

Lambda Architecture on Airline Data:

1. Understanding of Data:

I am using airline cancellation and delay data over different years. The dataset contains details of the flights which were canceled or delayed. Also, it maintains the arrival and destination time delay in minutes. It can be downloaded from Kaggle ([Airline Delay and Cancellation Data](#)).

EDA on data using jupyter notebook:

```
In [1]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

EDA on 2017 airline data

In [2]: #reading the csv file
df1 = pd.read_csv('2017.csv');
print (df1.shape)
df1.head(10)
```

(5674621, 28)

```
Out[2]:
```

	FL_DATE	OP_CARRIER	OP_CARRIER_FL_NUM	ORIGIN	DEST	CRS_DEP_TIME	DEP_TIME	DEP_DELAY	TAXI_OUT	WHEELS_OFF	...	CRS_ELAPSE
0	2017-01-01	AA	1	JFK	LAX	800	831.0	31.0	25.0	856.0	...	
1	2017-01-01	AA	2	LAX	JFK	900	934.0	34.0	34.0	1008.0	...	
2	2017-01-01	AA	4	LAX	JFK	1130	1221.0	51.0	20.0	1241.0	...	
3	2017-01-01	AA	5	DFW	HNL	1135	1252.0	77.0	19.0	1311.0	...	
4	2017-01-01	AA	6	OGG	DFW	1855	1855.0	0.0	16.0	1911.0	...	
5	2017-01-01	AA	7	DFW	OGG	940	1619.0	399.0	12.0	1631.0	...	

```
In [3]: df1.tail(10)
```

```
Out[3]:
```

	FL_DATE	OP_CARRIER	OP_CARRIER_FL_NUM	ORIGIN	DEST	CRS_DEP_TIME	DEP_TIME	DEP_DELAY	TAXI_OUT	WHEELS_OFF	...	CRS_ELAP
5674611	2017-12-31	UA	2415	EWB	PDX	825	850.0	25.0	14.0	904.0	...	
5674612	2017-12-31	UA	2417	PDX	EWB	2240	2238.0	-2.0	10.0	2248.0	...	
5674613	2017-12-31	UA	2418	RIC	DEN	1601	1600.0	-1.0	14.0	1614.0	...	

```
In [4]: # Total number of canceled flights
df1.CANCELLED.sum()
```

```
Out[4]: 82693.0
```

```
In [5]: # Let's explore column CANCELLED
df1.CANCELLED.unique()
```

```
Out[5]: array([0., 1.])
```

```
In [6]: # From above we see it's binary: 0 or 1, Let's see how it looks like
canceled = df1[(df1['CANCELLED'] > 0)]
```

```
In [7]: canceled.head(3)
```

```
Out[7]:
```

	FL_DATE	OP_CARRIER	OP_CARRIER_FL_NUM	ORIGIN	DEST	CRS_DEP_TIME	DEP_TIME	DEP_DELAY	TAXI_OUT	WHEELS_OFF	...	CRS_ELAPSED
80	2017-01-01	AA	106	PHX	JFK	957	NaN	NaN	NaN	NaN	...	
151	2017-01-01	AA	229	DFW	KOA	1030	NaN	NaN	NaN	NaN	...	
152	2017-01-01	AA	230	KOA	DFW	1840	NaN	NaN	NaN	NaN	...	

3 rows x 28 columns

```
In [8]: # Arrival delay data (in minutes)
df1.ARR_DELAY.head()
```

```
Out[8]: 0    27.0
1    42.0
2    42.0
3    97.0
4    42.0
Name: ARR_DELAY, dtype: float64
```

```
In [9]: # Check if FL_DATE is DateTime type
type(df1['FL_DATE'])
```

```
Out[9]: pandas.core.series.Series
```

```
In [10]: # Convert string to DateTime
```

```
pd.to_datetime(df1.FL_DATE)
```

```
Out[10]: 0    2017-01-01
1    2017-01-01
2    2017-01-01
3    2017-01-01
4    2017-01-01
...
5674616 2017-12-31
5674617 2017-12-31
5674618 2017-12-31
5674619 2017-12-31
5674620 2017-12-31
Name: FL_DATE, Length: 5674621, dtype: datetime64[ns]
```

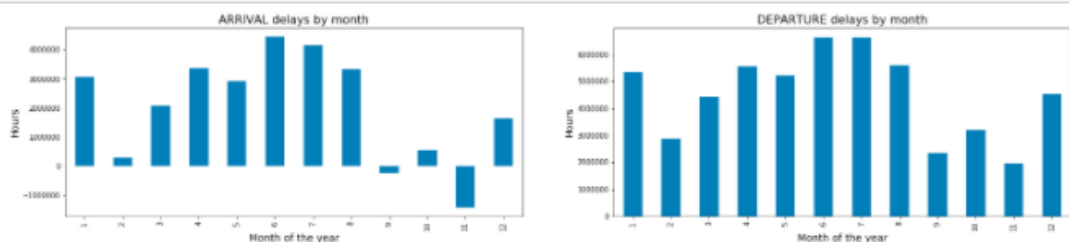
```
In [11]: # Month variable
df1['FL_DATE_month'] = pd.to_datetime(df1['FL_DATE']).dt.month
```

```
In [12]: # Arrival and departure delays by month of the year
plt.figure(figsize=(25, 12)).subplots_adjust(hspace = 0.5)

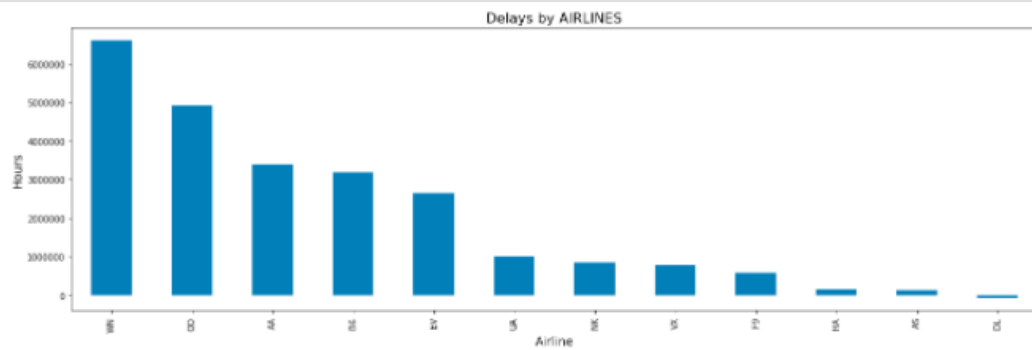
plt.subplot(2, 2, 1)
df1.groupby('FL_DATE_month').ARR_DELAY.sum().plot.bar().set_title('ARRIVAL delays by month')
plt.title('ARRIVAL delays by month', fontsize=16)
plt.ylabel('Hours', fontsize=14)
plt.xlabel('Month of the year', fontsize=14)

plt.subplot(2, 2, 2)
df1.groupby('FL_DATE_month').DEP_DELAY.sum().plot.bar().set_title('DEPARTURE delays by month')
plt.title('DEPARTURE delays by month', fontsize=16)
plt.ylabel('Hours', fontsize=14)
plt.xlabel('Month of the year', fontsize=14)

plt.show()
```



```
In [13]: # Delays by airlines
plt.figure(figsize=(20, 6))
df1.groupby('OP_CARRIER').ARR_DELAY.sum().sort_values(ascending=False).plot.bar()
plt.title('Delays by AIRLINES', fontsize=16)
plt.xlabel('Airline', fontsize=14)
plt.ylabel('Hours', fontsize=14)
plt.show()
```



```
In [14]: # Arrival and departure delays by month of the year
plt.figure(figsize=(25, 12)).subplots_adjust(hspace = 0.5)

plt.subplot(2, 2, 1)
plt.boxplot(df1.ARR_DELAY.dropna())
plt.title('ARRIVAL delays', fontsize=16)

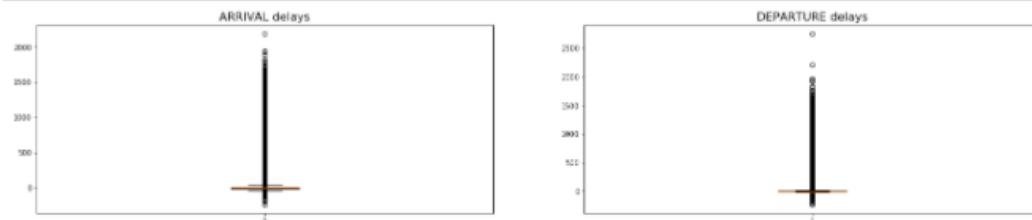
plt.subplot(2, 2, 2)
plt.boxplot(df1.DEP_DELAY.dropna())
plt.title('DEPARTURE delays', fontsize=16)

plt.show()
```

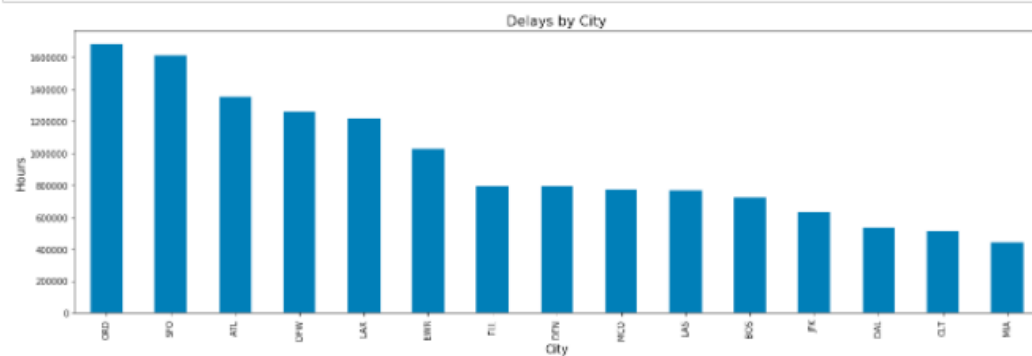


```
plt.subplot(2, 2, 2)
plt.boxplot(df1.DEP_DELAY.dropna())
plt.title('DEPARTURE delays', fontsize=16)

plt.show()
```



```
In [15]: # Delays by City
city_by_delay = df1.groupby('ORIGIN').ARR_DELAY.sum().sort_values(ascending=False)
plt.figure(figsize=(20, 6))
city_by_delay[:15].plot.bar()
plt.title('Delays by City', fontsize=16)
plt.xlabel('City', fontsize=14)
plt.ylabel('Hours', fontsize=14)
plt.show()
```



EDA on 2018 airline data

```
In [16]: #reading the csv file
df1 = pd.read_csv('2018.csv');
print (df1.shape)
df1.head(10)
```

```
(7213446, 28)
```

```
Out[16]:
```

