



Workshop Streaming Data with Amazon Kinesis

SIG Event Architecture 15 mei 2018

Inhoud

Prerequisites.....	4
Files	4
Reference	4
Software installation.....	5
Python 3.6.....	5
Pycharm Community Edition	5
Install AWS CLI	6
Using pip.....	6
Post install	6
Configuration	7
AWS console	11
Kinesis Datastreams	15
Kinesis Analytics	18
Create analytics application.....	18
Connect streaming data	19
Data producing.....	21
Running the local REST API service.....	24
Command line	24
Using PyCharm.....	25
Toubleshooting	27
Run streaming producer	27
Discover schema.....	28
Stopping an Kinesis Analytics Application	29
Performing some analytics	29
Simple streaming	29
Select aggregated data (Using GROUP BY).....	30
Select Most Frequently Occurring Values (top-K items).....	32
Reference data	33
Create a S3 bucket	33
Adding file to S3 bucket.....	36
IAM role.....	38
Add Reference Data Source to the Application Configuration.....	41
Using the reference data	43

Offloading data	43
Dataflow	43
Offload to S3.....	43
Stream Consumer.....	49

Prerequisites

For following this workshop you need the following:

- An Amazon AWS account with access to Kinesis, EC2, S3
 - Use your Amis AWS account if you have any
- AWS SDK for Python for running the sample code
- AWS command line interface CLI (see for more information: <https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-welcome.html>)
- For running Python you need Python 3.6.4 or up. Python can be found at <https://www.python.org/>
NOTE: When installing Python: Select Add Python to the Path option!
- An IDE for Python
 - for example: [Pycharm](#), download at <https://www.jetbrains.com/pycharm/download/> or use the version from the share

Files

Any support files can be found in the Amis Github repository: <https://github.com/AMIS-Services/sig-streaming-data-may-2018>.

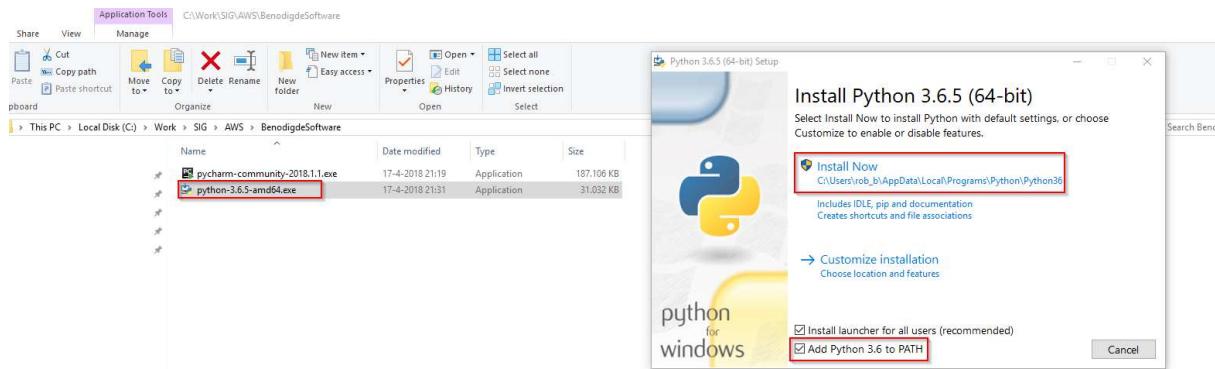
Reference

- Kinesis Developer Guide:
<https://docs.aws.amazon.com/streams/latest/dev/kinesis-dg.pdf>
- Kinesis Analytics Developer Guide:
<https://docs.aws.amazon.com/kinesisanalytics/latest/dev/analytics-api.pdf>

Software installation

Python 3.6

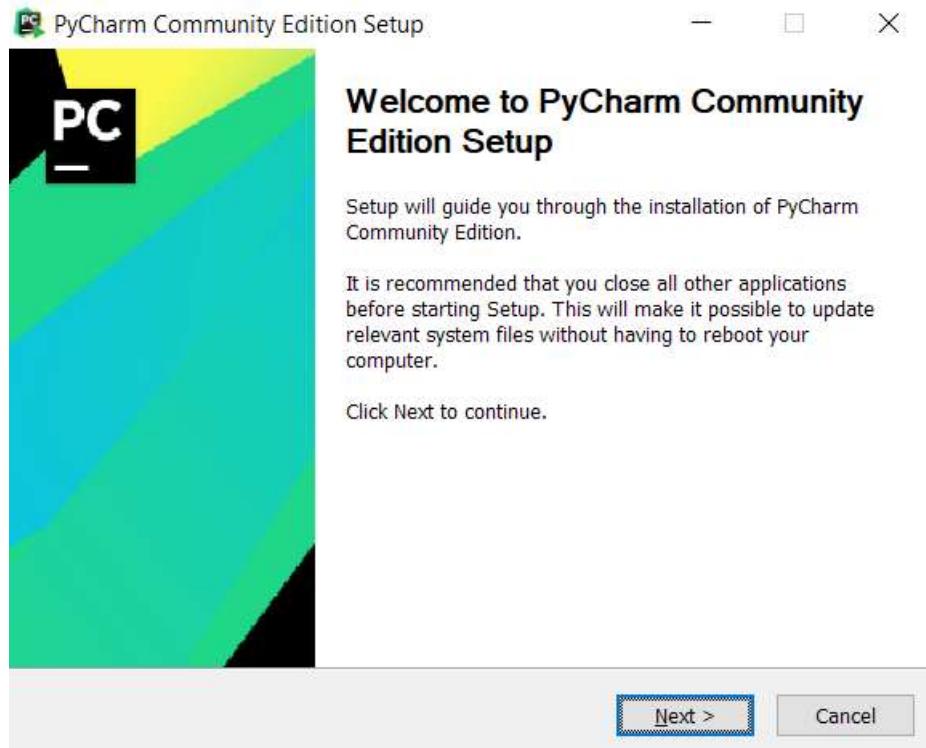
1. Download *python-3.6.5-amd64.exe*

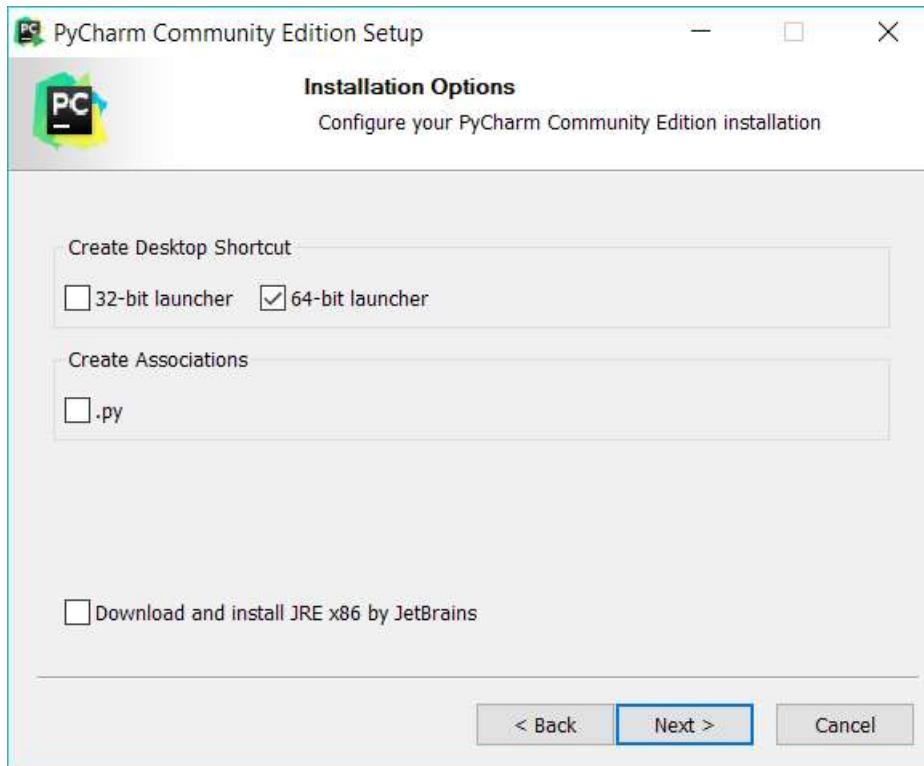


2. Run the installer *python-3.6.5-amd64.exe*. select the **Add Python 3.6 to PATH** checkbox!
3. Choose *Install Now*

Pycharm Community Edition

1. Run the install *pycharm-community-2018.1.1.exe* and choose the 64-bit launcher





Optionally check .py to start PyCharm when double clicking op *.py files.

Just press Next, select the directory and install.

Install AWS CLI

Using pip

The primary distribution of the AWS CLI is by using pip (the Python package installer). If you want to use this you need Python 2.6.5 or up or Python 3.3 or up. Just take the latest 3.6.x version.

If you have Python and pip installed, open a command shell (cmd or powershell on Windows) and type:

```
pip install awscli --upgrade --user
```

The `--upgrade` option tells pip to upgrade any requirements that are already installed. The `--user` option tells pip to install the program to a subdirectory of your user directory to avoid modifying libraries used by your operating system.

Post install

After installation you need to add the AWS CLI to your command line path. Follow the instructions here if you do not know how:

- Linux – [Adding the AWS CLI Executable to your Command Line Path](#)
- Windows – [Adding the AWS CLI Executable to your Command Line Path](#)
- macOS – [Adding the AWS CLI Executable to your Command Line Path](#)

Type: `aws --version` to check if the installation was successful.

Type: `pip install awscli --upgrade --user` to upgrade an existing installation.

Type: `pip uninstall awscli` to uninstall the AWS CLI.

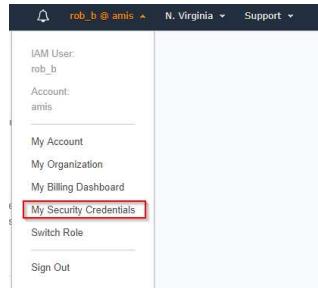
Configuration

For the AWS Credentials, you must login to the Amazon Console and go to IAM (if you have access to it).

We will provide the access key and secret access key. Most users will not have access to the IAM console to do this.

