BIG DATA PIPELINE FOR PASSENGER NAME RECORDS DATA

Subject: Big data storage and processing

Authors

1. Vu Trung Nghia - 20173284
2. Le Vu Loi - 20173240
3. Dang Lam San - 20170111

December 24, 2020

Outline

1. Introduction

2. System architecture

3. Execution results
   - Batch processing branch
   - Speed processing branch

4. Evaluation
   - KAFKA
   - HDFS
   - SPARK

5. Reference

1. Introduction
2. System architecture
3. Execution results
   - Batch processing branch
   - Speed processing branch
4. Evaluation
   - KAFKA
   - HDFS
   - SPARK
5. Reference

Problem formulation

- Passengers records often come in the form of time series data. On average, there are approximately 3.7 million bookings are performed each day on the targeted systems and therefore the traditional system is no longer sufficient.
- In this project, we built a distributed system and using docker to simulate. Our system has the following constraints:
  - Sufficient for storing and processing big data
  - Fault tolerant handling
  - Easy to expand
  - User can have two views: batch processing and real-time processing. In batch view, user can using SQL api and python to query and visualize data in HDFS, and in real-time view we simple logs the number of received records every 10 seconds.

Record specification

ArrivalTime – local time of arrival

BusinessLeisure – if the trip is for business or leisure

CabinCategory – cabin class

CreationDate –PNR creation date (Julian day)

CurrencyCode – 3-letter currency code of payment

DepartureTime – local time of departure

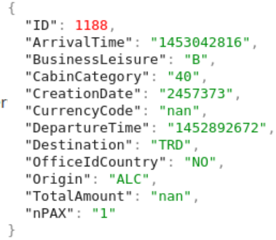
Destination – IATA code of arrival airport

OfficeIdCountry – country code of office placing the reser

Origin – IATA code of departure airport

TotalAmount – total reservation cost

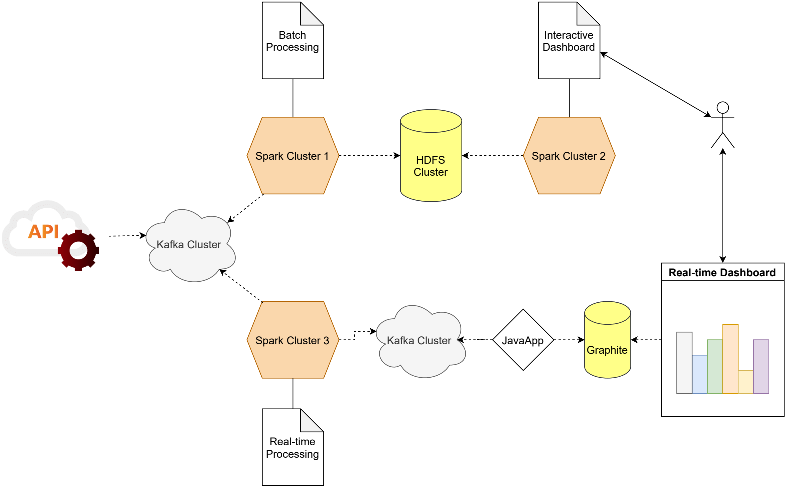
nPAX – number of passengers



```
{
  "ID": 1188,
  "ArrivalTime": "1453042816",
  "BusinessLeisure": "B",
  "CabinCategory": "40",
  "CreationDate": "2457373",
  "CurrencyCode": "nan",
  "DepartureTime": "1452892672",
  "Destination": "TRD",
  "OfficeIdCountry": "NO",
  "Origin": "ALC",
  "TotalAmount": "nan",
  "nPAX": "1"
}
```

1. Introduction
2. System architecture
3. Execution results
   - Batch processing branch
   - Speed processing branch
4. Evaluation
   - KAFKA
   - HDFS
   - SPARK
5. Reference

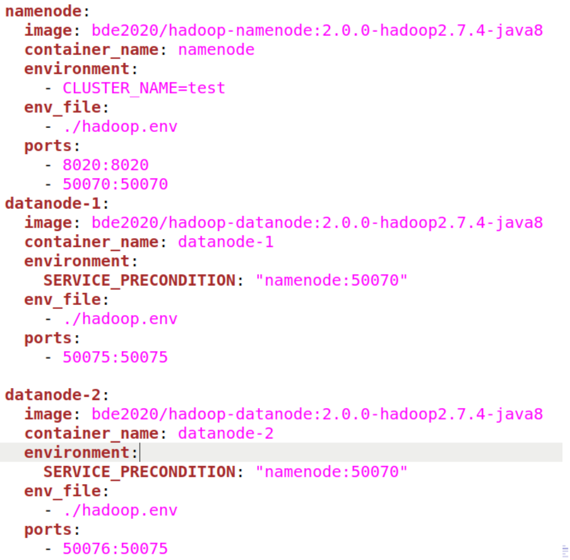
System architecture



System architecture

- PNR's API produces records to a topic in kafka cluster.
- Batch Processing program and Real-time Procesisng program subscribe to this topic, these programs use spark cluster 1 and spark cluster 3 as its computation resource.
- Batch Processing program processes data and stores to HDFS. Then user can use Interactive Dashboard which is a notebook to read data from HDFS and do some statistics. This notebook use spark cluster 2 as its computation resource.
- Real-time Processing program processes data and writes to another kafka topic, then the JavaApp subscribes to this topic to read data and write to Graphite database. Finally, Real-time Dashboard reads data from Graphite and visualizes.

HDFS Cluster



```
namenode:
  image: bde2020/hadoop-namenode:2.0.0-hadoop2.7.4-java8
  container_name: namenode
  environment:
    - CLUSTER_NAME=test
  env_file:
    - ./hadoop.env
  ports:
    - 8020:8020
    - 50070:50070
datanode-1:
  image: bde2020/hadoop-datanode:2.0.0-hadoop2.7.4-java8
  container_name: datanode-1
  environment:
    SERVICE_PRECONDITION: "namenode:50070"
  env_file:
    - ./hadoop.env
  ports:
    - 50075:50075

datanode-2:
  image: bde2020/hadoop-datanode:2.0.0-hadoop2.7.4-java8
  container_name: datanode-2
  environment:
    SERVICE_PRECONDITION: "namenode:50070"
  env_file:
    - ./hadoop.env
  ports:
    - 50076:50075
```

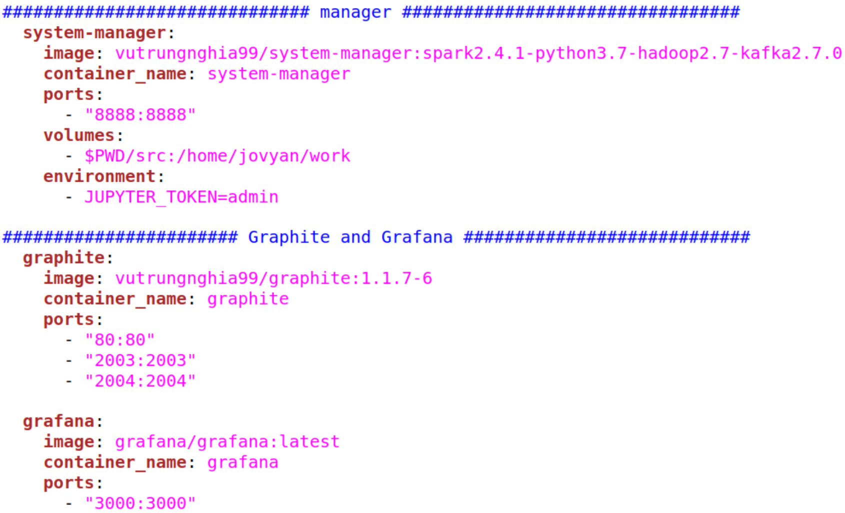
Kafka Cluster

```yaml
zookeeper:
  image: zookeeper:3.4.10
  container_name: zookeeper
  environment:
    ZOO_MY_ID: 1
    ZOO_SERVERS: server.1=0.0.0.0:2888:3888
    ZOO_TICK_TIME: 15000
  ports:
    - 2181:2181

kafka-broker-1:
  image: wurstmeister/kafka:2.12-2.2.1
  container_name: kafka-broker-1
  depends_on:
    - zookeeper
  ports:
    - "9092:9092"
  environment:
    - KAFKA_ZOOKEEPER_CONNECT=zookeeper:2181
    - ALLOW_PLAINTEXT_LISTENER=yes
    - KAFKA_ADVERTISED_LISTENERS=INSIDE://kafka-broker-1:9093,OUTSIDE://localhost:9092
    - KAFKA_LISTENER_SECURITY_PROTOCOL_MAP=INSIDE:PLAINTEXT,OUTSIDE:PLAINTEXT
    - KAFKA_LISTENERS=INSIDE://kafka-broker-1:9093,OUTSIDE://0.0.0.0:9092
    - KAFKA_INTER_BROKER_LISTENER_NAME=INSIDE
kafka-broker-2:
  image: wurstmeister/kafka:2.12-2.2.1
  container_name: kafka-broker-2
  depends_on:
    - zookeeper
  ports:
    - "9094:9094"
  environment:
    - KAFKA_ZOOKEEPER_CONNECT=zookeeper:2181
    - ALLOW_PLAINTEXT_LISTENER=yes
    - KAFKA_ADVERTISED_LISTENERS=INSIDE://kafka-broker-2:9093,OUTSIDE://localhost:9094
    - KAFKA_LISTENER_SECURITY_PROTOCOL_MAP=INSIDE:PLAINTEXT,OUTSIDE:PLAINTEXT
    - KAFKA_LISTENERS=INSIDE://kafka-broker-2:9093,OUTSIDE://0.0.0.0:9094
    - KAFKA_INTER_BROKER_LISTENER_NAME=INSIDE
```