	FL_DATE	OP_CARRIER	OP_CARRIER_FL_NUM	ORIGIN	DEST	CRS_DEP_TIME	DEP_TIME	DEP_DELAY	TAXI_OUT	WHEELS_OFF	...	CRS_ELAPSED
0	2018-01-01	UA	2429	EWR	DEN	1517	1512.0	-5.0	15.0	1527.0	...	
1	2018-01-01	UA	2427	LAS	SFO	1115	1107.0	-8.0	11.0	1118.0	...	
2	2018-01-01	UA	2426	SNA	DEN	1335	1330.0	-5.0	15.0	1345.0	...	
3	2018-01-01	UA	2425	RSW	ORD	1548	1552.0	6.0	19.0	1611.0	...	
4	2018-01-01	UA	2424	ORD	ALB	630	650.0	20.0	13.0	703.0	...	
5	2018-01-01	UA	2422	ORD	OMA	2241	2244.0	3.0	15.0	2259.0	...	

```
In [17]: df1.tail(10)
```

```
Out[17]:
```

	FL_DATE	OP_CARRIER	OP_CARRIER_FL_NUM	ORIGIN	DEST	CRS_DEP_TIME	DEP_TIME	DEP_DELAY	TAXI_OUT	WHEELS_OFF	...	CRS_ELAPSED
7213436	2018-12-31	AA	1812	PDX	PHX	1058	1100.0	2.0	17.0	1117.0	...	
7213437	2018-12-31	AA	1812	PHX	PDX	825	821.0	-4.0	15.0	836.0	...	
7213438	2018-12-31	AA	1813	CLT	ATL	2100	2100.0	0.0	12.0	2112.0	...	
7213439	2018-12-31	AA	1814	DFW	PHL	1955	2026.0	31.0	12.0	2038.0	...	
7213440	2018-12-31	AA	1815	CLT	DCA	1321	1320.0	-1.0	12.0	1332.0	...	
7213441	2018-12-31	AA	1815	DCA	CLT	1534	1530.0	-4.0	20.0	1550.0	...	
7213442	2018-12-31	AA	1816	CLT	DFW	1751	1757.0	6.0	18.0	1815.0	...	

```
In [18]: # Total number of cancelled flights
df1.CANCELLED.sum()
```

```
Out[18]: 116584.0
```

```
In [19]: # Let's explore column CANCELLED
df1.CANCELLED.unique()
```

```
Out[19]: array([0., 1.])
```

```
In [20]: # From above we see it's binary: 0 or 1, Let's see how it looks like
canceled = df1[(df1['CANCELLED'] > 0)]
```

```
In [21]: canceled.head(3)
```

```
Out[21]:
```

	FL_DATE	OP_CARRIER	OP_CARRIER_FL_NUM	ORIGIN	DEST	CRS_DEP_TIME	DEP_TIME	DEP_DELAY	TAXI_OUT	WHEELS_OFF	...	CRS_ELAPSED
178	2018-01-01	UA	2034	IAH	MFE	1440	NaN	NaN	NaN	NaN	...	
875	2018-01-01	UA	864	LAS	SFO	1744	NaN	NaN	NaN	NaN	...	
1244	2018-01-01	UA	488	MFE	IAH	1726	NaN	NaN	NaN	NaN	...	

```
3 rows x 28 columns
```

```
In [22]: # Arrival delay data (in minutes)
df1.ARR_DELAY.head()
```

```
Out[22]: 0    -23.0
1    -24.0
2    -13.0
3     -2.0
4     14.0
Name: ARR_DELAY, dtype: float64
```

```
In [23]: # Check if FL_DATE is DateTime type
type(df1['FL_DATE'])
```

```
Out[23]: pandas.core.series.Series
```

```
In [24]: # Convert string to DateTime
pd.to_datetime(df1.FL_DATE)
```

```
In [24]: # Convert string to datetime
pd.to_datetime(df1.FL_DATE)

Out[24]: 0      2018-01-01
1      2018-01-01
2      2018-01-01
3      2018-01-01
4      2018-01-01
...
7213441 2018-12-31
7213442 2018-12-31
7213443 2018-12-31
7213444 2018-12-31
7213445 2018-12-31
Name: FL_DATE, Length: 7213446, dtype: datetime64[ns]
```

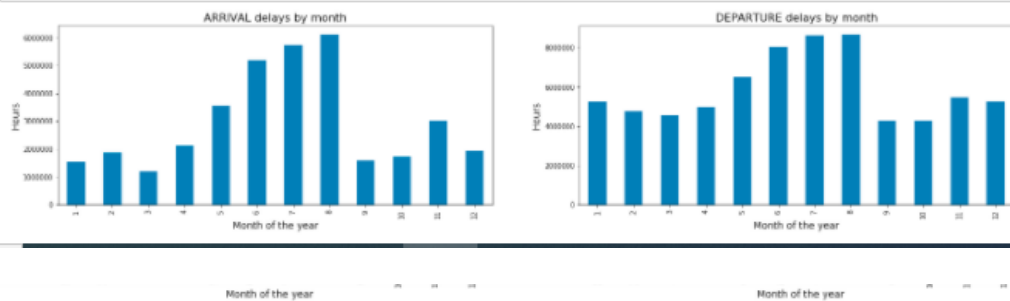
```
In [25]: # Month variable
df1['FL_DATE_month'] = pd.to_datetime(df1['FL_DATE']).dt.month
```

```
In [26]: # Arrival and departure delays by month of the year
plt.figure(figsize=(25, 12)).subplots_adjust(hspace = 0.5)

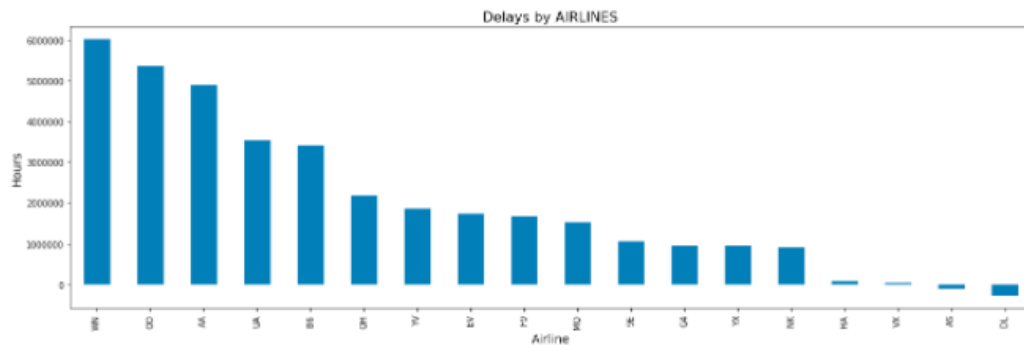
plt.subplot(2, 2, 1)
df1.groupby('FL_DATE_month').ARR_DELAY.sum().plot.bar().set_title('ARRIVAL delays by month')
plt.title('ARRIVAL delays by month', fontsize=16)
plt.ylabel('Hours', fontsize=14)
plt.xlabel('Month of the year', fontsize=14)

plt.subplot(2, 2, 2)
df1.groupby('FL_DATE_month').DEP_DELAY.sum().plot.bar()
plt.title('DEPARTURE delays by month', fontsize=16)
plt.ylabel('Hours', fontsize=14)
plt.xlabel('Month of the year', fontsize=14)

plt.show()
```



```
In [27]: # Delays by airlines
plt.figure(figsize=(20, 6))
df1.groupby('OP_CARRIER').ARR_DELAY.sum().sort_values(ascending=False).plot.bar()
plt.title('Delays by AIRLINES', fontsize=16)
plt.xlabel('Airline', fontsize=14)
plt.ylabel('Hours', fontsize=14)
plt.show()
```



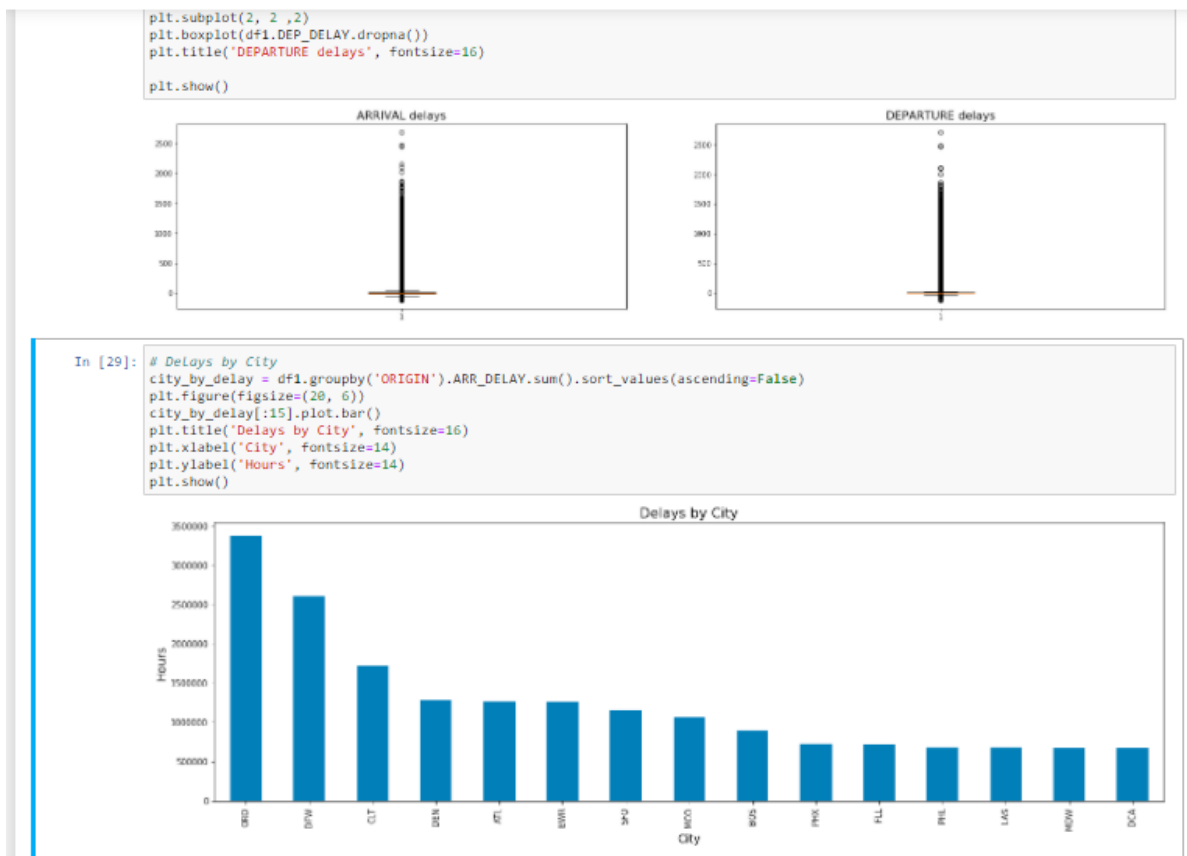
```
In [28]: # Arrival and departure delays by month of the year
plt.figure(figsize=(25, 12)).subplots_adjust(hspace = 0.5)

plt.subplot(2, 2, 1)
plt.boxplot(df1.ARR_DELAY.dropna())
plt.title('ARRIVAL delays', fontsize=16)

plt.subplot(2, 2, 2)
plt.boxplot(df1.DEP_DELAY.dropna())
plt.title('DEPARTURE delays', fontsize=16)

plt.show()
```