If you do have access: these are the steps to generate the access key:

Select in the topbar your name → Select My Security Credentials



Select Users from the left menu → select your user name from the list →

The screen below will be shown

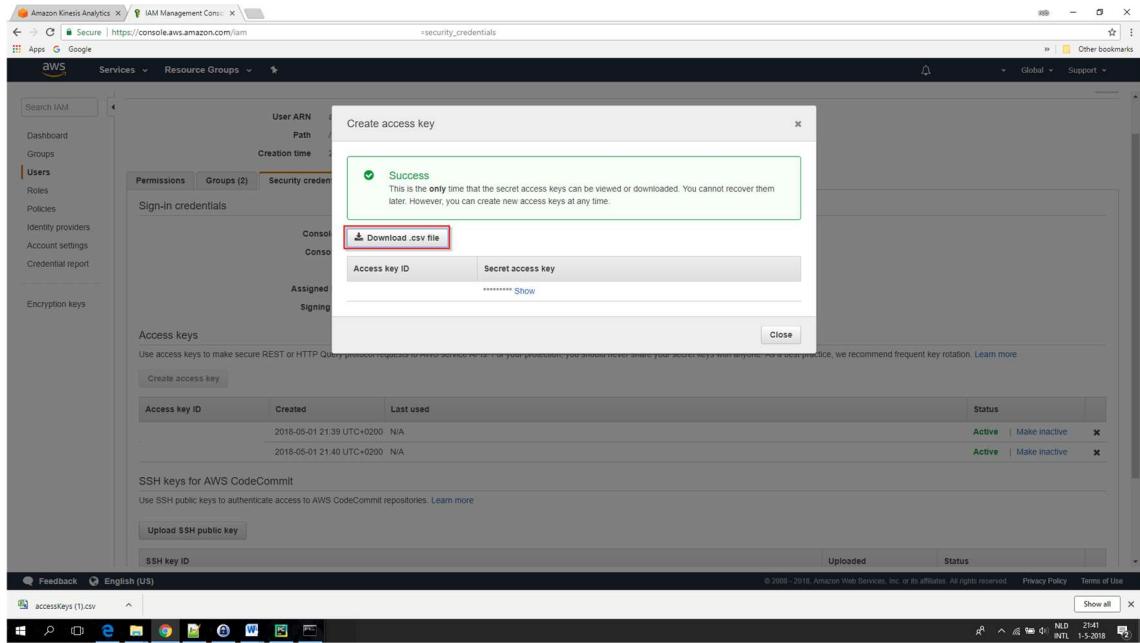
The screenshot shows the AWS IAM User Summary page for a user named 'rob_b'. The 'Security credentials' tab is active. Under 'Sign-in credentials', it shows:

Console password	Enabled	Manage password
Console login link		
Last login	2018-04-23 21:04 UTC+0200	
Assigned MFA device	No	
Signing certificates	None	

Under 'Access keys', there is a note: "Use access keys to make secure REST or HTTP Query protocol requests to AWS service APIs. For your protection, you should never share your keys with others." Below this is a table:

Access key ID	Created	Last used
A	2018-04-13 11:44 UTC+0200	N/A

You need to generate the access key again because the *secret access key* will only be shown during the creation of the key. Don't forget to save the keys in the csv-file, see screenshot below.



Or start the IAM service and open the Users page and locate your user id.

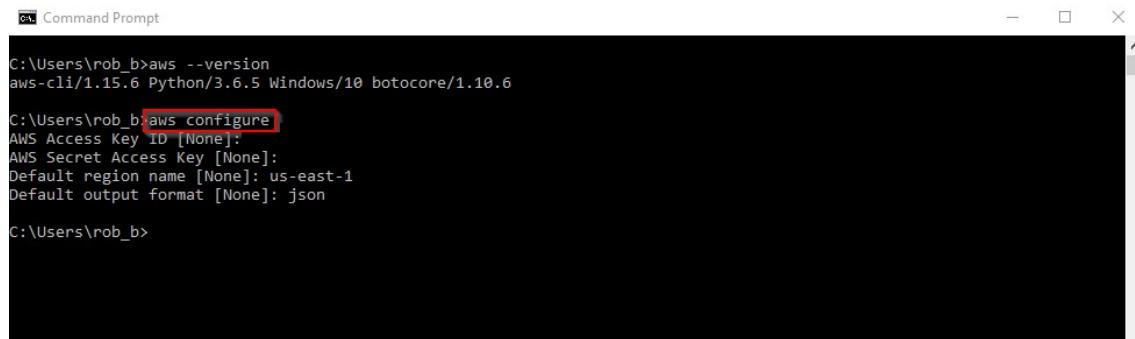
Access key ID	Created	Last used
A	2018-04-13 11:44 UTC+0200	N/A

Select the Security credentials tab and press Create accessKey.

Now you will have the option to show the Key and Secret. Save them and use it when running aws configure. After you leave the page, you won't be able to see the Secret anymore. Only the Access key ID will be shown.

Type: `aws configure` to start the configuration

You need to enter the AWS Access Key ID, AWS Secret Access Key, the default region name (`us-east-1`) and default output format (`json`).



A screenshot of a Windows Command Prompt window titled "Command Prompt". The window shows the following text:

```
C:\Users\rob_b>aws --version
aws-cli/1.15.6 Python/3.6.5 Windows/10 botocore/1.10.6

C:\Users\rob_b>aws configure
AWS Access Key ID [None]:
AWS Secret Access Key [None]:
Default region name [None]: us-east-1
Default output format [None]: json

C:\Users\rob_b>
```

Now your Security credentials are stored (in a local file in the subfolder `.aws` in your profile/user directory) so `awscli` automatically has access to this key.

AWS console

Log in op de AWS console: <https://console.aws.amazon.com/console/home>

Enter your AWS Account ID, IAM username and password.



Account ID or alias

IAM user name

Password

Sign In

[Sign-in using root account credentials](#)



English

[Terms of Use](#) [Privacy Policy](#) © 1996-2018, Amazon Web Services, Inc. or its affiliates.

You will arrive at the home page of the console.

aWS Services Resource Groups

AWS services Find a service by name or feature (for example, EC2, S3 or VM, storage.)

Recently visited services

- IAM
- Billing
- Kinesis
- EC2
- Elastic Beanstalk

All services

Build a solution

Get started with simple wizards and automated workflows.

- Launch a virtual machine With EC2 ~2-3 minutes
- Build a web app With Elastic Beanstalk ~4 minutes
- Build using virtual servers With Lightsail ~1-2 minutes
- Connect an IoT device With AWS IoT ~5 minutes
- Start a development project With CodeStar ~5 minutes
- Register a domain With Route 53 ~3 minutes

See more

Helpful tips

- Manage your costs Get real-time billing alerts based on your cost and usage budgets. [Start now](#)
- Create an organization Use AWS Organizations for policy-based management of multiple AWS accounts. [Start now](#)

Explore AWS

Amazon Relational Database Service (RDS)

RDS manages and scales your database for you. RDS supports Aurora, MySQL, PostgreSQL, MariaDB, Oracle, and SQL Server. [Learn more](#)

Real-Time Analytics with Amazon Kinesis

Stream and analyze real-time data, so you can get timely insights and react quickly. [Learn more](#)

Under recently visited services, you will find the services you last visited.

If you click on > All Services, you will see all other services. Note that you see every possible service, that does not mean you have access to it!

You can search services by start typing in the search box.

AWS services

The screenshot shows the AWS search interface with the search term 'kine' entered. Two results are displayed:

- Kinesis**: Work with Real-Time Streaming Data
- Kinesis Video Streams**: Capture, Process, and Store Video Streams for Analytics and Machine Learning

Below the search bar, there is a link to 'All services'.

Start typing kinesis.

Select Kinesis.

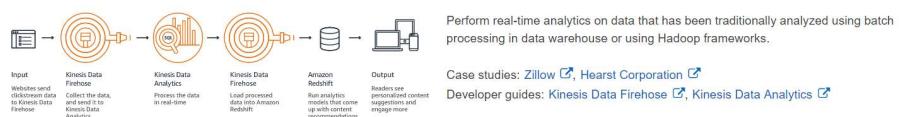
The Kinesis homepage is now shown. You also navigate to it directly by using the url:
<https://console.aws.amazon.com/kinesis>

If there is no Kinesis service created yet, you will see an informational page.

The screenshot shows the Amazon Kinesis homepage. It features the Kinesis logo and the heading "Amazon Kinesis". Below the heading, a sub-headline reads: "Easily collect, process, and analyze video and data streams in real time, so you can get timely insights and react quickly to new information." A "Get started" button is visible, along with a link "What is streaming data?"

What can you build with Amazon Kinesis?

Evolve from batch to real-time analytics



Select the correct region: us-east-1 (if this not already set):

The screenshot shows the AWS navigation bar with the following items:

- eventsig001 @ amis
- N. Virginia
- Support

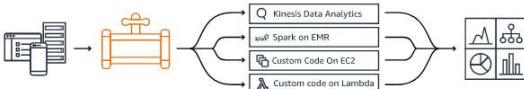
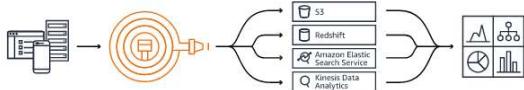
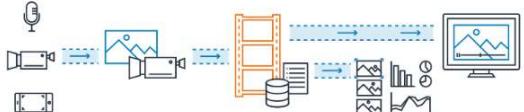
Below the navigation bar, a dropdown menu is open, showing two options:

- US East (N. Virginia)** (highlighted with a red border)
- US East (Ohio)

Press Get started.

Get started with Amazon Kinesis

To get started, choose an Amazon Kinesis resource to create.

<p>Ingest and process streaming data with Kinesis streams</p> <p>Process data with your own applications, or using AWS managed services like Amazon Kinesis Data Firehose, Amazon Kinesis Data Analytics, or AWS Lambda.</p>  <p>Create data stream</p>	<p>Deliver streaming data with Kinesis Firehose delivery streams</p> <p>Continuously collect, transform, and load streaming data into destinations such as Amazon S3 and Amazon Redshift.</p>  <p>Create delivery stream</p>
<p>Analyze streaming data with Kinesis analytics applications</p> <p>Run continuous SQL queries on streaming data from Kinesis data streams and Kinesis Firehose delivery streams.</p>  <p>Create analytics application</p>	<p>Ingest and process media streams with Kinesis video streams</p> <p>Build applications to process or analyze streaming media.</p>  <p>Create video stream</p>

Related CloudFormation templates

Choose an AWS CloudFormation template to get started building a complete streaming data solution. AWS CloudFormation templates offer quick and reliable provisioning of services and applications. [Learn more](#)

CloudFormation template	Description	Recommended for
Real-time insights on AWS account activity	Gain valuable insights on your AWS usage in seconds with a real-time dashboard. No coding required. Learn more	AWS customers with prior usage data
AWS connected vehicle	Implement secure vehicle connectivity services in the AWS Cloud, including local computing in vehicles, event rules, data processing, and storage. Learn more	Advanced AWS IoT customers

[Cancel](#)

If there are already streams or applications created, the Amazon Kinesis Dashboard looks like:

Amazon Kinesis dashboard

Amazon Kinesis makes it easy to collect, process, and analyze video and data streams in real time, so you can get timely insights and react quickly to new information. [What is streaming data?](#)

Kinesis data streams (2)

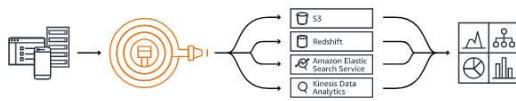
Name	Status
DEMO_LAMBDA1	Active
kinesis-analytics-demo-stream	Active

[View all](#) [Create data stream](#)

Kinesis Firehose delivery streams

Continuously collect, transform, and load streaming data into destinations such as Amazon S3 and Amazon Redshift.

[Learn more](#)



[Create delivery stream](#)

Kinesis analytics applications (2)

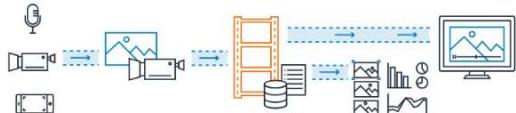
Name	Status
ExampleApp	Ready
demo-lambda-1	Ready

[View all](#) [Create analytics application](#)

Kinesis video streams

Build applications to process or analyze streaming media.

[Learn more](#)



[Create video stream](#)

Related CloudFormation templates

Choose an AWS CloudFormation template to get started building a complete streaming data solution. AWS CloudFormation templates offer quick and reliable provisioning of services and applications. [Learn more](#)

CloudFormation template	Description	Recommended for
Real-time insights on AWS account activity	Gain valuable insights on your AWS usage in seconds with a real-time dashboard. No coding required. Learn more	AWS customers with prior usage data
AWS connected vehicle	Implement secure vehicle connectivity services in the AWS Cloud, including local computing in vehicles, event rules, data processing, and storage. Learn more	Advanced AWS IoT customers

14/49

Kinesis Datastreams

On the Kinesis dashboard, choose Create datastream.