Spark Cluster

```yaml
post-batch-processing-spark-master:
  image: vutrungnghia99/spark-master:spark2.4.1-python3.7-hadoop2.7
  container_name: post-batch-processing-spark-master
  ports:
    - "8083:8080"
    - "7078:7077"
  environment:
    - INIT_DAEMON_STEP=setup_spark
post-batch-processing-spark-worker-1:
  image: vutrungnghia99/spark-worker:spark2.4.1-python3.7-hadoop2.7
  container_name: post-batch-processing-spark-worker-1
  depends_on:
    - post-batch-processing-spark-master
  environment:
    - "SPARK_MASTER=spark://post-batch-processing-spark-master:7077"
    - "SPARK_WORKER_CORES=1"
    - "SPARK_WORKER_MEMORY=1G"
    - "SPARK_DRIVER_MEMORY=128m"
    - "SPARK_EXECUTOR_MEMORY=256m"
post-batch-processing-spark-worker-2:
  image: vutrungnghia99/spark-worker:spark2.4.1-python3.7-hadoop2.7
  container_name: post-batch-processing-spark-worker-2
  depends_on:
    - post-batch-processing-spark-master
  environment:
    - "SPARK_MASTER=spark://post-batch-processing-spark-master:7077"
    - "SPARK_WORKER_CORES=1"
    - "SPARK_WORKER_MEMORY=1G"
    - "SPARK_DRIVER_MEMORY=128m"
    - "SPARK_EXECUTOR_MEMORY=256m"
```

Systen manager - Graphite - Grafana

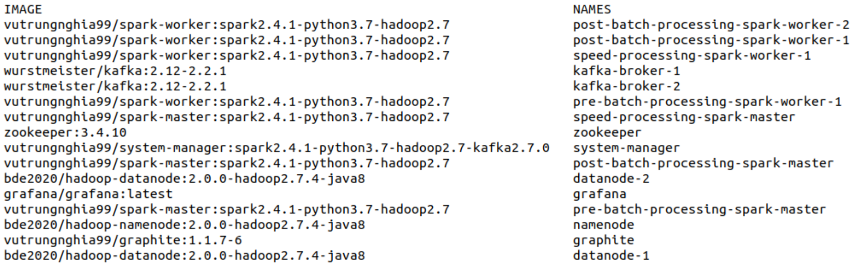


```
############################## manager #################################
  system-manager:
    image: vutrungnghia99/system-manager:spark2.4.1-python3.7-hadoop2.7-kafka2.7.0
    container_name: system-manager
    ports:
      - "8888:8888"
    volumes:
      - $PWD/src:/home/jovyan/work
    environment:
      - JUPYTER_TOKEN=admin

####################### Graphite and Grafana ############################
  graphite:
    image: vutrungnghia99/graphite:1.1.7-6
    container_name: graphite
    ports:
      - "80:80"
      - "2003:2003"
      - "2004:2004"

  grafana:
    image: grafana/grafana:latest
    container_name: grafana
    ports:
      - "3000:3000"
```

1. Introduction
2. System architecture
3. Execution results
   - Batch processing branch
   - Speed processing branch
4. Evaluation
   - KAFKA
   - HDFS
   - SPARK
5. Reference

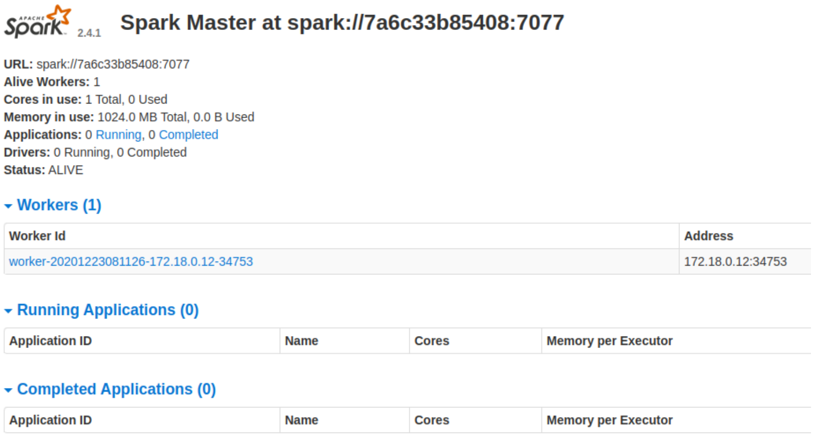
1. Introduction
2. System architecture
3. Execution results
   - Batch processing branch
   - Speed processing branch
4. Evaluation
   - KAFKA
   - HDFS
   - SPARK
5. Reference

List of containers [16]



```
IMAGE                                                                     NAMES
vutrungnghia99/spark-worker:spark2.4.1-python3.7-hadoop2.7                post-batch-processing-spark-worker-2
vutrungnghia99/spark-worker:spark2.4.1-python3.7-hadoop2.7                post-batch-processing-spark-worker-1
vutrungnghia99/spark-worker:spark2.4.1-python3.7-hadoop2.7                speed-processing-spark-worker-1
wurstmeister/kafka:2.12-2.2.1                                             kafka-broker-1
wurstmeister/kafka:2.12-2.2.1                                             kafka-broker-2
vutrungnghia99/spark-worker:spark2.4.1-python3.7-hadoop2.7                pre-batch-processing-spark-worker-1
vutrungnghia99/spark-master:spark2.4.1-python3.7-hadoop2.7                speed-processing-spark-master
zookeeper:3.4.10                                                          zookeeper
vutrungnghia99/system-manager:spark2.4.1-python3.7-hadoop2.7-kafka2.7.0   system-manager
vutrungnghia99/spark-master:spark2.4.1-python3.7-hadoop2.7                post-batch-processing-spark-master
bde2020/hadoop-datanode:2.0.0-hadoop2.7.4-java8                           datanode-2
grafana/grafana:latest                                                    grafana
vutrungnghia99/spark-master:spark2.4.1-python3.7-hadoop2.7                pre-batch-processing-spark-master
bde2020/hadoop-namenode:2.0.0-hadoop2.7.4-java8                           namenode
vutrungnghia99/graphite:1.1.7-6                                           graphite
bde2020/hadoop-datanode:2.0.0-hadoop2.7.4-java8                           datanode-1
```

Spark cluster 1



Spark Master at spark://7a6c33b85408:7077

URL: spark://7a6c33b85408:7077
Alive Workers: 1
Cores in use: 1 Total, 0 Used
Memory in use: 1024.0 MB Total, 0.0 B Used
Applications: 0 Running, 0 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers (1)

Worker Id	Address
worker-20201223081126-172.18.0.12-34753	172.18.0.12:34753

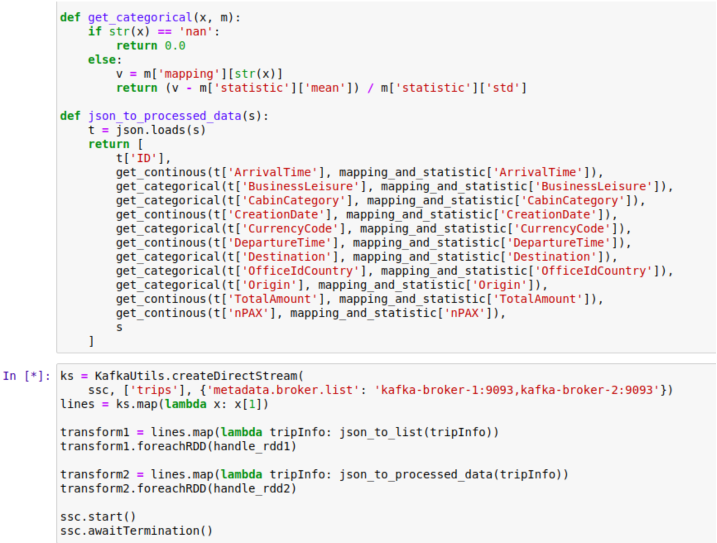
Running Applications (0)

Application ID	Name	Cores	Memory per Executor

Completed Applications (0)

Application ID	Name	Cores	Memory per Executor

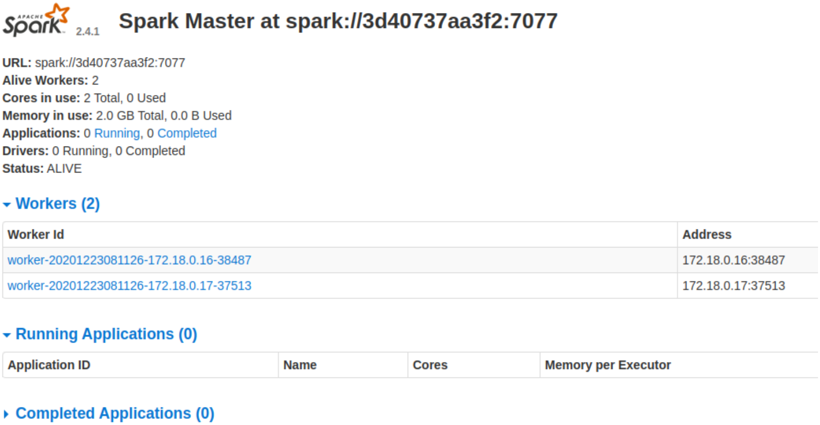
BatchProcessing program



```
def get_categorical(x, m):
    if str(x) == 'nan':
        return 0.0
    else:
        v = m['mapping'][str(x)]
        return (v - m['statistic']['mean']) / m['statistic']['std']

def json_to_processed_data(s):
    t = json.loads(s)
    return [
        t['ID'],
        get_continous(t['ArrivalTime'], mapping_and_statistic['ArrivalTime']),
        get_categorical(t['BusinessLeisure'], mapping_and_statistic['BusinessLeisure']),
        get_categorical(t['CabinCategory'], mapping_and_statistic['CabinCategory']),
        get_continous(t['CreationDate'], mapping_and_statistic['CreationDate']),
        get_categorical(t['CurrencyCode'], mapping_and_statistic['CurrencyCode']),
        get_continous(t['DepartureTime'], mapping_and_statistic['DepartureTime']),
        get_categorical(t['Destination'], mapping_and_statistic['Destination']),
        get_categorical(t['OfficeIdCountry'], mapping_and_statistic['OfficeIdCountry']),
        get_categorical(t['Origin'], mapping_and_statistic['Origin']),
        get_continous(t['TotalAmount'], mapping_and_statistic['TotalAmount']),
        get_continous(t['nPAX'], mapping_and_statistic['nPAX']),
        s
    ]
```

```
In [*]: ks = KafkaUtils.createDirectStream(
            ssc, ['trips'], {'metadata.broker.list': 'kafka-broker-1:9093,kafka-broker-2:9093'})
        lines = ks.map(lambda x: x[1])

        transform1 = lines.map(lambda tripInfo: json_to_list(tripInfo))
        transform1.foreachRDD(handle_rdd1)

        transform2 = lines.map(lambda tripInfo: json_to_processed_data(tripInfo))
        transform2.foreachRDD(handle_rdd2)

        ssc.start()
        ssc.awaitTermination()
```