2. Domain Knowledge

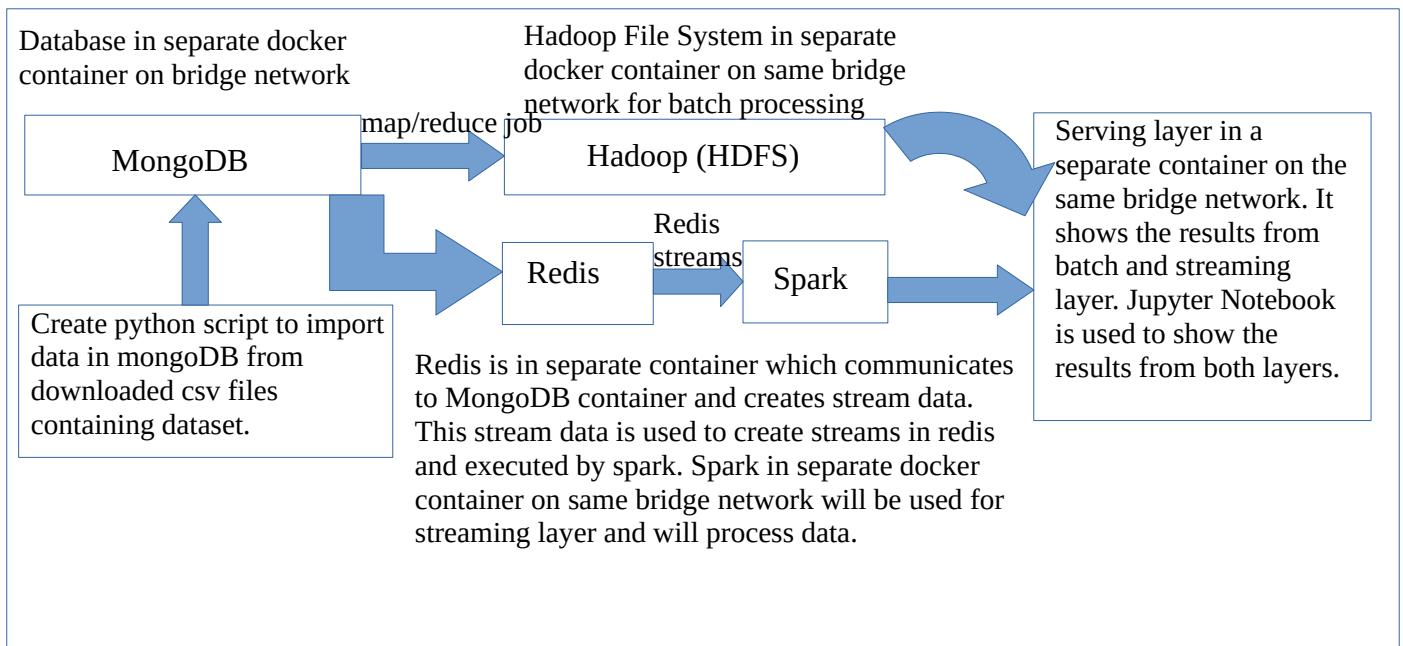
For the batch processing, I am using airline data for years 2017 and 2018. The data exists in tabular form which is around 1.2 GB. I will be executing a hadoop map/reduce job to calculate the average delay time in flights over the years. Firstly, the delay time can be analyzed by filtering the canceled flights. That is; removing the rows with 'CANCELED' column as 1. Next, a delayed flight can be defined as arriving late to the destination so the 'ARR_DELAY' column value will be positive and greater than 0.

For streaming layer, spark will be used which will take data streams from redis. It will execute a script to calculate average flights delay in a month. Each data stream from redis will be for each month and spark will read it execute the script. The output is save to redis table which consists of month and the average delay. The script is executed with the above analysis that a delay flight will have 'CANCELLED' value 0 and positive delay values are considered for average.

The serving layer will show the results from batch and streaming layer.

3. Methodology

Architecture Diagram:



4. Technical Details

Initially I worked on Redis lab commands which are:

Redis is an open-source in-memory key-value data store. It can be used as a database, cache and, message broker, and supports various data structures such as Strings, Hashes, Lists, Sets, and more.

Installation in Ubuntu:

```
tazeen@tazeen-HP-15-Notebook-PC:~$ sudo apt update
Hit:1 https://download.docker.com/linux/ubuntu focal InRelease
Hit:2 http://pk.archive.ubuntu.com/ubuntu focal InRelease
Hit:3 http://security.ubuntu.com/ubuntu focal-security InRelease
Reading package lists... Done
Building dependency tree
Reading state information... Done
115 packages can be upgraded. Run 'apt list --upgradable' to see them.
tazeen@tazeen-HP-15-Notebook-PC:~$ sudo apt install redis-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libhiredis0.14 libjemalloc2 liblua5.1-0 lua-bitop lua-cjson redis-tools
Suggested packages:
  ruby-redis
The following NEW packages will be installed:
  libhiredis0.14 libjemalloc2 liblua5.1-0 lua-bitop lua-cjson redis-server
  redis-tools
0 upgraded, 7 newly installed, 0 to remove and 115 not upgraded.
Need to get 915 kB of archives.
After this operation, 4,077 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://pk.archive.ubuntu.com/ubuntu focal/universe amd64 libhiredis0.14 amd64 0.14.0-6 [30.2 kB]
Get:2 http://pk.archive.ubuntu.com/ubuntu focal/universe amd64 libjemalloc2 amd64 5.2.1-1ubuntu1 [235 kB]
Get:3 http://pk.archive.ubuntu.com/ubuntu focal/universe amd64 liblua5.1-0 amd64 5.1.5-8.1build4 [99.9 kB]
Get:4 http://pk.archive.ubuntu.com/ubuntu focal/universe amd64 lua-bitop amd64 1.0.2-5 [6,680 B]
Get:5 http://pk.archive.ubuntu.com/ubuntu focal/universe amd64 lua-cjson amd64 2.1.0+dfsg-2.1 [17.4 kB]
Get:6 http://pk.archive.ubuntu.com/ubuntu focal/universe amd64 redis-tools amd64 5:5.0.7-2 [489 kB]
Get:7 http://pk.archive.ubuntu.com/ubuntu focal/universe amd64 redis-server amd64 5:5.0.7-2 [37.3 kB]
Fetched 915 kB in 3s (331 kB/s)
Selecting previously unselected package libhiredis0.14:amd64.
(Reading database ... 178989 files and directories currently installed.)
Preparing to unpack .../0-libhiredis0.14_0.14.0-6_amd64.deb ...
Unpacking libhiredis0.14:amd64 (0.14.0-6) ...
Selecting previously unselected package libjemalloc2:amd64.
Preparing to unpack .../1-libjemalloc2_5.2.1-1ubuntu1_amd64.deb ...
Unpacking libjemalloc2:amd64 (5.2.1-1ubuntu1) ...
Selecting previously unselected package liblua5.1-0:amd64.
```

```

Selecting previously unselected package liblua5.1-0:amd64.
Preparing to unpack .../2-liblua5.1-0_5.1.5-8.1build4_amd64.deb ...
Unpacking liblua5.1-0:amd64 (5.1.5-8.1build4) ...
Selecting previously unselected package lua-bitop:amd64.
Preparing to unpack .../3-lua-bitop_1.0.2-5_amd64.deb ...
Unpacking lua-bitop:amd64 (1.0.2-5) ...
Selecting previously unselected package lua-cjson:amd64.
Preparing to unpack .../4-lua-cjson_2.1.0+dfsg-2.1_amd64.deb ...
Unpacking lua-cjson:amd64 (2.1.0+dfsg-2.1) ...
Selecting previously unselected package redis-tools.
Preparing to unpack .../5-redis-tools_5:3a5.0.7-2_amd64.deb ...
Unpacking redis-tools (5:5.0.7-2) ...
Selecting previously unselected package redis-server.
Preparing to unpack .../6-redis-server_5:3a5.0.7-2_amd64.deb ...
Unpacking redis-server (5:5.0.7-2) ...
Setting up libjemalloc2:amd64 (5.2.1-1ubuntu1) ...
Setting up lua-cjson:amd64 (2.1.0+dfsg-2.1) ...
Setting up lua-bitop:amd64 (1.0.2-5) ...
Setting up liblua5.1-0:amd64 (5.1.5-8.1build4) ...
Setting up libhiredis0:14:amd64 (0.14.0-6) ...
Setting up redis-tools (5:5.0.7-2) ...
Setting up redis-server (5:5.0.7-2) ...
Created symlink /etc/systemd/system/redis.service → /lib/systemd/system/redis-server.service.
Created symlink /etc/systemd/system/multi-user.target.wants/redis-server.service → /lib/systemd/system/redis-server.service.
Processing triggers for systemd (245.4-4ubuntu3.2) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libc-bin (2.31-0ubuntu9) ...
tazeen@tazeen-HP-15-Notebook-PC:~$ sudo systemctl status redis-server
● redis-server.service - Advanced key-value store
   Loaded: loaded (/lib/systemd/system/redis-server.service; enabled; vendor
   Active: active (running) since Wed 2020-12-09 21:13:09 PKT; 19s ago
     Docs: http://redis.io/documentation,
           man:redis-server(1)
   Main PID: 4126 (redis-server)
     Tasks: 4 (limit: 14229)
    Memory: 2.2M
   CGroup: /system.slice/redis-server.service

```

```

21:24 9 ديسمبر
Terminal
tazeen@tazeen-HP-15-Notebook-PC: ~
21:13:09 09 فسمير tazeen-HP-15-Notebook-PC systemd[1]: Starting Advanced key-value store.
21:13:09 09 فسمير tazeen-HP-15-Notebook-PC systemd[1]: redis-server.service: Can't open PID file /run/redis/redis-server.pid (yet?) after start: Op
21:13:09 09 فسمير tazeen-HP-15-Notebook-PC systemd[1]: Started Advanced key-value store.
lines 1-14/14 (END)...skipping...
● redis-server.service - Advanced key-value store
   Loaded: loaded (/lib/systemd/system/redis-server.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2020-12-09 21:13:09 PKT; 19s ago
     Docs: http://redis.io/documentation,
           man:redis-server(1)
  Main PID: 4126 (redis-server)
    Tasks: 4 (limit: 14229)
   Memory: 2.2M
   CGroup: /system.slice/redis-server.service
           └─4126 /usr/bin/redis-server 127.0.0.1:6379

21:13:09 09 فسمير tazeen-HP-15-Notebook-PC systemd[1]: Starting Advanced key-value store...
21:13:09 09 فسمير tazeen-HP-15-Notebook-PC systemd[1]: redis-server.service: Can't open PID file /run/redis/redis-server.pid (yet?) after start: Op
21:13:09 09 فسمير tazeen-HP-15-Notebook-PC systemd[1]: Started Advanced key-value store.

