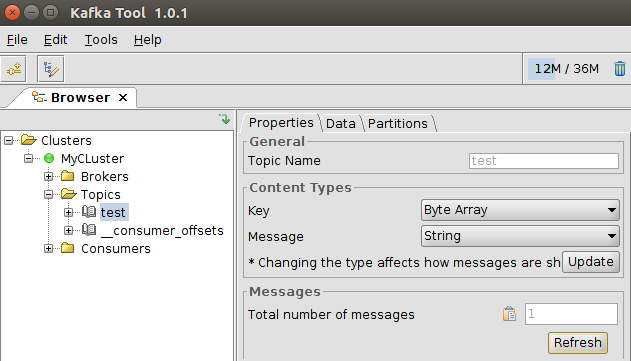
Scroll down and click the test topic



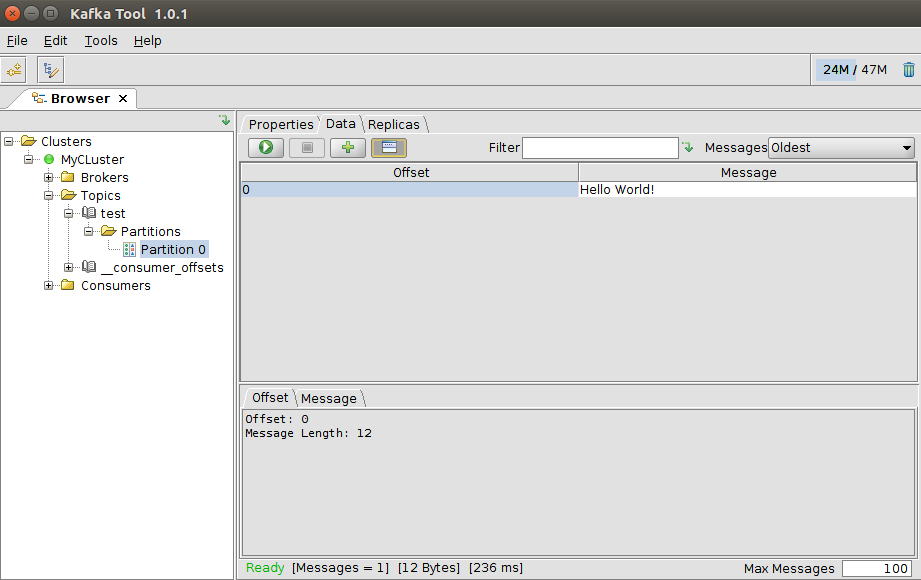
Scroll down again and confirm the latest offset is empty. No consumer has picked up messages yet.



Open kafkatool. Indicate the messages on the topic are string. Click the update button. Click the refresh button to view how many messages are present on the topic. Confirm the number is 1.



Open the partition under the test topic. Click the green play button. Confirm the Hello World! message has arrived.



### Consume a message

Consume a message

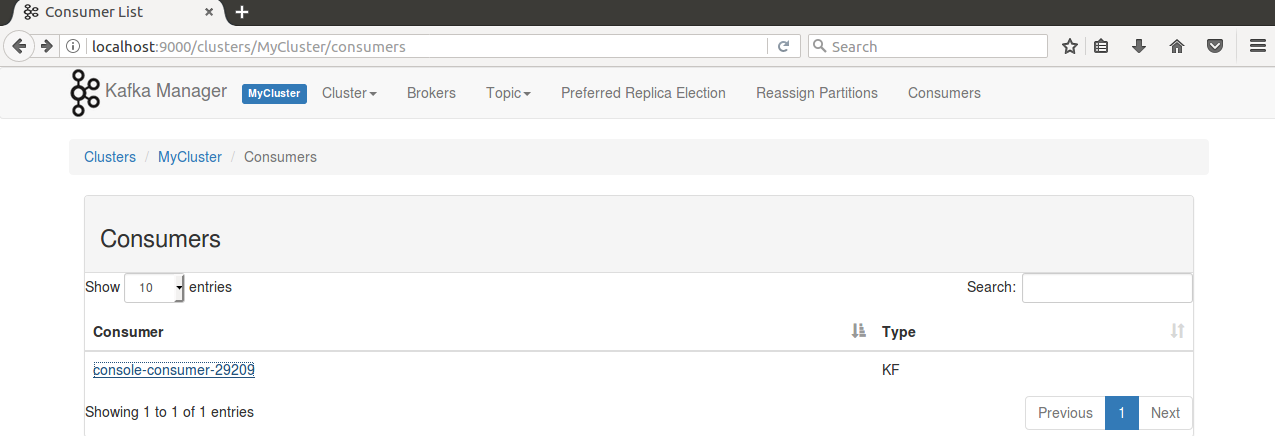
kafka-console-consumer --bootstrap-server localhost:9092 --topic test --from-beginning



Confirm that the previously posted message has been consumed.

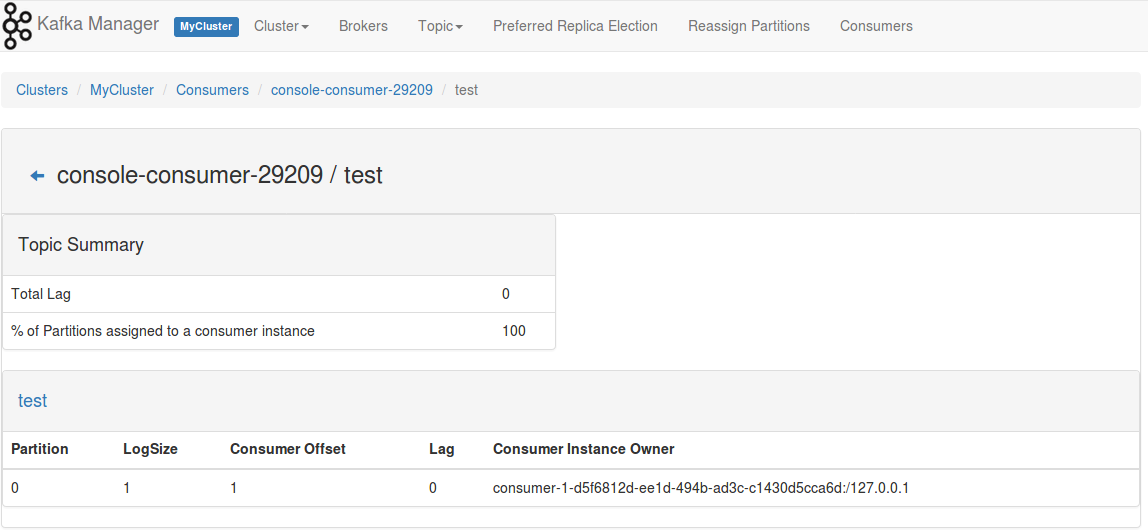
Look at kafka-manager: <http://localhost:9000>

Click on MyCluster, Consumers



Confirm the console consumer is visible

Click the consumer name. Next click the topic test.



Confirm that this consumer has an offset of 1.

End the console consumer by pressing ctrl-C in the terminal window where the consumer is running. End the console producer by pressing ctrl-C in the terminal window where the producer is running.

### Availability

Start a consumer

kafka-console-consumer --consumer-property group.id=TestConsumer --bootstrap-server localhost:9092 --topic test

Start a producer

kafka-console-producer --broker-list localhost:9092 --topic test

Send a message from the producer

Hello World!

Confirm the consumer receives the message.

End the consumer by pressing ctrl-C in the consumer console window

Produce a new message

Hello World!

Start the consumer

Confirm the consumer receives the message it had missed while being down.

# Oracle Stream Analytics (minimal sample)

Start Stream Explorer

cd ~/Oracle/Middleware12212osa/Oracle\_Home/user\_projects/domains/defaultdomain/defaultserver