Spark cluster 2



Spark Master at spark://3d40737aa3f2:7077

URL: spark://3d40737aa3f2:7077
Alive Workers: 2
Cores in use: 2 Total, 0 Used
Memory in use: 2.0 GB Total, 0.0 B Used
Applications: 0 Running, 0 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers (2)

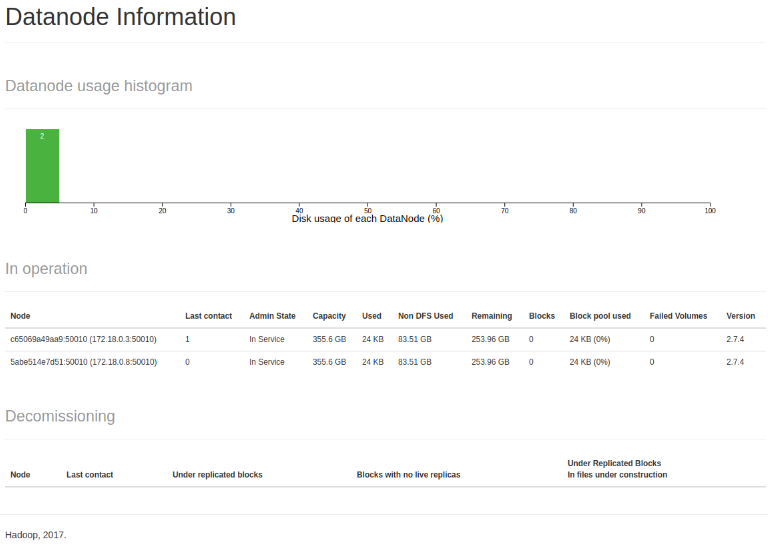
Worker Id	Address
worker-20201223081126-172.18.0.16-38487	172.18.0.16:38487
worker-20201223081126-172.18.0.17-37513	172.18.0.17:37513

Running Applications (0)

Application ID	Name	Cores	Memory per Executor

Completed Applications (0)

HDFS



Datanode Information

Datanode usage histogram

In operation

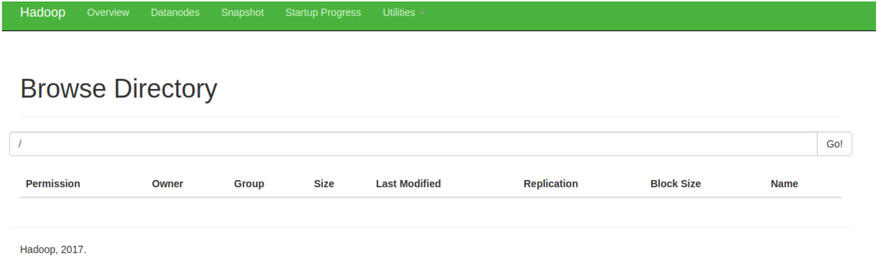
Node	Last contact	Admin State	Capacity	Used	Non DFS Used	Remaining	Blocks	Block pool used	Failed Volumes	Version
c65069a49aa9:50010 (172.18.0.3:50010)	1	In Service	355.6 GB	24 KB	83.51 GB	253.96 GB	0	24 KB (0%)	0	2.7.4
5abe514e7d51:50010 (172.18.0.8:50010)	0	In Service	355.6 GB	24 KB	83.51 GB	253.96 GB	0	24 KB (0%)	0	2.7.4

Decomissioning

Node	Last contact	Under replicated blocks	Blocks with no live replicas	Under Replicated Blocks In files under construction

Hadoop, 2017.

HDFS



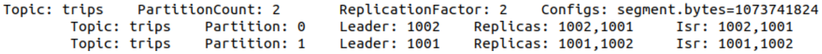
Browse Directory

/

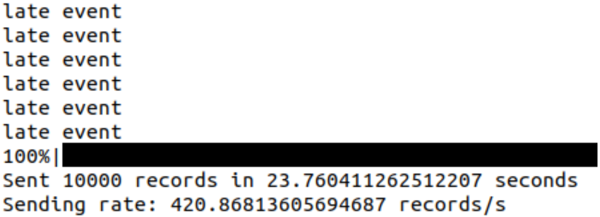
Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name

Hadoop, 2017.

Send 10000 records from local machine to topic "trips"

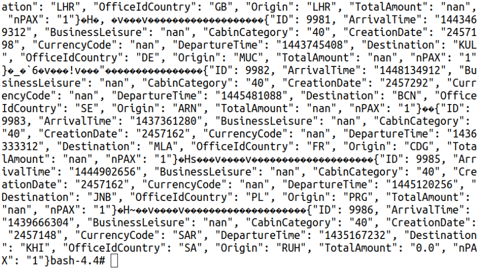


```
Topic: trips    PartitionCount: 2       ReplicationFactor: 2    Configs: segment.bytes=1073741824
        Topic: trips    Partition: 0    Leader: 1002    Replicas: 1002,1001     Isr: 1002,1001
        Topic: trips    Partition: 1    Leader: 1001    Replicas: 1001,1002     Isr: 1001,1002
```

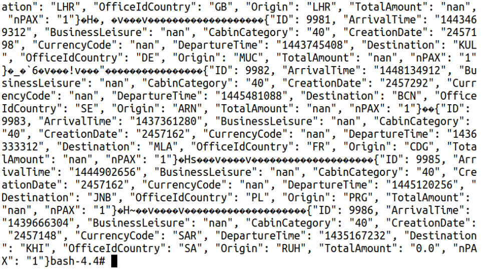


```
late event
late event
late event
late event
late event
late event
100%|████████████████████████████████████████
Sent 10000 records in 23.760411262512207 seconds
Sending rate: 420.86813605694687 records/s
```

Partition 0 trên hai brokers

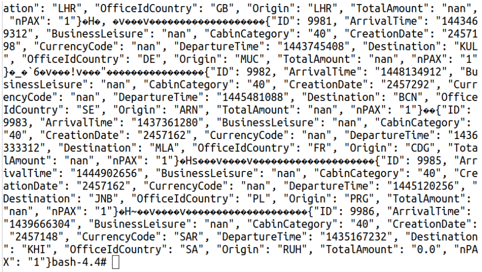


```
ation": "LHR", "OfficeIdCountry": "GB", "Origin": "LHR", "TotalAmount": "nan",
 "nPAX": "1"}�H�, �v���v�������������������{"ID": 9981, "ArrivalTime": "144346
9312", "BusinessLeisure": "nan", "CabinCategory": "40", "CreationDate": "24571
98", "CurrencyCode": "nan", "DepartureTime": "1443745408", "Destination": "KUL
", "OfficeIdCountry": "DE", "Origin": "MUC", "TotalAmount": "nan", "nPAX": "1"
}�_�`6�v���!v���"����������������{"ID": 9982, "ArrivalTime": "1448134912", "Bu
sinessLeisure": "nan", "CabinCategory": "40", "CreationDate": "2457292", "Curr
encyCode": "nan", "DepartureTime": "1445481088", "Destination": "BCN", "Office
IdCountry": "SE", "Origin": "ARN", "TotalAmount": "nan", "nPAX": "1"}��{"ID":
9983, "ArrivalTime": "1437361280", "BusinessLeisure": "nan", "CabinCategory":
"40", "CreationDate": "2457162", "CurrencyCode": "nan", "DepartureTime": "1436
333312", "Destination": "MLA", "OfficeIdCountry": "FR", "Origin": "CDG", "Tota
lAmount": "nan", "nPAX": "1"}�Hs���v����v��������������������{"ID": 9985, "Arr
ivalTime": "1444902656", "BusinessLeisure": "nan", "CabinCategory": "40", "Cre
ationDate": "2457162", "CurrencyCode": "nan", "DepartureTime": "1445120256", "
Destination": "JNB", "OfficeIdCountry": "PL", "Origin": "PRG", "TotalAmount":
"nan", "nPAX": "1"}�H~��v����v��������������������{"ID": 9986, "ArrivalTime":
"1439666304", "BusinessLeisure": "nan", "CabinCategory": "40", "CreationDate":
 "2457148", "CurrencyCode": "SAR", "DepartureTime": "1435167232", "Destination
": "KHI", "OfficeIdCountry": "SA", "Origin": "RUH", "TotalAmount": "0.0", "nPA
X": "1"}bash-4.4# 
```

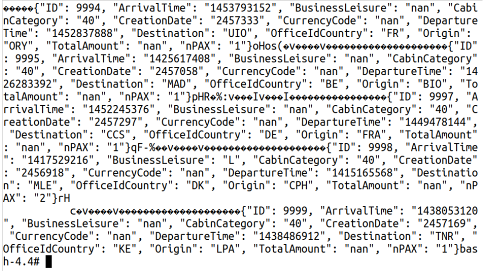


```
ation": "LHR", "OfficeIdCountry": "GB", "Origin": "LHR", "TotalAmount": "nan",
 "nPAX": "1"}�H�, �v���v�������������������{"ID": 9981, "ArrivalTime": "144346
9312", "BusinessLeisure": "nan", "CabinCategory": "40", "CreationDate": "24571
98", "CurrencyCode": "nan", "DepartureTime": "1443745408", "Destination": "KUL
", "OfficeIdCountry": "DE", "Origin": "MUC", "TotalAmount": "nan", "nPAX": "1"
}�_�`6�v���!v���"����������������{"ID": 9982, "ArrivalTime": "1448134912", "Bu
sinessLeisure": "nan", "CabinCategory": "40", "CreationDate": "2457292", "Curr
encyCode": "nan", "DepartureTime": "1445481088", "Destination": "BCN", "Office
IdCountry": "SE", "Origin": "ARN", "TotalAmount": "nan", "nPAX": "1"}��{"ID":
9983, "ArrivalTime": "1437361280", "BusinessLeisure": "nan", "CabinCategory":
"40", "CreationDate": "2457162", "CurrencyCode": "nan", "DepartureTime": "1436
333312", "Destination": "MLA", "OfficeIdCountry": "FR", "Origin": "CDG", "Tota
lAmount": "nan", "nPAX": "1"}�Hs���v����v��������������������{"ID": 9985, "Arr
ivalTime": "1444902656", "BusinessLeisure": "nan", "CabinCategory": "40", "Cre
ationDate": "2457162", "CurrencyCode": "nan", "DepartureTime": "1445120256", "
Destination": "JNB", "OfficeIdCountry": "PL", "Origin": "PRG", "TotalAmount":
"nan", "nPAX": "1"}�H~��v����v��������������������{"ID": 9986, "ArrivalTime":
"1439666304", "BusinessLeisure": "nan", "CabinCategory": "40", "CreationDate":
 "2457148", "CurrencyCode": "SAR", "DepartureTime": "1435167232", "Destination
": "KHI", "OfficeIdCountry": "SA", "Origin": "RUH", "TotalAmount": "0.0", "nPA
X": "1"}bash-4.4# 
```