```

Redis cli: Check if redis is running

```
ies Terminal 21:51 9 دسمبر
tazeen@tazeen-HP-15-Notebook-PC: ~
tazeen@tazeen-HP-15-Notebook-PC:~$ redis-cli
127.0.0.1:6379> ping
PONG
127.0.0.1:6379> 
```

Get and Set key value:

```
ies Terminal 21:53 9 دسمبر
tazeen@tazeen-HP-15-Notebook-PC: ~
tazeen@tazeen-HP-15-Notebook-PC:~$ redis-cli
127.0.0.1:6379> set myKey testValue
OK
127.0.0.1:6379> get myKey
"testValue"
127.0.0.1:6379> 
```

Commands to information about client connected:

Lambda Architecture:

I- Database (mongodb layer)

Creating docker-compose.yml file for mongo db container

```
tazeen@tazeen-HP-15-Notebook-PC:~$ docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
3da3ccfed5d1        bridge             bridge              local
def85b3aa983        host               host                local
3f1c406c8fc5        none              null                local
tazeen@tazeen-HP-15-Notebook-PC:~$ docker network inspect bridge
[
  {
    "Name": "bridge",
    "Id": "3da3ccfed5d13b18c3c72290d364541ccdb54f34925fc7cb46c553812a6b402b",
    "Created": "2020-12-21T21:32:42.138834963+05:00",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": null,
      "Config": [
        {
          "Subnet": "172.17.0.0/16",
          "Gateway": "172.17.0.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {},
    "Options": {
      "com.docker.network.bridge.default_bridge": "true",
      "com.docker.network.bridge.enable_icc": "true",
      "com.docker.network.bridge.enable_ip_masquerade": "true",
      "com.docker.network.bridge.host_binding_ipv4": "0.0.0.0",
      "com.docker.network.bridge.name": "docker0",
      "com.docker.network.driver.mtu": "1500"
    }
  }
]
```

```
tazeen@tazeen-HP-15-Notebook-PC:~$ sudo apt-get install bridge-utils
[sudo] password for tazeen:
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  ifupdown
The following NEW packages will be installed:
  bridge-utils
0 upgraded, 1 newly installed, 0 to remove and 126 not upgraded.
Need to get 30.5 kB of archives.
After this operation, 112 kB of additional disk space will be used.
Get:1 http://pk.archive.ubuntu.com/ubuntu focal/main amd64 bridge-utils amd64 1.6-2ubuntu1 [30.5 kB]
Fetched 30.5 kB in 1s (24.2 kB/s)
Selecting previously unselected package bridge-utils.
(Reading database ... 179070 files and directories currently installed.)
Preparing to unpack .../bridge-utils 1.6-2ubuntu1_amd64.deb ...
Unpacking bridge-utils (1.6-2ubuntu1) ...
Setting up bridge-utils (1.6-2ubuntu1) ...
Processing triggers for man-db (2.9.1-1) ...
tazeen@tazeen-HP-15-Notebook-PC:~$ brctl show
bridge name      bridge id        STP enabled      interfaces
docker0          8000.02421518cc18  no                
```

```
tazeen@tazeen-HP-15-Notebook-PC:~$ sudo curl -L "https://github.com/docker/compose/releases/download/1.27.4/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 651 100 651 0 0 1493 0 --:--:-- --:--:-- --:--:-- 1496
100 11.6M 100 11.6M 0 0 317k 0 0:00:37 0:00:37 --:--:-- 315k
tazeen@tazeen-HP-15-Notebook-PC:~$
tazeen@tazeen-HP-15-Notebook-PC:~$ sudo chmod +x /usr/local/bin/docker-compose
tazeen@tazeen-HP-15-Notebook-PC:~$ docker-compose --version
docker-compose version 1.27.4, build 40524192
```

```
tazeen@tazeen-HP-15-Notebook-PC:~$ sudo nano docker-compose.yml
[sudo] password for tazeen:
```

```
GNU nano 4.8 docker-compose.yml
version: "3.2"
services:
  py-mongo:
    # build the image from Dockerfile
    build:
      context: ${PWD}
    volumes:
      - $PWD/mongo-data:/data/db
      - $PWD/mongo-app:/var/www/html
    ports:
      - "27017:27017"
    environment:
      - MONGO_INITDB_ROOT_USERNAME=root
      - MONGO_INITDB_ROOT_PASSWORD=1234
```

Read 14 lines

Get Help Write Out Where Is Cut Text Justify Cur Pos M-U Undo M-A Mark Text M-] To Bracket
Exit Read File Replace Paste Text To Spell Go To Line M-E Redo M-C Copy Text M-_ Where Was

```
tazeen@tazeen-HP-15-Notebook-PC:~$ sudo nano Dockerfile
tazeen@tazeen-HP-15-Notebook-PC:~$
```

```
GNU nano 4.8 Dockerfile
FROM mongo:latest

# install Python 3
RUN apt-get update && apt-get install -y python3 python3-pip
RUN apt-get -y install python3.7-dev
RUN pip3 install pymongo

EXPOSE 27017

Get Help  Write Out  Where Is  Cut Text  Read 8 lines  Justify  Cur Pos  M-U Undo  M-A Mark Text  M-] To Bracket
Exit      Read File  Replace  Paste Text  To Spell  Go To Line  M-E Redo  M-C Copy Text  M-_ Where Was
```

```
Building py-mongo
Step 1/5 : FROM mongo:latest
latest: Pulling from library/mongo
f22ccc0b8772: Pull complete
3cf8fb62ba5f: Pull complete
e80c964ece6a: Pull complete
329e632c35b3: Pull complete
3e1bd1325a3d: Pull complete
4aa6e3d64a4a: Pull complete
035bca87b778: Pull complete
874e4e43cb00: Pull complete
08cb97662b8b: Pull complete
f623ce2ba1e1: Pull complete
f100ac278196: Pull complete
6f5539f9b3ee: Pull complete
Digest: sha256:02e9941ddcb949424fa4eb01f9d235da91a5b7b64feb5887eab77e1ef84a3bad
Status: Downloaded newer image for mongo:latest
--> 3068f6bb852e
Step 2/5 : RUN apt-get update && apt-get install -y python3 python3-pip
--> Running in 9123a9b802de
Get:1 http://archive.ubuntu.com/ubuntu bionic InRelease [242 kB]
Get:2 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
Ign:3 http://repo.mongodb.org/apt/ubuntu bionic/mongodb-org/4.4 InRelease
Get:4 http://repo.mongodb.org/apt/ubuntu bionic/mongodb-org/4.4 Release [5391 B]
Get:5 http://repo.mongodb.org/apt/ubuntu bionic/mongodb-org/4.4 Release.gpg [801 B]
Get:6 http://repo.mongodb.org/apt/ubuntu bionic/mongodb-org/4.4/multiverse amd64 Packages [7139 B]
Get:7 http://archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Get:8 http://archive.ubuntu.com/ubuntu bionic-backports InRelease [74.6 kB]
Get:9 http://security.ubuntu.com/ubuntu bionic-security/universe amd64 Packages [1372 kB]
Get:10 http://archive.ubuntu.com/ubuntu bionic/restricted amd64 Packages [13.5 kB]
Get:11 http://archive.ubuntu.com/ubuntu bionic/main amd64 Packages [1344 kB]
Get:12 http://archive.ubuntu.com/ubuntu bionic/universe amd64 Packages [11.3 MB]
Get:13 http://security.ubuntu.com/ubuntu bionic-security/multiverse amd64 Packages [15.3 kB]
Get:14 http://security.ubuntu.com/ubuntu bionic-security/main amd64 Packages [1816 kB]
Get:15 http://security.ubuntu.com/ubuntu bionic-security/restricted amd64 Packages [237 kB]
Get:16 http://archive.ubuntu.com/ubuntu bionic/multiverse amd64 Packages [186 kB]
Get:17 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 Packages [2136 kB]
```

```
es Terminal 22:03 23 دسمبر
tazeen@tazeen-HP-15-Notebook-PC: ~
,log=(enabled=true,archive=true,path=journal,compressor=snappy),file_manager=(close_idle_time=100000,close_scan_interval=10,close_handle_minim
um=250),statistics_log=(wait=0),verbose=[recovery_progress,checkpoint_progress,compact_progress])}
py-mongo_1 | {"t":{"$date":"2020-12-23T17:00:29.855+00:00"},"s":"I", "c":"STORAGE", "id":22430, "ctx":"initandlisten","msg":"WiredTiger m
essage","attr":{"message":["1608742829:855525][1:0x7f3dbc9e5ac0], txn-recover: [WT_VERB_RECOVERY_PROGRESS] Recovering log 1 through 2"}}
py-mongo_1 | {"t":{"$date":"2020-12-23T17:00:29.957+00:00"},"s":"I", "c":"STORAGE", "id":22430, "ctx":"initandlisten","msg":"WiredTiger m
essage","attr":{"message":["1608742829:957450][1:0x7f3dbc9e5ac0], txn-recover: [WT_VERB_RECOVERY_PROGRESS] Recovering log 2 through 2"}}
py-mongo_1 | {"t":{"$date":"2020-12-23T17:00:30.054+00:00"},"s":"I", "c":"STORAGE", "id":22430, "ctx":"initandlisten","msg":"WiredTiger m
essage","attr":{"message":["1608742830:54971][1:0x7f3dbc9e5ac0], txn-recover: [WT_VERB_RECOVERY | WT_VERB_RECOVERY_PROGRESS] Main recovery loo
p: starting at 1/29952 to 2/256"}}
py-mongo_1 | {"t":{"$date":"2020-12-23T17:00:30.158+00:00"},"s":"I", "c":"STORAGE", "id":22430, "ctx":"initandlisten","msg":"WiredTiger m
essage","attr":{"message":["1608742830:158106][1:0x7f3dbc9e5ac0], txn-recover: [WT_VERB_RECOVERY_PROGRESS] Recovering log 1 through 2"}}
py-mongo_1 | {"t":{"$date":"2020-12-23T17:00:30.285+00:00"},"s":"I", "c":"STORAGE", "id":22430, "ctx":"initandlisten","msg":"WiredTiger m
essage","attr":{"message":["1608742830:285128][1:0x7f3dbc9e5ac0], txn-recover: [WT_VERB_RECOVERY_PROGRESS] Recovering log 2 through 2"}}
py-mongo_1 | {"t":{"$date":"2020-12-23T17:00:30.351+00:00"},"s":"I", "c":"STORAGE", "id":22430, "ctx":"initandlisten","msg":"WiredTiger m
essage","attr":{"message":["1608742830:351984][1:0x7f3dbc9e5ac0], txn-recover: [WT_VERB_RECOVERY | WT_VERB_RECOVERY_PROGRESS] Set global recov
ery timestamp: (0, 0)}}
py-mongo_1 | {"t":{"$date":"2020-12-23T17:00:30.352+00:00"},"s":"I", "c":"STORAGE", "id":22430, "ctx":"initandlisten","msg":"WiredTiger m
essage","attr":{"message":["1608742830:352043][1:0x7f3dbc9e5ac0], txn-recover: [WT_VERB_RECOVERY | WT_VERB_RECOVERY_PROGRESS] Set global oldest
timestamp: (0, 0)}}
py-mongo_1 | {"t":{"$date":"2020-12-23T17:00:30.934+00:00"},"s":"I", "c":"STORAGE", "id":4795906, "ctx":"initandlisten","msg":"WiredTiger o
pened","attr":{"durationMillis":1919}}
py-mongo_1 | {"t":{"$date":"2020-12-23T17:00:30.934+00:00"},"s":"I", "c":"RECOVERY", "id":23987, "ctx":"initandlisten","msg":"WiredTiger r
ecoveryTimestamp","attr":{"recoveryTimestamp":{"$timestamp":{"t":0,"i":0}}}}
py-mongo_1 | {"t":{"$date":"2020-12-23T17:00:30.937+00:00"},"s":"I", "c":"STORAGE", "id":4366408, "ctx":"initandlisten","msg":"No table log
ging settings modifications are required for existing WiredTiger tables","attr":{"loggingEnabled":true}}
py-mongo_1 | {"t":{"$date":"2020-12-23T17:00:30.938+00:00"},"s":"I", "c":"STORAGE", "id":22262, "ctx":"initandlisten","msg":"Timestamp no
nitor starting"}}
py-mongo_1 | {"t":{"$date":"2020-12-23T17:00:31.027+00:00"},"s":"I", "c":"STORAGE", "id":20536, "ctx":"initandlisten","msg":"Flow Control
is enabled on this deployment"}}
py-mongo_1 | {"t":{"$date":"2020-12-23T17:00:31.030+00:00"},"s":"I", "c":"FTDC", "id":20625, "ctx":"initandlisten","msg":"Initializing
full-time diagnostic data capture","attr":{"dataDirectory":"/data/db/diagnostic.data"}}
py-mongo_1 | {"t":{"$date":"2020-12-23T17:00:31.033+00:00"},"s":"I", "c":"NETWORK", "id":23015, "ctx":"listener","msg":"Listening on","at
tr":{"address":"/tmp/mongodb-27017.sock"}}
py-mongo_1 | {"t":{"$date":"2020-12-23T17:00:31.033+00:00"},"s":"I", "c":"NETWORK", "id":23015, "ctx":"listener","msg":"Listening on","at
tr":{"address":"0.0.0.0"}}
py-mongo_1 | {"t":{"$date":"2020-12-23T17:00:31.034+00:00"},"s":"I", "c":"NETWORK", "id":23016, "ctx":"listener","msg":"Waiting for conne
ctions","attr":{"port":27017,"ssl":"off"}}
[]
```

