

Building big data pipeline on AWS documentation

Sr. No.	Updated By	Updated On	Comments
1.	Divya Sistla	22/04/2020	First Draft

Version Information:

Table of Contents

Version Information:	2
Overview	4
Project execution guidelines	5
Dataflow Orchestration using Airflow	18
Tableau visualisations	Error! Bookmark not defined.
Tools used	25
Known errors and resolutions	25

Overview

The purpose is to collect the real time streaming data from COVID19 open API for every 5 minutes into the ecosystem using NiFi and to process it and store it in the data lake on AWS. Data processing includes parsing the data from complex JSON format to csv format then publishing to Kafka for persistent delivery of messages into PySpark for further processing. The processed data is then fed into output Kafka topic which is in turn consumed by NiFi and stored in HDFS. A Hive external table is created on top of HDFS processed data for which the process is Orchestrated using Airflow to run for every time interval. Finally KPIs are visualised in tableau.

Project execution guidelines

- As soon as we login into the EC2 instance , we need to start the following jps services using the commands .
- Note that everytime we stop and start the instance these services need to be restarted if they are not running as Daemon processes.

To run Hadoop:

```
root@ip-172-31-23-142:/home/ubuntu/hadoop_2.7.1# sbin/start-all.sh
```

Or

```
root@ip-172-31-23-142:/home/ubuntu/hadoop_2.7.1# sbin /start-dfs.sh
```

```
root@ip-172-31-23-142:/home/ubuntuhadoop_2.7.1# sbin/start-yarn.sh
```

To run nifi :

```
root@ip-172-31-23-142:/home/ubuntu# bin/nifi.sh start
```

To run NiFi in the Browser:

After running the vncserver, open the browser Then type

<http://localhost:9999/nifi>

To run kafka:

```
root@ip-172-31-23-142:/home/ubuntu/kafka/ bin/kafka-server-start.sh  
config/server.properties
```

Before run the NiFi GUI browser we start the vnc server

To start vncserver :

```
root@ip-172-31-23-142:/home/ubuntu# vncserver :1
```

To run Airflow:

Running Airflow needs two terminals to be opened in parallel .

Need to login into the ec-2 instance into both the terminals .

Then go into the root folder by specifying:

```
root@ip-172-31-23-142: cd ~
```

Then into airflow folder by:

```
root@ip-172-31-23-142: cd Airflow
```

Then run each of these commands in different terminals and keep them running

```
root@ip-172-31-23-142: airflow webserver -p 8080
```

```
root@ip-172-31-23-142:airflow scheduler
```

Once all the processes are started , we can check the command jps in another terminal

```
root@ip-172-31-23-142:/home/ubuntu/kafka_2.11-2.4.0# bin/kafka-server-start.sh -daemon config/server.properties
root@ip-172-31-23-142:/home/ubuntu/kafka_2.11-2.4.0# jps
7488 Jps      Instance: i-08798744262275934   Public DNS: ec2-3-21-156-70.us-east-2.compute.amazonaws.com
7185 NameNode
8052 NodeManager
32358 NiFi
1670 QuorumPeerMain
32327 RunNiFi
7736 ResourceManager
7354 DataNode
7565 SecondaryNameNode
7455 -- main class information unavailable
root@ip-172-31-23-142:/home/ubuntu/kafka_2.11-2.4.0# jps
7185 NameNode
8052 NodeManager
32358 NiFi
1670 QuorumPeerMain
32327 RunNiFi
7736 ResourceManager
7354 DataNode
7627 Jps
7565 SecondaryNameNode
7455 Kafka
root@ip-172-31-23-142:/home/ubuntu/kafka_2.11-2.4.0#
```

Instance ID	i-08798744262275934	Public DNS (IPv4)	ec2-3-21-156-70.us-east-2.c
Instance state	running	IPv4 Public IP	3.21.156.70
Instance type	t2.xlarge	IPv6 IPs	-
Finding	Opt-in to AWS Compute Optimizer for recommendations. Learn more	Elastic IPs	-
Private DNS	ip-172-31-23-142.us-east-2.compute.internal	Availability zone	us-east-2b
Private IPs	172.31.23.142	Security groups	launch-wizard-1, view inbound
Scheduled events	No scheduled events	AMI ID	ubuntu/images/hvm-ssd/ubun
AMI ID	ubuntu/images/hvm-ssd/ubun	Platform details	-
Subnet ID	subnet-095bfe73	Usage operation	-
Network interfaces	eth0	Source/dest. check	True
IAM role	-	T2/T3 Unlimited	Disabled
Key pair name	dezyre-keypair	EBS-optimized	False
Owner	143178219551	Root device type	efs
Launch time	April 21, 2020 at 6:59:37 PM UTC+5:30 (less than one hour)	Root device	/dev/sda1
Termination protection	False	Block devices	/dev/sda1
Lifecycle	normal	Elastic Graphics ID	-
Monitoring	basic	Elastic Inference accelerator ID	-
Capacity Reservation	-		

The next step is to open the VNC server and Firefox browser in it, and give localhost:9999/nifi to open the NiFi UI

Here, as a first step , we extract the data from Corona Open API endpoint using

- **InvokeHttp processor** -Drag and drop a new processor and search for Invokehttp and click on configure and specify details as follows:

Configure Processor

Stopped

SETTINGS SCHEDULING PROPERTIES COMMENTS

Required field +

Property	Value
HTTP Method	GET
Remote URL	https://api.covid19api.com/summary
SSL Context Service	No value set
Connection Timeout	5 secs
Read Timeout	15 secs
Include Date Header	True
Follow Redirects	True
Attributes to Send	No value set
Basic Authentication Username	No value set
Basic Authentication Password	No value set
Proxy Configuration Service	No value set
Proxy Host	No value set

CANCEL APPLY

- Then run the processor and the raw data is received in nested JSON format (JSON array and JSON objects) from the COVID 19 API endpoint for every 5 minutes as scheduled in the processor.

- Then, evaluate the fields which are as JSON object using EvaluateJSON processor with the following configuration

Configure Processor

Stopped

SETTINGS

SCHEDULING

PROPERTIES

COMMENTS

Required field

+

Property	Value	
Destination	<div>?</div> flowfile-attribute	
Return Type	<div>?</div> auto-detect	
Path Not Found Behavior	<div>?</div> ignore	
Null Value Representation	<div>?</div> empty string	
Global_new_confirmed	<div>?</div> \$.Global.NewConfirmed	<div>🗑</div>
Global_new_deaths	<div>?</div> \$.Global.NewDeaths	<div>🗑</div>
Global_new_recovered	<div>?</div> \$.Global.NewRecovered	<div>🗑</div>
Global_total_confirmed	<div>?</div> \$.Global.TotalConfirmed	<div>🗑</div>
Global_total_deaths	<div>?</div> \$.Global.TotalDeaths	<div>🗑</div>
Global_total_recovered	<div>?</div> \$.Global.TotalRecovered	<div>🗑</div>