Partition 0,1 trên broker 0

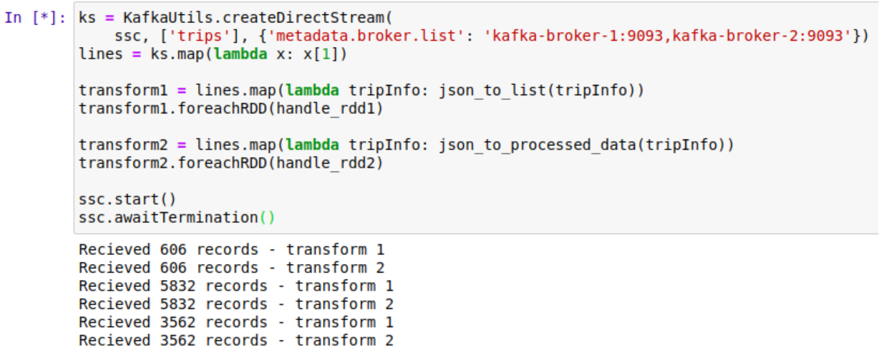


```
ation": "LHR", "OfficeIdCountry": "GB", "Origin": "LHR", "TotalAmount": "nan",
 "nPAX": "1"}�H�, �V���V��������������������{"ID": 9981, "ArrivalTime": "144346
9312", "BusinessLeisure": "nan", "CabinCategory": "40", "CreationDate": "24571
98", "CurrencyCode": "nan", "DepartureTime": "1443745408", "Destination": "KUL
", "OfficeIdCountry": "DE", "Origin": "MUC", "TotalAmount": "nan", "nPAX": "1"
}�_�`6�V���!V���"����������������{"ID": 9982, "ArrivalTime": "1448134912", "Bu
sinessLeisure": "nan", "CabinCategory": "40", "CreationDate": "2457292", "Curr
encyCode": "nan", "DepartureTime": "1445481088", "Destination": "BCN", "Office
IdCountry": "SE", "Origin": "ARN", "TotalAmount": "nan", "nPAX": "1"}��{"ID":
9983, "ArrivalTime": "1437361280", "BusinessLeisure": "nan", "CabinCategory":
"40", "CreationDate": "2457162", "CurrencyCode": "nan", "DepartureTime": "1436
333312", "Destination": "MLA", "OfficeIdCountry": "FR", "Origin": "CDG", "Tota
lAmount": "nan", "nPAX": "1"}�Hs���V����V����������������������{"ID": 9985, "Arr
ivalTime": "1444902656", "BusinessLeisure": "nan", "CabinCategory": "40", "Cre
ationDate": "2457162", "CurrencyCode": "nan", "DepartureTime": "1445120256", "
Destination": "JNB", "OfficeIdCountry": "PL", "Origin": "PRG", "TotalAmount":
"nan", "nPAX": "1"}�H~��V����V�����������������������{"ID": 9986, "ArrivalTime":
"1439666304", "BusinessLeisure": "nan", "CabinCategory": "40", "CreationDate":
 "2457148", "CurrencyCode": "SAR", "DepartureTime": "1435167232", "Destination
": "KHI", "OfficeIdCountry": "SA", "Origin": "RUH", "TotalAmount": "0.0", "nPA
X": "1"}bash-4.4# 
```



```
�����{"ID": 9994, "ArrivalTime": "1453793152", "BusinessLeisure": "nan", "Cabi
nCategory": "40", "CreationDate": "2457333", "CurrencyCode": "nan", "Departure
Time": "1452837888", "Destination": "UIO", "OfficeIdCountry": "FR", "Origin":
"ORY", "TotalAmount": "nan", "nPAX": "1"}oHos(�V����V�����������������������{"ID"
: 9995, "ArrivalTime": "1425617408", "BusinessLeisure": "nan", "CabinCategory"
: "40", "CreationDate": "2457058", "CurrencyCode": "nan", "DepartureTime": "14
26283392", "Destination": "MAD", "OfficeIdCountry": "BE", "Origin": "BIO", "To
talAmount": "nan", "nPAX": "1"}pHR�%:V���IV���I�����������������{"ID": 9997, "A
rrivalTime": "1452245376", "BusinessLeisure": "nan", "CabinCategory": "40", "C
reationDate": "2457297", "CurrencyCode": "nan", "DepartureTime": "1449478144",
 "Destination": "CCS", "OfficeIdCountry": "DE", "Origin": "FRA", "TotalAmount"
: "nan", "nPAX": "1"}qF-%��V����V����������������������{"ID": 9998, "ArrivalTime
": "1417529216", "BusinessLeisure": "L", "CabinCategory": "40", "CreationDate"
: "2456918", "CurrencyCode": "nan", "DepartureTime": "1415165568", "Destinatio
n": "MLE", "OfficeIdCountry": "DK", "Origin": "CPH", "TotalAmount": "nan", "nP
AX": "2"}rH
           c�V����V����������������������{"ID": 9999, "ArrivalTime": "1438053120
", "BusinessLeisure": "nan", "CabinCategory": "40", "CreationDate": "2457169",
 "CurrencyCode": "nan", "DepartureTime": "1438486912", "Destination": "TNR", "
OfficeIdCountry": "KE", "Origin": "LPA", "TotalAmount": "nan", "nPAX": "1"}bas
h-4.4# 
```

The records have been received and processed at Batch Processing Notebook

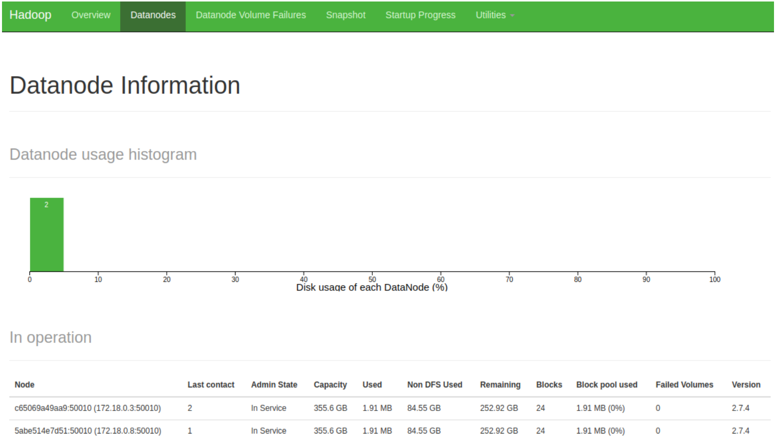


In [*]:

```
ks = KafkaUtils.createDirectStream(
    ssc, ['trips'], {'metadata.broker.list': 'kafka-broker-1:9093,kafka-broker-2:9093'})
lines = ks.map(lambda x: x[1])

transform1 = lines.map(lambda tripInfo: json_to_list(tripInfo))
transform1.foreachRDD(handle_rdd1)

transform2 = lines.map(lambda tripInfo: json_to_processed_data(tripInfo))
transform2.foreachRDD(handle_rdd2)

ssc.start()
ssc.awaitTermination()
```

```
Recieved 606 records - transform 1
Recieved 606 records - transform 2
Recieved 5832 records - transform 1
Recieved 5832 records - transform 2
Recieved 3562 records - transform 1
Recieved 3562 records - transform 2
```

Data has been stored in HDFS



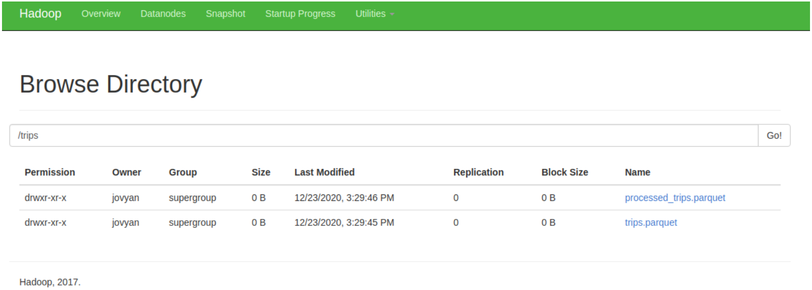
Datanode Information

Datanode usage histogram

In operation

Node	Last contact	Admin State	Capacity	Used	Non DFS Used	Remaining	Blocks	Block pool used	Failed Volumes	Version
c65069a49aa9:50010 (172.18.0.3:50010)	2	In Service	355.6 GB	1.91 MB	84.55 GB	252.92 GB	24	1.91 MB (0%)	0	2.7.4
5abe514e7d51:50010 (172.18.0.8:50010)	1	In Service	355.6 GB	1.91 MB	84.55 GB	252.92 GB	24	1.91 MB (0%)	0	2.7.4

Data has been stored in HDFS



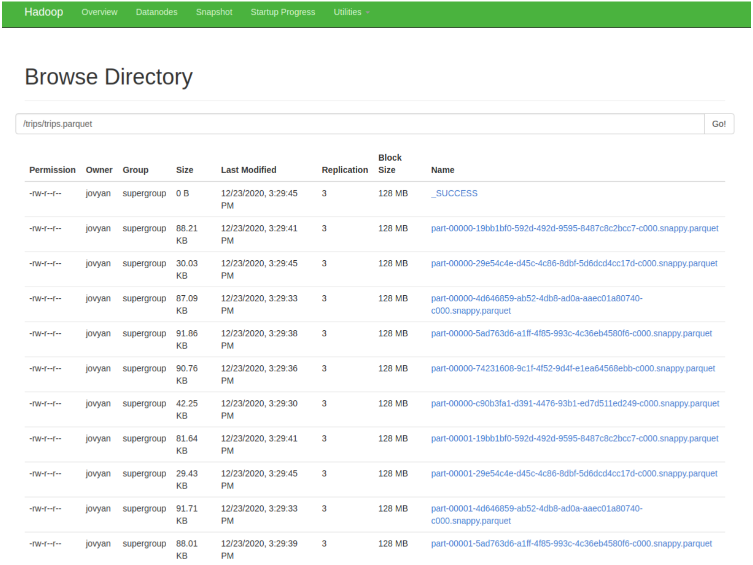
Browse Directory

/trips

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	jovyan	supergroup	0 B	12/23/2020, 3:29:46 PM	0	0 B	processed_trips.parquet
drwxr-xr-x	jovyan	supergroup	0 B	12/23/2020, 3:29:45 PM	0	0 B	trips.parquet

Hadoop, 2017.