Create Kinesis stream

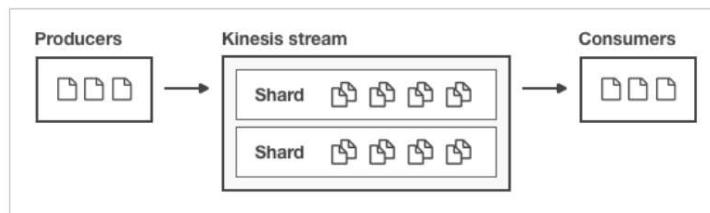


Kinesis stream name*

Acceptable characters are uppercase and lowercase letters, numbers, underscores, hyphens, and periods.

Shards

A shard is a unit of throughput capacity. Each shard ingests up to 1MB/sec and 1000 records/sec, and emits up to 2MB/sec. To accommodate for higher or lower throughput, the number of shards can be modified after the Kinesis stream is created using the API. [Learn more](#)



▶ Estimate the number of shards you'll need

Number of shards*

You can provision up to 500 more shards before hitting your account limit of 500.
[Learn more](#) or [request a shard limit increase for this account](#)

Total stream capacity

Values are calculated based on the number of shards entered above.

Write MB per second

Records per second

Read MB per second

* Required

[Cancel](#)

[Create Kinesis stream](#)

Note: where you see {your alias}, enter an alias to distinguish the different objects created by the different users!

Give the stream a name: **{your alias}-traffic-message-stream**

Number of shards: **1**

The total stream capacity is shown.

Press Create Kinesis stream

The stream is now being created.

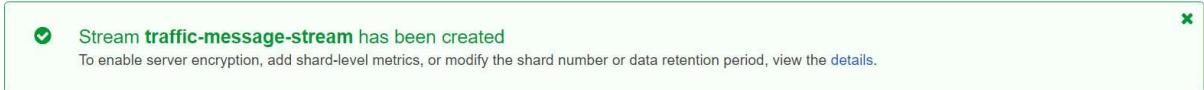


Creating stream traffic-message-stream

Creating a stream generally takes up to a minute.



When ready (it will not take long), a message is shown that the stream has been created:



Press on the created stream in the table:

Create Kinesis stream			Connect to Firehose	Actions	
			Filter or search by stream name	Viewing 1 - 3 of 3 Kinesis streams	
	Kinesis stream name	Number of shards		Status	
<input type="checkbox"/>	DEMO_LAMBDA1	1		Active	
<input type="checkbox"/>	kinesis-analytics-demo-stream	1		Active	
<input type="checkbox"/>	traffic-message-stream	1		Active	

The details page is shown.

Streams > traffic-message-stream

traffic-message-stream

[Connect to Firehose](#)

[Delete Kinesis stream](#)

To send data to the stream, configure producers using the Kinesis Streams PUT API operation or the Amazon Kinesis Producer Library (KPL). [Learn more](#). For custom processing and analyzing of real-time, streaming data, use the Kinesis Client Library (KCL).

[Details](#) [Monitoring](#) [Tags](#)

Stream ARN

Status ACTIVE

Sending data to the stream Configure producers to send data using the Streams PUT API operation or the Amazon Kinesis Producer Library (KPL). [Learn more](#)

Receiving and processing data **Kinesis Streams:** Build a custom application to process or analyze streaming data using the Kinesis Client Library. [Learn more](#)

Kinesis Firehose: Connect your Kinesis stream to a Firehose delivery stream. [Go to Kinesis Firehose](#)

Kinesis Analytics: Analyze streaming data from Kinesis Streams in real time using SQL. [Go to Kinesis Analytics](#)

Shards

[Edit](#)

In order to adapt to changes in the rate of data flow through the stream, Amazon Kinesis Streams supports scaling, which enables you to adjust the number of shards in your stream. [Learn more](#)

Open shards 1 [Edit](#)

Closed shards 0 [Edit](#)

Server-side encryption

[Edit](#)

Enable server-side encryption to encrypt sensitive data in the Kinesis stream with an AWS KMS master key. [Learn more](#)

Server-side encryption Disabled

Data retention period

[Edit](#)

The data retention period can be increased from 24 hours up to 168 hours for an additional cost. Go to [Kinesis Streams pricing](#).

Data retention period 24 hours [Edit](#)

Shard level metrics

[Edit](#)

Shard-level metrics are valuable for identifying the distribution of data throughput; data is available in 1-minute periods for an additional cost. Go to [CloudWatch pricing](#).

Shard level metrics No shard level metrics enabled

Now we have a datastream on which data can be received.

On the monitoring tab, you see several metrics about the datastream. But currently we do not have any data yet, so nothing is shown.

Kinesis Analytics

Create analytics application

Go back to the Dashboard.

Amazon Kinesis

Dashboard

Data Streams

Data Firehose

Data Analytics

Video Streams

External resources

[What's new](#)

Now create an analytics application so we can do something with the data.

Press Create analytics application.

Kinesis Analytics - Create application

Kinesis Analytics applications continuously read and analyze data from a connected streaming source in real-time. To enable interactivity with your data during configuration you will be prompted to run your application. Kinesis Analytics resources are not covered under the [AWS Free Tier](#), and [usage-based charges apply](#). For more information, see [Kinesis Analytics pricing](#).

Application name*

|

Description

* Required

[Cancel](#)

[Create application](#)

Give the application a name: **{your alias}-Analyze-traffic**

And optional a description.

Press Create application.

Analyze-traffic

Application ARN:

Application version ID: 1 [?](#)



Successfully created Application **Analyze-traffic**

Next, choose **Connect streaming data**.



Source

Streaming data

Connect to an existing Kinesis stream or Firehose delivery stream, or easily create and connect to a new demo Kinesis stream. Each application can connect to one streaming data source. [Learn more](#)

[Connect streaming data](#)



Real time analytics

Author your own SQL queries or add SQL from templates to easily analyze your source data.



Destination

(Optional) Connect an in-application stream to a Kinesis stream, or to a Firehose delivery stream, to continuously deliver SQL results to AWS destinations. The limit is three destinations for each application. [Learn more](#)

[Exit to Kinesis Analytics applications](#)

Connect streaming data

To be able to do anything, we first need to add the source.

Press connect streaming data.



Connect to source

Choose from your Kinesis streams and Firehose delivery streams, or quickly configure a demo Kinesis stream that can be used to explore Kinesis Analytics.

[Choose source](#)

[Configure a new stream](#)

Source*

Kinesis stream ⓘ
 Kinesis Firehose delivery stream ⓘ

Kinesis stream*

In-application stream name In your SQL queries, refer to this source as:
SOURCE_SQL_STREAM_001

Record pre-processing with AWS Lambda

Kinesis Analytics can invoke your Lambda function to pre-process records before they are used in this application. To pre-process records, your Lambda function must be compliant with the required record transformation output model. [Learn more](#)

Record pre-processing* Disabled
 Enabled

Access to chosen resources

Create or choose IAM role with the required permissions. [Learn more](#)

Access to chosen resources* Create / update IAM role **kinesis-analytics-Analyze-traffic-us-east-1**
 Choose from IAM roles that Kinesis Analytics can assume

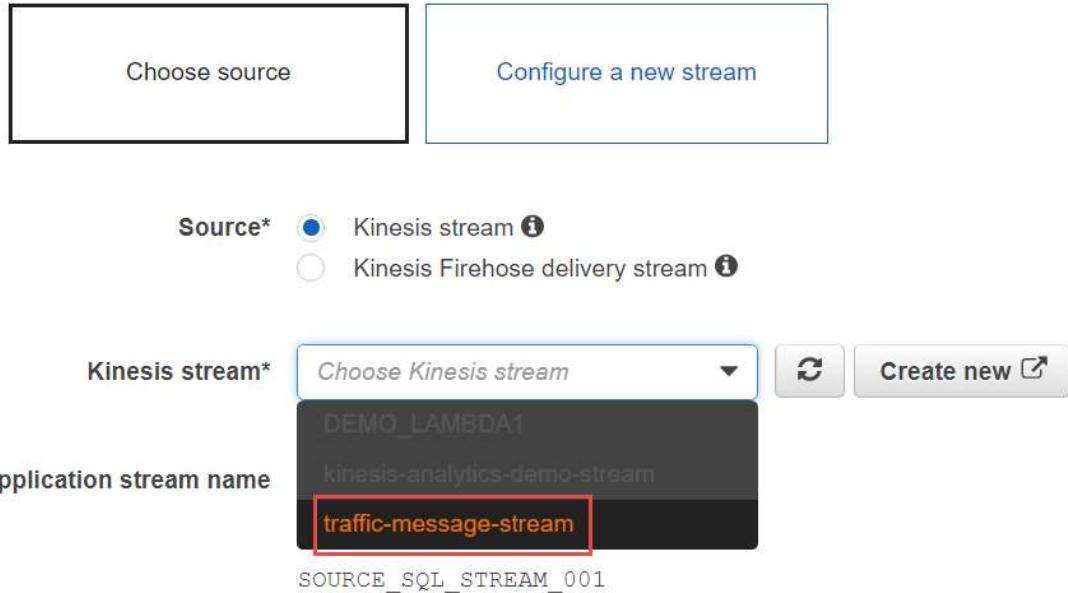
Schema

Schema discovery can generate a schema using recent records from the source. Schema column names are the same as in the source, unless they contain special characters, repeated column names, or reserved keywords. [Learn more](#)

[Discover schema](#)

[Cancel](#) [Save and continue](#)

We already created a stream before, so select this one from the drop down list.



To be able understand the message coming in, we need to set the schema of the data stream.

Data producing

But remember: we did not have any data yet in the stream, so there discovering will fail.

We now first need to send data to the created datastream.

We will use a client on the local pc to send the data to the Kinesis datastream.

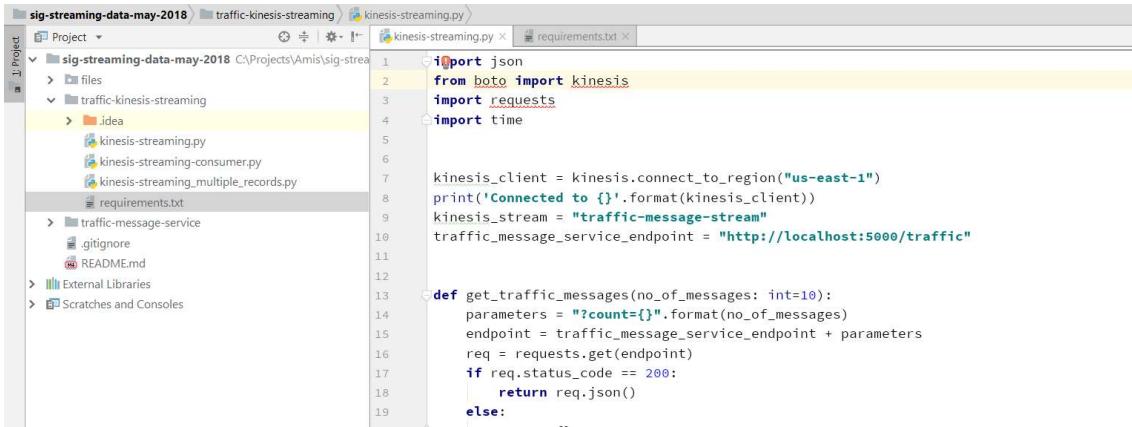
Before we can do this, we first need to install the client sample code.

Clone or download the following repository:

<https://github.com/AMIS-Services/sig-streaming-data-may-2018>

This repository contains a simple client which can sent data to Kinesis inside of the traffic-kinesis-streaming subfolder (<https://github.com/AMIS-Services/sig-streaming-data-may-2018/tree/master/traffic-kinesis-streaming>).