Get the docker container id and inspect it to see the IP address

```
es Terminal 01:36 24 دسمبر
tazeen@tazeen-HP-15-Notebook-PC: ~
tazeen@tazeen-HP-15-Notebook-PC:~$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED            STATUS              PORTS               NAMES
90c3a25dffbf1      tazeen_py-mongo    "docker-entrypoint.s..." 3 hours ago        Up 3 hours          0.0.0.0:27017->27017/tcp   tazeen_py-mongo_1
tazeen@tazeen-HP-15-Notebook-PC:~$ docker inspect 90c3a25dffbf1
[
  {
    "Id": "90c3a25dffbf1312ee5a802ca40497fe576e1c68d3fd4cfc9006a212c40bd23ff",
    "Created": "2020-12-23T17:40:02.032629196Z",
    "Path": "docker-entrypoint.sh",
    "Args": [
      "mongod"
    ],
    "State": {
      "Status": "running",
      "Running": true,
      "Paused": false,
      "Restarting": false,
      "OOMKilled": false,
      "Dead": false,
      "Pid": 13816,
      "ExitCode": 0,
      "Error": "",
      "StartedAt": "2020-12-23T17:40:05.440689578Z",
      "FinishedAt": "0001-01-01T00:00:00Z"
    },
    "Image": "sha256:575d0cd1ce89a06cc857aa7acb3cab8eb4a964954d60f820c72d84c0f25702d",
    "ResolvConfPath": "/var/lib/docker/containers/90c3a25dffbf1312ee5a802ca40497fe576e1c68d3fd4cfc9006a212c40bd23ff/resolv.conf",
    "HostnamePath": "/var/lib/docker/containers/90c3a25dffbf1312ee5a802ca40497fe576e1c68d3fd4cfc9006a212c40bd23ff/hostname",
    "HostsPath": "/var/lib/docker/containers/90c3a25dffbf1312ee5a802ca40497fe576e1c68d3fd4cfc9006a212c40bd23ff/hosts",
    "LogPath": "/var/lib/docker/containers/90c3a25dffbf1312ee5a802ca40497fe576e1c68d3fd4cfc9006a212c40bd23ff/90c3a25dffbf1312ee5a802ca40497fe576e1c68d3fd4cfc9006a212c40bd23ff.json.log",
    "Name": "/tazeen_py-mongo_1",
    "RestartCount": 0,
    "Driver": "overlay2",
    "Platform": "linux",
    "MountLabel": "",
    "ProcessLabel": "",
    "AppArmorProfile": "docker-default"
  }
]
```

```
Terminal 01:37 24 دسمبر
tazeen@tazeen-HP-15-Notebook-PC: ~

{
  "HostIp": "0.0.0.0",
  "HostPort": "27017"
}
]
},
"SandboxKey": "/var/run/docker/netns/c732369d4800",
"SecondaryIPAddresses": null,
"SecondaryIPv6Addresses": null,
"EndpointID": "",
"Gateway": "",
"GlobalIPv6Address": "",
"GlobalIPv6PrefixLen": 0,
"IPAddress": "",
"IPPrefixLen": 0,
"IPv6Gateway": "",
"MacAddress": "",
"Networks": {
  "tazeen_default": {
    "IPAMConfig": null,
    "Links": null,
    "Aliases": [
      "py-mongo",
      "90c3a25dfffb1"
    ],
    "NetworkID": "93f176d001f83a0589b0751dddec10adcc77ec9b066f1eb865bcf68174cdb7fd8",
    "EndpointID": "b635c8a715a3e1dcea3f532e049518aca5b42a4db986f8d4b48655b2a6f779d9",
    "Gateway": "172.18.0.1",
    "IPAddress": "172.18.0.2",
    "IPPrefixLen": 16,
    "IPv6Gateway": "",
    "GlobalIPv6Address": "",
    "GlobalIPv6PrefixLen": 0,
    "MacAddress": "02:42:ac:12:00:02",
    "DriverOpts": null
  }
}
}
```

```
Terminal 01:44 24 دسمبر
tazeen@tazeen-HP-15-Notebook-PC: ~/mongo-app

tazeen@tazeen-HP-15-Notebook-PC:~$ ls -la
.          .bashrc      Desktop      Downloads    mongo-app    Pictures      .rediscli_history  Templates
..         .cache       docker-compose.yml  get-docker.sh  mongo-data   .profile     snap               .thunderbird
.bash_history  .config     Dockerfile    .gnupg      .mozilla    Public        .ssh               Videos
.bash_logout  database-connection  Documents    .local      Music        .python_history  .sudo_as_admin_successful

tazeen@tazeen-HP-15-Notebook-PC:~$ cd mongo-app
tazeen@tazeen-HP-15-Notebook-PC:~/mongo-app$ sudo nano database-connection
tazeen@tazeen-HP-15-Notebook-PC:~/mongo-app$
```

```
Terminal 01:44 24 دسمبر
tazeen@tazeen-HP-15-Notebook-PC: ~/mongo-app

GNU nano 4.8 database-connection
#!/usr/bin/env python3
#-*- coding: utf-8 -*-

# import the MongoClient class
from pymongo import MongoClient, errors

# global variables for MongoDB host (default port is 27017)
DOMAIN = '172.18.0.2'
PORT = 27017

# use a try-except indentation to catch MongoClient() errors
try:
    # try to instantiate a client instance
    client = MongoClient(
        host = [ str(DOMAIN) + ":" + str(PORT) ],
        serverSelectionTimeoutMS = 3000, # 3 second timeout
        username = "root",
        password = "1234",
    )

    # print the version of MongoDB server if connection successful
    print ("server version:", client.server_info()["version"])

    # get the database_names from the MongoClient()
    database_names = client.list_database_names()

except errors.ServerSelectionTimeoutError as err:
    # set the client and DB name list to 'None' and '[]' if exception
    client = None
    database_names = []

    # catch pymongo.errors.ServerSelectionTimeoutError
    print ("pymongo ERROR:", err)

^G Get Help      ^O Write Out    ^W Where Is     ^K Cut Text     ^J Justify      ^C Cur Pos      ^U Undo         ^A Mark Text    ^_ To Bracket
^X Exit          ^R Read File    ^H Replace      ^U Paste Text  ^T To Spell    ^G Go To Line  ^E Redo        ^M Copy Text    ^Q Where Was
```

Execute the python script from docker to test the mongodb connection

```
Terminal 01:48 24 دسمبر
root@45f45046f2bc: /var/www/html

tazeen@tazeen-HP-15-Notebook-PC:~/mongo-app$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
45f45046f2bc       tazeen_py-mongo    "docker-entrypoint.s..." 4 minutes ago       Up 4 minutes       0.0.0.0:27017->27017/tcp   tazeen_py-mongo_1

tazeen@tazeen-HP-15-Notebook-PC:~/mongo-app$ docker exec -it 45f45046f2bc /bin/bash
root@45f45046f2bc:/# python3 -V
bash: python3-V: command not found
root@45f45046f2bc:/# python3
Python 3.6.9 (default, Oct 8 2020, 12:12:24)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>> exit()
root@45f45046f2bc:/# python3 --version
Python 3.6.9
root@45f45046f2bc:/# pip3 --version
pip 9.0.1 from /usr/lib/python3/dist-packages (python 3.6)
root@45f45046f2bc:/# mongo --version
MongoDB shell version v4.4.2
build info: {
  "version": "4.4.2",
  "gitVersion": "15e73dc5738d2278b688f8929aee605fe4279b0e",
  "opensslVersion": "OpenSSL 1.1.1 11 Sep 2018",
  "modules": [],
  "allocator": "tcmalloc",
  "environment": {
    "distmod": "ubuntu1804",
    "distarch": "x86_64",
    "target_arch": "x86_64"
  }
}

root@45f45046f2bc:/# cd /var/www/html
root@45f45046f2bc:/var/www/html# ls
database-connection
root@45f45046f2bc:/var/www/html# python3 database-connection
server version: 4.4.2
databases: ['admin', 'config', 'local']
root@45f45046f2bc:/var/www/html#
```