Data has been stored in HDFS

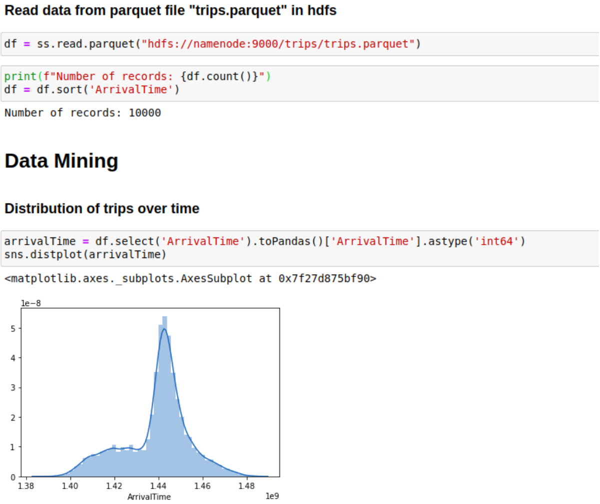


Browse Directory

/trips/trips.parquet

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	jovyan	supergroup	0 B	12/23/2020, 3:29:45 PM	3	128 MB	_SUCCESS
-rw-r--r--	jovyan	supergroup	88.21 KB	12/23/2020, 3:29:41 PM	3	128 MB	part-00000-19bb1bf0-592d-492d-9595-8487c8c2bcc7-c000.snappy.parquet
-rw-r--r--	jovyan	supergroup	30.03 KB	12/23/2020, 3:29:45 PM	3	128 MB	part-00000-29e54c4e-d45c-4c86-8dbf-5d6dcd4cc17d-c000.snappy.parquet
-rw-r--r--	jovyan	supergroup	87.09 KB	12/23/2020, 3:29:33 PM	3	128 MB	part-00000-4d646859-ab52-4db8-ad0a-aaec01a80740-c000.snappy.parquet
-rw-r--r--	jovyan	supergroup	91.86 KB	12/23/2020, 3:29:38 PM	3	128 MB	part-00000-5ad763d6-a1ff-4f85-993c-4c36eb4580f6-c000.snappy.parquet
-rw-r--r--	jovyan	supergroup	90.76 KB	12/23/2020, 3:29:36 PM	3	128 MB	part-00000-74231608-9c1f-4f52-9d4f-e1ea64568ebb-c000.snappy.parquet
-rw-r--r--	jovyan	supergroup	42.25 KB	12/23/2020, 3:29:30 PM	3	128 MB	part-00000-c90b3fa1-d391-4476-93b1-ed7d511ed249-c000.snappy.parquet
-rw-r--r--	jovyan	supergroup	81.64 KB	12/23/2020, 3:29:41 PM	3	128 MB	part-00001-19bb1bf0-592d-492d-9595-8487c8c2bcc7-c000.snappy.parquet
-rw-r--r--	jovyan	supergroup	29.43 KB	12/23/2020, 3:29:45 PM	3	128 MB	part-00001-29e54c4e-d45c-4c86-8dbf-5d6dcd4cc17d-c000.snappy.parquet
-rw-r--r--	jovyan	supergroup	91.71 KB	12/23/2020, 3:29:33 PM	3	128 MB	part-00001-4d646859-ab52-4db8-ad0a-aaec01a80740-c000.snappy.parquet
-rw-r--r--	jovyan	supergroup	88.01 KB	12/23/2020, 3:29:39 PM	3	128 MB	part-00001-5ad763d6-a1ff-4f85-993c-4c36eb4580f6-c000.snappy.parquet

Read data from HDFS and visualize



Read data from parquet file "trips.parquet" in hdfs

```
df = ss.read.parquet("hdfs://namenode:9000/trips/trips.parquet")
```

```
print(f"Number of records: {df.count()}")
df = df.sort('ArrivalTime')
```

```
Number of records: 10000
```

Data Mining

Distribution of trips over time

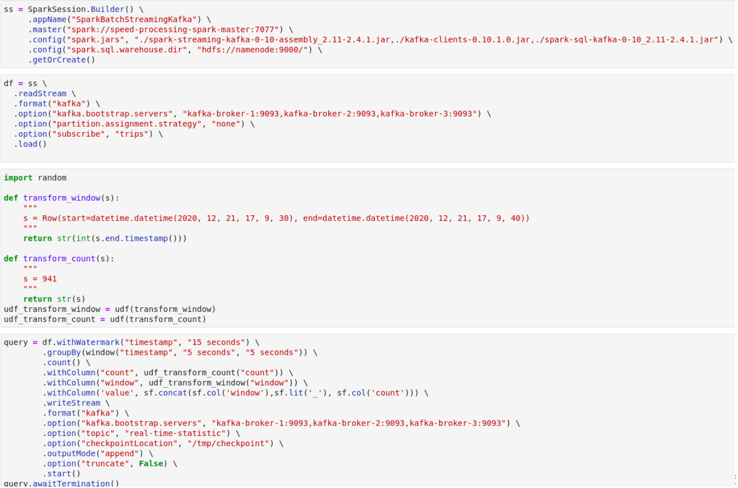
```
arrivalTime = df.select('ArrivalTime').toPandas()['ArrivalTime'].astype('int64')
sns.distplot(arrivalTime)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f27d875bf90>
```

1. Introduction
2. System architecture
3. Execution results
   - Batch processing branch
   - Speed processing branch
4. Evaluation
   - KAFKA
   - HDFS
   - SPARK
5. Reference

Kafka topic => kafka topic

- Program that consumes data from topic "trips" and produces to topic "real-time-statistic"



```
ss = SparkSession.Builder() \
     .appName("SparkBatchStreamingKafka") \
     .master("spark://speed-processing-spark-master:7077") \
     .config("spark.jars", "./spark-streaming-kafka-0-10-assembly_2.11-2.4.1.jar,./kafka-clients-0.10.1.0.jar,./spark-sql-kafka-0-10_2.11-2.4.1.jar") \
     .config("spark.sql.warehouse.dir", "hdfs://namenode:9000/") \
     .getOrCreate()
```

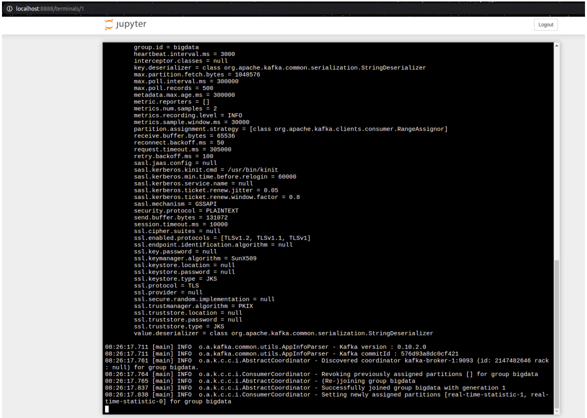
```
df = ss \
  .readStream \
  .format("kafka") \
  .option("kafka.bootstrap.servers", "kafka-broker-1:9093,kafka-broker-2:9093,kafka-broker-3:9093") \
  .option("partition.assignment.strategy", "none") \
  .option("subscribe", "trips") \
  .load()
```

```
import random

def transform_window(s):
    """
    s = Row(start=datetime.datetime(2020, 12, 21, 17, 9, 30), end=datetime.datetime(2020, 12, 21, 17, 9, 40))
    """
    return str(int(s.end.timestamp()))

def transform_count(s):
    """
    s = 941
    """
    return str(s)
udf_transform_window = udf(transform_window)
udf_transform_count = udf(transform_count)
```

```
query = df.withWatermark("timestamp", "15 seconds") \
        .groupBy(window("timestamp", "5 seconds", "5 seconds")) \
        .count() \
        .withColumn("count", udf_transform_count("count")) \
        .withColumn("window", udf_transform_window("window")) \
        .withColumn('value', sf.concat(sf.col('window'),sf.lit('_'), sf.col('count'))) \
        .writeStream \
        .format("kafka") \
        .option("kafka.bootstrap.servers", "kafka-broker-1:9093,kafka-broker-2:9093,kafka-broker-3:9093") \
        .option("topic", "real-time-statistic") \
        .option("checkpointLocation", "/tmp/checkpoint") \
        .outputMode("append") \
        .option("truncate", False) \
        .start()
query.awaitTermination()
```