startwlevs.sh

Start a Kafka broker

kafka-server-start /etc/kafka/server.properties

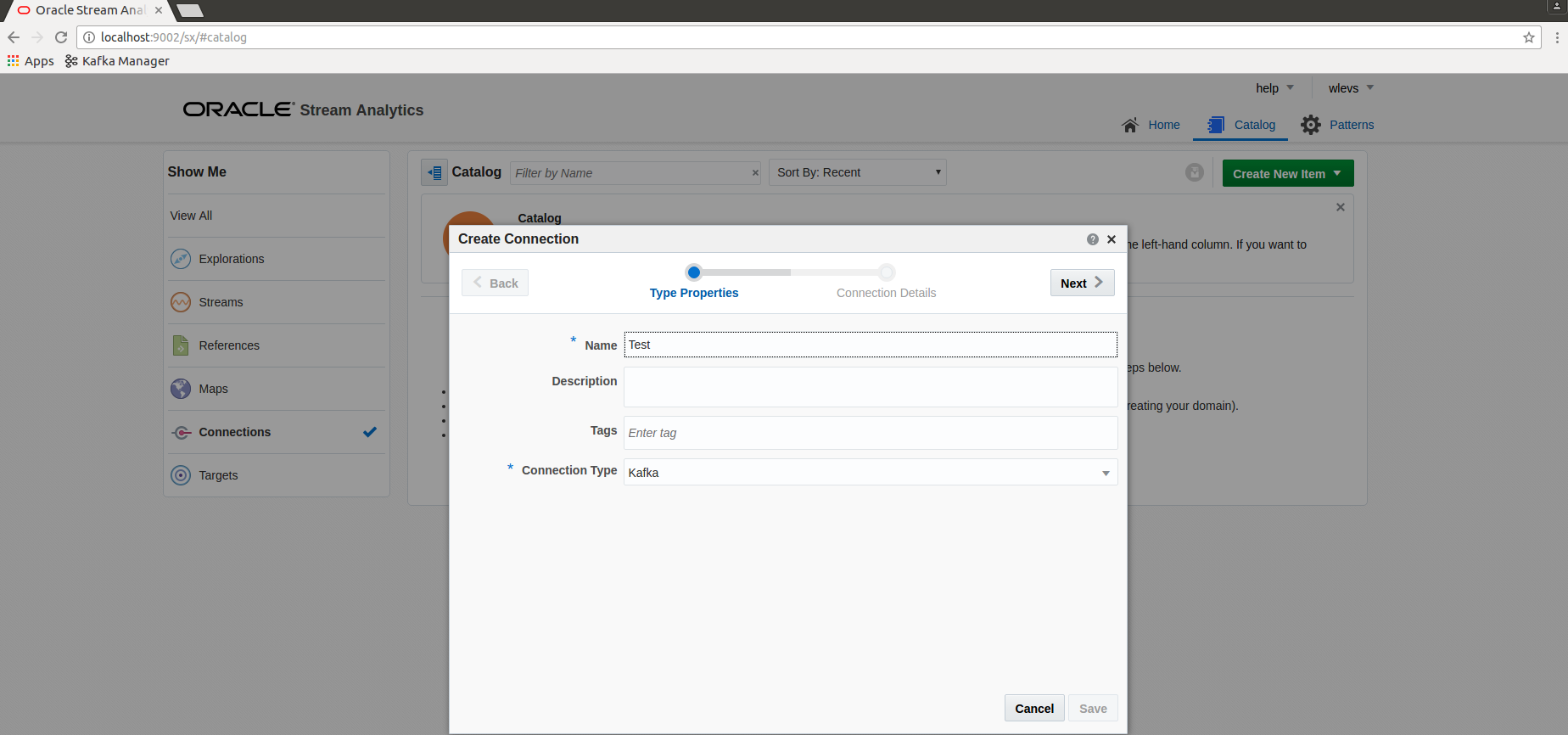
Start a console producer

kafka-console-producer --broker-list localhost:9092 --topic test

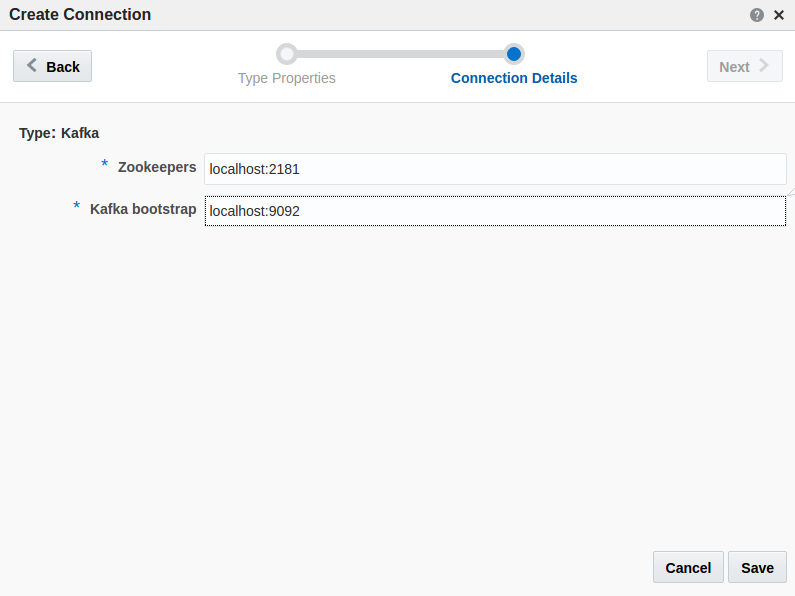
Start a browser and go to <http://localhost:9002/sx>. Login with user wlevs, Welcome01

## Create a connection

Go to Catalog, create New Item, Connection and fill in the fields as shown below



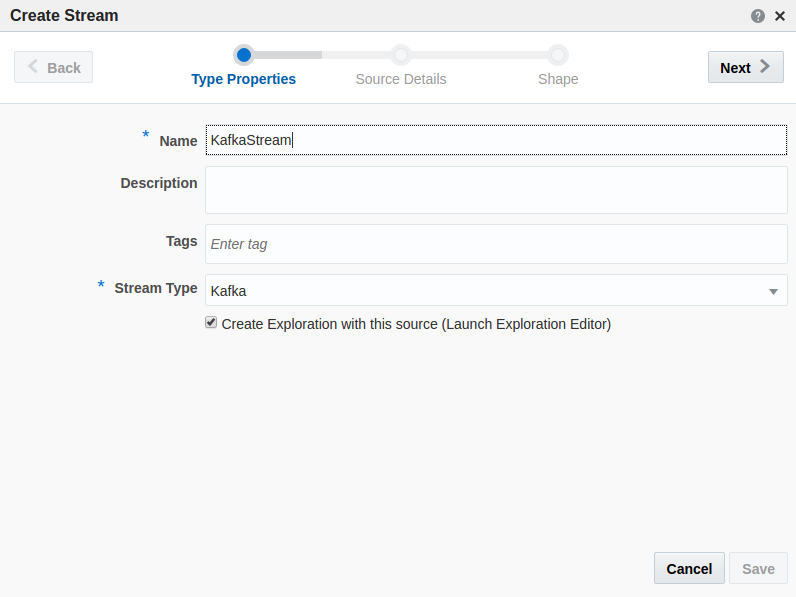
Click next and fill in the Zookeeper URL and bootstrap broker.

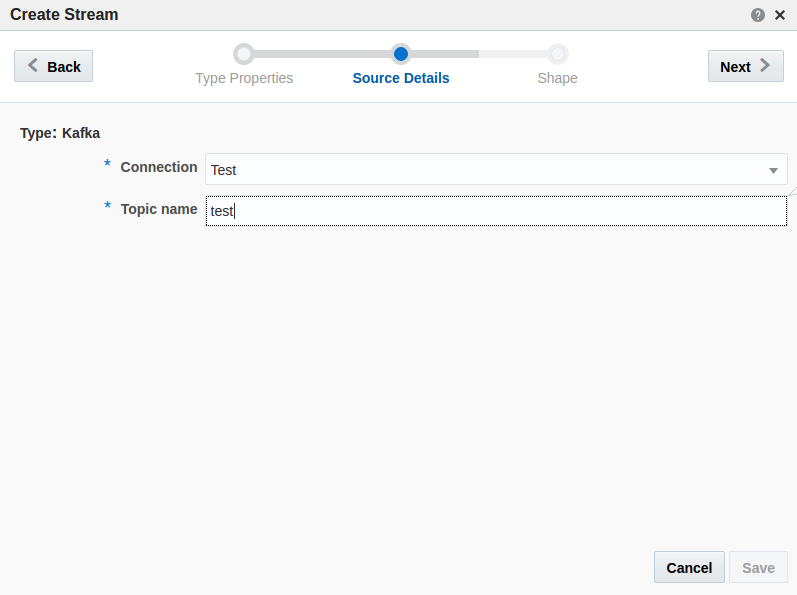


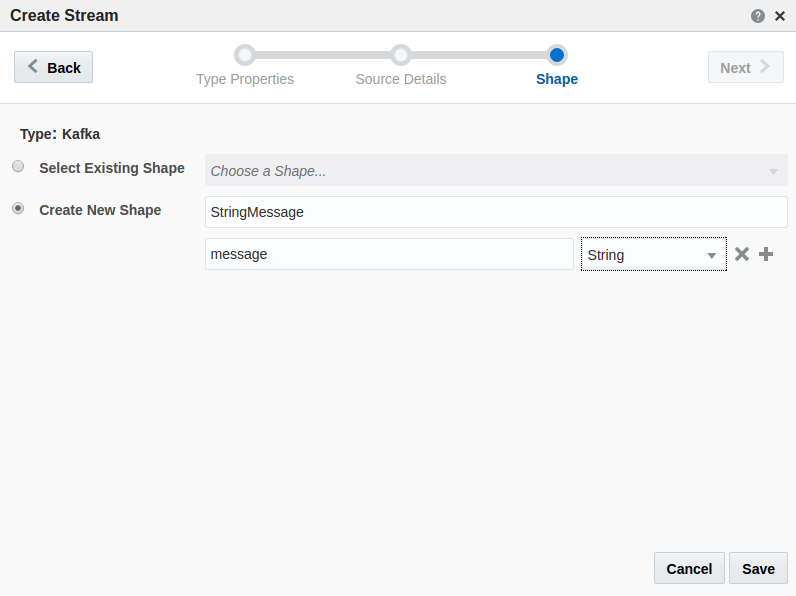
Click Save.