CANCEL

APPLY

- To evaluate fields which are as JSON arrays , splitJSON processor is used

Configure Processor

■ Stopped

SETTINGS

SCHEDULING

PROPERTIES

COMMENTS

Required field



Property		Value
JsonPath Expression		\$..Countries
Null Value Representation		empty string

CANCEL

APPLY

- Then we use evalauteJSON again to extract the individual fields from JSON array which was split by the array

Configure Processor

Stopped

SETTINGS SCHEDULING **PROPERTIES** COMMENTS

Required field +

Property		Value	
Path Not Found Behavior	?	Ignore	
Null Value Representation	?	empty string	
Country_code	?	\$.CountryCode	🗑
Country_name	?	\$.Country	🗑
Country_new_deaths	?	\$.NewDeaths	🗑
Country_new_recovered	?	\$.NewRecovered	🗑
Country_newconfirmed	?	\$.NewConfirmed	🗑
Country_slug	?	\$.Slug	🗑
Country_total_confirmed	?	\$.TotalConfirmed	🗑
Country_total_deaths	?	\$.TotalDeaths	🗑
Country_total_recovered	?	\$.TotalRecovered	🗑
Extracted_timestamp	?	\$.Date	🗑

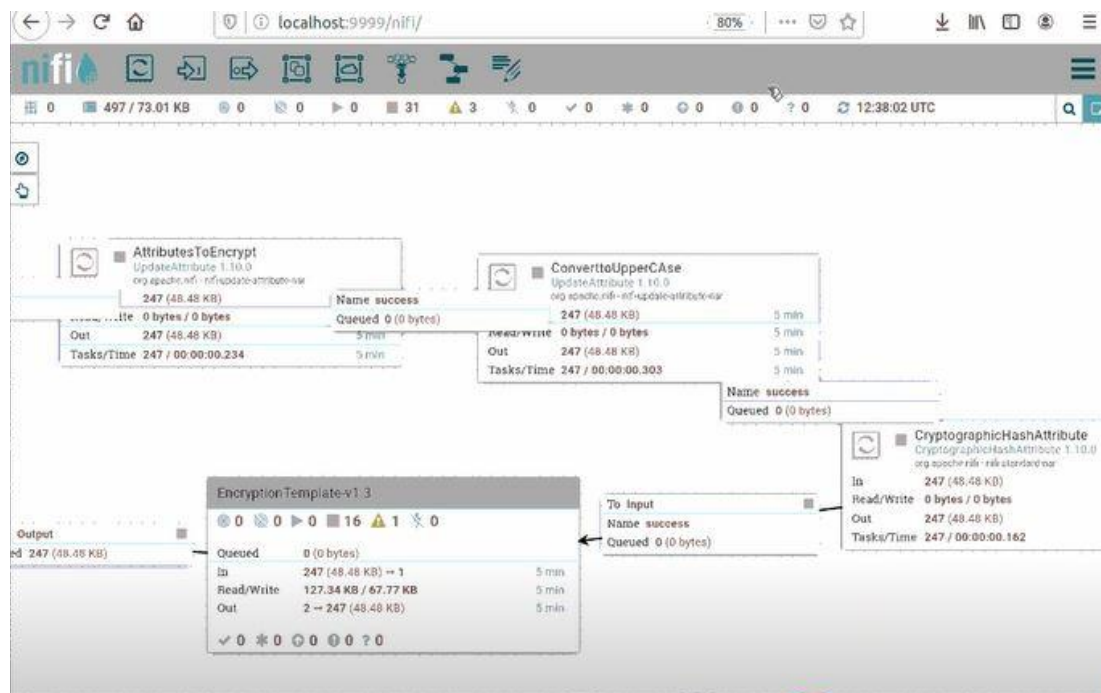
CANCEL APPLY

Encryption of Pii fields in the Data using Nifi

Data encryption of Pii fields is necessary for Data security purposes in large scale distribution environments. So, we've encrypted one of the fields as a sample to show how the encryption can be carried out using NiFi .

We would need three processors for the encryption purpose

- AttributestoEncrypt
- Cryptographic Hash Attribute
- EncryptContent
- Encryption template which needs to be created using set of processors for encrypting the data which needs to be imported as XML into NiFi from settings->templates->upload template and we need to upload the given XML here->drag and drop the template icon in NiFi and choose the latest uploaded template .



- Now, parse the values into CSV format using ReplaceText processor with commas as follows:

Configure Processor

Stopped

SETTINGS SCHEDULING PROPERTIES COMMENTS

Required field

Property

Search Value

Replacement Value

Character Set

Maximum Buffer Size

Replacement Strategy

Evaluation Mode

Line-by-Line Evaluation Mode

1

`1 ${Global_new_confirmed},${Global_new_deaths},${Global_new_recovered},${Global_total_confirmed},${Global_total_deaths},${Global_total_recovered}.`

☐ Set empty string

CANCEL OK

Configure Processor

Stopped

SETTINGS SCHEDULING PROPERTIES COMMENTS

Required field

Property

Value

Search Value

Replacement Value

Character Set

Maximum Buffer Size

Replacement Strategy

Evaluation Mode

Line-by-Line Evaluation Mode

(?s)(^,.*\$)

`${Global_new_confirmed},${Global_new_deaths},${Global_new_recovered},${Global_total_confirmed},${Global_total_deaths},${Global_total_recovered}.`

UTF-8

1 MB

Regex Replace

Entire text

All

☐ Set empty string

CANCEL APPLY

- Now the parsed data in parallel goes into HDFS as well as Kafka for ease of use . For that we use PublishKafka and PutHDFS processors and are configured as follows :

PublishKafka

Configure Processor

Stopped

SETTINGS SCHEDULING **PROPERTIES** COMMENTS

Required field +

Property	Value
Kafka Brokers	localhost:9092
Security Protocol	PLAINTEXT
Kerberos Service Name	No value set
SSL Context Service	No value set
Topic Name	dezyre_data_csv
Delivery Guarantee	Best Effort
Kafka Key	No value set
Key Attribute Encoding	UTF-8 Encoded
Message Demarcator	No value set
Max Request Size	1 MB
Acknowledgment Wait Time	5 secs
Max Metadata Wait Time	5 sec
Default Properties	Default Properties

CANCEL APPLY

PutHDFS

Configure Processor

Stopped

SETTINGS SCHEDULING **PROPERTIES** COMMENTS

Required field +

Property	Value
Hadoop Configuration Resources	/home/ubuntu/hadoop-2.7.3/etc/hadoop/core-site.x...
Kerberos Credentials Service	No value set
Kerberos Principal	No value set
Kerberos Keytab	No value set
Kerberos Relogin Period	4 hours
Additional Classpath Resources	No value set
Directory	/dezyre_data/corana-table
Conflict Resolution Strategy	replace
Block Size	No value set
IO Buffer Size	No value set
Replication	No value set
Permissions umask	No value set
Default Properties	Default Properties

CANCEL APPLY

- Once the data is published to Kafka topic , we can view the data in the console using the following command:
- Navigate to bin folder where Kafka was installed .Then run the following command

bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --from-beginning --topic topic name