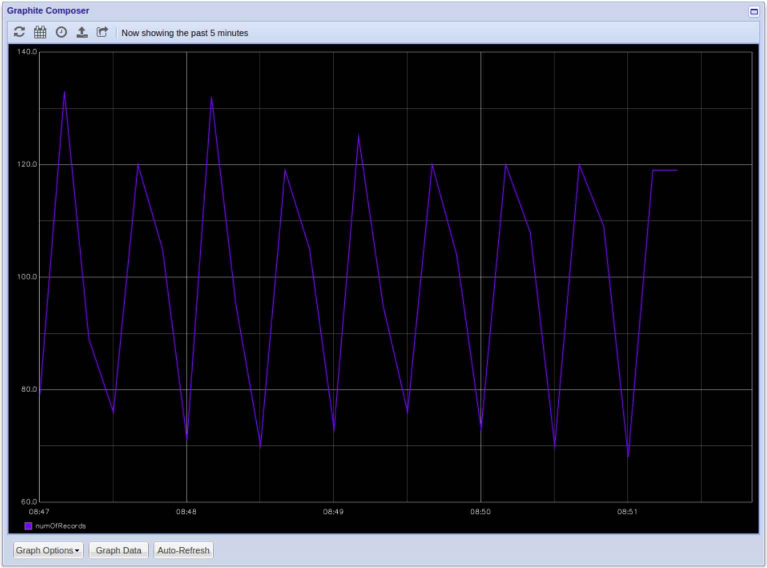
Kafka topic => graphite

- Program that consumes topic "real-time-statistic" and write to graphite

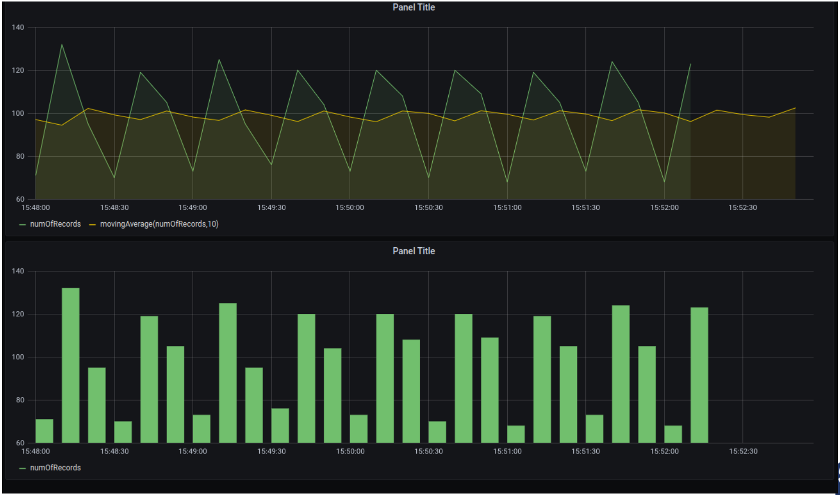


```
        group.id = bigdata
        heartbeat.interval.ms = 3000
        interceptor.classes = null
        key.deserializer = class org.apache.kafka.common.serialization.StringDeserializer
        max.partition.fetch.bytes = 1048576
        max.poll.interval.ms = 300000
        max.poll.records = 500
        metadata.max.age.ms = 300000
        metric.reporters = []
        metrics.num.samples = 2
        metrics.recording.level = INFO
        metrics.sample.window.ms = 30000
        partition.assignment.strategy = [class org.apache.kafka.clients.consumer.RangeAssignor]
        receive.buffer.bytes = 65536
        reconnect.backoff.ms = 50
        request.timeout.ms = 305000
        retry.backoff.ms = 100
        sasl.jaas.config = null
        sasl.kerberos.kinit.cmd = /usr/bin/kinit
        sasl.kerberos.min.time.before.relogin = 60000
        sasl.kerberos.service.name = null
        sasl.kerberos.ticket.renew.jitter = 0.05
        sasl.kerberos.ticket.renew.window.factor = 0.8
        sasl.mechanism = GSSAPI
        security.protocol = PLAINTEXT
        send.buffer.bytes = 131072
        session.timeout.ms = 10000
        ssl.cipher.suites = null
        ssl.enabled.protocols = [TLSv1.2, TLSv1.1, TLSv1]
        ssl.endpoint.identification.algorithm = null
        ssl.key.password = null
        ssl.keymanager.algorithm = SunX509
        ssl.keystore.location = null
        ssl.keystore.password = null
        ssl.keystore.type = JKS
        ssl.protocol = TLS
        ssl.provider = null
        ssl.secure.random.implementation = null
        ssl.trustmanager.algorithm = PKIX
        ssl.truststore.location = null
        ssl.truststore.password = null
        ssl.truststore.type = JKS
        value.deserializer = class org.apache.kafka.common.serialization.StringDeserializer

08:26:17.711 [main] INFO  o.a.kafka.common.utils.AppInfoParser - Kafka version : 0.10.2.0
08:26:17.711 [main] INFO  o.a.kafka.common.utils.AppInfoParser - Kafka commitId : 576d93a8dc0cf421
08:26:17.761 [main] INFO  o.a.k.c.c.i.AbstractCoordinator - Discovered coordinator kafka-broker-1:9093 (id: 2147482646 rack
: null) for group bigdata.
08:26:17.764 [main] INFO  o.a.k.c.c.i.ConsumerCoordinator - Revoking previously assigned partitions [] for group bigdata
08:26:17.765 [main] INFO  o.a.k.c.c.i.AbstractCoordinator - (Re-)joining group bigdata
08:26:17.837 [main] INFO  o.a.k.c.c.i.AbstractCoordinator - Successfully joined group bigdata with generation 1
08:26:17.838 [main] INFO  o.a.k.c.c.i.ConsumerCoordinator - Setting newly assigned partitions [real-time-statistic-1, real-
time-statistic-0] for group bigdata
```

Real-time data in graphite



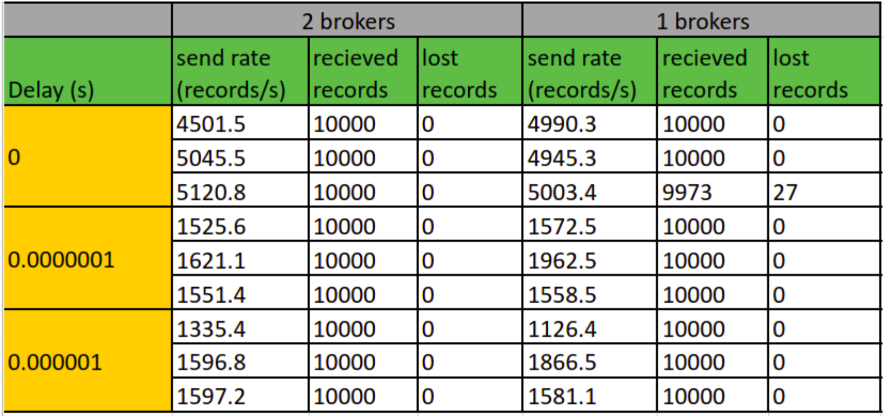
Real-time data in grafana



1. Introduction
2. System architecture
3. Execution results
   - Batch processing branch
   - Speed processing branch
4. Evaluation
   - KAFKA
   - HDFS
   - SPARK
5. Reference

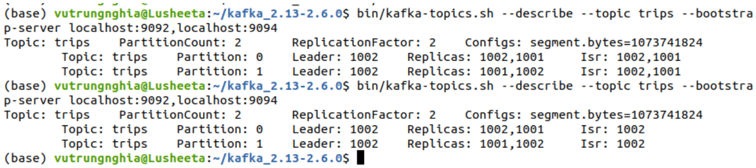
1. Introduction
2. System architecture
3. Execution results
   - Batch processing branch
   - Speed processing branch
4. Evaluation
   - KAFKA
   - HDFS
   - SPARK
5. Reference

Stress testing

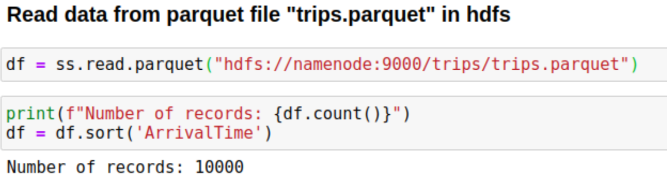


	2 brokers			1 brokers		
Delay (s)	send rate (records/s)	recieved records	lost records	send rate (records/s)	recieved records	lost records
0	4501.5	10000	0	4990.3	10000	0
	5045.5	10000	0	4945.3	10000	0
	5120.8	10000	0	5003.4	9973	27
0.0000001	1525.6	10000	0	1572.5	10000	0
	1621.1	10000	0	1962.5	10000	0
	1551.4	10000	0	1558.5	10000	0
0.000001	1335.4	10000	0	1126.4	10000	0
	1596.8	10000	0	1866.5	10000	0
	1597.2	10000	0	1581.1	10000	0

Fault tolerant



```
(base) vutrungnghia@Lusheeta:~/kafka_2.13-2.6.0$ bin/kafka-topics.sh --describe --topic trips --bootstra
p-server localhost:9092,localhost:9094
Topic: trips    PartitionCount: 2       ReplicationFactor: 2    Configs: segment.bytes=1073741824
        Topic: trips    Partition: 0    Leader: 1002    Replicas: 1002,1001     Isr: 1002,1001
        Topic: trips    Partition: 1    Leader: 1002    Replicas: 1001,1002     Isr: 1002,1001
(base) vutrungnghia@Lusheeta:~/kafka_2.13-2.6.0$ bin/kafka-topics.sh --describe --topic trips --bootstra
p-server localhost:9092,localhost:9094
Topic: trips    PartitionCount: 2       ReplicationFactor: 2    Configs: segment.bytes=1073741824
        Topic: trips    Partition: 0    Leader: 1002    Replicas: 1002,1001     Isr: 1002
        Topic: trips    Partition: 1    Leader: 1002    Replicas: 1001,1002     Isr: 1002
(base) vutrungnghia@Lusheeta:~/kafka_2.13-2.6.0$ 
```



Read data from parquet file "trips.parquet" in hdfs

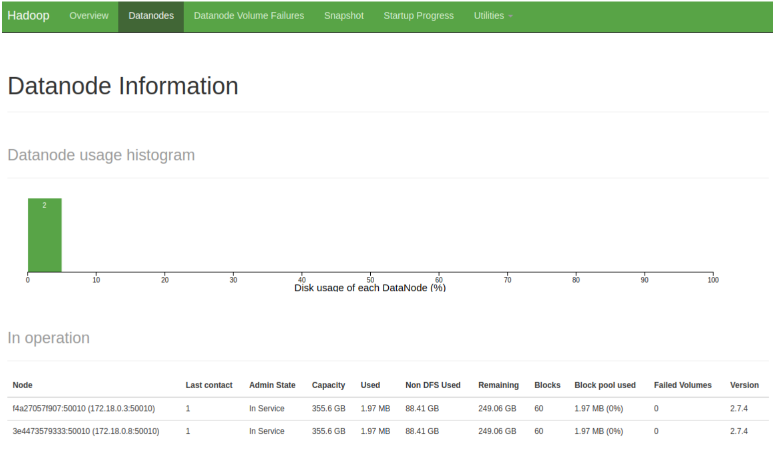
```
df = ss.read.parquet("hdfs://namenode:9000/trips/trips.parquet")
```

```
print(f"Number of records: {df.count()}")
df = df.sort('ArrivalTime')
```

```
Number of records: 10000
```

1. Introduction
2. System architecture
3. Execution results
   - Batch processing branch
   - Speed processing branch
4. Evaluation
   - KAFKA
   - HDFS
   - SPARK
5. Reference