Open the whole project in PyCharm to see the code.



```

1 import json
2 from boto import kinesis
3 import requests
4 import time
5
6
7 kinesis_client = kinesis.connect_to_region("us-east-1")
8 print("Connected to {}".format(kinesis_client))
9 kinesis_stream = "traffic-message-stream"
10 traffic_message_service_endpoint = "http://localhost:5000/traffic"
11
12
13 def get_traffic_messages(no_of_messages: int=10):
14     parameters = "?count={}".format(no_of_messages)
15     endpoint = traffic_message_service_endpoint + parameters
16     req = requests.get(endpoint)
17     if req.status_code == 200:
18         return req.json()
19     else:
19

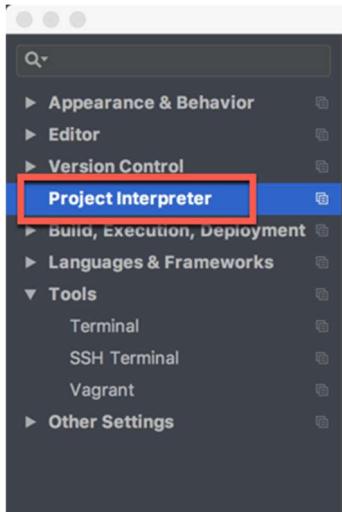
```

If you see the red underlining, this means that the required Python modules are not yet available. First a virtualenv must be created (or selected).

To run the Python script, first create a Python virtualenv (if you did not do that already):

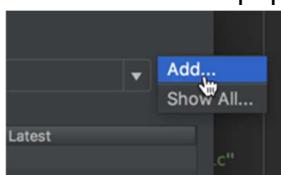
In the menu select File > Default Settings.

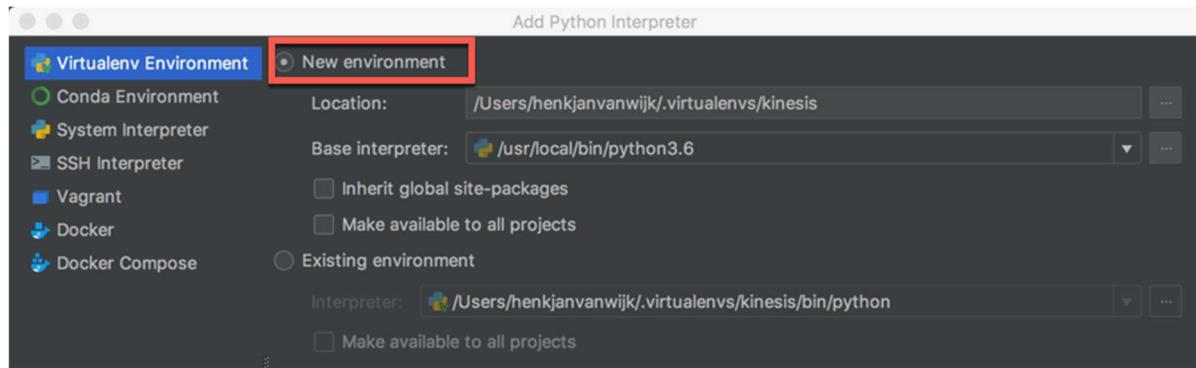
Navigate to Project Interpreter.



In the right pane, click on the wheel icon 

Select Add in the popup menu.





Select Virtualenv Environment, New environment and enter a location, eg. **kinesis** or **kinesis-workshop**.

The last folder (in the screenshot: kinesis) is the name of the virtual environment.
Press OK.

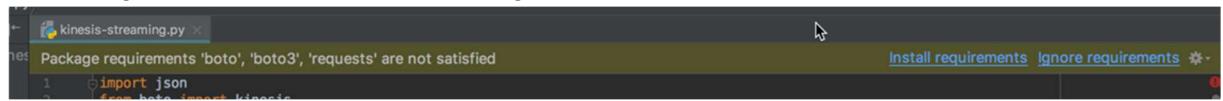
A new virtual environment is created and linked to the project.

If you already created a virtualenv, then you need to select the already created virtualenv:

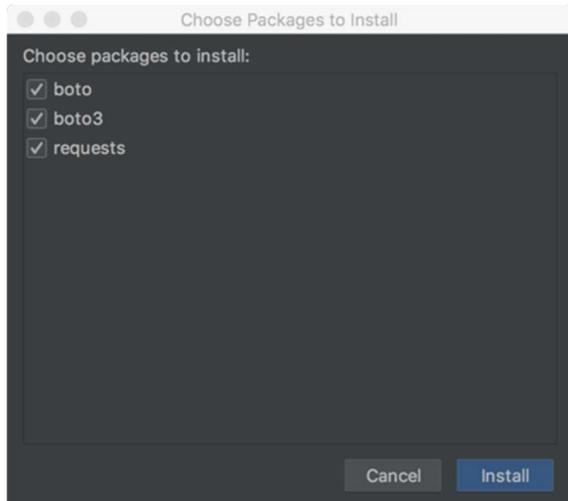


Select an existing virtualenv or add one to the list using the plus icon.

You will get an error that certain package requirements are not satisfied.

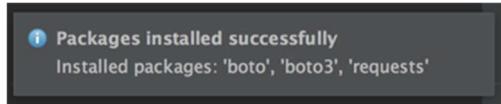


Press install requirements.



Choose all packages (the default) and press Install.

The packages are now being downloaded and installed (using pip).



In the script the variable `traffic_message_service_endpoint` contains the url to the traffic message API.

Use either:

- `localhost:8080/traffic`
- <http://traffic-message-service.s7whvi2kjr.us-east-1.elasticbeanstalk.com/traffic>

The first you can use when running the API locally or the second one using an already installed API on AWS Elastic Beanstalk.

Running the local REST API service

In the folder `traffic-message-service` you will find the (simple) implementation of the REST API service for generating traffic containing random license plates.

This API is implemented using Python Flask.

To be able to run this API locally, you can either run the script from a command line or using PyCharm.

Command line

Open a command prompt and navigate to the `traffic-message-service` folder.

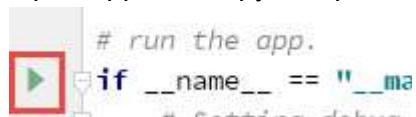
Activate the virtualenv and start the server: `python application.py`

```
traffic-message-service) PS C:\Projects\Amis\sig-streaming-data-may-2018\traffic-message-service> python application.py
* Restarting with stat
* Debugger is active!
* Debugger PIN: 231-292-656
* Running on http://0.0.0.0:8080/ (Press CTRL+C to quit)
```

The used port is being displayed.

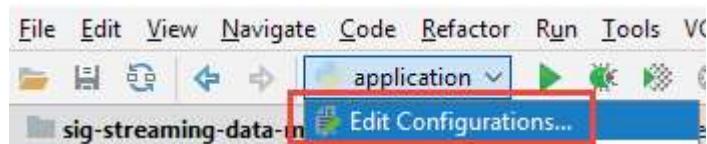
Using PyCharm

Open application.py and press the run icon in the margin of the editor

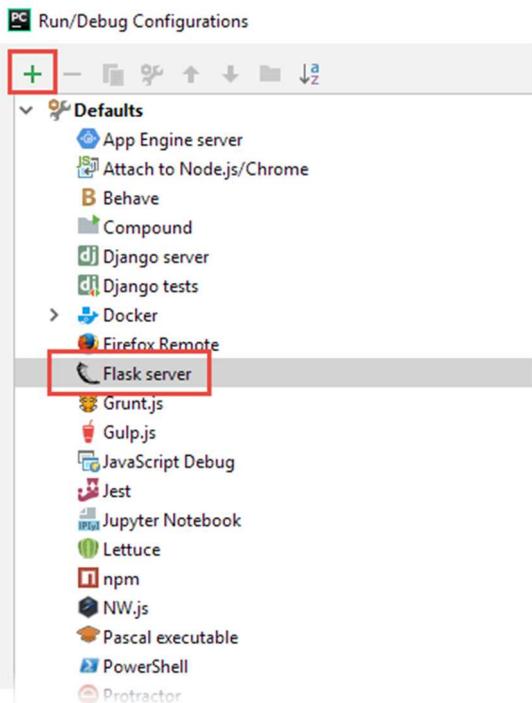


```
# run the app.  
if __name__ == "__ma  
    ...
```

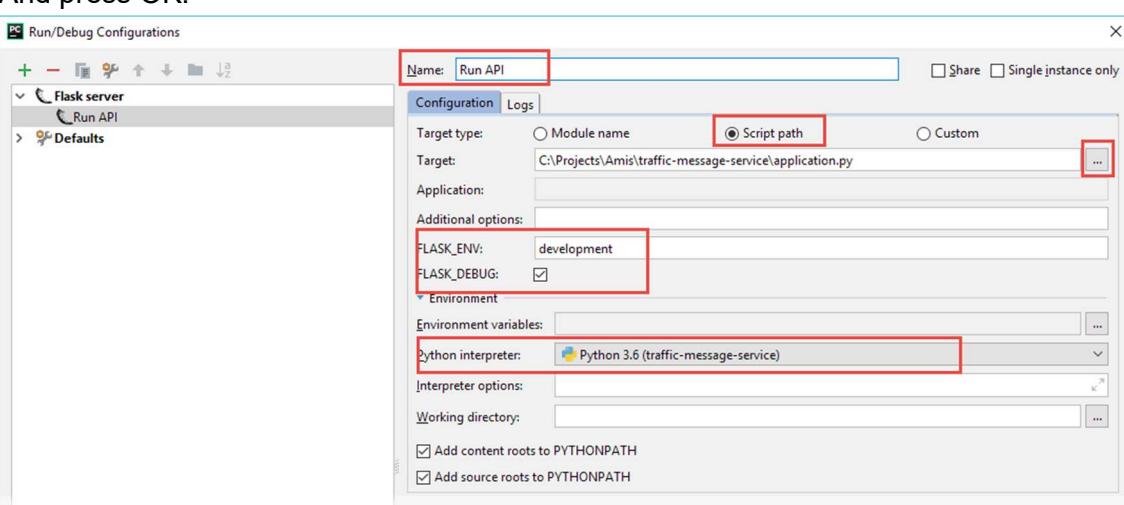
Or add a Run configuration.



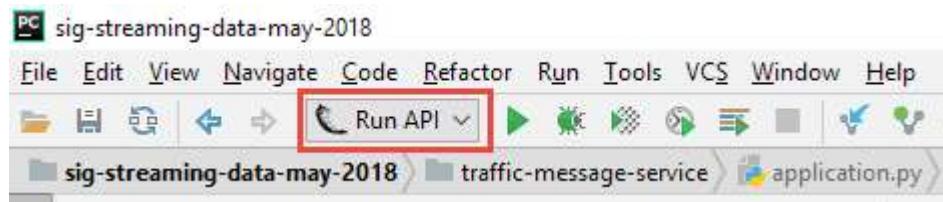
Press the plus icon and choose Flask Server



Enter a name, choose Script path and select the application.py file.
Make sure the correct virtualenv is selected under Python interpreter.
And press OK.



Now the option is added to the drop down list next to the Run button.

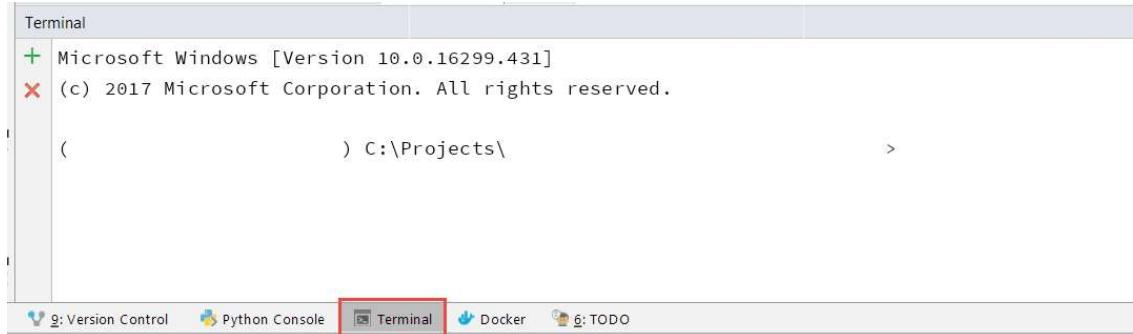


Press the Run button to start the API.