- The above command shows the data which is published into the topic using PublishKafka in NiFi

```
root@ip-172-31-23-142: /home/ubuntu/kafka_2.11-2.4.0# bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic dezyre_data_csv --from-beginning
70871,4940,21835,2470922,169952,645094,AX,ALA Aland Islands,0,0,0,ala-aland-islands,0,0,0,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,AF,Afghanistan,3,4,30,afghanistan,1026,36,135,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,AL,Albania,0,13,22,albania,584,26,327,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,DZ,Algeria,9,52,89,algeria,2718,384,1099,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,AS,American Samoa,0,0,0,american-samoa,0,0,0,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,AD,Andorra,1,13,4,andorra,717,37,248,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,AO,Angola,0,0,0,angola,24,2,6,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,AI,Anguilla,0,0,0,anguilla,0,0,0,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,AQ,Antarctica,0,0,0,antarctica,0,0,0,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,AG,Antigua and Barbuda,0,0,0,antigua-and-barbuda,23,3,3,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,AR,Argentina,4,28,102,argentina,2941,136,737,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,AM,Armenia,2,35,48,armenia,1339,22,580,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,AW,Aruba,0,0,0,aruba,0,0,0,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,AU,Australia,0,0,0,australia,6547,67,4124,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,AT,Austria,18,130,46,austria,14795,470,10631,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,AZ,Azerbaijan,0,79,38,azerbaijan,1436,19,791,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,BS,Bahamas,0,1,5,bahamas,60,9,11,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,BH,Bahrain,0,10,26,bahrain,1907,7,769,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,BO,Bangladesh,10,10,492,bangladesh,2948,101,85,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,BB,Barbados,0,2,0,barbados,75,5,19,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,BY,Belarus,4,20,1485,belarus,6264,51,514,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,BE,Belgium,145,138,1487,belgium,39983,5828,8895,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,BZ,Belize,0,0,0,belize,18,2,2,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,BJ,Benin,0,9,19,benin,54,1,27,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,BM,Bermuda,0,0,0,bermuda,0,0,0,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,BT,Bhutan,0,0,0,bhutan,5,0,2,2020-04-21T14:49:09Z
70871,4940,21835,2470922,169952,645094,BO,Bolivia,1,0,44,bolivia,564,33,31,2020-04-21T14:49:09Z
```

- The data which is now in Kafka topic , we extract this streaming data messages into PySpark for streaming data processing
- Python code can be found in the link below:
- https://docs.google.com/document/d/17gIkV58tIzTpw44ijJKcKUkiDDXMyVlw5v_8ESI4M54/edit?usp=sharing

- Before submitting the python code , ensure that you have downloaded the required jars in the below command and copy them into your ec2-instance from your local using scp command .
- Then run the python code in the spark bin folder using:

```
bin/spark-submit --packages org.apache.spark:spark-sql-kafka-0-10_2.11:2.2.0,org.apache.spark:spark-streaming-kafka-0-8-assembly_2.11:2.3.0 -
-jars /home/ubuntu/spark-streaming-kafka-0-10-assembly_2.11-2.4.5.jar,/home/ubuntu/spark-sql-kafka-0-10_2.11-2.4.5.jar,kafka-clients-2.3.0.jar
--master local[2] /home/ubuntu/test.py
```

```
root@ip-172-31-23-142:/home/ubuntu/spark-2.4.5-bin-hadoop2.7# bin/spark-submit --packages org.apache.spark:spark-sql-kafka-0-10_2.11:2.2.0,org.apache.spark:spark-streaming-kafka-0-8-assembly_2.11:2.3.0 --jars /home/ubuntu/spark-streaming-kafka-0-10-assembly_2.11-2.4.5.jar,/home/ubuntu/spark-sql-kafka-0-10_2.11-2.4.5.jar,kafka-clients-2.3.0.jar --master local[2] /home/ubuntu/test.py

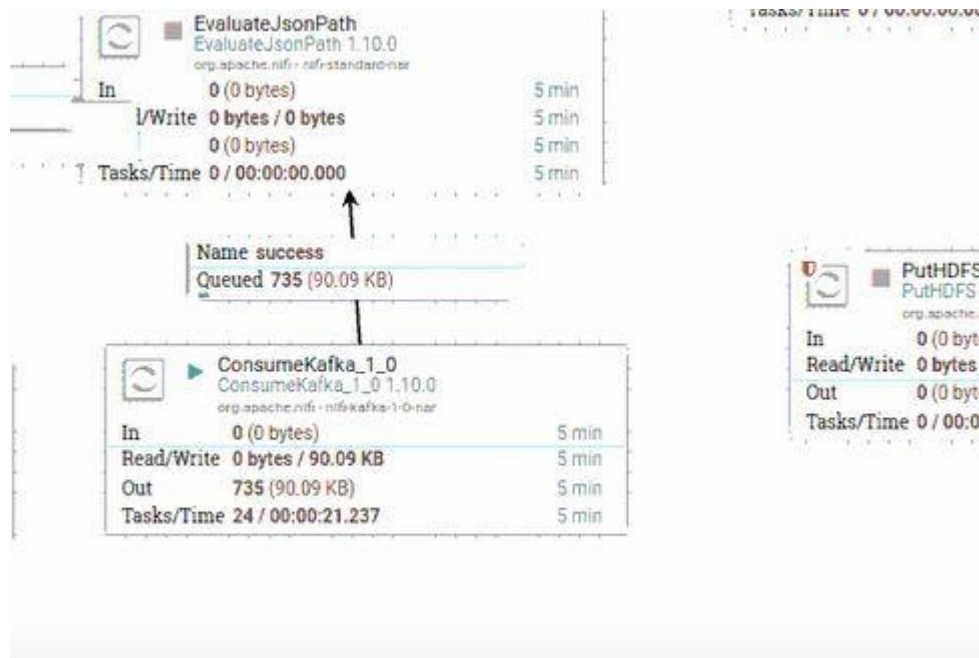
Ivy Default Cache set to: /root/.ivy2/cache
The jars for the packages stored in: /root/.ivy2/jars
:: loading settings :: url = jar:file:/home/ubuntu/spark-2.4.5-bin-hadoop2.7/jars/ivy-2.4.0.jar!/org/apache/ivy/core/settings/ivysettings.xml
org.apache.spark#spark-sql-kafka-0-10_2.11 added as a dependency
org.apache.spark#spark-streaming-kafka-0-8-assembly_2.11 added as a dependency
:: resolving dependencies :: org.apache.spark#spark-submit-parent-11434798-b6e0-46fb-89a6-662e0e1688f2;1.0
  confs: [default]
  found org.apache.spark#spark-sql-kafka-0-10_2.11;2.2.0 in central
  found org.apache.kafka#kafka-clients;0.10.0.1 in central
  found net.jpountz.lz4#lz4;1.3.0 in central
  found org.xerial.snappy#snappy-java;1.1.2.6 in central
  found org.slf4j#slf4j-api;1.7.16 in central
  found org.spark-project.spark#unused;1.0.0 in central
  found org.apache.spark#spark-streaming-kafka-0-8-assembly_2.11;2.3.0 in central
:: resolution report :: resolve 2586ms :: artifacts dl 72ms
  modules in use:
    net.jpountz.lz4#lz4;1.3.0 from central in [default]
    org.apache.kafka#kafka-clients;0.10.0.1 from central in [default]
    org.apache.spark#spark-sql-kafka-0-10_2.11;2.2.0 from central in [default]
    org.apache.spark#spark-streaming-kafka-0-8-assembly_2.11;2.3.0 from central in [default]
    org.slf4j#slf4j-api;1.7.16 from central in [default]
    org.spark-project.spark#unused;1.0.0 from central in [default]
    org.xerial.snappy#snappy-java;1.1.2.6 from central in [default]
  -----
  |                  |  sq.setmodules rel('WAR') artifacts  |
```