Fault tolerant



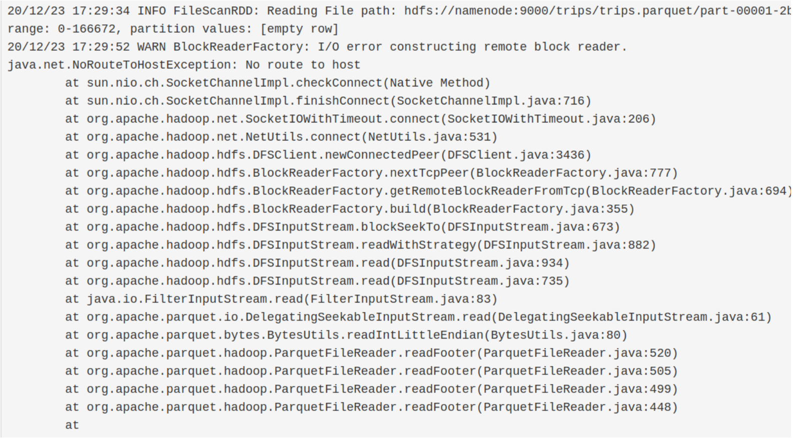
Datanode Information

Datanode usage histogram

In operation

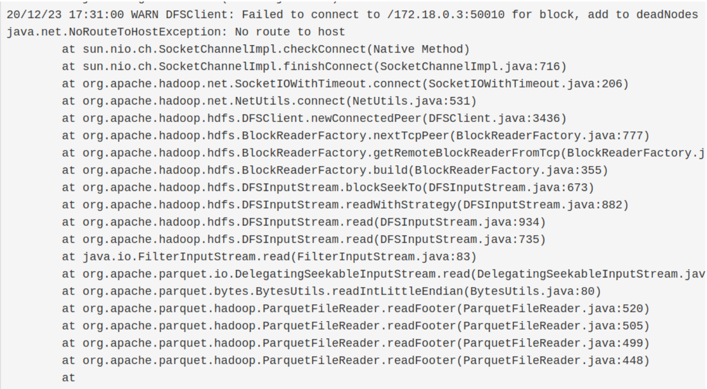
Node	Last contact	Admin State	Capacity	Used	Non DFS Used	Remaining	Blocks	Block pool used	Failed Volumes	Version
f4a27057f907:50010 (172.18.0.3:50010)	1	In Service	355.6 GB	1.97 MB	88.41 GB	249.06 GB	60	1.97 MB (0%)	0	2.7.4
3e4473579333:50010 (172.18.0.8:50010)	1	In Service	355.6 GB	1.97 MB	88.41 GB	249.06 GB	60	1.97 MB (0%)	0	2.7.4

Fault tolerant



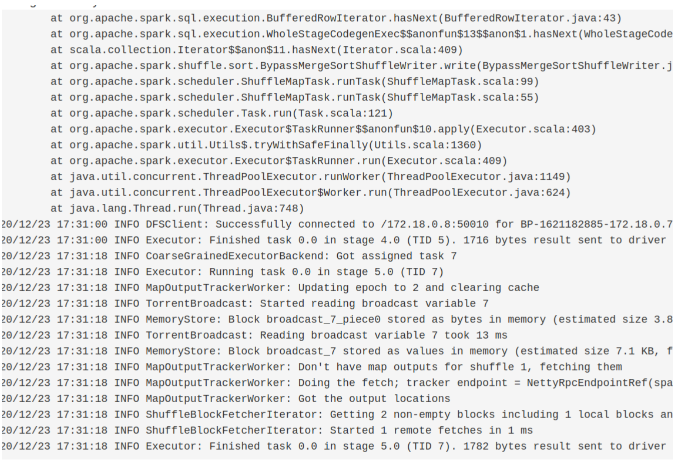
```
20/12/23 17:29:34 INFO FileScanRDD: Reading File path: hdfs://namenode:9000/trips/trips.parquet/part-00001-2b
range: 0-166672, partition values: [empty row]
20/12/23 17:29:52 WARN BlockReaderFactory: I/O error constructing remote block reader.
java.net.NoRouteToHostException: No route to host
        at sun.nio.ch.SocketChannelImpl.checkConnect(Native Method)
        at sun.nio.ch.SocketChannelImpl.finishConnect(SocketChannelImpl.java:716)
        at org.apache.hadoop.net.SocketIOWithTimeout.connect(SocketIOWithTimeout.java:206)
        at org.apache.hadoop.net.NetUtils.connect(NetUtils.java:531)
        at org.apache.hadoop.hdfs.DFSClient.newConnectedPeer(DFSClient.java:3436)
        at org.apache.hadoop.hdfs.BlockReaderFactory.nextTcpPeer(BlockReaderFactory.java:777)
        at org.apache.hadoop.hdfs.BlockReaderFactory.getRemoteBlockReaderFromTcp(BlockReaderFactory.java:694)
        at org.apache.hadoop.hdfs.BlockReaderFactory.build(BlockReaderFactory.java:355)
        at org.apache.hadoop.hdfs.DFSInputStream.blockSeekTo(DFSInputStream.java:673)
        at org.apache.hadoop.hdfs.DFSInputStream.readWithStrategy(DFSInputStream.java:882)
        at org.apache.hadoop.hdfs.DFSInputStream.read(DFSInputStream.java:934)
        at org.apache.hadoop.hdfs.DFSInputStream.read(DFSInputStream.java:735)
        at java.io.FilterInputStream.read(FilterInputStream.java:83)
        at org.apache.parquet.io.DelegatingSeekableInputStream.read(DelegatingSeekableInputStream.java:61)
        at org.apache.parquet.bytes.BytesUtils.readIntLittleEndian(BytesUtils.java:80)
        at org.apache.parquet.hadoop.ParquetFileReader.readFooter(ParquetFileReader.java:520)
        at org.apache.parquet.hadoop.ParquetFileReader.readFooter(ParquetFileReader.java:505)
        at org.apache.parquet.hadoop.ParquetFileReader.readFooter(ParquetFileReader.java:499)
        at org.apache.parquet.hadoop.ParquetFileReader.readFooter(ParquetFileReader.java:448)
        at
```

Fault tolerant



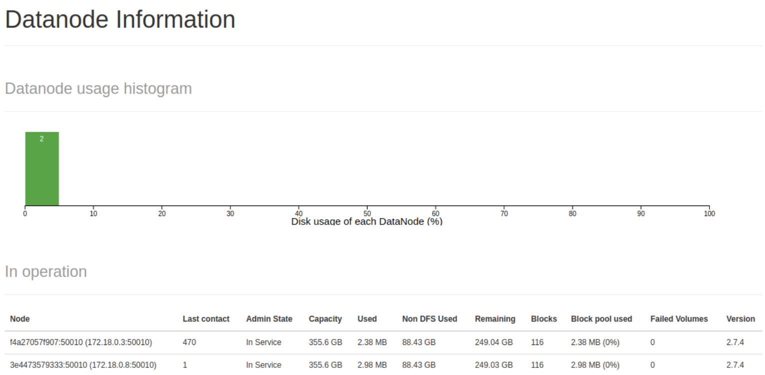
```
20/12/23 17:31:00 WARN DFSClient: Failed to connect to /172.18.0.3:50010 for block, add to deadNodes
java.net.NoRouteToHostException: No route to host
        at sun.nio.ch.SocketChannelImpl.checkConnect(Native Method)
        at sun.nio.ch.SocketChannelImpl.finishConnect(SocketChannelImpl.java:716)
        at org.apache.hadoop.net.SocketIOWithTimeout.connect(SocketIOWithTimeout.java:206)
        at org.apache.hadoop.net.NetUtils.connect(NetUtils.java:531)
        at org.apache.hadoop.hdfs.DFSClient.newConnectedPeer(DFSClient.java:3436)
        at org.apache.hadoop.hdfs.BlockReaderFactory.nextTcpPeer(BlockReaderFactory.java:777)
        at org.apache.hadoop.hdfs.BlockReaderFactory.getRemoteBlockReaderFromTcp(BlockReaderFactory.j
        at org.apache.hadoop.hdfs.BlockReaderFactory.build(BlockReaderFactory.java:355)
        at org.apache.hadoop.hdfs.DFSInputStream.blockSeekTo(DFSInputStream.java:673)
        at org.apache.hadoop.hdfs.DFSInputStream.readWithStrategy(DFSInputStream.java:882)
        at org.apache.hadoop.hdfs.DFSInputStream.read(DFSInputStream.java:934)
        at org.apache.hadoop.hdfs.DFSInputStream.read(DFSInputStream.java:735)
        at java.io.FilterInputStream.read(FilterInputStream.java:83)
        at org.apache.parquet.io.DelegatingSeekableInputStream.read(DelegatingSeekableInputStream.jav
        at org.apache.parquet.bytes.BytesUtils.readIntLittleEndian(BytesUtils.java:80)
        at org.apache.parquet.hadoop.ParquetFileReader.readFooter(ParquetFileReader.java:520)
        at org.apache.parquet.hadoop.ParquetFileReader.readFooter(ParquetFileReader.java:505)
        at org.apache.parquet.hadoop.ParquetFileReader.readFooter(ParquetFileReader.java:499)
        at org.apache.parquet.hadoop.ParquetFileReader.readFooter(ParquetFileReader.java:448)
        at
```