Creating python script to save data in mongodb

```
Terminal
21:42 25 دسمبر
tazeen@tazeen-HP-15-Notebook-PC: ~/mongo-app
File Edit View Search Terminal Help
tazeen@tazeen-HP-15-Notebook-PC:~$ cd mongo-app
tazeen@tazeen-HP-15-Notebook-PC:~/mongo-app$ mv ../project-data/2017.csv .
mv: cannot move '../project-data/2017.csv' to './2017.csv': Permission denied
tazeen@tazeen-HP-15-Notebook-PC:~/mongo-app$ sudo mv ../project-data/2017.csv .
[sudo] password for tazeen:
tazeen@tazeen-HP-15-Notebook-PC:~/mongo-app$ ls
2017.csv 2018.csv database-connection database-conn.py
tazeen@tazeen-HP-15-Notebook-PC:~/mongo-app$ sudo nano database-conn.py
tazeen@tazeen-HP-15-Notebook-PC:~/mongo-app$ sudo nano database-conn.py
tazeen@tazeen-HP-15-Notebook-PC:~/mongo-app$
```

Saving data from 2018.csv and 2017.csv which is placed in mongo-app folder. These csv files are header based so starting with first index and inserting data into database.

```
Terminal
23:37 29 دسمبر
tazeen@tazeen-HP-15-Notebook-PC: ~/mongo-app
GNU nano 4.8 database-conn.py
from pymongo import MongoClient, errors
import csv

# global variables for MongoDB host (default port is 27017)
DOMAIN = '172.18.0.2'
PORT = 27017

# use a try-except indentation to catch MongoClient() errors
try:
    # try to instantiate a client instance
    client = MongoClient(
        host = [ str(DOMAIN) + ":" + str(PORT) ],
        serverSelectionTimeoutMS = 3000, # 3 second timeout
        username = "root",
        password = "1234",
    )
    db=client.airline

    with open('2018.csv', 'r') as csvfile:
        header = ["FL_DATE","OP_CARRIER","OP_CARRIER_FL_NUM","ORIGIN","DEST","CRS_DEP_TIME","DEP_TIME","DEP_DELAY","TAXI_OUT","WHEELS_OFF","WHEE
        reader = csv.reader(csvfile)
        headerRow = next(reader)
        for row in reader:
            doc={}
            for n in range(0,len(header)):
                doc[header[n]] = row[n]
            db.flights.insert(doc)

    with open('2017.csv', 'r') as csvfile:
        header = ["FL_DATE","OP_CARRIER","OP_CARRIER_FL_NUM","ORIGIN","DEST","CRS_DEP_TIME","DEP_TIME","DEP_DELAY","TAXI_OUT","WHEELS_OFF","WHEE
        reader = csv.reader(csvfile)
        headerRow = next(reader)
        for row in reader:
            doc={}

Read 42 Lines
Get Help Write Out Where Is Cut Text Justify Cur Pos Undo Mark Text To Bracket
Exit Read File Replace Paste Text To Spell Go To Line Redo Copy Text Where Was
```

```
GNU nano 4.8 database-conn.py
# try to instantiate a client instance
client = MongoClient(
    host = [ str(DOMAIN) + ":" + str(PORT) ],
    serverSelectionTimeoutMS = 3000, # 3 second timeout
    username = "root",
    password = "1234",
)
db=client.airline

with open('2018.csv', 'r') as csvfile:
    header = ["FL_DATE", "OP_CARRIER", "OP_CARRIER_FL_NUM", "ORIGIN", "DEST", "CRS_DEP_TIME", "DEP_TIME", "DEP_DELAY", "TAXI_OUT", "WHEELS_OFF", "WHEELS_ON"]
    reader = csv.reader(csvfile)
    headerRow = next(reader)
    for row in reader:
        doc={}
        for n in range(0,len(header)):
            doc[header[n]] = row[n]
        db.flights.insert(doc)

with open('2017.csv', 'r') as csvfile:
    header = ["FL_DATE", "OP_CARRIER", "OP_CARRIER_FL_NUM", "ORIGIN", "DEST", "CRS_DEP_TIME", "DEP_TIME", "DEP_DELAY", "TAXI_OUT", "WHEELS_OFF", "WHEELS_ON"]
    reader = csv.reader(csvfile)
    headerRow = next(reader)
    for row in reader:
        doc={}
        for n in range(0,len(header)):
            doc[header[n]] = row[n]
        db.flights.insert(doc)

# mongoimport --db=airline-db --collection=flights --type=csv --file=2018.csv --headerline --username=root --password=1234 --uri "mongodb://root:1234@localhost:27017/"
except errors.ServerSelectionTimeoutError as err:
    # catch pymongo.errors.ServerSelectionTimeoutError
    print ("pymongo ERROR:", err)

```

```
root@45f45046f2bc: /var/www/html
tazeen@tazeen-HP-15-Notebook-PC:~$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS
45f45046f2bc       tazeen_py-mongo    "docker-entrypoint.s..." 44 hours ago        Up 2 hours          0.0.0.0:27017->27017/tcp
03c4a52c7579       cloudera/quickstart "/usr/bin/docker-qu..." 3 weeks ago         Exited (137) 2 weeks ago
gitated_raman

tazeen@tazeen-HP-15-Notebook-PC:~$ docker exec -it 45f45046f2bc /bin/bash
root@45f45046f2bc:/# cd /var/www/html
bash: cd: /var/www/html: No such file or directory
root@45f45046f2bc:/# cd /var/www/html
root@45f45046f2bc:/var/www/html# ls
2017.csv 2018.csv database-conn.py database-connection
root@45f45046f2bc:/var/www/html# python3 database-conn.py

```

Connecting to mongodb to check that data is inserted in airline database.

```
Terminal 02:10 30 دسمبر
root@60bcea4e3233: /var/www/html

root@60bcea4e3233:/var/www/html# mongo --username=root --password=1234
MongoDB shell version v4.4.2
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("544d3d66-a03c-4cdd-b0e7-46a65e1bb9bf") }
MongoDB server version: 4.4.2
---
The server generated these startup warnings when booting:
  2020-12-29T17:35:54.075+00:00: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://dochub.mongodb.org/core/prodnotes-filesystem
---
---
  Enable MongoDB's free cloud-based monitoring service, which will then receive and display
  metrics about your deployment (disk utilization, CPU, operation statistics, etc).

  The monitoring data will be available on a MongoDB website with a unique URL accessible to you
  and anyone you share the URL with. MongoDB may use this information to make product
  improvements and to suggest MongoDB products and deployment options to you.

  To enable free monitoring, run the following command: db.enableFreeMonitoring()
  To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
> show dbs;
admin      0.000GB
airline    1.865GB
config     0.000GB
local      0.000GB
>
```

II- Batch processing layer:

Pulling image for hadoop container

```
Terminal 16:38 26 دسمبر
tazeen@tazeen-HP-15-Notebook-PC: ~

tazeen@tazeen-HP-15-Notebook-PC:~$ docker pull sequenceiq/hadoop-docker:2.7.0
2.7.0: Pulling from sequenceiq/hadoop-docker
Image docker.io/sequenceiq/hadoop-docker:2.7.0 uses outdated schema1 manifest format. Please upgrade to a schema2 image for better future comp
atibility. More information at https://docs.docker.com/registry/spec/deprecated-schema-v1/
b253335dcf03: Pulling fs layer
a3ed95cae02: Pulling fs layer
69623ef05416: Pulling fs layer
63aebddf4bce: Pulling fs layer
46305a4cda1d: Pulling fs layer
70ff65ec2366: Pulling fs layer
72accdc282f3: Pulling fs layer
5298ddb3b339: Pulling fs layer
ec461d25c2ea: Pulling fs layer
315b476b23a4: Pull complete
6e6acc31f8b1: Pull complete
38a227158d97: Pull complete
319a3b8afa25: Pull complete
11e1e16af8f3: Pull complete
834533551a37: Pull complete
c24255b6d9f4: Pull complete
8b4ea3c67dc2: Pull complete
40ba2c2cdf73: Pull complete
5424a04bc240: Pull complete
7df43f09096d: Pull complete
b34787ee2fde: Pull complete
4eaa47927d15: Pull complete
cb95b9da9646: Pull complete
e495e287a108: Pull complete
3158ca49a54c: Pull complete
33b5a5de9544: Pull complete
d6f46cf55f0f: Pull complete
40c19fb76cfd: Pull complete
018a1f3d7249: Pull complete
40f52c973507: Pull complete
49dca4de47eb: Pull complete
d26082bd2aa9: Pull complete
c4f97d87af86: Pull complete
fb839f93fc0f: Pull complete
43661864505e: Pull complete
```

```
Terminal
16:38 26 دسمبر
tazeen@tazeen-HP-15-Notebook-PC: ~

319a3b8afa25: Pull complete
11e1e16af8f3: Pull complete
834533551a37: Pull complete
c24255b6d9f4: Pull complete
8b4ea3c67dc2: Pull complete
40ba2c2cdf73: Pull complete
5424a04bc240: Pull complete
7df43f09096d: Pull complete
b34787ee2fde: Pull complete
4eaa47927d15: Pull complete
cb95b9da9646: Pull complete
e495e287a108: Pull complete
3158ca49a54c: Pull complete
33b5a5de9544: Pull complete
d6f46cf55f0f: Pull complete
40c19fb76cfd: Pull complete
018a1f3d7249: Pull complete
40f52c973507: Pull complete
49dca4de47eb: Pull complete
d26082bd2aa9: Pull complete
c4f97d87af86: Pull complete
fb839f93fc0f: Pull complete
43661864505e: Pull complete
d8908a83648e: Pull complete
af8b686deb23: Pull complete
c1214abd7b96: Pull complete
9d00f27ba8d2: Pull complete
09f787a7573b: Pull complete
4e86267d5247: Pull complete
3876cba35aed: Pull complete
23df48ffdb39: Pull complete
646aedbc2bb6: Pull complete
60a65f8179cf: Pull complete
046b321f8081: Pull complete
Digest: sha256:a40761746eca036fee6aafdf9fddbd6878ac3dd9a7cd83c0f3f5d8a0e6350c76a
Status: Downloaded newer image for sequenceiq/hadoop-docker:2.7.0
docker.io/sequenceiq/hadoop-docker:2.7.0
tazeen@tazeen-HP-15-Notebook-PC:~$
```