- The above python code is writing data to Kafka output topic in JSON format and it can be verified in console via Kafka console consumer
- `bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --from-beginning --topic topic name`

```
root@ip-172-31-23-142: /home/ubuntu# cd kafka_2.11-2.4.0/
root@ip-172-31-23-142: /home/ubuntu/kafka_2.11-2.4.0# bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic dezyre_out --from-beginning

{"Country_code": "AQ", "Country_name": "Antarctica", "Country_total_deaths": "0", "Extracted_timestamp": "2020-04-20T14:45:38.000Z"}
{"Country_code": "AU", "Country_name": "Australia", "Country_total_deaths": "67", "Extracted_timestamp": "1970-01-01T00:00:00.021Z"}
{"Country_code": "LY", "Country_name": "Libya", "Country_total_deaths": "1", "Extracted_timestamp": "2020-04-20T15:25:59.000Z"}
{"Country_code": "KR", "Country_name": "Korea (South)", "Country_total_deaths": "234", "Extracted_timestamp": "2020-04-20T14:45:38.000Z"}
{"Country_code": "AR", "Country_name": "Argentina", "Country_total_deaths": "129", "Extracted_timestamp": "1970-01-01T00:00:00.021Z"}
{"Country_code": "BB", "Country_name": "Barbados", "Country_total_deaths": "5", "Extracted_timestamp": "2020-04-20T14:45:38.000Z"}
{"Country_code": "TK", "Country_name": "Tokelau", "Country_total_deaths": "0", "Extracted_timestamp": "2020-04-20T14:45:38.000Z"}
{"Country_code": "BG", "Country_name": "Bulgaria", "Country_total_deaths": "42", "Extracted_timestamp": "2020-04-20T15:25:59.000Z"}
{"Country_code": "SA", "Country_name": "Saudi Arabia", "Country_total_deaths": "97", "Extracted_timestamp": "2020-04-20T15:25:59.000Z"}
{"Country_code": "SL", "Country_name": "Sierra Leone", "Country_total_deaths": "0", "Extracted_timestamp": "2020-04-20T15:25:59.000Z"}
{"Country_code": "HU", "Country_name": "Hungary", "Country_total_deaths": "189", "Extracted_timestamp": "2020-04-20T14:45:38.000Z"}
{"Country_code": "SO", "Country_name": "Somalia", "Country_total_deaths": "7", "Extracted_timestamp": "2020-04-20T14:45:38.000Z"}
{"Country_code": "CR", "Country_name": "Costa Rica", "Country_total_deaths": "5", "Extracted_timestamp": "2020-04-20T14:45:38.000Z"}
{"Country_code": "SY", "Country_name": "Syrian Arab Republic (Syria)", "Country_total_deaths": "3", "Extracted_timestamp": "2020-04-20T15:25:59.000Z"}
{"Country_code": "UZ", "Country_name": "Uzbekistan", "Country_total_deaths": "15", "Extracted_timestamp": "2020-04-20T14:45:38.000Z"}
{"Country_code": "DO", "Country_name": "Dominican Republic", "Country_total_deaths": "226", "Extracted_timestamp": "2020-04-20T14:45:38.000Z"}
{"Country_code": "SG", "Country_name": "Singapore", "Country_total_deaths": "11", "Extracted_timestamp": "1970-01-01T00:00:00.018Z"}
{"Country_code": "IN", "Country_name": "India", "Country_total_deaths": "521", "Extracted_timestamp": "1970-01-01T00:00:00.020Z"}
{"Country_code": "DO", "Country_name": "Dominican Republic", "Country_total_deaths": "226", "Extracted_timestamp": "2020-04-20T15:25:59.000Z"}
{"Country_code": "PE", "Country_name": "Peru", "Country_total_deaths": "348", "Extracted_timestamp": "1970-01-01T00:00:00.017Z"}
{"Country_code": "CH", "Country_name": "Switzerland", "Country_total_deaths": "1368", "Extracted_timestamp": "1970-01-01T00:00:00.018Z"}
{"Country_code": "HN", "Country_name": "Honduras", "Country_total_deaths": "46", "Extracted_timestamp": "2020-04-20T14:45:38.000Z"}
{"Country_code": "SN", "Country_name": "Senegal", "Country_total_deaths": "3", "Extracted_timestamp": "1970-01-01T00:00:00.020Z"}
{"Country_code": "KH", "Country_name": "Cambodia", "Country_total_deaths": "0", "Extracted_timestamp": "2020-04-20T15:25:59.000Z"}
{"Country_code": "TL", "Country_name": "Timor-Leste", "Country_total_deaths": "0", "Extracted_timestamp": "1970-01-01T00:00:00.019Z"}
{"Country_code": "BZ", "Country_name": "Belize", "Country_total_deaths": "2", "Extracted_timestamp": "2020-04-20T14:45:38.000Z"}
{"Country_code": "BW", "Country_name": "Botswana", "Country_total_deaths": "1", "Extracted_timestamp": "1970-01-01T00:00:00.022Z"}
{"Country_code": "CV", "Country_name": "Cape Verde", "Country_total_deaths": "1", "Extracted_timestamp": "1970-01-01T00:00:00.022Z"}
{"Country_code": "EE", "Country_name": "Estonia", "Country_total_deaths": "40", "Extracted_timestamp": "2020-04-20T15:25:59.000Z"}
{"Country_code": "VU", "Country_name": "Vanuatu", "Country_total_deaths": "0", "Extracted_timestamp": "2020-04-20T14:45:38.000Z"}
{"Country_code": "KH", "Country_name": "Cambodia", "Country_total_deaths": "0", "Extracted_timestamp": "1970-01-01T00:00:00.019Z"}
{"Country_code": "LC", "Country_name": "Saint Lucia", "Country_total_deaths": "0", "Extracted_timestamp": "1970-01-01T00:00:00.017Z"}
{"Country_code": "CH", "Country_name": "Switzerland", "Country_total_deaths": "1393", "Extracted_timestamp": "2020-04-20T14:45:38.000Z"}
{"Country_code": "BY", "Country_name": "Belarus", "Country_total_deaths": "47", "Extracted_timestamp": "2020-04-20T14:45:38.000Z"}
{"Country_code": "MY", "Country_name": "Malaysia", "Country_total_deaths": "89", "Extracted_timestamp": "2020-04-20T14:45:38.000Z"}
```