## Create a Stream

Create a stream using the connection.

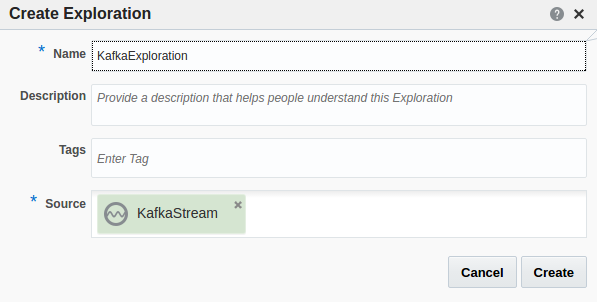






## Create an exploration

Create an exploration based on the stream



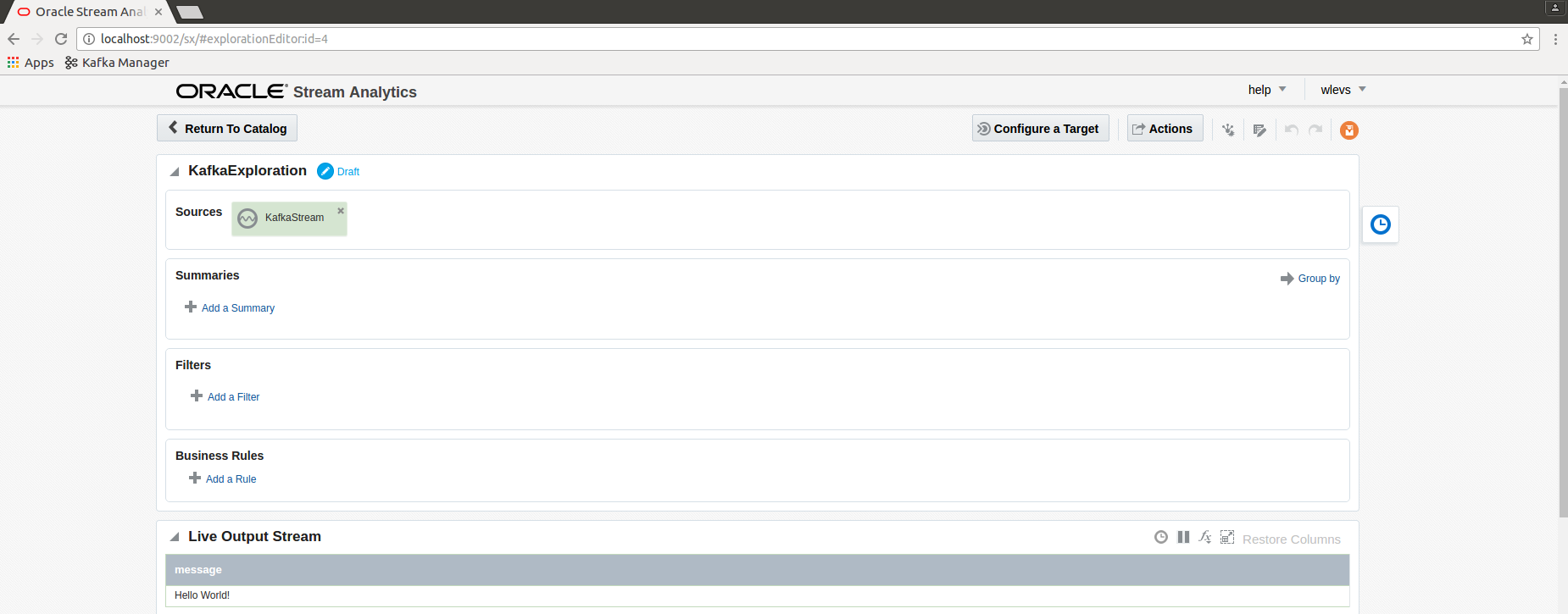
Click the create button

## Confirm OSA receives messages

Next produce a message. Go to the console of the producer and type:

{"message":"Hello World!"}

Confirm you can see the message in your exploration.