Toubleshooting

If you do not get the message for installing the missing modules, you can also do it by hand in a terminal window.

Press the terminal icon in the footerbar.



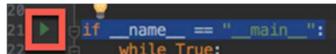
Between the brackets in the terminal window you should see the name of your virtualenv. This means the virtualenv is activated.

From the rootfolder of the repository type:

```
pip install --upgrade -r traffic-kinesis-streaming\requirements.txt
pip install --upgrade -r traffic-message-service\requirements.txt
```

Run streaming producer

Press the small run button next to if __name__ == "__main__":



The client will now be run.

Take a look at the code. You will notice that the client is sending the data one record at a time. This can be done more efficiently (if you do not need the messages to be sent one at a time), by combining them in a multi record list.

Open the file '*kinesis_streaming_multiple_records.py*' to see the example for this type of message sending. If you look carefully at the difference between the 2 scripts, you will see that the single record script uses **put_record** and the multi record script uses **put_records**. And that is not the only difference. The first script uses the boto library and the second one the boto3 library. Both libraries are maintained by Amazon. Boto3 is the newest. Boto is more legacy code and has a class for each of the AWS services. Boto3 is more generic.

```
kinesis_client = kinesis.connect_to_region("us-east-1")
vs
kinesis_client = boto3.client('kinesis')
```

After starting the client, data will be coming into the Kinesis stream.

Discover schema

Go back to the AWS page you left.

Scroll down to Schema.

Press Discover schema.

Schema

Schema discovery can generate a schema using recent records from the source. Schema column names are the same as in the source, unless they contain special characters, repeated column names, or reserved keywords. [Learn more](#)

Discover schema

 Discovering schema for stream **traffic-message-stream** with starting point 'NOW'

Wait for a while until the schema has been analyzed from the incoming data.

 **Schema discovery successful**

Detected JSON format and applied schema

- To define a custom schema, choose "Edit schema" in the stream sample below.
- To capture a new stream sample from the selected source for discovery, choose **Retry schema discovery** below.

(Optional) Send AWS a sample of your data to help improve schema discovery in Amazon Kinesis Analytics

[Help improve schema discovery](#)

Edit schema **Retry schema discovery**

Raw	Lambda output	Formatted																																																												
<input type="text" value="Filter by column name or column type"/>																																																														
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>color VARCHAR(8)</th> <th>licenseplate VARCHAR(8)</th> <th>manufacturer VARCHAR(16)</th> <th>model VARCHAR(8)</th> <th>latitude VARCHAR(16)</th> <th>longitude DOUBLE</th> <th>roadNumber VARCHAR(4)</th> <th>speed INTEGER</th> <th>COL_timestamp VARCHAR(32)</th> </tr> </thead> <tbody> <tr> <td>black</td> <td>PC-90-13</td> <td>BMW</td> <td>7 series</td> <td>52.0604741</td> <td>5.100614</td> <td>A12</td> <td>136</td> <td>2018-04-19T21:08:06</td> </tr> <tr> <td>green</td> <td>56-66-GW</td> <td>Renault</td> <td>Twingo</td> <td>52.0604741</td> <td>5.100614</td> <td>A12</td> <td>132</td> <td>2018-04-19T21:07:56</td> </tr> <tr> <td>white</td> <td>7-RTR-58</td> <td>Renault</td> <td>Talisman</td> <td>52.0604741</td> <td>5.100614</td> <td>A12</td> <td>100</td> <td>2018-04-19T21:08:05</td> </tr> <tr> <td>black</td> <td>HN-FH-87</td> <td>Volkswagen</td> <td>Polo</td> <td>52.0604741</td> <td>5.100614</td> <td>A12</td> <td>101</td> <td>2018-04-19T21:08:06</td> </tr> <tr> <td>white</td> <td>7-THN-97</td> <td>Volvo</td> <td>V90</td> <td>52.0604741</td> <td>5.100614</td> <td>A12</td> <td>80</td> <td>2018-04-19T21:08:05</td> </tr> </tbody> </table>									color VARCHAR(8)	licenseplate VARCHAR(8)	manufacturer VARCHAR(16)	model VARCHAR(8)	latitude VARCHAR(16)	longitude DOUBLE	roadNumber VARCHAR(4)	speed INTEGER	COL_timestamp VARCHAR(32)	black	PC-90-13	BMW	7 series	52.0604741	5.100614	A12	136	2018-04-19T21:08:06	green	56-66-GW	Renault	Twingo	52.0604741	5.100614	A12	132	2018-04-19T21:07:56	white	7-RTR-58	Renault	Talisman	52.0604741	5.100614	A12	100	2018-04-19T21:08:05	black	HN-FH-87	Volkswagen	Polo	52.0604741	5.100614	A12	101	2018-04-19T21:08:06	white	7-THN-97	Volvo	V90	52.0604741	5.100614	A12	80	2018-04-19T21:08:05
color VARCHAR(8)	licenseplate VARCHAR(8)	manufacturer VARCHAR(16)	model VARCHAR(8)	latitude VARCHAR(16)	longitude DOUBLE	roadNumber VARCHAR(4)	speed INTEGER	COL_timestamp VARCHAR(32)																																																						
black	PC-90-13	BMW	7 series	52.0604741	5.100614	A12	136	2018-04-19T21:08:06																																																						
green	56-66-GW	Renault	Twingo	52.0604741	5.100614	A12	132	2018-04-19T21:07:56																																																						
white	7-RTR-58	Renault	Talisman	52.0604741	5.100614	A12	100	2018-04-19T21:08:05																																																						
black	HN-FH-87	Volkswagen	Polo	52.0604741	5.100614	A12	101	2018-04-19T21:08:06																																																						
white	7-THN-97	Volvo	V90	52.0604741	5.100614	A12	80	2018-04-19T21:08:05																																																						

Check if the schema is OK.

Press Edit schema and make any necessary changes, such as datatype and data length.

After you made the necessary changes, save the changes.

Cancel **Save schema and update stream samples**

The stream is now checked against the updated schema.

After a while new rows will be shown (if the schema is ok).

When you are done, press Exit (done).

Save and continue

Press

After successful discovery of the schema, stop the client of producing data (to minimize the running costs of using Kinesis).

Start the client again when you need some new data.

Also stop a running Kinesis Analytics Application if you do not need it.

Stopping an Kinesis Analytics Application

You can only create and delete an application in the console. You need to use the AWS CLI to stop a running application:

```
aws --region us-east-1 kinesisanalytics stop-application --application-name
Analyze-traffic
```

Enter the correct region and application name.

Performing some analytics

Now we can do some analysis.

Press Go to SQL editor.



Simple streaming

Let start with a 'simple' query to show the streaming data.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
    carname varchar(20),
    carmodel varchar(20),
    licenseplate varchar(10),
    country varchar(20));

CREATE OR REPLACE PUMP "STREAM_PUMP" AS INSERT INTO
"DESTINATION_SQL_STREAM"
SELECT STREAM "manufacturer", "model", "licenseplate", "country"
FROM "SOURCE_SQL_STREAM_001"
```

All real time analytics queries look like this one.

We start with creation a destination stream (kind of table) in which you define the column the stream will have and its datatypes.

Next you define a pump which will take data from the source data stream you created earlier and insert the data into then destination stream.

The pump and the stream must match of course.

Select aggregated data (Using GROUP BY)

A more advanced query is the one we use to answer the following question:

- We are interested in the total number of cars grouped by manufacturer within a one-minute window.

The query below answers our question:

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (manufacturer
VARCHAR(8), manufacturer_count INTEGER);

-- CREATE OR REPLACE PUMP to insert into output
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
    INSERT INTO "DESTINATION_SQL_STREAM"
        SELECT STREAM "manufacturer", COUNT(*) AS manufacturer_count
    FROM "SOURCE_SQL_STREAM_001"
        GROUP BY "manufacturer",
                STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '60'
SECOND);
```

Select the query above paste it in *SQL editor* and press the *Save and run SQL button*

The result could be something like below:

Real-time analytics

[Save and run SQL](#) [Add SQL from templates](#) [Download SQL](#) [SQL reference guide](#) [Kinesis data generator tool](#)

```
1 CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (manufacturer VARCHAR(8), manufacturer_count INTEGER);
2
3 -- CREATE OR REPLACE PUMP to insert into output
4 CREATE OR REPLACE PUMP "STREAM_PUMP" AS
5   INSERT INTO "DESTINATION_SQL_STREAM"
6     SELECT STREAM "manufacturer", COUNT(*) AS manufacturer_count
7   FROM "SOURCE_SQL_STREAM_001"
8     GROUP BY "manufacturer",
9           STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '60' SECOND);
10
```

Source data Real-time analytics Destination Application status: RUNNING

In-application streams: [Start streaming results](#) [Download CSV](#)

DESTINATION_SQL_STREAM	Filter by column name		
	ROWTIME	MANUFACTURER	MANUFACTURER_COUNT
error_stream	2018-05-01 17:47:00.0	Renault	22
	2018-05-01 17:48:00.0	Volvo	20
	2018-05-01 17:48:00.0	Renault	24
	2018-05-01 17:48:00.0	Mercedes	16
	2018-05-01 17:48:00.0	SEAT	12
	2018-05-01 17:48:00.0	SKODA	23
	2018-05-01 17:48:00.0	Toyota	8
	2018-05-01 17:48:00.0	Volkswag	8
	2018-05-01 17:48:00.0	Citroen	19
	2018-05-01 17:48:00.0	BMW	24

Select Most Frequently Occurring Values (top-K items)

In our message we have a car manufacturer. Suppose we want to know what the two most occurring car manufacturers are within a specific time window(tumbling).

From the *SQL editor* window select the *Add SQL from templates* button and choose the *Approximate top-K items template*. Select the *Use this SQL instead* button (this will overwrite your previous query!).