- Now the data in the output Kafka topic is ready to consume by consumers like Nifi.
- We use ConsumeKafka processor to get the data for further processing



- Once the data is consumed by NiFi, We parse the data into CSV again and store it in HDFS.
- Then create a Hive external table using the below script in hive.
- We login into Hive by entering into hive terminal by going into the folder where apache-hive is installed and give the following command
- bin/hive which enters into Hive shell

```
releases.
hive> use default;
OK
Time taken: 0.777 seconds
hive> show tables;
OK
airflow_hive
sample
Time taken: 0.163 seconds, Fetched: 2 row(s)
hive> select * from sample;
OK
Time taken: 1.134 seconds
hive> select * from airflow_hive;
OK
1      bigdata
Time taken: 0.074 seconds, Fetched: 1 row(s)
hive>
```

<https://docs.google.com/document/d/1f79QCJc6n7nJaIqfxyF18u0-SNnULRy2GGiP6Qc/edit?usp=sharing>

We create external table to point to HDFS location.

- The Hive script to create table can be found at:

<https://docs.google.com/document/d/1f79QCJc6n7nJaIqfxyF18u0-SNnULRy2GGiP6Q91F1c/edit?usp=sharing>

- We create an external table to point to the HDFS location. Further processed tables could also be created as a managed table.

Dataflow Orchestration using Airflow

As a first step, we need to start the Airflow webserver and scheduler in separate terminals and keep them running

To run the airflow:

```
root@ip-172-31-23-142:~/airflow: airflow webserver -p 8080
```

```
root@ip-172-31-23-142:~/airflow: airflow scheduler
```

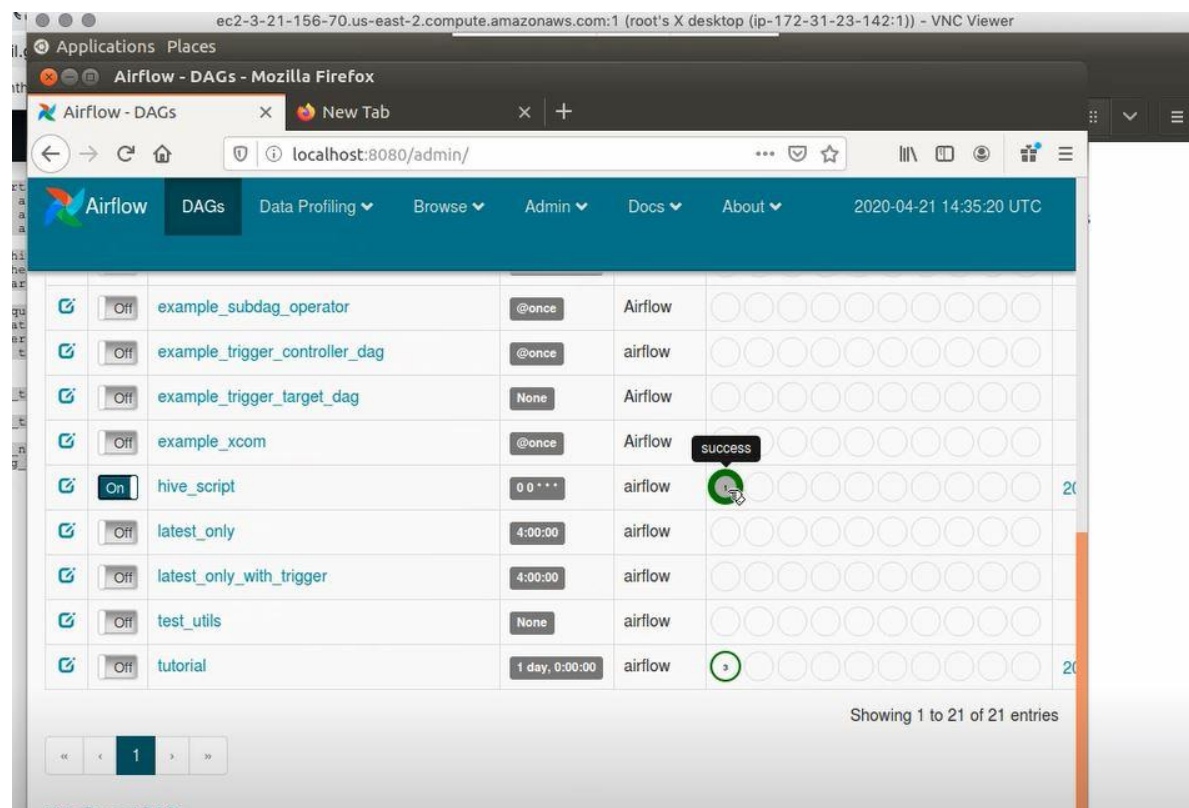
```
root@ip-172-31-23-142:~/airflow: airflow worker
```

To start the Airflow GUI:

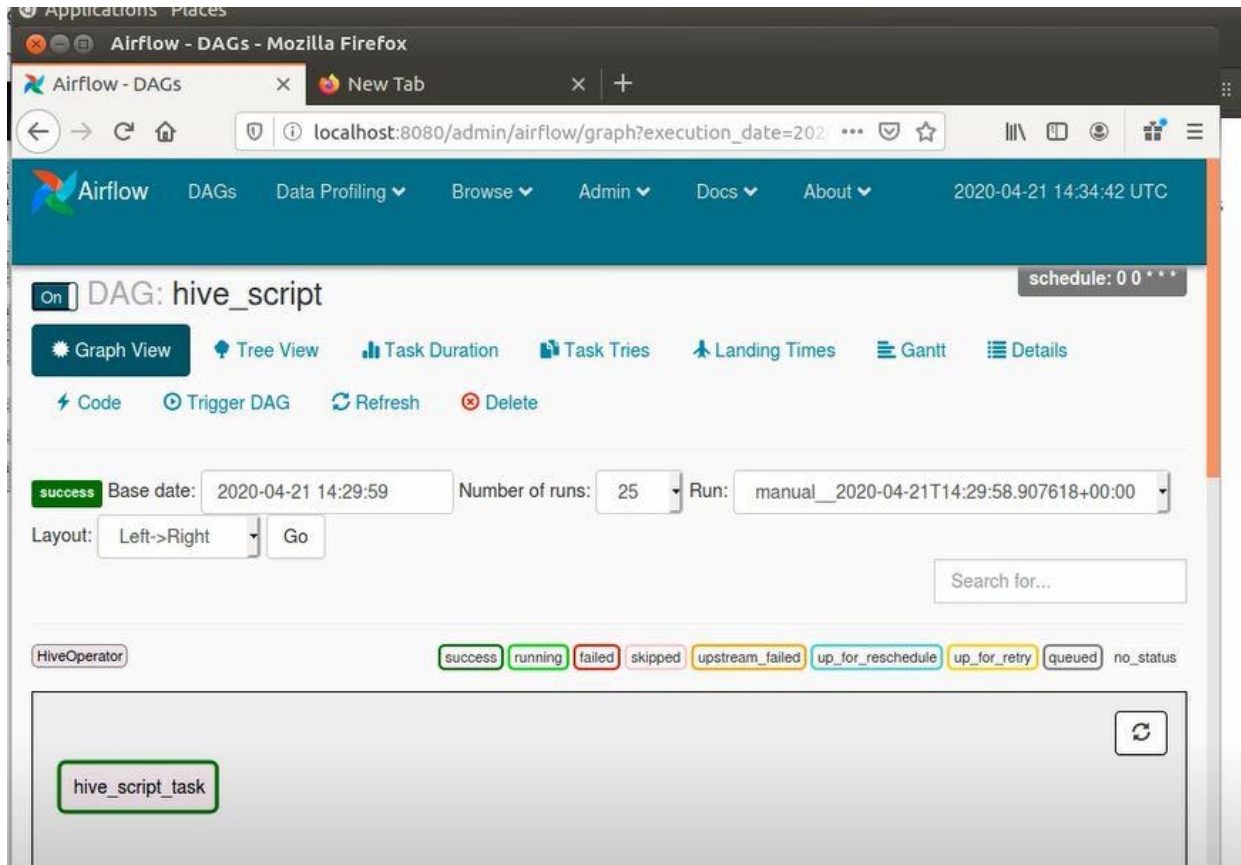
Goto the browser and then type <http://localhost:8080/admin>