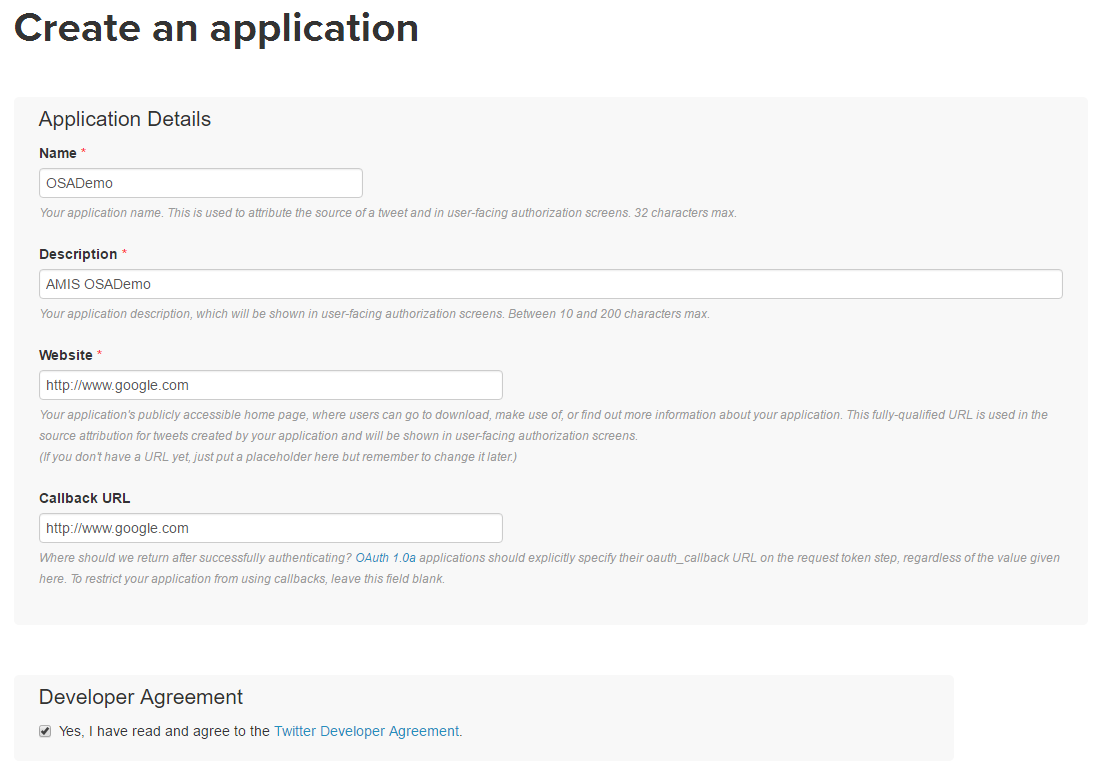
# OSA with Twitter and Service Bus

Start the Kafka broker

kafka-server-start /etc/kafka/server.properties

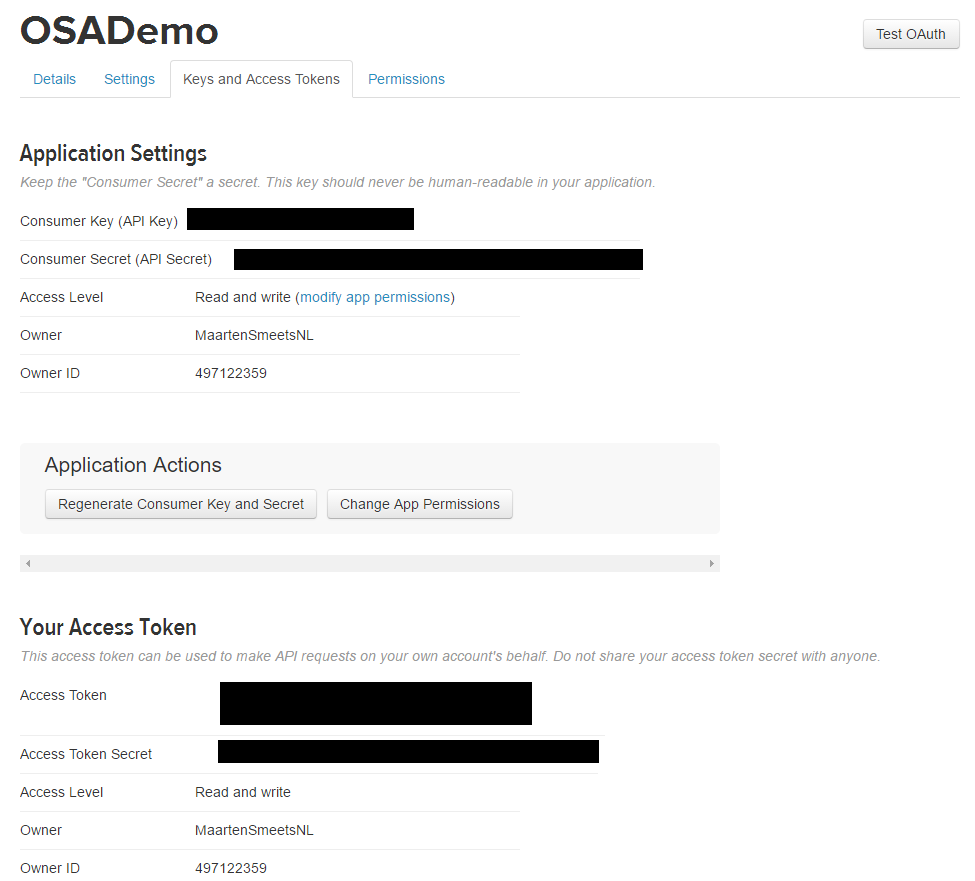
## Obtain requires data from Twitter

Go to dev.twitter.com, My app and create a new application



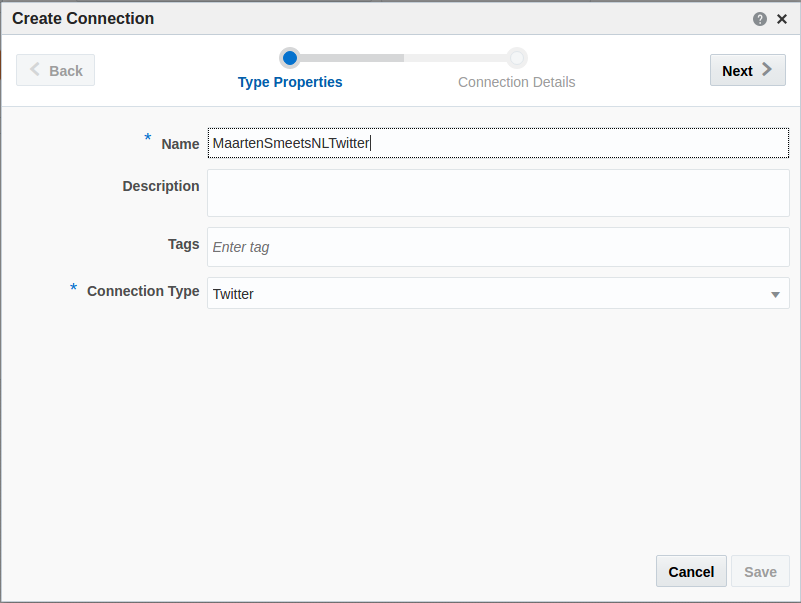
On the next page obtain:

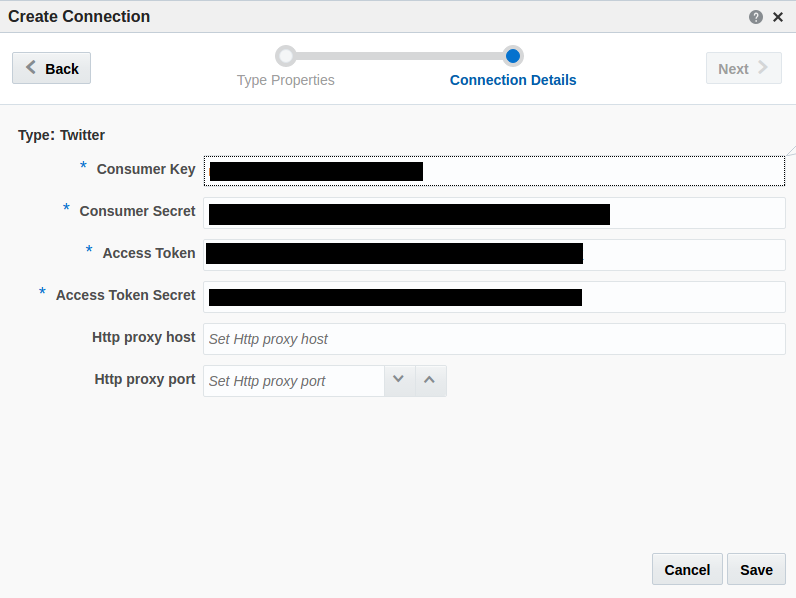
Consumer Key, Consumer Secret, Access Token, Access Token Secret