Change the query so it looks like the query below:

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (ITEM VARCHAR(1024),
ITEM_COUNT DOUBLE);
CREATE OR REPLACE PUMP "STREAM_PUMP" AS INSERT INTO
"DESTINATION_SQL_STREAM"
SELECT STREAM ITEM, ITEM_COUNT FROM TABLE(TOP_K_ITEMS_TUMBLING(
CURSOR(SELECT STREAM * FROM "SOURCE_SQL_STREAM_001"),
'manufacturer', -- name of column in single quotes
2, -- number of top items
60 -- tumbling window size in seconds
)
);
```

NOTE: The bold text shows the changes you have to make.

If the changes are made, press the *Save and run SQL button*. The result could be something like below:

Kinesis Analytics applications > Analyze-traffic > SQL editor



Real-time analytics

[Save and run SQL](#) [Add SQL from templates](#) [Download SQL](#) [SQL reference guide](#) [Kinesis data generator tool](#)

```
1 -- Approximate top-K items - Finds the most frequently occurring values in a stream using the Space Saving algorithm.
2 -- Returns the approximate top-K most frequently
3 -- occurring values in a specified column over a
4 -- tumbling window
5 CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (ITEM VARCHAR(1024), ITEM_COUNT DOUBLE);
6 CREATE OR REPLACE PUMP "STREAM_PUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"
7 SELECT STREAM ITEM, ITEM_COUNT FROM TABLE(TOP_K_ITEMS_TUMBLING(
8 CURSOR(SELECT STREAM * FROM "SOURCE_SQL_STREAM_001"),
9 'manufacturer', -- name of column in single quotes
10 2, -- number of top items
11 60 -- tumbling window size in seconds
12 )
13 );
14 );
```

Source data [Real-time analytics](#) [Destination](#) Application status: RUNNING

In-application streams: [Pause results](#) New results are added every 2-10 seconds. The results below are sampled. [?](#)

DESTINATION_SQL_STREAM Scroll to bottom when new results arrive.

error_stream

Filter by column name		
ROWTIME	ITEM	ITEM_COUNT
2018-04-30 19:50:06.713	BMW	32.0
2018-04-30 19:51:06.713	BMW	31.0
2018-04-30 19:51:06.713	Renault	27.0

[Close](#)

Based on the testdata we use, the result above could be correct. The manufacturers that are shown belong to the two most frequently occurring car manufacturers.

Further explanation about tumbling windows see documentation:

- <https://docs.aws.amazon.com/kinesisanalytics/latest/dev/tumbling-window-concepts.html>
- Tumbling windows are even better explained in the Microsoft documentation:
<https://docs.microsoft.com/nl-nl/azure/stream-analytics/stream-analytics-window-functions>

Reference data

In our message we have a country code, but suppose you want to use the country name instead.

We can do this by adding a reference table and perform a join.

As a reference table we will use a csv file with the necessary translation.

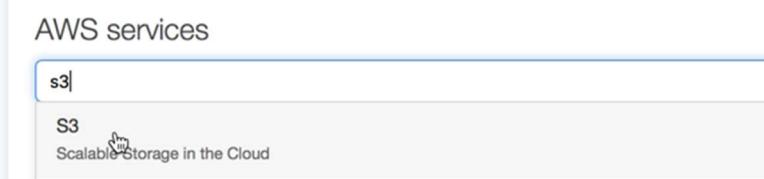
This csv file must be uploaded to a S3 bucket and given access to Kinesis.

We prepared the S3 bucket already for you. You will only need to add the reference file.

But if you want to know how the S3 bucket is created, see the following paragraph, otherwise skip to “Adding file to S3 bucket” (page 36).

Create a S3 bucket

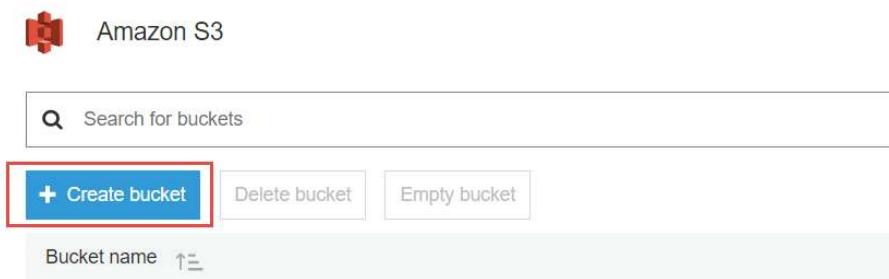
Go to the AWS console homepage and type s3 in the search box.



The screenshot shows the AWS services search interface. A search bar at the top contains the text "s3". Below the search bar, a list of services is displayed, with "S3" being the first item. The "S3" entry includes a small icon of a red cube and the text "Scalable Storage in the Cloud".

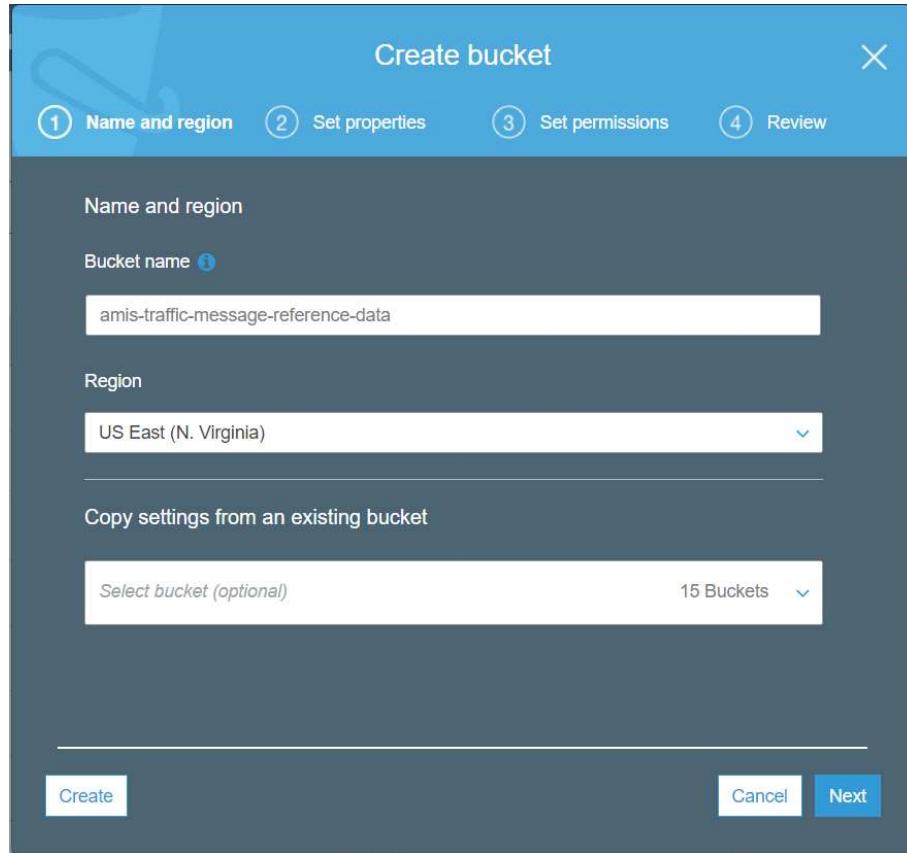
Select S3.

Press Create Bucket.



The screenshot shows the "Amazon S3" service page. At the top, there is a search bar labeled "Search for buckets". Below the search bar, there are three buttons: a blue "+ Create bucket" button, a white "Delete bucket" button, and a white "Empty bucket" button. At the bottom of the screen, there is a text input field labeled "Bucket name" with a small upward arrow icon next to it.

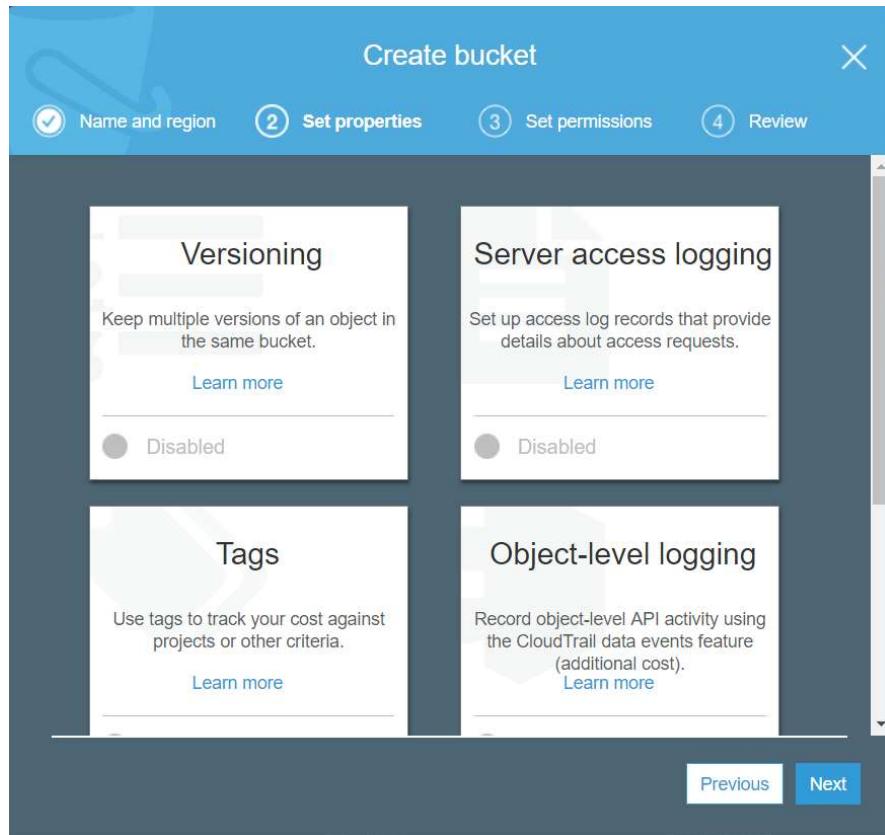
traffic-message-reference-data



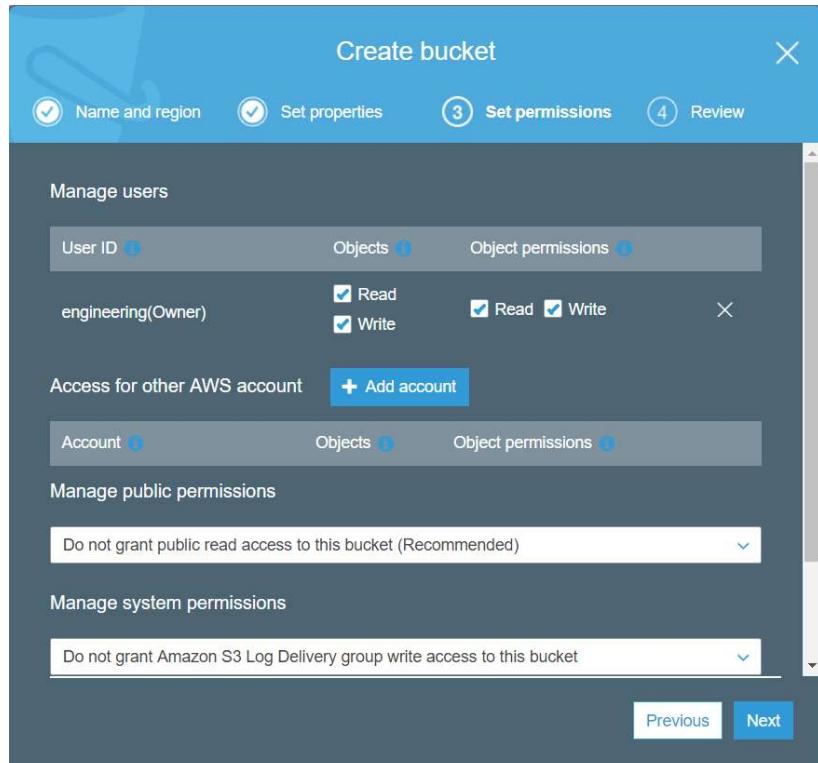
Give the bucket a name: **amis-traffic-message-reference-data**

Select a Region: US East (N. Virginia)

Press Next



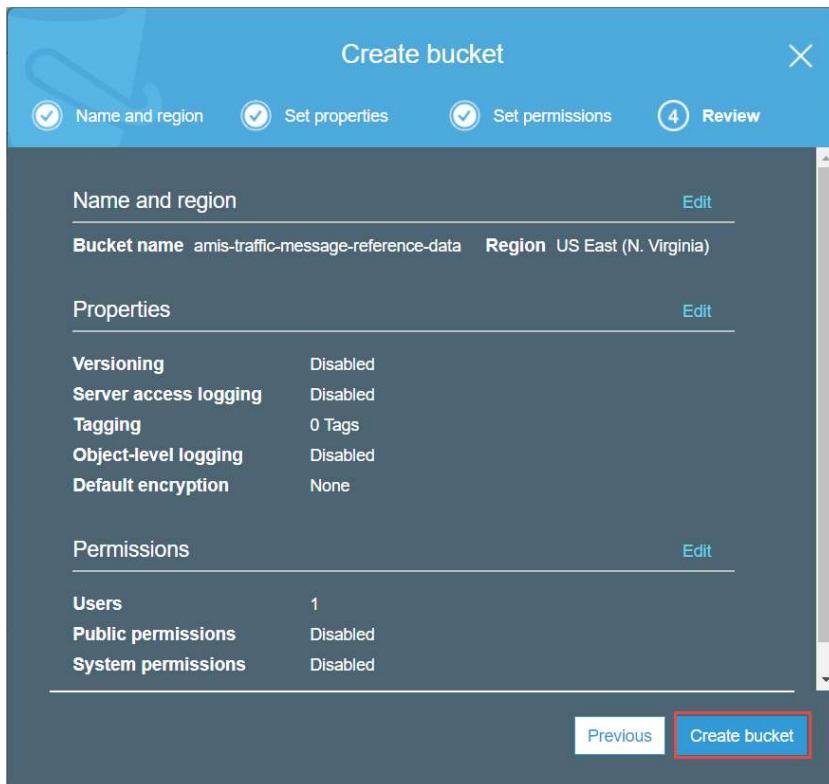
No properties are needed, so press Next again.



On the permission tab, leave the default options.

No public permissions.

Press Next.



Press Create bucket to create the bucket.

Adding file to S3 bucket

Navigate to the S3 service (if you are not there already after creating the bucket).

Bucket name		Access	Region	Date created
amis-traffic-message-reference-data	Not public	US East (N. Virginia)	May 2, 2018 7:14:45 PM GMT+0200	

Find the bucket.

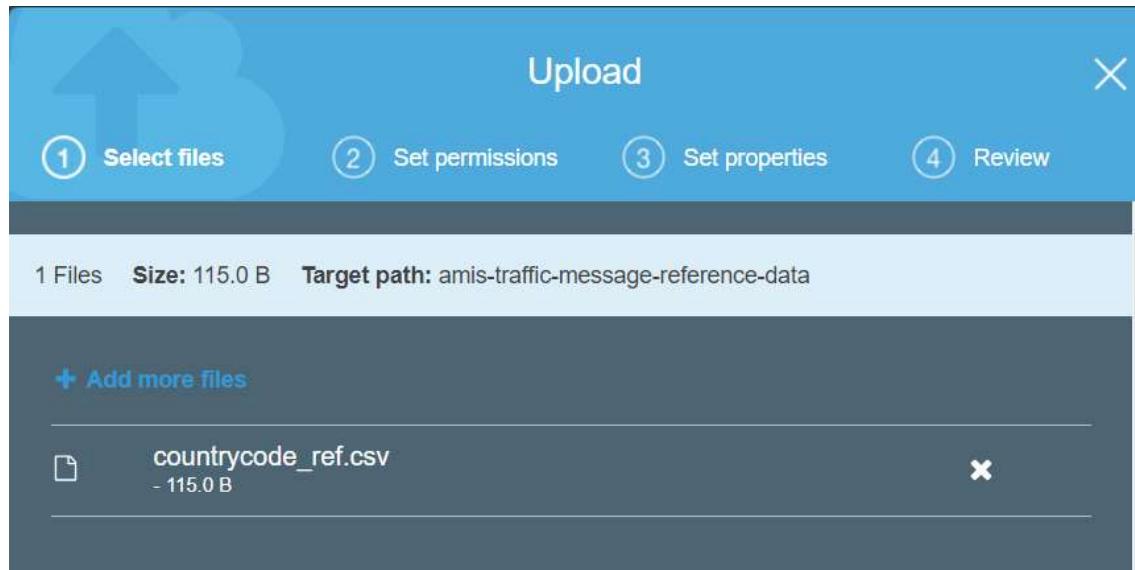
Hover over the row and click on the link (notice the underlining!).

Clicking on the row only will show the properties of the bucket, not the contents.

Press the Upload button.

The screenshot shows the AWS S3 console interface. The top navigation bar has 'Overview' and 'Properties' tabs, with 'Properties' being the active tab. Below the tabs is a search bar with placeholder text 'Type a prefix and press Enter to search. Press ESC to clear.' Underneath the search bar are three buttons: 'Upload' (highlighted with a red box), '+ Create folder', and 'More'. A large modal window titled 'Upload' is displayed. The modal has four numbered steps: 1. Select files, 2. Set permissions, 3. Set properties, and 4. Review. Step 1 is currently selected. The main area of the modal contains a blue folder icon with a white document icon inside, followed by the text 'Drag and drop here OR' and a blue 'Add files' button. At the bottom of the modal are two buttons: 'Upload' on the left and 'Next' on the right.

Copy the file '`countrycode_ref.csv`' from the share to your local PC and rename the file to **{your alias}_countrycode_ref.csv**. And drop the renamed file on the window or use the Add files button.



Press Upload.

The file will appear in the table.

The screenshot shows the AWS S3 console with the 'Properties' tab selected. The URL is 'Amazon S3 > amis-traffic-message-reference-data'. A search bar contains the placeholder 'Type a prefix and press Enter to search. Press ESC to clear.' Below the search bar are buttons for 'Upload', 'Create folder', and 'More'. A table lists a single file: 'countrycode_ref.csv'. The table columns are 'Name' and 'Last modified'. The file was last modified on May 2, 2018, at 7:21:01 PM GMT+0200.

Name	Last modified
countrycode_ref.csv	May 2, 2018 7:21:01 PM GMT+0200

Unfortunately it is not possible to set the permission for accessing this file from the Kinesis application using the console. We need to use the AWS CLI for this.

IAM role

But first we need a new IAM role called **KinesisAnalytics-ReadS3Object** if this not already created. We did this already for you.

But if you want to know the steps involved, then see the following paragraph. Otherwise skip to paragraph "Add Reference Data Source to the Application Configuration" on page 41.

Create a new IAM role

Go to the IAM page.

Select Roles.

Press Create Role.

Create role

1 2 3

Select type of trusted entity

 AWS service EC2, Lambda and others	 Another AWS account Belonging to you or 3rd party	 Web identity Cognito or any OpenID provider	 SAML 2.0 federation Your corporate directory
---	--	--	---

Allows AWS services to perform actions on your behalf. [Learn more](#)

Choose the service that will use this role

EC2 Allows EC2 instances to call AWS services on your behalf.	Lambda Allows Lambda functions to call AWS services on your behalf.	API Gateway	Config	Elastic Container Service	Lex	SWF
AppSync	DMS			Elastic Transcoder	Machine Learning	SageMaker

2

Select AWS service.

Select Lambda.

Press Next: Permissions

Do not add any policies.

Press Next: Review

Create role

1 2 3

Review

Provide the required information below and review this role before you create it.

Role name*	<input type="text"/>
Use alphanumeric and '+,-,@-' characters. Maximum 64 characters.	
Role description	Allows Lambda functions to call AWS services on your behalf.
Maximum 1000 characters. Use alphanumeric and '+,-,@-' characters.	

Trusted entities AWS service: lambda.amazonaws.com

Policies

Enter the Role name: **KinesisAnalytics-ReadS3Object**

Press Create role

After creating the role, select the role you just created.

The screenshot shows the AWS IAM Roles page. At the top, there are two buttons: "Create role" (in blue) and "Delete role". Below them is a search bar containing the text "KinesisAnalytics-ReadS3Object". A table follows, with columns "Role name" and "Description". There is one row in the table, showing "KinesisAnalytics-ReadS3Object" and a description: "Allows Lambda functions to call AWS services on your behalf.".

Go to the Trust relationships tab

Roles > KinesisAnalytics-ReadS3Object

Summary

The screenshot shows the "Summary" tab for the "KinesisAnalytics-ReadS3Object" role. It displays various details about the role, such as its ARN, description, instance profile ARNs, path, creation time, and maximum session duration. Below this, there are tabs for "Permissions", "Trust relationships" (which is highlighted with a red box), "Access Advisor", and "Revoke sessions". A note says "You can view the trusted entities that can assume the role and the access conditions for the role. Show policy document". Below this, there is a "Edit trust relationship" button and sections for "Trusted entities" and "Conditions".

Press Edit trust relationship.

Enter:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "kinesisanalytics.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Edit Trust Relationship

You can customize trust relationships by editing the following access control policy document.

Policy Document