Note: will give the python script dag file in the ~/airflow/dags folder.

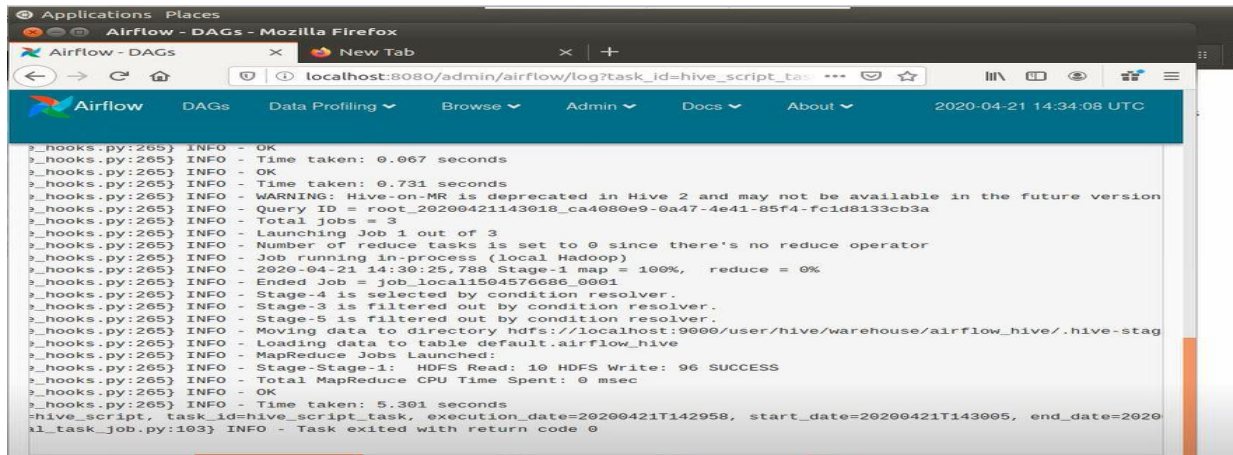
Then navigate to the Airflow UI in the VNC server browser by giving localhost:8080/admin which opens the airflow UI as follows :



Here in the above screen as you our dag appears in the browser the will the file and trigger the dag option as below pic.



And the logs for the task can be traced by clicking on the hive_script_task and view logs option in the pop up .The logs will be shown as follows:



The screenshot shows the Airflow web interface in a Mozilla Firefox browser. The address bar displays 'localhost:8080/admin/airflow/log?task_id=hive_script_ta...'. The interface includes a top navigation bar with 'Airflow', 'DAGs', 'Data Profiling', 'Browse', 'Admin', 'Docs', and 'About' menus, along with a timestamp '2020-04-21 14:34:08 UTC'. The main content area displays a log stream for a task. The log entries are as follows:

```

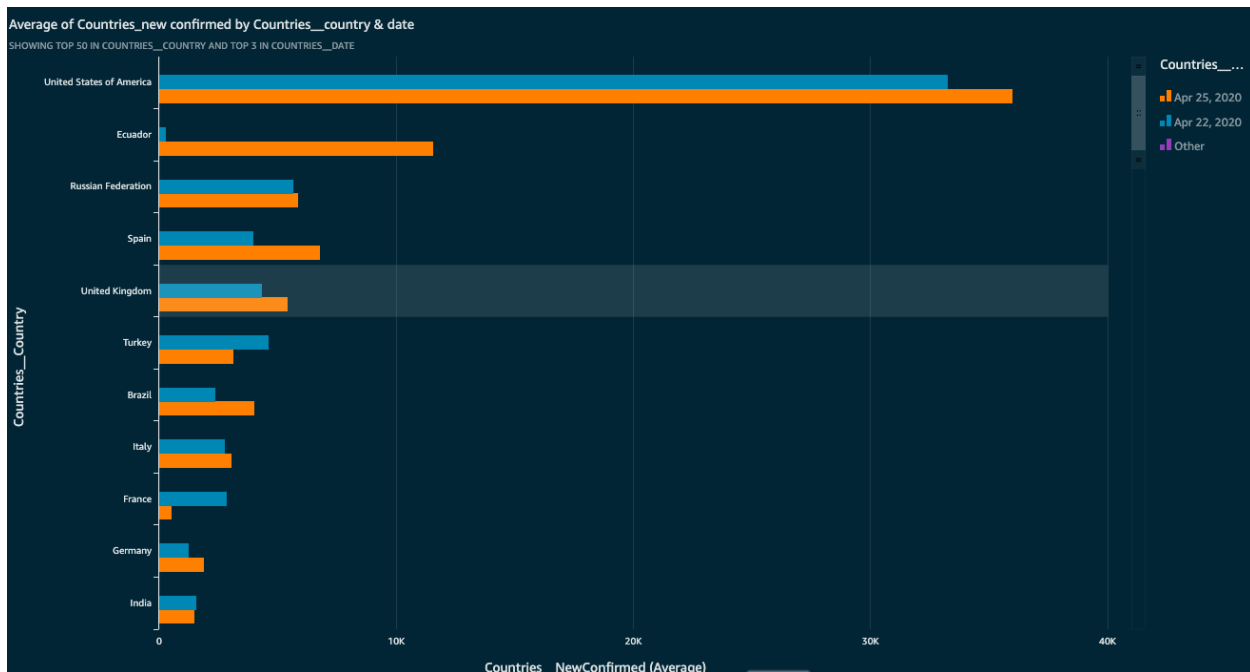
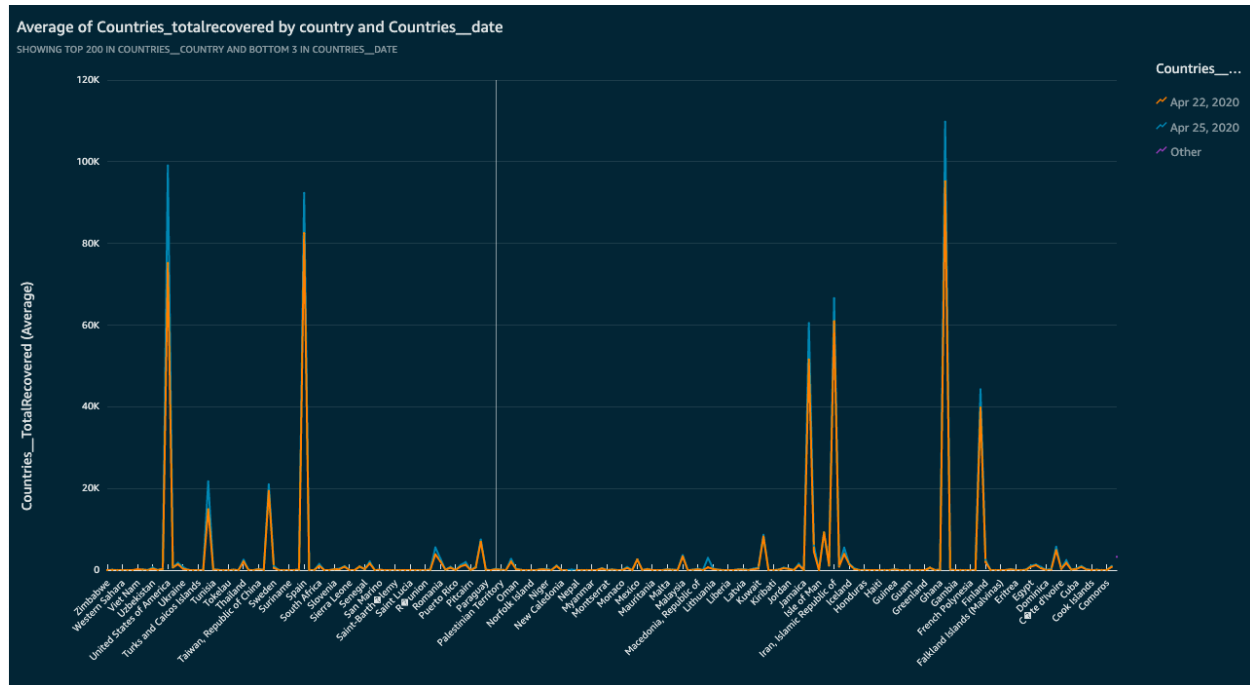
2_hooks.py:265} INFO - OK
2_hooks.py:265} INFO - Time taken: 0.067 seconds
2_hooks.py:265} INFO - OK
2_hooks.py:265} INFO - Time taken: 0.731 seconds
2_hooks.py:265} INFO - WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future version
2_hooks.py:265} INFO - Query ID = root_20200421143018_ca4080e9-0a47-4e41-85f4-fc1d8133cb3a
2_hooks.py:265} INFO - Total jobs = 3
2_hooks.py:265} INFO - Launching Job 1 out of 3
2_hooks.py:265} INFO - Number of reduce tasks is set to 0 since there's no reduce operator
2_hooks.py:265} INFO - Job running in-process (local Hadoop)
2_hooks.py:265} INFO - 2020-04-21 14:30:25,788 Stage-1 map = 100%, reduce = 0%
2_hooks.py:265} INFO - Ended Job = job_local1504576686_0001
2_hooks.py:265} INFO - Stage-4 is selected by condition resolver.
2_hooks.py:265} INFO - Stage-3 is filtered out by condition resolver.
2_hooks.py:265} INFO - Stage-5 is filtered out by condition resolver.
2_hooks.py:265} INFO - Moving data to directory hdfs://localhost:9000/user/hive/warehouse/airflow_hive/.hive-stag
2_hooks.py:265} INFO - Loading data to table default.airflow_hive
2_hooks.py:265} INFO - MapReduce Jobs Launched:
2_hooks.py:265} INFO - Stage-Stage-1: HDFS Read: 10 HDFS Write: 96 SUCCESS
2_hooks.py:265} INFO - Total MapReduce CPU Time Spent: 0 msec
2_hooks.py:265} INFO - OK
2_hooks.py:265} INFO - Time taken: 5.301 seconds
hive_script, task_id=hive_script_task, execution_date=20200421T142958, start_date=20200421T143005, end_date=2020
11_task_job.py:103} INFO - Task exited with return code 0

```

- The DAG should be deployed as a python file with the required tasks specified (Hive task, python task email task etc)
- Below is the link for the airflow python code which needs to be placed in airflow/dags folder

<https://docs.google.com/document/d/1cprVhn9lOVUimvs2A3nUPFP8i8BvwrZCONbah-5LmYc/edit?usp=sharing>