Updating docker compose to add another container for hadoop which links to pymongo container

```
Terminal
04:43 28 دسمبر
tazeen@tazeen-HP-15-Notebook-PC: ~

tazeen@tazeen-HP-15-Notebook-PC:~$ sudo nano docker-compose.yml
tazeen@tazeen-HP-15-Notebook-PC:~$ sudo docker-compose up
WARNING: The PWD variable is not set. Defaulting to a blank string.
Creating network "tazeen_bridge" with the default driver
Creating tazeen_py-mongo_1 ... done
Creating tazeen_hdfs_1 ... done
Attaching to tazeen_py-mongo_1, tazeen_hdfs_1
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:18.231+00:00"},"s":"I", "c":"CONTROL", "id":23285, "ctx":"main","msg":"Automatically disabling TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'"
hdfs_1 | /
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:18.234+00:00"},"s":"W", "c":"ASIO", "id":22601, "ctx":"main","msg":"No TransportLayer configured during NetworkInterface startup"}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:18.235+00:00"},"s":"I", "c":"NETWORK", "id":4648601, "ctx":"main","msg":"Implicit TCP FastOpen unavailable. If TCP FastOpen is required, set tcpFastOpenServer, tcpFastOpenClient, and tcpFastOpenQueueSize."}
hdfs_1 | Starting sshd: [ OK ]
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:18.235+00:00"},"s":"I", "c":"STORAGE", "id":4615611, "ctx":"initandlisten","msg":"MongoDB starting", "attr":{"pid":1, "port":27017, "dbPath":"/data/db", "architecture":"64-bit", "host":"1f7d90ced38e"}}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:18.235+00:00"},"s":"I", "c":"CONTROL", "id":23403, "ctx":"initandlisten","msg":"Build Info", "attr":{"buildInfo":{"version":"4.4.2","gitVersion":"15e73dc5738d2278b688f8929aee605fe4279b0e","openSSLVersion":"OpenSSL 1.1.1 11 Sep 2018","modules":[],"allocator":"tcmalloc","environment":{"distmod":"ubuntu1804","distarch":"x86_64","target_arch":"x86_64"}}}}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:18.235+00:00"},"s":"I", "c":"CONTROL", "id":51765, "ctx":"initandlisten","msg":"Operating System", "attr":{"os":{"name":"Ubuntu", "version":"18.04"}}}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:18.235+00:00"},"s":"I", "c":"CONTROL", "id":21951, "ctx":"initandlisten","msg":"Options set by command line", "attr":{"options":{"net":{"bindIp":"*"},"security":{"authorization":"enabled"}}}}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:18.237+00:00"},"s":"I", "c":"STORAGE", "id":22270, "ctx":"initandlisten","msg":"Storage engine to use detected by data files", "attr":{"dbpath":"/data/db", "storageEngine":"wiredTiger"}}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:18.237+00:00"},"s":"I", "c":"STORAGE", "id":22297, "ctx":"initandlisten","msg":"Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://dochub.mongodb.org/core/prodnotes-filesystem", "tags":["startupWarnings"]}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:18.237+00:00"},"s":"I", "c":"STORAGE", "id":22315, "ctx":"initandlisten","msg":"Opening WiredTiger", "attr":{"config":"create,cache_size=5444M,session_max=33000,eviction=(threads_min=4,threads_max=4),config_base=false,statistics=(fast),log=(enabled=true,archive=true,path=journal,compressor=snappy),file_manager=(close_idle_time=100000,close_scan_interval=10,close_handle_minimum=250),statistics_log=(wait=0),verbose=[recovery_progress,checkpoint_progress,compact_progress]}"}}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:19.268+00:00"},"s":"I", "c":"STORAGE", "id":22430, "ctx":"initandlisten","msg":"WiredTiger message", "attr":{"message":["[1609112239:268056][1:0x7f0d75734ac0], txn-recover: [WT_VERB_RECOVERY_PROGRESS] Recovering log 3 through 4"]}}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:19.326+00:00"},"s":"I", "c":"STORAGE", "id":22430, "ctx":"initandlisten","msg":"WiredTiger message", "attr":{"message":["[1609112239:326158][1:0x7f0d75734ac0], txn-recover: [WT_VERB_RECOVERY_PROGRESS] Recovering log 4 through 4"]}}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:19.417+00:00"},"s":"I", "c":"STORAGE", "id":22430, "ctx":"initandlisten","msg":"WiredTiger message", "attr":{"message":["[1609112239:417644][1:0x7f0d75734ac0], txn-recover: [WT_VERB_RECOVERY | WT_VERB_RECOVERY_PROGRESS] Main recovery lo
```

```

es Terminal 04:43 28 دسمبر
tazeen@tazeen-HP-15-Notebook-PC: ~
op: starting at 3/14464 to 4/256}}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:19.536+00:00"},"s":"I", "c":"STORAGE", "id":22430, "ctx":"initandlisten","msg":"WiredTiger message","attr":{"message":"[1609112239:536855][1:0x7f0d75734ac0], txn-recover: [WT_VERB_RECOVERY_PROGRESS] Recovering log 3 through 4"}}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:19.631+00:00"},"s":"I", "c":"STORAGE", "id":22430, "ctx":"initandlisten","msg":"WiredTiger message","attr":{"message":"[1609112239:631734][1:0x7f0d75734ac0], txn-recover: [WT_VERB_RECOVERY_PROGRESS] Recovering log 4 through 4"}}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:19.713+00:00"},"s":"I", "c":"STORAGE", "id":22430, "ctx":"initandlisten","msg":"WiredTiger message","attr":{"message":"[1609112239:713227][1:0x7f0d75734ac0], txn-recover: [WT_VERB_RECOVERY | WT_VERB_RECOVERY_PROGRESS] Set global recovery timestamp: (0, 0)}}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:19.713+00:00"},"s":"I", "c":"STORAGE", "id":22430, "ctx":"initandlisten","msg":"WiredTiger message","attr":{"message":"[1609112239:713292][1:0x7f0d75734ac0], txn-recover: [WT_VERB_RECOVERY | WT_VERB_RECOVERY_PROGRESS] Set global oldest timestamp: (0, 0)}}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:20.689+00:00"},"s":"I", "c":"STORAGE", "id":4795906, "ctx":"initandlisten","msg":"WiredTiger opened","attr":{"durationMillis":2452}}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:20.689+00:00"},"s":"I", "c":"RECOVERY", "id":23987, "ctx":"initandlisten","msg":"WiredTiger recoveryTimestamp","attr":{"recoveryTimestamp":{"timestamp":{"t":0,"i":0}}}}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:20.691+00:00"},"s":"I", "c":"STORAGE", "id":4366408, "ctx":"initandlisten","msg":"No table logging settings modifications are required for existing WiredTiger tables","attr":{"loggingEnabled":true}}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:20.692+00:00"},"s":"I", "c":"STORAGE", "id":22262, "ctx":"initandlisten","msg":"Timestamp monitor starting"}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:20.771+00:00"},"s":"I", "c":"STORAGE", "id":20536, "ctx":"initandlisten","msg":"Flow Control is enabled on this deployment"}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:20.773+00:00"},"s":"I", "c":"FTDC", "id":20625, "ctx":"initandlisten","msg":"Initializing full-time diagnostic data capture","attr":{"dataDirectory":"/data/db/diagnostic.data"}}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:20.776+00:00"},"s":"I", "c":"NETWORK", "id":23015, "ctx":"listener","msg":"Listening on","attr":{"address":"/tmp/mongod-27017.sock"}}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:20.776+00:00"},"s":"I", "c":"NETWORK", "id":23015, "ctx":"listener","msg":"Listening on","attr":{"address":"0.0.0.0"}}
py-mongo_1 | {"t":{"$date":"2020-12-27T23:37:20.776+00:00"},"s":"I", "c":"NETWORK", "id":23016, "ctx":"listener","msg":"Waiting for connections","attr":{"port":27017,"ssl":"off"}}
hdfs_1 | Starting namenodes on [51cdf46f4742]
hdfs_1 | 51cdf46f4742: starting namenode, logging to /usr/local/hadoop/logs/hadoop-root-namenode-51cdf46f4742.out
hdfs_1 | localhost: starting datanode, logging to /usr/local/hadoop/logs/hadoop-root-datanode-51cdf46f4742.out
hdfs_1 | Starting secondary namenodes [0.0.0.0]
hdfs_1 | 0.0.0.0: starting secondarynamenode, logging to /usr/local/hadoop/logs/hadoop-root-secondarynamenode-51cdf46f4742.out
hdfs_1 | starting yarn daemons
hdfs_1 | starting resourcemanager, logging to /usr/local/hadoop/logs/yarn--resourcemanager-51cdf46f4742.out
hdfs_1 | localhost: starting nodemanager, logging to /usr/local/hadoop/logs/yarn-root-nodemanager-51cdf46f4742.out

```

```

es Terminal 18:52 29 دسمبر
tazeen@tazeen-HP-15-Notebook-PC: ~
GNU nano 4.8 docker-compose.yml
version: "3.2"
services:
  hdfs:
    image: sequenceiq/hadoop-docker:2.7.0
    depends_on:
      - py-mongo
    networks:
      - default_bridge
    volumes:
      - ./mongo-app:/var/www/html
  py-mongo:
    # build the image from Dockerfile
    build:
      context: .
    volumes:
      - ./mongo-data:/data/db
      - ./mongo-app:/var/www/html
    ports:
      - "27017:27017"
    environment:
      - MONGO_INITDB_ROOT_USERNAME=root
      - MONGO_INITDB_ROOT_PASSWORD=1234
    networks:
      - default_bridge

networks:
  default_bridge:
    name: tazeen_bridge
volumes:
  mongo-app:

```

Hadoop container is correctly set up which can be viewed as:


```
Terminal 20:37 29 دسمبر
tazeen@tazeen-HP-15-Notebook-PC: ~/mongo-app

{
  "SandboxKey": "/var/run/docker/netns/c8e84133c070",
  "SecondaryIPAddresses": null,
  "SecondaryIPv6Addresses": null,
  "EndpointID": "",
  "Gateway": "",
  "GlobalIPv6Address": "",
  "GlobalIPv6PrefixLen": 0,
  "IPAddress": "",
  "IPPrefixLen": 0,
  "IPv6Gateway": "",
  "MacAddress": "",
  "Networks": {
    "tazeen_bridge": {
      "IPAMConfig": null,
      "Links": null,
      "Aliases": [
        "py-mongo",
        "60bcea4e3233"
      ],
      "NetworkID": "02014f58a32fb5e8163e33eacff1782f8279a40ba353db237b6bd790b66b22b4",
      "EndpointID": "c967331fa810b36be9def589d674bcf078467986d52ad2e4d7ec02138e54c7ca",
      "Gateway": "172.28.0.1",
      "IPAddress": "172.28.0.2",
      "IPPrefixLen": 16,
      "IPv6Gateway": "",
      "GlobalIPv6Address": "",
      "GlobalIPv6PrefixLen": 0,
      "MacAddress": "02:42:ac:1c:00:02",
      "DriverOpts": null
    }
  }
}

tazeen@tazeen-HP-15-Notebook-PC:~$ cd mongo-app
tazeen@tazeen-HP-15-Notebook-PC:~/mongo-app$ sudo nano batch-data.py
[sudo] password for tazeen:
tazeen@tazeen-HP-15-Notebook-PC:~/mongo-app$
```