```

1  {
2    "Version": "2012-10-17",
3    "Statement": [
4      {
5        "Effect": "Allow",
6        "Principal": {
7          "Service": "kinesisanalytics.amazonaws.com"
8        },
9        "Action": "sts:AssumeRole"
10      }
11    ]
12 }
```

Press Update Trust Policy

[Cancel](#) [Update Trust Policy](#)

Go to the Permissions tab and attach an AWS managed policy called **AmazonS3ReadOnlyAccess**.

This grants the role permissions to read an S3 object.

The screenshot shows the IAM Roles page with the 'Permissions' tab selected. Under the 'Attached policies' section, there is a single policy named 'AmazonS3ReadOnlyAccess'. This policy is identified as an 'AWS managed policy'. There is also a link to 'Add inline policy'.

Press attach policy.

[Add Reference Data Source to the Application Configuration](#)

Navigate to the IAM Roles and find the role KinesisAnalytics-ReadS3Object.

The select the role and copy the Role ARN.

Summary

Role ARN	arn:aws:iam:	'KinesisAnalytics-ReadS3Object'
Role description	Allows Lambda functions to call AWS services on your behalf. Edit	
Instance Profile ARNs		

Save it somewhere. You will need it in a minute.

Now we are going to add the reference data source to the Kinesis Analytics Application.

We will use the AWS CLI.

Open a command prompt or Powershell window.

Run the describe-application command to get the application details.

```
aws kinesisanalytics describe-application --region us-east-1 --application-name application-name
```

Replace application-name with the name of the Kinesis Analytics Application.

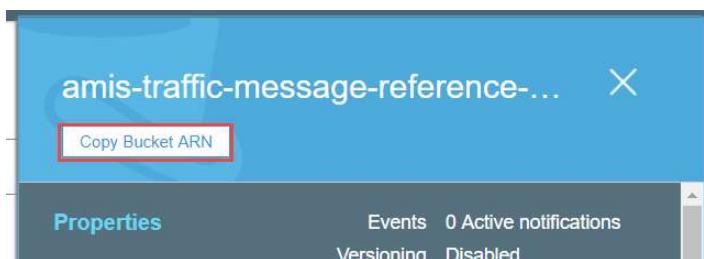
Note the application version Id. We will need that in a minute.

You also need the S3 bucket ARN.

Go to the S3 console, select the bucket (not the link!).

The properties are shown.

Press Copy Bucket ARN.



Save it somewhere.

On the command prompt / shell, type:

```
aws kinesisanalytics add-application-reference-data-source --endpoint https://kinesisanalytics.us-east-1.amazonaws.com --region us-east-1 --application-name traffic-stream-demo --reference-data-source '{"TableName":"Countries","S3ReferenceDataSource": {"BucketARN": "arn:aws:s3:::traffic-message-reference-", "FileKey": "countrycode_ref.csv", "ReferenceRoleARN": "arn:aws:iam::#####:role/KinesisAnalytics-ReadS3Object"}, "ReferenceSchema": {"RecordFormat": {"RecordFormatType": "CSV", "MappingParameters": {"CSVMappingParameters": {"RecordRowDelimiter": "\n", "RecordColumnDelimiter": ",", "Name": "countrycode", "SqlType": "VARCHAR(5)", "Name": "countryname", "SqlType": "VARCHAR(64)"}}}, "RecordEncoding": "UTF-8", "RecordColumns": [{"Name": "countrycode", "SqlType": "VARCHAR(5)}, {"Name": "countryname", "SqlType": "VARCHAR(64)"}]}' --current-application-version-id 7
```

Note the **red marked** entries. Replace them with your own values.

Note the **blue marked** entries. These are the table name as it will be known in the Kinesis Analytics Application and the used filename.

Using the reference data

Create a new Kinesis Analytics Applications or just replace one.

Enter the following query:

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (carname varchar(20),
carmodel varchar(20),
licenseplate varchar(10),
country varchar(20)
);

CREATE OR REPLACE PUMP "STREAM_PUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"
SELECT STREAM "manufacturer", "model", "licenseplate", "c"."countryname"
FROM "SOURCE_SQL_STREAM_001" left join "Countries" as "c"
ON "SOURCE_SQL_STREAM_001"."country" = "c"."countrycode"
```

This is similar to the first simple streaming query, but now with the country name instead of the code.