Fault tolerant



```
        at org.apache.spark.sql.execution.BufferedRowIterator.hasNext(BufferedRowIterator.java:43)
        at org.apache.spark.sql.execution.WholeStageCodegenExec$$anonfun$13$$anon$1.hasNext(WholeStageCode
        at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:409)
        at org.apache.spark.shuffle.sort.BypassMergeSortShuffleWriter.write(BypassMergeSortShuffleWriter.j
        at org.apache.spark.scheduler.ShuffleMapTask.runTask(ShuffleMapTask.scala:99)
        at org.apache.spark.scheduler.ShuffleMapTask.runTask(ShuffleMapTask.scala:55)
        at org.apache.spark.scheduler.Task.run(Task.scala:121)
        at org.apache.spark.executor.Executor$TaskRunner$$anonfun$10.apply(Executor.scala:403)
        at org.apache.spark.util.Utils$.tryWithSafeFinally(Utils.scala:1360)
        at org.apache.spark.executor.Executor$TaskRunner.run(Executor.scala:409)
        at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
        at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
        at java.lang.Thread.run(Thread.java:748)
20/12/23 17:31:00 INFO DFSClient: Successfully connected to /172.18.0.8:50010 for BP-1621182885-172.18.0.7
20/12/23 17:31:00 INFO Executor: Finished task 0.0 in stage 4.0 (TID 5). 1716 bytes result sent to driver
20/12/23 17:31:18 INFO CoarseGrainedExecutorBackend: Got assigned task 7
20/12/23 17:31:18 INFO Executor: Running task 0.0 in stage 5.0 (TID 7)
20/12/23 17:31:18 INFO MapOutputTrackerWorker: Updating epoch to 2 and clearing cache
20/12/23 17:31:18 INFO TorrentBroadcast: Started reading broadcast variable 7
20/12/23 17:31:18 INFO MemoryStore: Block broadcast_7_piece0 stored as bytes in memory (estimated size 3.8
20/12/23 17:31:18 INFO TorrentBroadcast: Reading broadcast variable 7 took 13 ms
20/12/23 17:31:18 INFO MemoryStore: Block broadcast_7 stored as values in memory (estimated size 7.1 KB, f
20/12/23 17:31:18 INFO MapOutputTrackerWorker: Don't have map outputs for shuffle 1, fetching them
20/12/23 17:31:18 INFO MapOutputTrackerWorker: Doing the fetch; tracker endpoint = NettyRpcEndpointRef(spa
20/12/23 17:31:18 INFO MapOutputTrackerWorker: Got the output locations
20/12/23 17:31:18 INFO ShuffleBlockFetcherIterator: Getting 2 non-empty blocks including 1 local blocks an
20/12/23 17:31:18 INFO ShuffleBlockFetcherIterator: Started 1 remote fetches in 1 ms
20/12/23 17:31:18 INFO Executor: Finished task 0.0 in stage 5.0 (TID 7). 1782 bytes result sent to driver
```

Fault tolerant



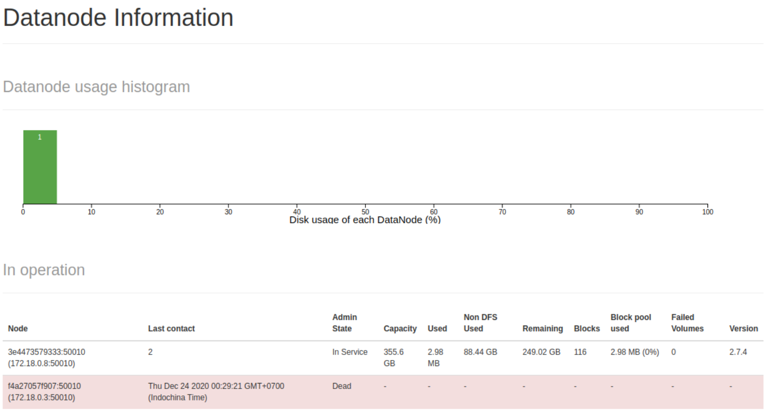
Datanode Information

Datanode usage histogram

In operation

Node	Last contact	Admin State	Capacity	Used	Non DFS Used	Remaining	Blocks	Block pool used	Failed Volumes	Version
f4a27057f907:50010 (172.18.0.3:50010)	470	In Service	355.6 GB	2.38 MB	88.43 GB	249.04 GB	116	2.38 MB (0%)	0	2.7.4
3e4473579333:50010 (172.18.0.8:50010)	1	In Service	355.6 GB	2.98 MB	88.43 GB	249.03 GB	116	2.98 MB (0%)	0	2.7.4

Fault tolerant



Datanode Information

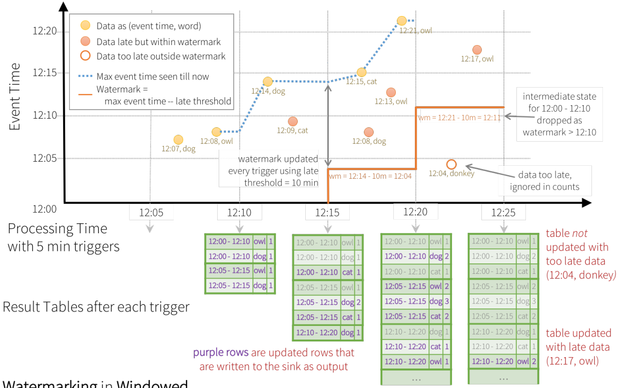
Datanode usage histogram

In operation

Node	Last contact	Admin State	Capacity	Used	Non DFS Used	Remaining	Blocks	Block pool used	Failed Volumes	Version
3e4473579333:50010 (172.18.0.8:50010)	2	In Service	355.6 GB	2.98 MB	88.44 GB	249.02 GB	116	2.98 MB (0%)	0	2.7.4
f4a27057f907:50010 (172.18.0.3:50010)	Thu Dec 24 2020 00:29:21 GMT+0700 (Indochina Time)	Dead	-	-	-	-	-	-	-	-

1. Introduction
2. System architecture
3. Execution results
    - Batch processing branch
    - Speed processing branch
4. Evaluation
    - KAFKA
    - HDFS
    - SPARK
5. Reference

Late messages handling


Data as (event time, word)
Data late but within watermark
Data too late outside watermark
Max event time seen till now
Watermark =
max event time -- late threshold
Event Time
12:20
12:15
12:10
12:05
12:00
12:05
12:10
12:15
12:20
12:25
12:07, dog
12:08, owl
12:14, dog
12:09, cat
12:15, cat
12:08, dog
12:13, owl
12:21, owl
12:17, owl
12:04, donkey
watermark updated every trigger using late threshold = 10 min
wm = 12:14 - 10m = 12:04
wm = 12:21 - 10m = 12:11
intermediate state for 12:00 - 12:10 dropped as watermark > 12:10
data too late, ignored in counts
Processing Time with 5 min triggers
Result Tables after each trigger
table *not* updated with too late data (12:04, donkey)
table updated with late data (12:17, owl)
purple rows are updated rows that are written to the sink as output

12:10

12:00 - 12:10	owl	1
12:00 - 12:10	dog	1
12:05 - 12:15	owl	1
12:05 - 12:15	dog	1

12:15

12:00 - 12:10	owl	1
12:00 - 12:10	dog	1
12:00 - 12:10	cat	1
12:05 - 12:15	owl	1
12:05 - 12:15	dog	2
12:05 - 12:15	cat	1
12:10 - 12:20	dog	1

12:20

12:00 - 12:10	owl	1
12:00 - 12:10	dog	2
12:00 - 12:10	cat	1
12:05 - 12:15	owl	2
12:05 - 12:15	dog	3
12:05 - 12:15	cat	2
12:10 - 12:20	dog	1
12:10 - 12:20	cat	1
12:10 - 12:20	owl	1
...		

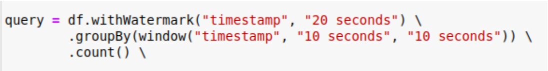
12:25

12:00 - 12:10	owl	1
12:00 - 12:10	dog	2
12:00 - 12:10	cat	1
12:05 - 12:15	owl	2
12:05 - 12:15	dog	3
12:05 - 12:15	cat	2
12:10 - 12:20	dog	1
12:10 - 12:20	cat	1
12:10 - 12:20	owl	2
...		

Watermarking in Windowed
Grouped Aggregation with Update Mode

(*) source: https://spark.apache.org

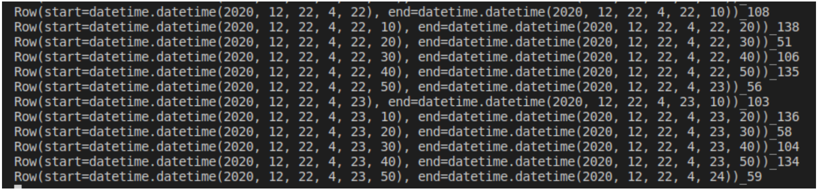
Late messages handling



```
query = df.withWatermark("timestamp", "20 seconds") \
          .groupBy(window("timestamp", "10 seconds", "10 seconds")) \
          .count() \
```

- To handle late event, Structured Spark supports Window Grouped Aggregation and Watermarking.
- Wartermarking excludes event that is too late (for some threshold).

Late messages handling



```
Row(start=datetime.datetime(2020, 12, 22, 4, 22), end=datetime.datetime(2020, 12, 22, 4, 22, 10))_108
Row(start=datetime.datetime(2020, 12, 22, 4, 22, 10), end=datetime.datetime(2020, 12, 22, 4, 22, 20))_138
Row(start=datetime.datetime(2020, 12, 22, 4, 22, 20), end=datetime.datetime(2020, 12, 22, 4, 22, 30))_51
Row(start=datetime.datetime(2020, 12, 22, 4, 22, 30), end=datetime.datetime(2020, 12, 22, 4, 22, 40))_106
Row(start=datetime.datetime(2020, 12, 22, 4, 22, 40), end=datetime.datetime(2020, 12, 22, 4, 22, 50))_135
Row(start=datetime.datetime(2020, 12, 22, 4, 22, 50), end=datetime.datetime(2020, 12, 22, 4, 23))_56
Row(start=datetime.datetime(2020, 12, 22, 4, 23), end=datetime.datetime(2020, 12, 22, 4, 23, 10))_103
Row(start=datetime.datetime(2020, 12, 22, 4, 23, 10), end=datetime.datetime(2020, 12, 22, 4, 23, 20))_136
Row(start=datetime.datetime(2020, 12, 22, 4, 23, 20), end=datetime.datetime(2020, 12, 22, 4, 23, 30))_58
Row(start=datetime.datetime(2020, 12, 22, 4, 23, 30), end=datetime.datetime(2020, 12, 22, 4, 23, 40))_104
Row(start=datetime.datetime(2020, 12, 22, 4, 23, 40), end=datetime.datetime(2020, 12, 22, 4, 23, 50))_134
Row(start=datetime.datetime(2020, 12, 22, 4, 23, 50), end=datetime.datetime(2020, 12, 22, 4, 24))_59
```

Figure: Count the number of records for the last 10 seconds

- Using KafkaProducer to simulate late event.
- After 20s (watermarking threshold), half of the records are assigned as late event.

Reference

- https://github.com/haiphucnguyen/BigDataDemo
- http://www.diva-portal.org/smash/get/diva2:897808/FULLTE pdf
- https://blog.softwaremill.com/7-mistakes-when-using-apa
- https://spark.apache.org/docs/1.5.2/sql-programming-guic html
- https://spark.apache.org/docs/latest/structured-streamir html
- https://medium.com/dev-genius/an-in-depth-look-at-zooke
- https://spark.apache.org/docs/latest/streaming-programmi html