Visualisation of Data in AWS Quicksight



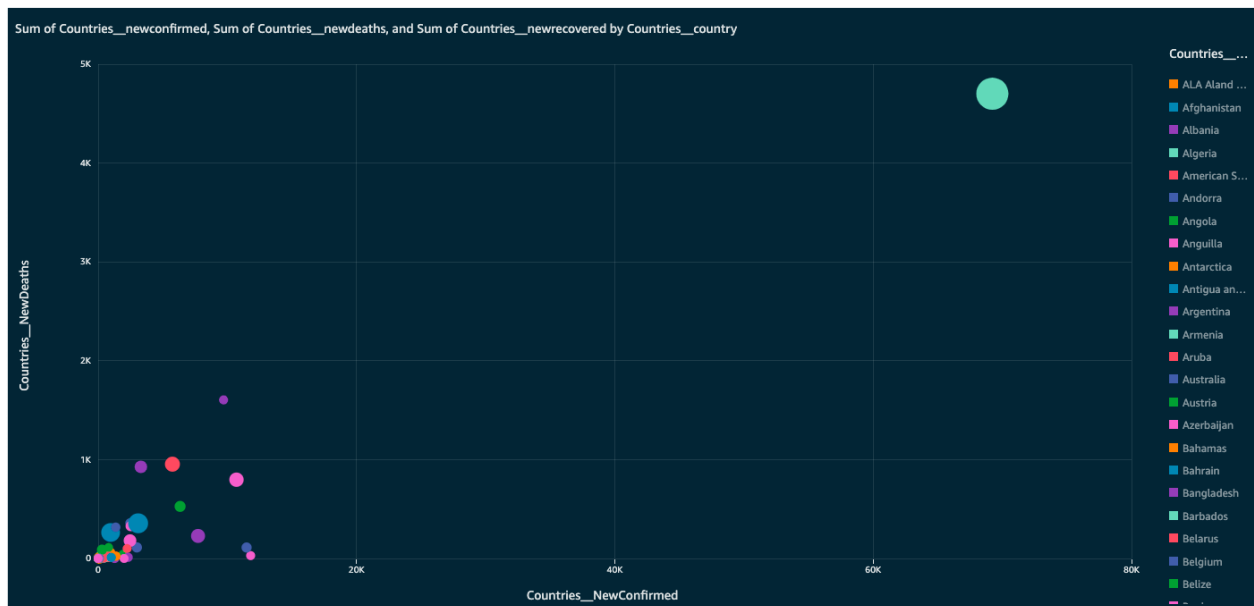
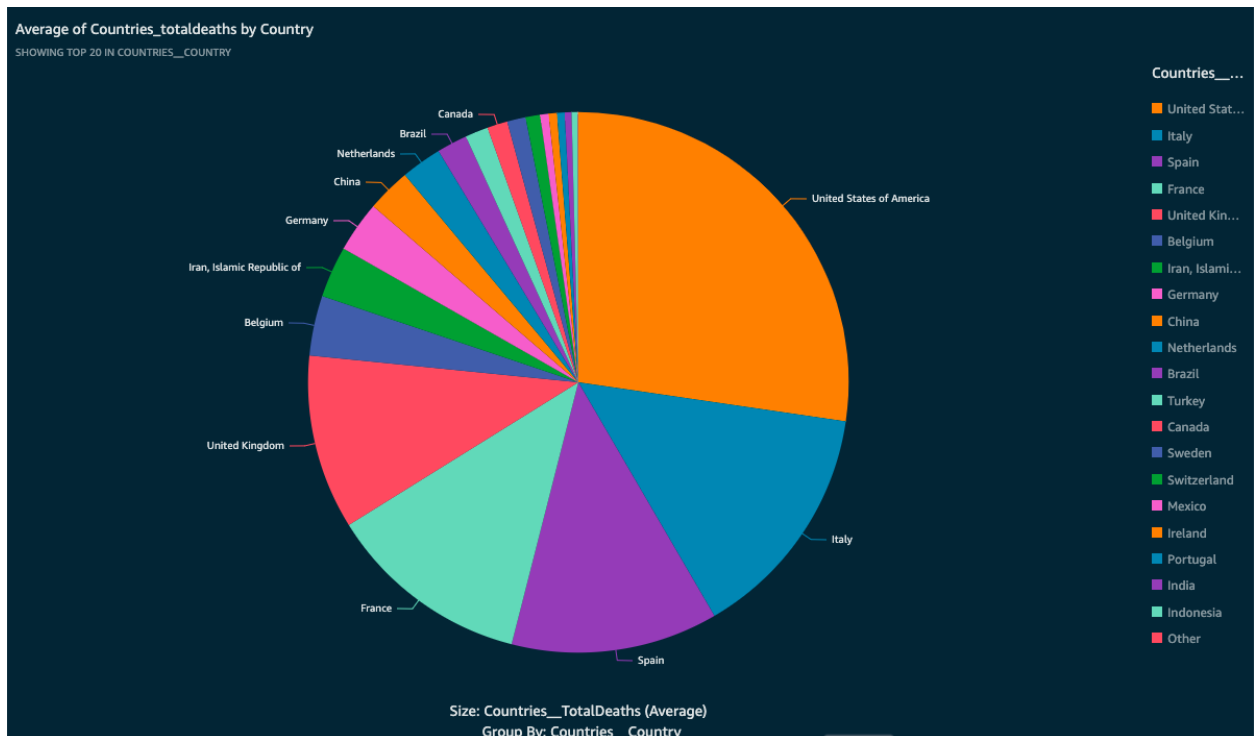
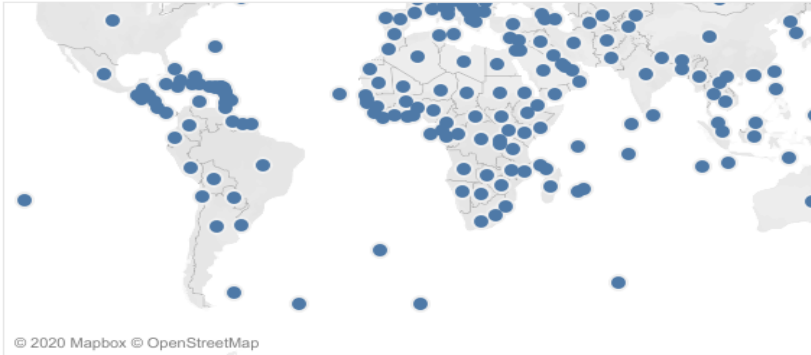
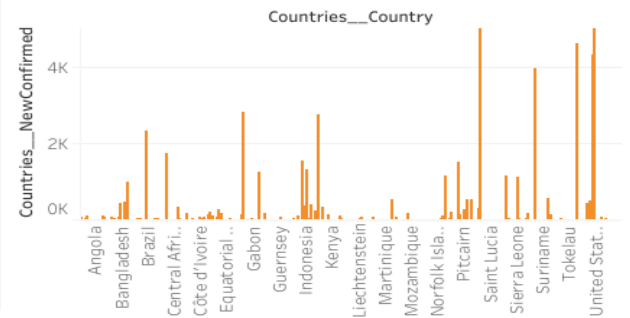


Tableau visualisations (Tableau online)

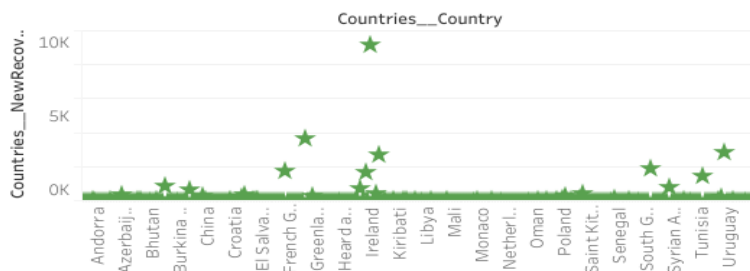
Countrywise total deaths



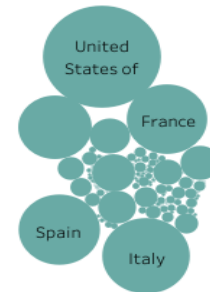
Countrywise new confirmed cases



Countrywise new deaths



Country analysis



Tools used

1. Nifi -nifi-1.10.0
2. Hadoop -hadoop_2.7.3
3. Hive-apache-hive-2.1.0
4. Spark-spark-2.4.5
5. Zookeeper-zookeeper-2.3.5
6. Kafka-kafka_2.11-2.4.0
7. Airflow-airflow-1.8.1
8. Tableau-

Known errors and resolutions

- Everytime consume kafka processor is run, the group id attribute needs to be changed and can be given some random text value
- When we see the below error while starting Hive :

Caused by: java.net.URISyntaxException: Relative path in absolute URI:
\${system:java.io.tmpdir%7D/\${%7Bsystem:user.name%7D

Then we need to add the following property at the beginning of hive-site.xml file

```
<property>

  <name>system:java.io.tmpdir</name>

  <value>/user/local/hive/tmp/java</value>

</property> <property>

  <name>system:user.name</name>

  <value>${user.name}</value>

</property>
```


THE END