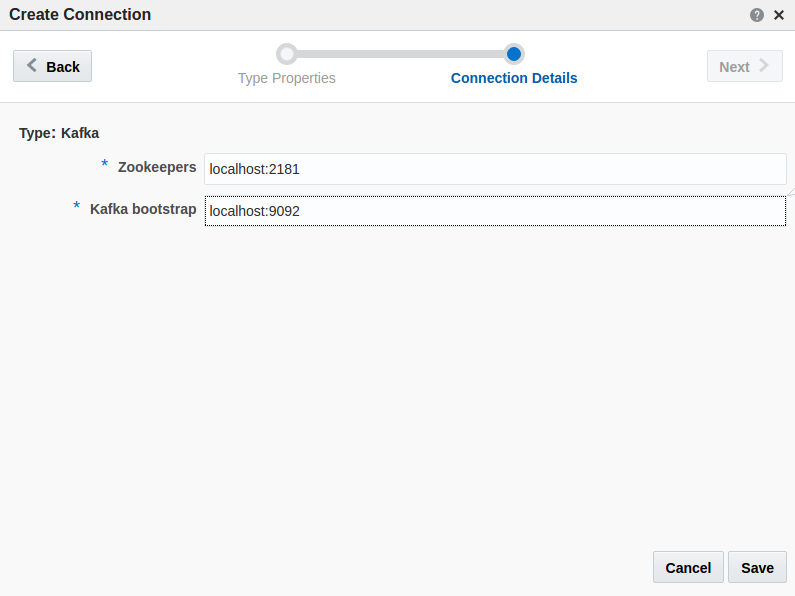
## OSA Configuration

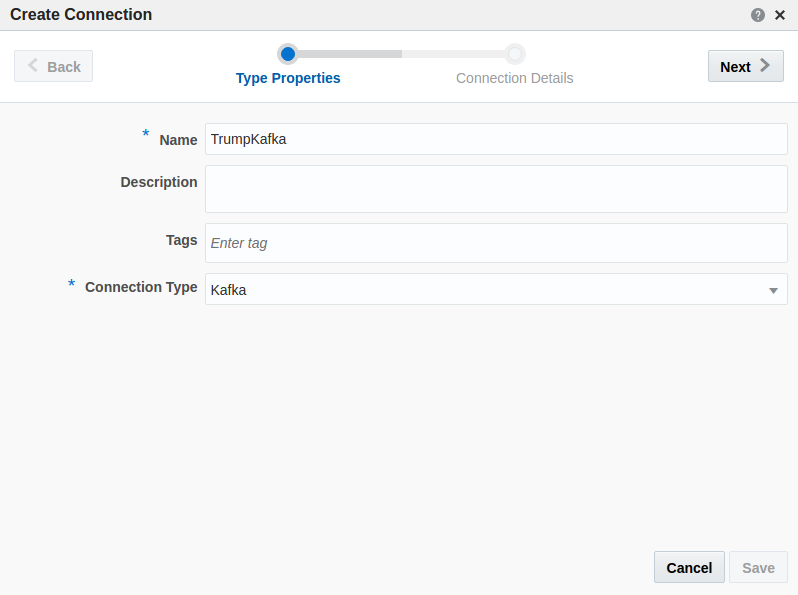
Create a new Twitter connection



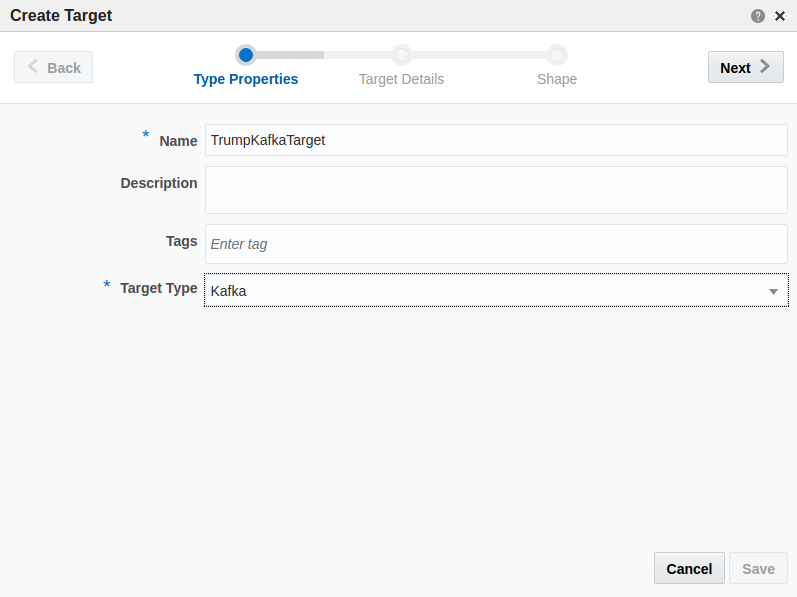


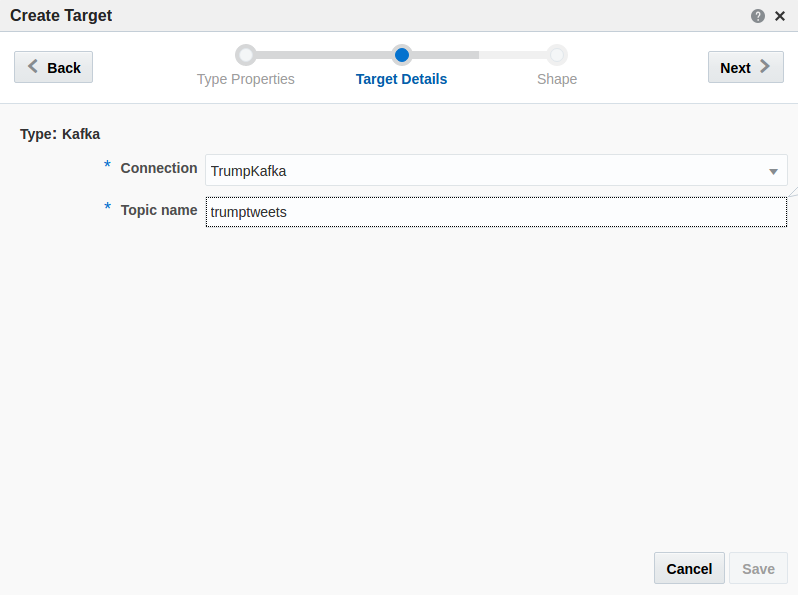
Create a new connection to Kafka or use an existing one.