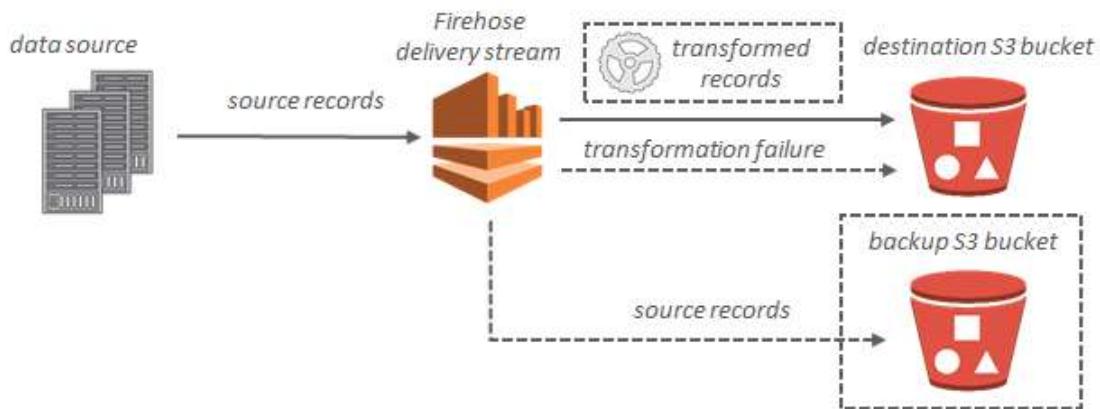
Offloading data

Offloading data in Kinesis can be done by adding a Kinesis Firehose delivery stream. Firehose can offload data to S3 buckets, Redshift, Amazon Elastic Search and to another Kinesis Analytics Application.

Dataflow

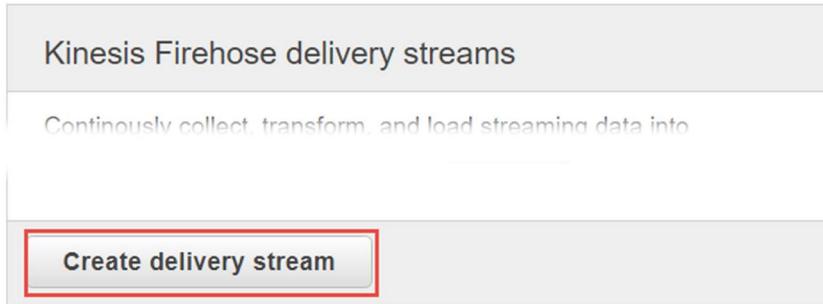
Offload to S3

For Amazon S3 destinations, streaming data is delivered to your S3 bucket. If data transformation is enabled, you can optionally back up source data to another Amazon S3 bucket.



Let's start with creating a new Firehose Delivery stream.

Press Create delivery stream on the Kinesis dashboard.



New delivery stream



Delivery streams load data to the destinations that you specify, automatically and continuously. Kinesis Firehose resources are not covered under the [AWS Free Tier](#), and **usage-based charges apply**. For more information, see [Kinesis Firehose pricing](#).

Delivery stream name*	<input type="text" value="my-traffic-message-data"/>
<small>Acceptable characters are uppercase and lowercase letters, numbers, underscores, hyphens, and periods.</small>	

Enter a Delivery stream name: **{your alias}-traffic-message-data**

For the source there are 2 options:

1. Direct PUT or other sources
2. Kinesis stream

The first option allows you to directly put records on using a custom producer or from AWS IoT, Cloudwatch logs or events.

Using the second option we take an existing Kinesis stream as source.

Choose the second option.

Source*	<input type="radio"/> Direct PUT or other sources <small>Choose this option to send records directly to the delivery stream, or to send records from AWS IoT, CloudWatch Logs, or CloudWatch events.</small>
	<input checked="" type="radio"/> Kinesis stream
Kinesis stream*	<input type="button" value="Choose Kinesis stream"/> <input type="button" value="Create new"/>

Select your stream from the dropdown box.

Press Next.

Transform records with AWS Lambda



Kinesis Firehose can invoke a Lambda function and transform source records before delivery. To return transformed records to Kinesis Firehose, your Lambda function must be compliant with the required record transformation output model.

Record transformation*	<input checked="" type="radio"/> Disabled <input type="radio"/> Enabled
------------------------	--

* Required

[Cancel](#)

[Previous](#)

[Next](#)

You can optionally transform the records using AWS Lambda before the records are delivered on the destination.

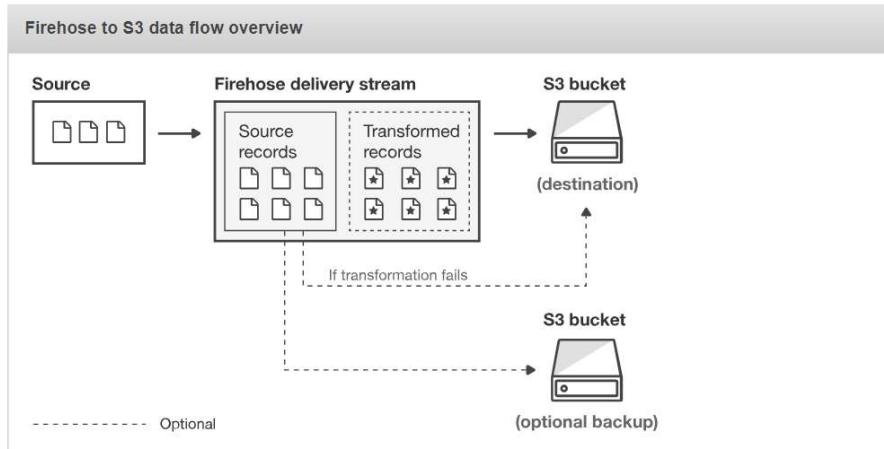
Select Disabled and press Next.

Select destination



Destination*	<input checked="" type="radio"/> Amazon S3 <input type="radio"/> Amazon Redshift <input type="radio"/> Amazon Elasticsearch Service <input type="radio"/> Splunk
--------------	---

Firehose to S3 data flow overview



S3 destination

S3 bucket*	<input style="border: 1px solid #ccc; padding: 2px 5px;" type="button" value="Choose bucket"/> ▼	<input style="border: 1px solid #0070C0; background-color: #0070C0; color: white; padding: 2px 5px;" type="button" value="Create new"/>
View bucket in S3 console		
Prefix	<input style="width: 150px; border: 1px solid #ccc; padding: 2px;" type="text" value="Specify prefix"/> 	

* Required

[Cancel](#)

[Previous](#)

[Next](#)

Now select the desired destination. In our case we select Amazon S3.

Select as S3 bucket: **amis-traffic-message-data**

S3 destination

S3 bucket* amis-traffic-message-data

Prefix i Firehose automatically appends the "YYYY/MM/DD/HH/" UTC prefix to delivered S3 files. You can also specify an extra prefix in front of the time format and add "/" to the end to have it appear as a folder in the S3 console.

Enter as Prefix: **{your alias}/**

Add the slash at the end to create a subfolder in the S3 bucket with your alias.

Press Next

Configure settings



Configure buffer, compression, logging, and IAM role settings for your delivery stream.

S3 buffer conditions

Firehose buffers incoming records before delivering them to your S3 bucket. Record delivery will be triggered once either of these conditions has been satisfied. [Learn more](#)

Buffer size* MB

Specify a buffer size between 1-128MB

Buffer interval* seconds

Specify a buffer interval between 60-900 seconds

S3 compression and encryption

Firehose can compress records before delivering them to your S3 bucket. Compressed records can also be encrypted in the S3 bucket using a KMS master key. [Learn more](#)

S3 compression* Disabled
 GZIP
 Snappy
 Zip

S3 encryption* Disabled
 Enabled

Error logging

Firehose can log record delivery errors to CloudWatch Logs. If enabled, a CloudWatch log group and corresponding log streams are created on your behalf. [Learn more](#)

Error logging* Disabled
 Enabled

IAM role

Firehose uses an IAM role to access your specified resources, such as the S3 bucket and KMS key. [Learn more](#)

IAM role*

* Required

Cancel

Previous

Next

Choose the buffer conditions. These are the conditions under which data is being offloaded.

The default settings is 5 MB or 5 minutes of data.

Just leave the default settings.

We will do no compression or encryption.

Leave Error logging enabled.

A (new) IAM role is to be created or chosen. The role is used to grant Kinesis Data Firehose access to your S3 bucket, AWS KMS key (if data encryption is enabled), and Lambda function (if data transformation is enabled).

Note the IAM role is create for you (you need IAM privileges to create the role and edit the policy).

Press the button.

Select the IAM role: **kinesis_firehose_delivery_role**

Select the Policy Name: (the only one)

Press Allow.

Press Next.

Press Create delivery stream

You can send some data to the stream and after at least 5 minutes, check the S3 bucket amis-traffic-message-data: <https://s3.console.aws.amazon.com/s3/buckets/the-bucket/?region=us-east-1&tab=overview>.

You will see a folder with your alias.

Name	Last modified	Size	Storage class
my	--	--	--

Click on the folder.

Then you will see a year folder (2018) and in that a month folder (05) and in that a day folder and then a hour folder (UTC time) and then finally your data.

The screenshot shows the Amazon S3 console with the path 'Amazon S3 > amis-traffic-message-data / my / 2018 / 05 / 02 / 20'. The 'Overview' tab is selected. A search bar contains the placeholder 'Type a prefix and press Enter to search. Press ESC to clear.' Below it, there are buttons for 'Upload', '+ Create folder', and 'More'. The location 'US East (N. Virginia)' is shown with a link icon. A table lists one file: 'my-traffic-message-data-1-2018-05-02-20-34-21-ae1cfeae-6315-44f7-a414-e0639dea8710'. The table includes columns for Name, Last modified, Size, and Storage class. The file was last modified on May 2, 2018, at 10:39:23 PM GMT+0200, has a size of 71.0 KB, and is stored in the Standard storage class. The status 'Viewing 1 to 1' is displayed at the bottom.

Click on the file.

The screenshot shows the file properties for the selected file. The tabs at the top are 'Overview' (selected), 'Properties' (highlighted in dark blue), 'Permissions', and 'Select from'. Below are buttons for 'Open', 'Download', 'Download as', 'Make public', and 'Copy path'. The file details are as follows:

- Owner:** engineering
- Last modified:** May 2, 2018 10:39:23 PM GMT+0200
- Etag:** f1543d4037097c10befa558bdf97845c
- Storage class:** Standard
- Server-side encryption:** None
- Size:** 72665
- Link:** <https://s3.amazonaws.com/amis-traffic-message-data/my/2018/05/02/20/my-traffic-message-data-1-2018-05-02-20-34-21-ae1cfeae-6315-44f7-a414-e0639dea8710>

Press Open or Download to see the contents of the file.

In the file you will see records which have been sent to your Kinesis Stream.

Stream Consumer

It is possible to create your own custom code to read directly from a Kinesis stream. This is called a stream consumer. In the traffic-kinesis-stream repository there is a file 'kinesis-stream-consumer.py'. You can see this Python script for an example to dump the data from the stream to a local json file.