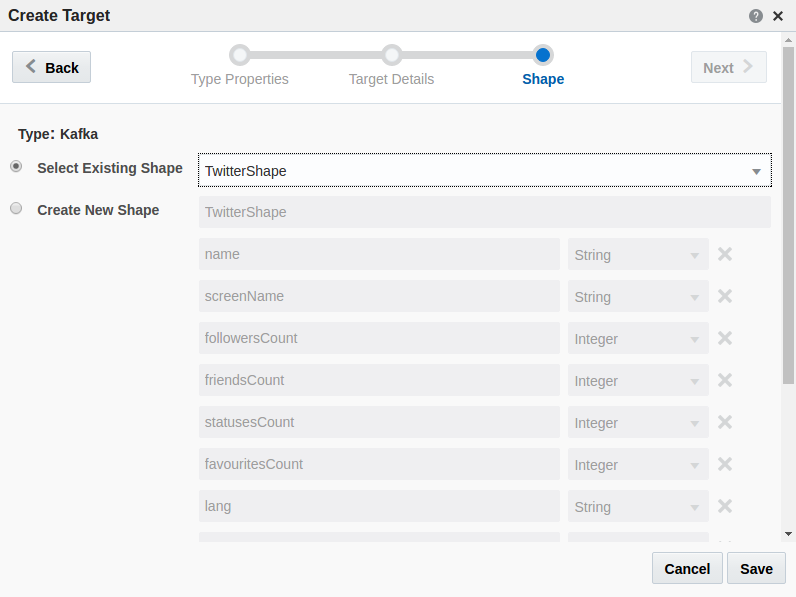




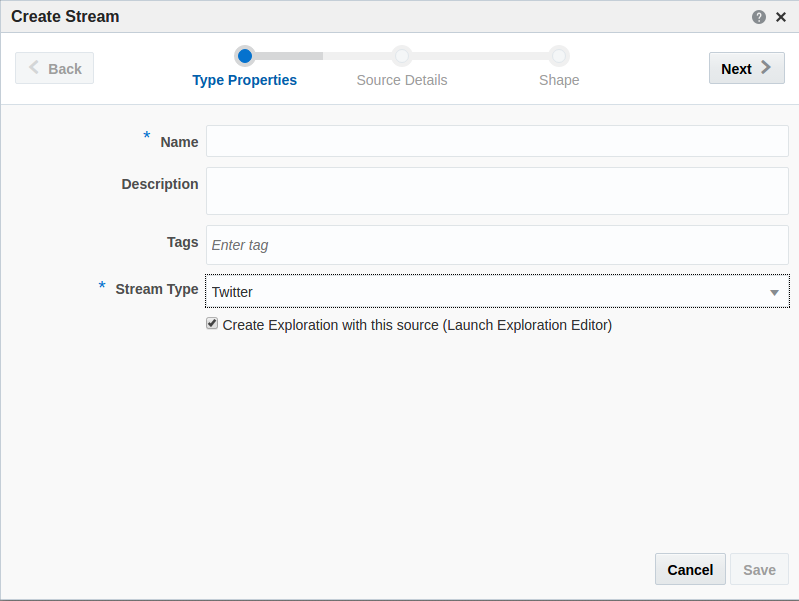
Create a new target

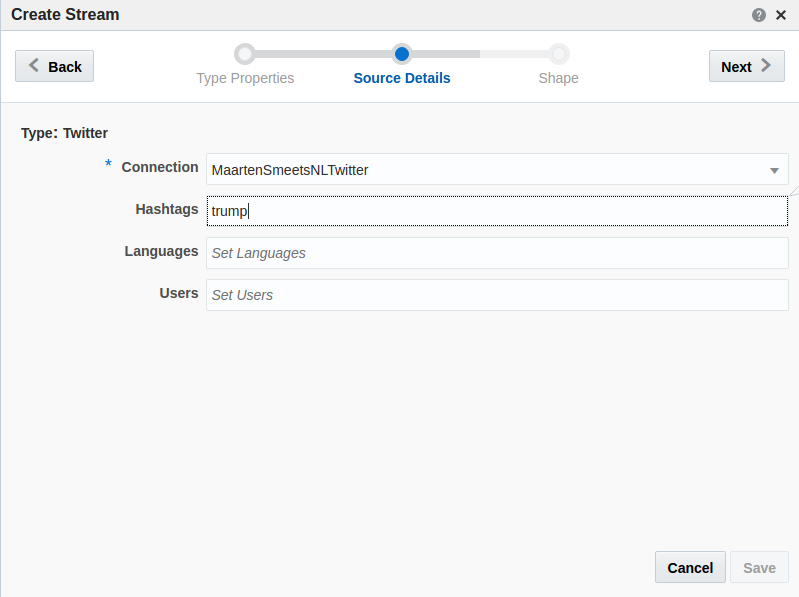


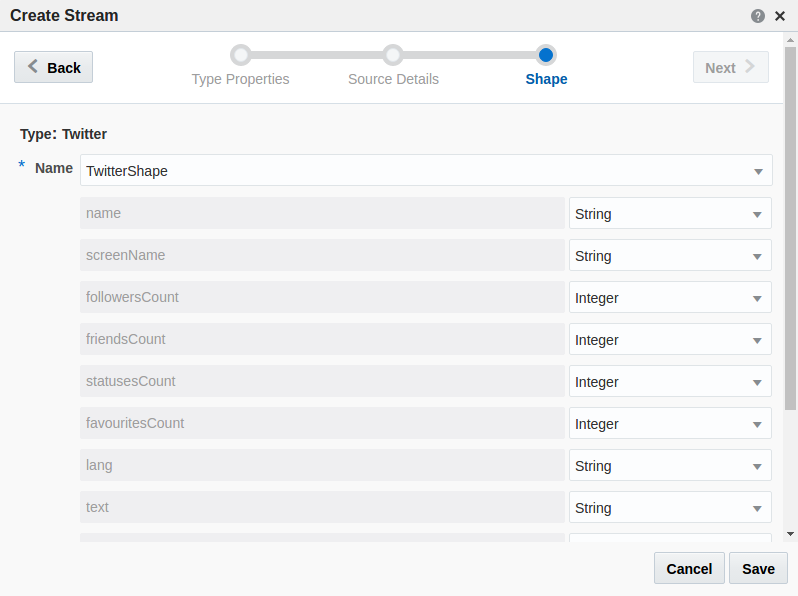




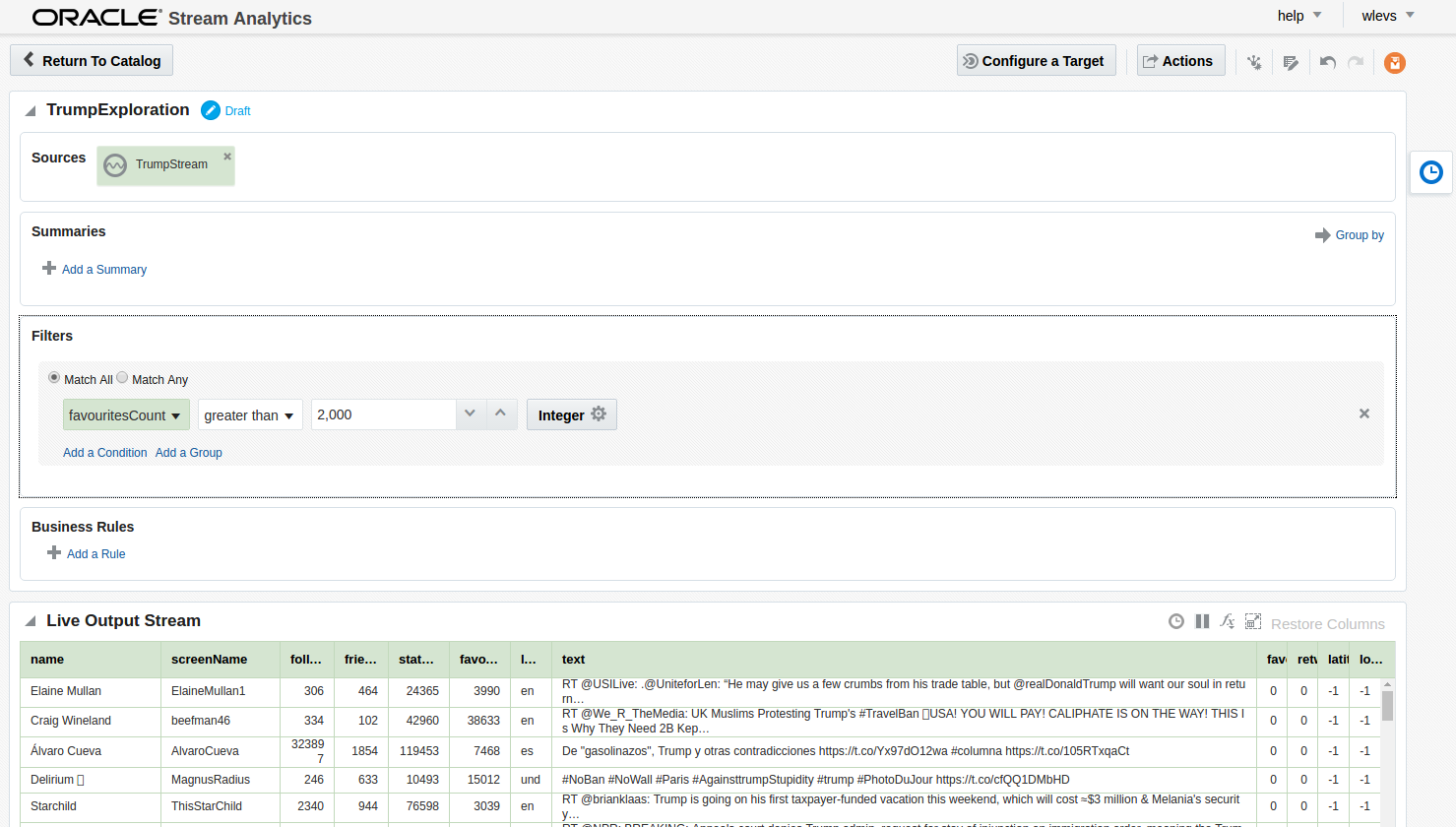
Create a new Stream based on the Twitter connection



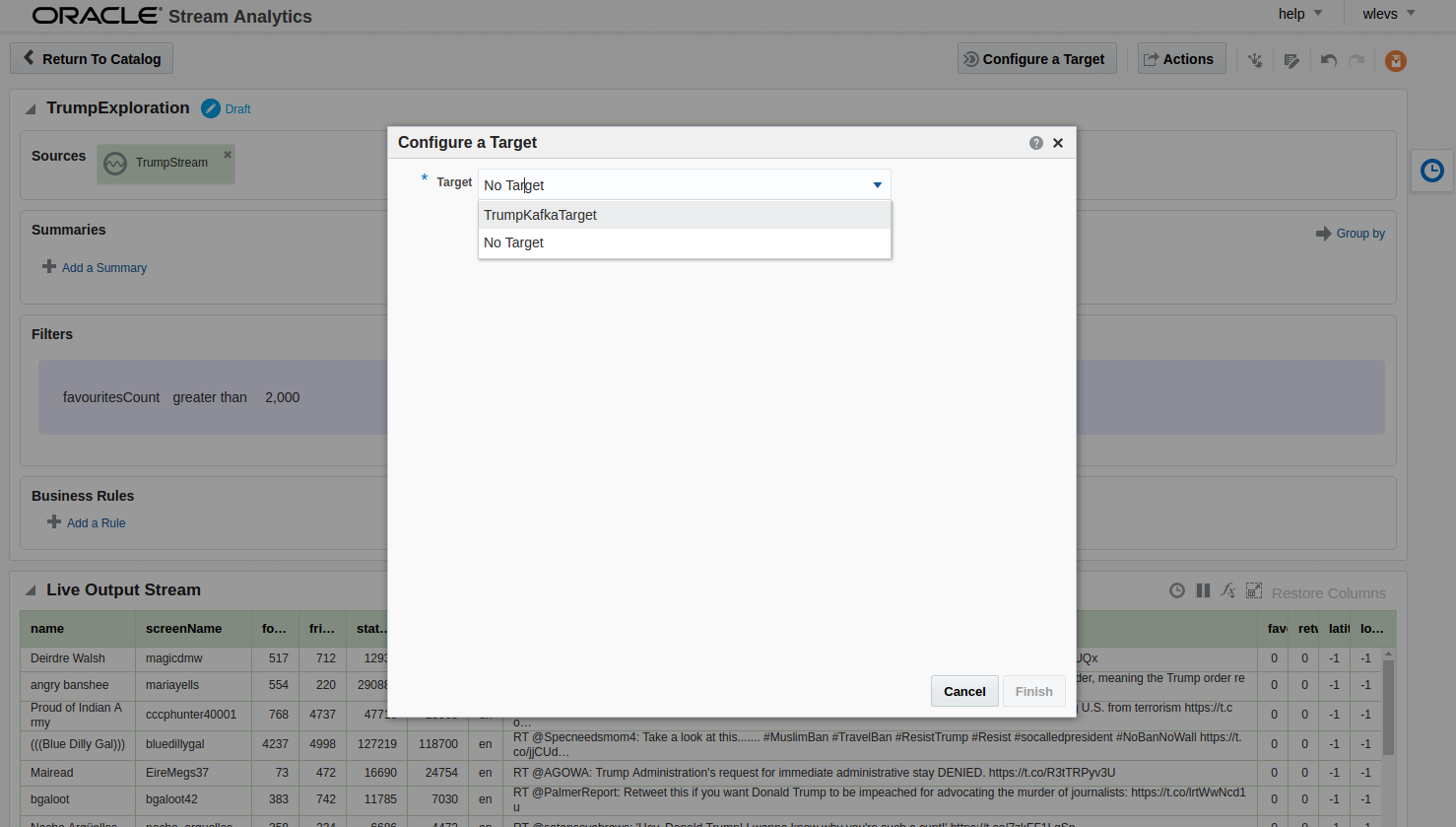




Create an exploration based on the stream

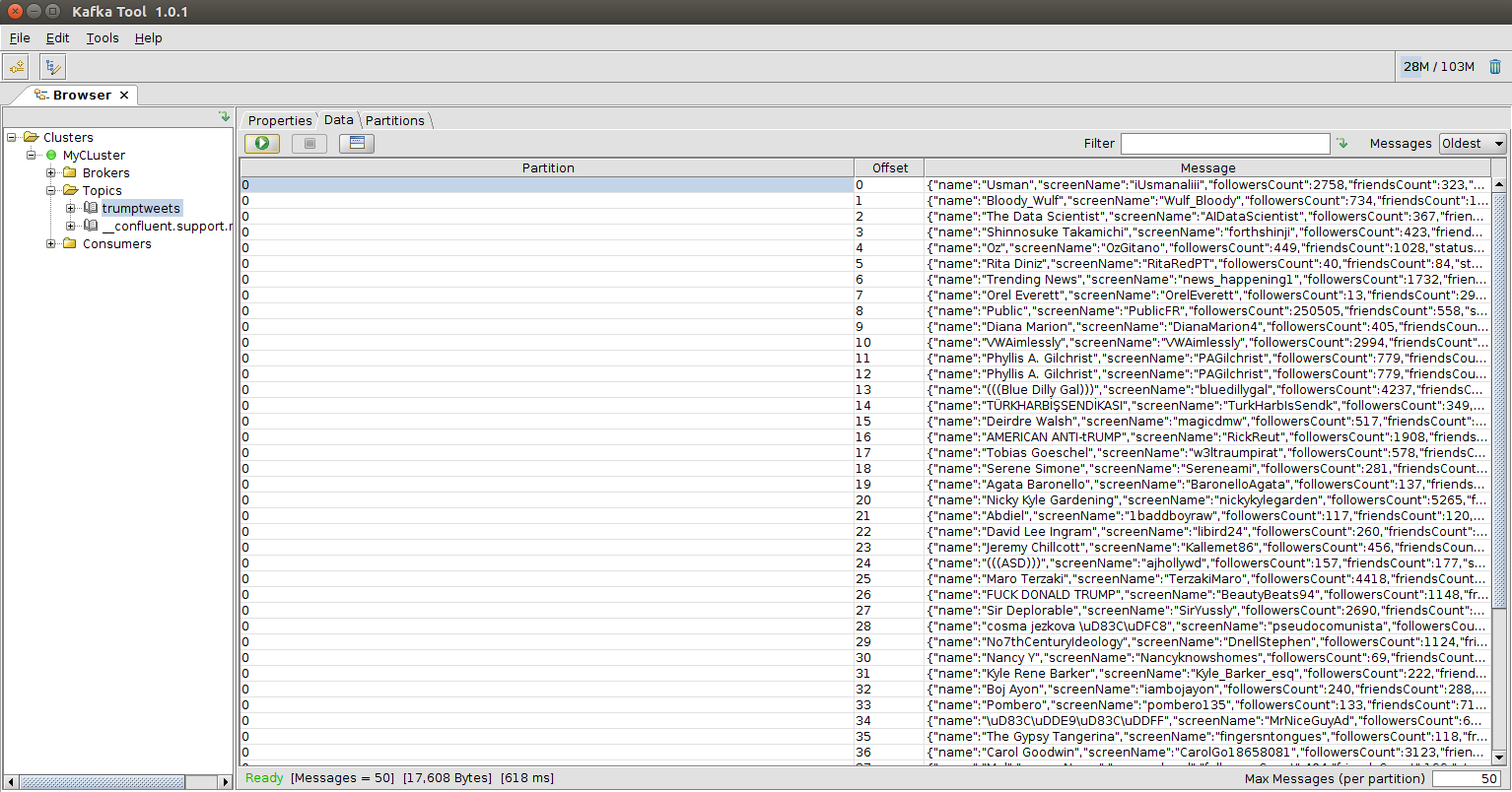


Set the target



Publish the Exploration

Confirm messages are put on the topic with kafkatool



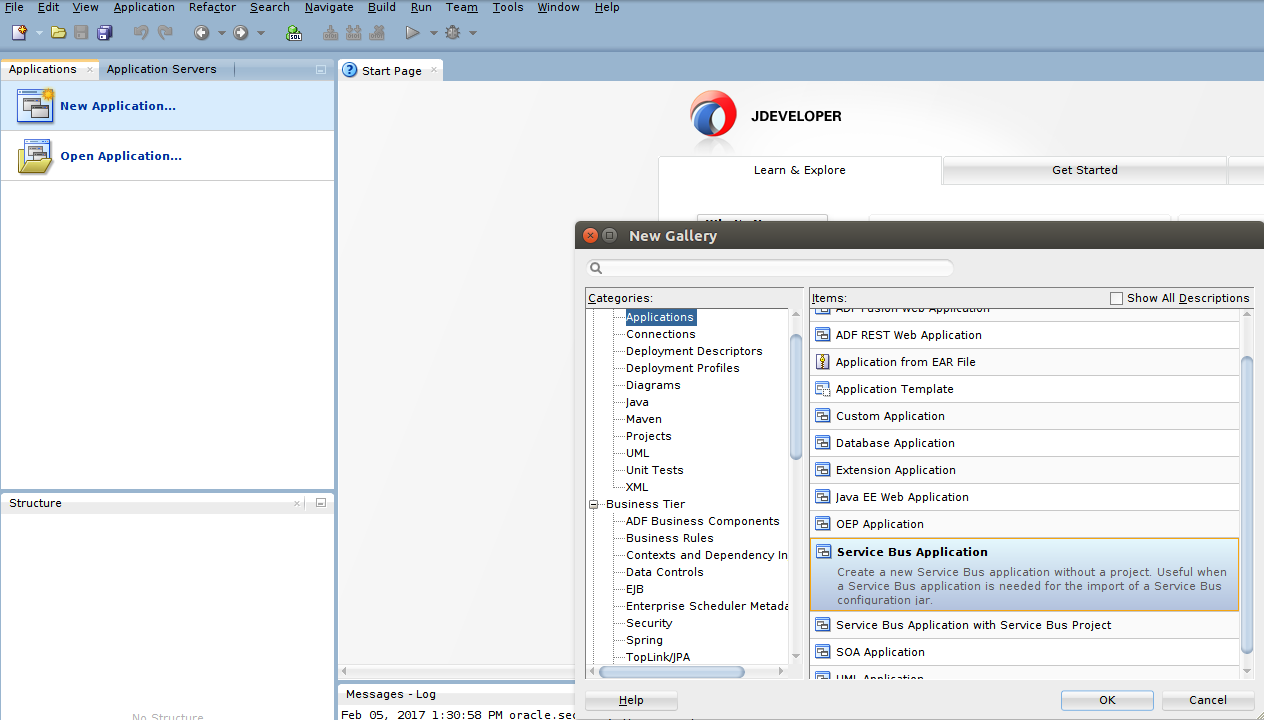
## Create a Service Bus process

Start JDeveloper

cd /home/kafka/Oracle/Middleware12212soa/Oracle\_Home/jdeveloper/jdev/bin/

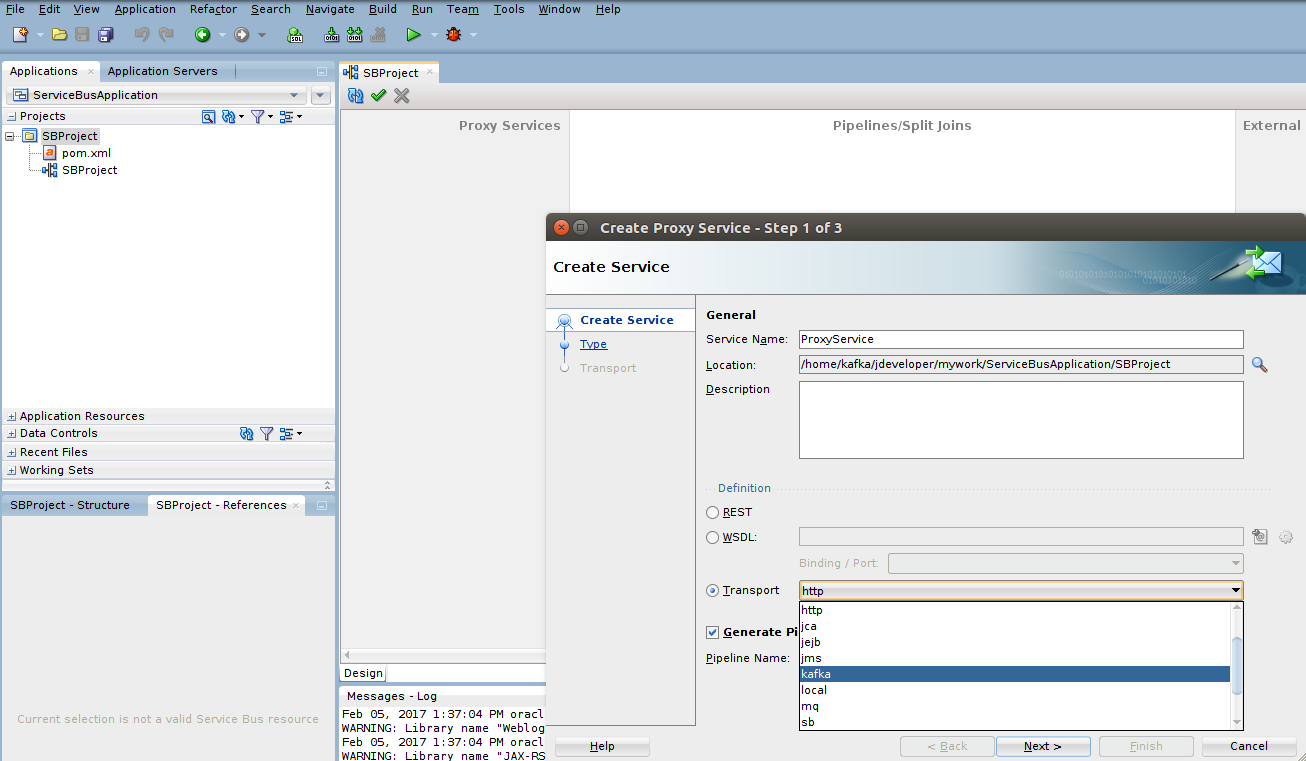
./jdev

Create a new application



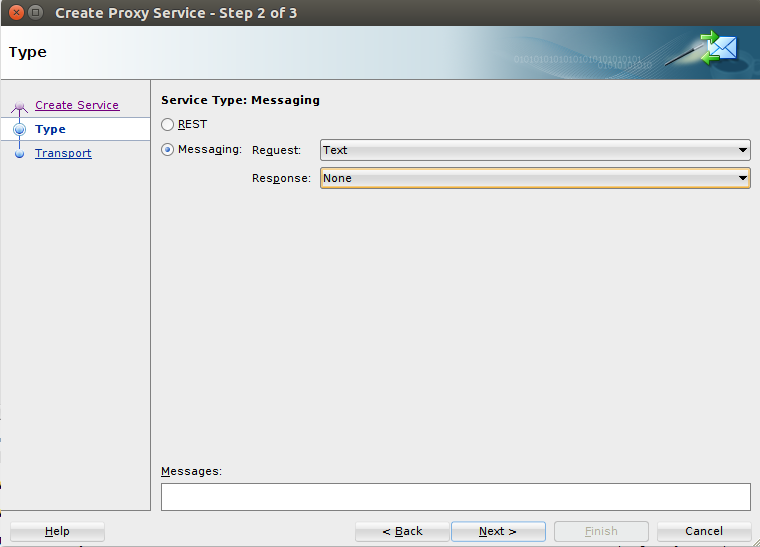
Add a Service Bus project to the application

Add a new proxy service. The Kafka transport cannot be selected from the component palette

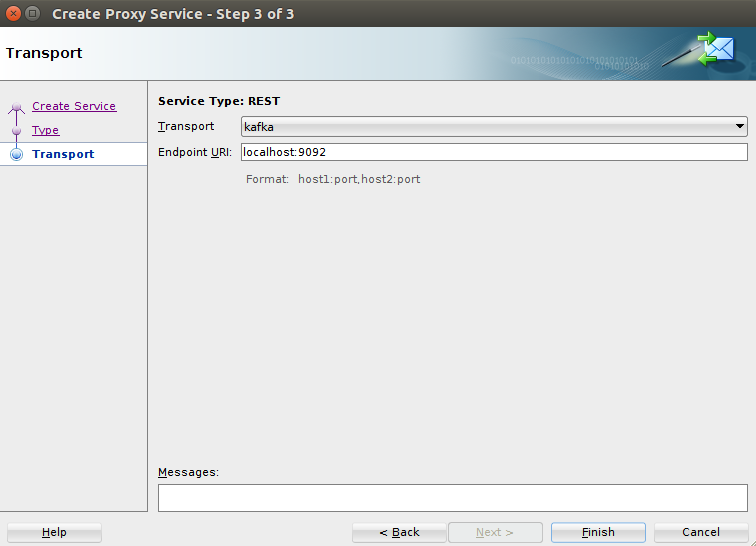


Define the transport

In the Type screen specify messaging

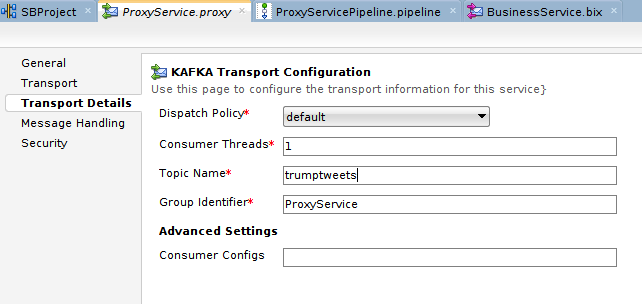


Specify the broker



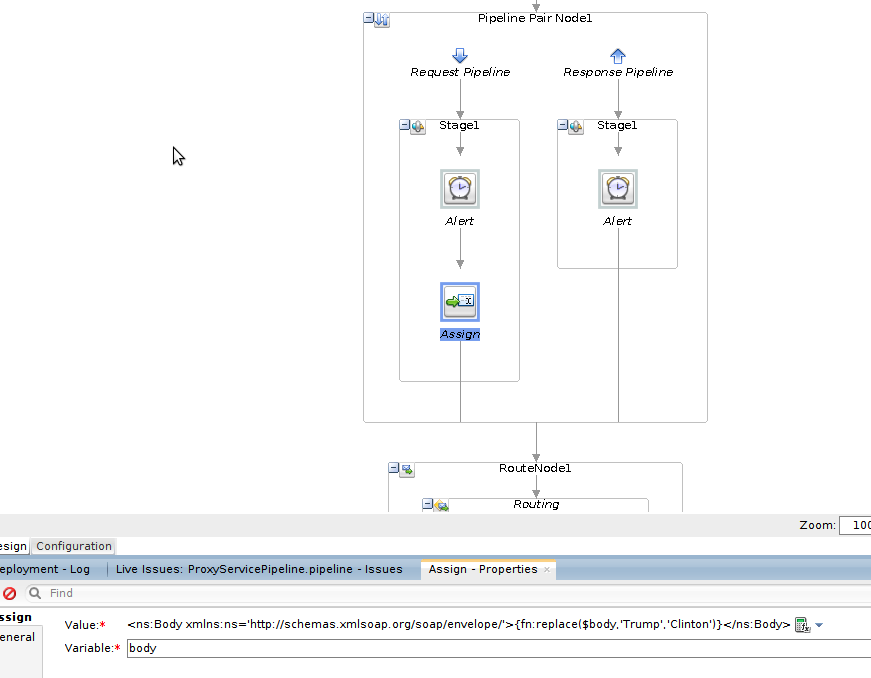
Click finish

Update the transport details of the Proxy Service and specify the topic name



For outbound, create a Kafka Business Service and have it put messages on topic clintontweets.

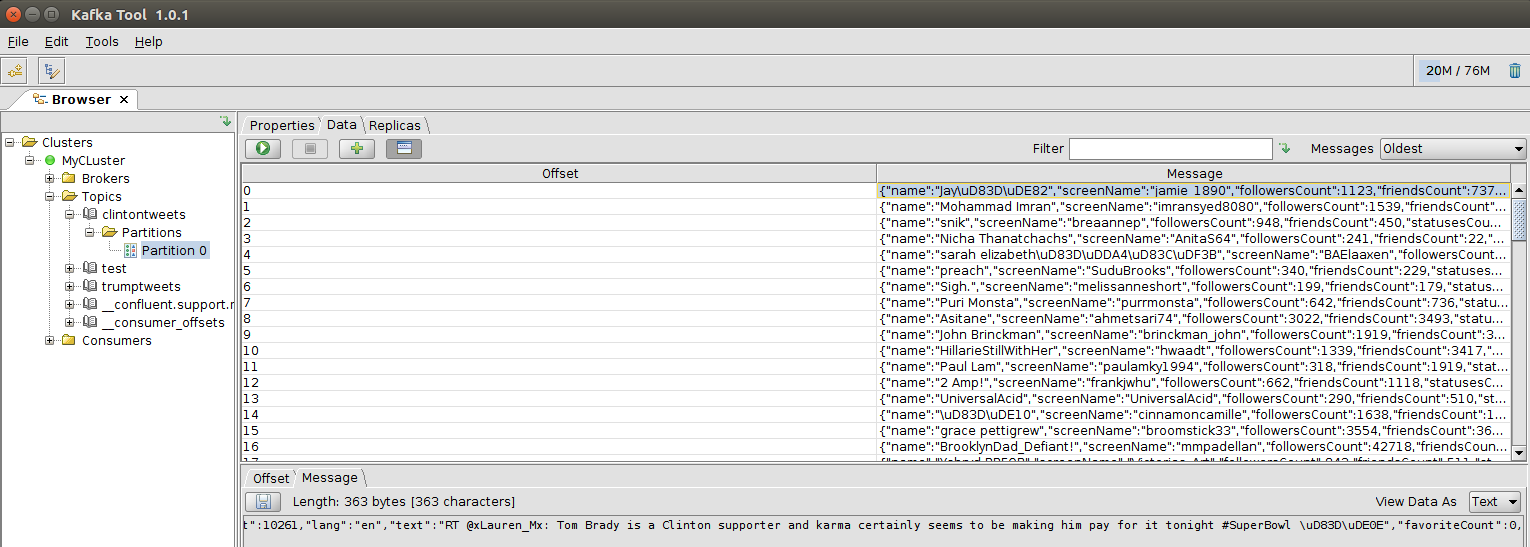
Create an assign operation in the Service Bus (the pipeline alerts in the below image are not required):



The value of the Assign should be:

<ns:Body xmlns:ns='http://schemas.xmlsoap.org/soap/envelope/'>{fn:replace($body,'Trump','Clinton')}</ns:Body>

Check in Kafkatool the clintontweets topic



# References

Kafka OSA:

<https://www.rittmanmead.com/blog/2016/07/stream-analytics-processing-kafka-oracle-stream-analytics/>

Network access

<https://technology.amis.nl/2017/01/29/network-access-to-ubuntu-virtual-box-vm-from-host-laptop/>