Creating multiple python scripts to export mongodb data for hadoop container. Because scripts take a lot of time in processing and stuck due to less resources. Each script creates text file data for specific months, total data is used from 1/1/2017 – 31/5/2017 and 1/1/2018 – 31/5/2018. The screenshots for batch-data.py is added below, on it's execution mongodb data is exported in text files shared using same bridge network.

```
Terminal 02:51 5 جنوری
tazeen@tazeen-HP-15-Notebook-PC: ~
root@cd01b0d988fa: /var/www/html mongo-app/batch-data.py
GNU nano 4.8
from pymongo import MongoClient, errors
import csv
import datetime
import json

# global variables for MongoDB host (default port is 27017)
DOMAIN = '172.18.0.2'
PORT = 27017

# use a try-except indentation to catch MongoClient() errors
try:
    # try to instantiate a client instance
    client = MongoClient(
        host = [ str(DOMAIN) + ":" + str(PORT) ],
        serverSelectionTimeoutMS = 3000, # 3 second timeout
        username = "root",
        password = "1234",
    )
    db = client.airline;
    batchData = db.flights.find({'$or': [
        {'$and': [{'FL_DATE': {'$gte': '2018-01-01'}}, {'FL_DATE': {'$lte': '2018-03-31'}}]},
        {'$and': [{'FL_DATE': {'$gte': '2017-01-01'}}, {'FL_DATE': {'$lte': '2017-03-31'}}]}
    ]}, {'CANCELLED':1, 'ARR_DELAY':1, 'FL_DATE':1, '_id':0 });
    batchStr = str();
    for document in batchData:
        print (document);
        if batchStr:
            batchStr = batchStr + ",";
        batchStr = batchStr + json.dumps(document);
        #batchJson.append(document);
    with open('batch-data.txt', 'w') as outfile:
        outfile.write(batchStr)

Get Help Write Out Where Is Cut Text Justify Cur Pos Undo Mark Text To Bracket
Exit Read File Replace Paste Text To Spell Go To Line Redo Copy Text Where Was
```

```
es Terminal 02:51 5 جنوري
tazeen@tazeen-HP-15-Notebook-PC: ~
root@c0d1b0d988fa: /var/www/html mongo-app/batch-data.py
GNU nano 4.8
# global variables for MongoDB host (default port is 27017)
DOMAIN = '172.18.0.2'
PORT = 27017

# use a try-except indentation to catch MongoClient() errors
try:
    # try to instantiate a client instance
    client = MongoClient(
        host = [ str(DOMAIN) + ":" + str(PORT) ],
        serverSelectionTimeoutMS = 3000, # 3 second timeout
        username = "root",
        password = "1234",
    )
    db = client.airline;
    batchData = db.flights.find({'$or':[
        {'$and':[{'FL_DATE':{'$gte': '2018-01-01'}},{'FL_DATE':{'$lte':'2018-03-31'}}]},
        {'$and':[{'FL_DATE':{'$gte': '2017-01-01'}},{'FL_DATE':{'$lte':'2017-03-31'}}]}
    ]}, {'CANCELLED':1, 'ARR_DELAY':1, 'FL_DATE':1, '_id':0 });
    batchStr = str();
    for document in batchData:
        print (document);
        if batchStr:
            batchStr = batchStr + ";";
            batchStr = batchStr + json.dumps(document);
            #batchJson.append(document);
    with open('batch-data.txt', 'w') as outfile:
        outfile.write(batchStr)

except errors.ServerSelectionTimeoutError as err:
    # catch pymongo.errors.ServerSelectionTimeoutError
    print ("pymongo ERROR:", err)

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos ^U Undo ^A Mark Text ^_ To Bracket
^X Exit ^R Read File ^\ Replace ^P Paste Text ^T To Spell ^_ Go To Line ^E Redo ^O Copy Text ^Q Where Was
```

After execution, same file is updated for different data.

```
es Terminal 02:51 5 جنوري
tazeen@tazeen-HP-15-Notebook-PC: ~
root@c0d1b0d988fa: /var/www/html mongo-app/batch-data.py Modified
GNU nano 4.8
from pymongo import MongoClient, errors
import csv
import datetime
import json

# global variables for MongoDB host (default port is 27017)
DOMAIN = '172.18.0.2'
PORT = 27017

# use a try-except indentation to catch MongoClient() errors
try:
    # try to instantiate a client instance
    client = MongoClient(
        host = [ str(DOMAIN) + ":" + str(PORT) ],
        serverSelectionTimeoutMS = 3000, # 3 second timeout
        username = "root",
        password = "1234",
    )
    db = client.airline;
    batchData = db.flights.find({'$or':[
        {'$and':[{'FL_DATE':{'$gte': '2018-04-01'}},{'FL_DATE':{'$lte':'2018-05-31'}}]},
        {'$and':[{'FL_DATE':{'$gte': '2017-04-01'}},{'FL_DATE':{'$lte':'2017-05-31'}}]}
    ]}, {'CANCELLED':1, 'ARR_DELAY':1, 'FL_DATE':1, '_id':0 });
    batchStr = str();
    for document in batchData:
        print (document);
        if batchStr:
            batchStr = batchStr + ";";
            batchStr = batchStr + json.dumps(document);
            #batchJson.append(document);
    with open('batch-data1.txt', 'w') as outfile:
        outfile.write(batchStr)

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos ^U Undo ^A Mark Text ^_ To Bracket
^X Exit ^R Read File ^\ Replace ^P Paste Text ^T To Spell ^_ Go To Line ^E Redo ^O Copy Text ^Q Where Was
```

```
GNU nano 4.8 mongo-app/batch-data.py Modified
# global variables for MongoDB host (default port is 27017)
DOMAIN = '172.18.0.2'
PORT = 27017

# use a try-except indentation to catch MongoClient() errors
try:
    # try to instantiate a client instance
    client = MongoClient(
        host = [ str(DOMAIN) + ":" + str(PORT) ],
        serverSelectionTimeoutMS = 3000, # 3 second timeout
        username = "root",
        password = "1234",
    )
    db = client.airline;
    batchData = db.flights.find({'$or':[
        {'$and':[{'FL_DATE':{'$gte':'2018-04-01'}},{'FL_DATE':{'$lte':'2018-05-31'}}]},
        {'$and':[{'FL_DATE':{'$gte':'2017-04-01'}},{'FL_DATE':{'$lte':'2017-05-31'}}]}
    ]}, {'CANCELLED':1, 'ARR_DELAY':1, 'FL_DATE':1, '_id':0 });
    batchStr = str();
    for document in batchData:
        print (document);
        if batchStr:
            batchStr = batchStr + ";";
        batchStr = batchStr + json.dumps(document);
        #batchJson.append(document);
    with open('batch-data1.txt', 'w') as outfile:
        outfile.write(batchStr)

except errors.ServerSelectionTimeoutError as err:
    # catch pymongo.errors.ServerSelectionTimeoutError
    print ("pymongo ERROR:", err)
[]

^G Get Help      ^O Write Out    ^W Where Is     ^K Cut Text     ^J Justify      ^C Cur Pos      ^U Undo         ^A Mark Text    ^_ To Bracket
^X Exit          ^R Read File    ^H Replace      ^U Paste Text   ^T To Spell     ^_ Go To Line   ^E Redo         ^G Copy Text    ^Q Where Was
```

Executing the scripts by the following command

```
03:05 30 دسمبر
root@60bcea4e3233: /var/www/html

tazeen@tazeen-HP-15-Notebook-PC:~$ docker ps
dockerCONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              NAMES              PORTS
167c92585cc2             sequenceiq/hadoop-docker:2.7.0   "/etc/bootstrap.sh -d"   45 hours ago       Up 11 minutes      tazeen_hdfs_1      2122/tcp, 8030-8033/tcp, 8040/tcp, 8042/tcp, 8088/tcp, 19888/tcp, 49707/tcp, 50010/tcp, 50020/tcp, 50070/tcp, 50075/tcp, 50090/tcp
60bcea4e3233             tazeen_py-mongo    "docker-entrypoint.s..."   46 hours ago       Up 11 minutes      tazeen_py-mongo_1  0.0.0.0:27017->27017/tcp

tazeen@tazeen-HP-15-Notebook-PC:~$ docker exec -it 60bcea4e3233 /bin/bash
root@60bcea4e3233:/# cd /var/www/html
root@60bcea4e3233:/var/www/html# python3 batch-data.py []
```

```
Terminal 17:52 3 جنوری
root@79f4d4bad0c8: /var/www/html

SED_TIME': '176.0', 'AIR_TIME': '148.0', 'DISTANCE': '1189.0', 'CARRIER_DELAY': '', 'WEATHER_DELAY': '', 'NAS_DELAY': '', 'SECURITY_DELAY': ''
, 'LATE_AIRCRAFT_DELAY': '', 'Unnamed': '' }
{ 'FL_DATE': '2018-06-30', 'OP_CARRIER': 'B6', 'OP_CARRIER_FL_NUM': '2945', 'ORIGIN': 'LAS', 'DEST': 'JFK', 'CRS_DEP_TIME': '1554', 'DEP_TIME':
'1554.0', 'DEP_DELAY': '0.0', 'TAXI_OUT': '12.0', 'WHEELS_OFF': '1606.0', 'WHEELS_ON': '2336.0', 'TAXI_IN': '6.0', 'CRS_ARR_TIME': '2357', 'A
RR_TIME': '2342.0', 'ARR_DELAY': '-15.0', 'CANCELLED': '0.0', 'CANCELLATION_CODE': '', 'DIVERTED': '0.0', 'CRS_ELAPSED_TIME': '303.0', 'ACTUAL
ELAPSED_TIME': '288.0', 'AIR_TIME': '270.0', 'DISTANCE': '2248.0', 'CARRIER_DELAY': '', 'WEATHER_DELAY': '', 'NAS_DELAY': '', 'SECURITY_DELAY
': '', 'LATE_AIRCRAFT_DELAY': '', 'Unnamed': '' }
{ 'FL_DATE': '2018-06-30', 'OP_CARRIER': 'B6', 'OP_CARRIER_FL_NUM': '2949', 'ORIGIN': 'BOS', 'DEST': 'SAV', 'CRS_DEP_TIME': '1500', 'DEP_TIME':
'1534.0', 'DEP_DELAY': '34.0', 'TAXI_OUT': '11.0', 'WHEELS_OFF': '1545.0', 'WHEELS_ON': '1753.0', 'TAXI_IN': '8.0', 'CRS_ARR_TIME': '1737', 'A
RR_TIME': '1801.0', 'ARR_DELAY': '28.0', 'CANCELLED': '0.0', 'CANCELLATION_CODE': '', 'DIVERTED': '0.0', 'CRS_ELAPSED_TIME': '153.0', 'ACTUAL
ELAPSED_TIME': '147.0', 'AIR_TIME': '128.0', 'DISTANCE': '901.0', 'CARRIER_DELAY': '0.0', 'WEATHER_DELAY': '0.0', 'NAS_DELAY': '28.0', 'SECUR
ITY_DELAY': '0.0', 'LATE_AIRCRAFT_DELAY': '0.0', 'Unnamed': '' }
{ 'FL_DATE': '2018-06-30', 'OP_CARRIER': 'B6', 'OP_CARRIER_FL_NUM': '2950', 'ORIGIN': 'SAV', 'DEST': 'BOS', 'CRS_DEP_TIME': '1810', 'DEP_TIME':
'1833.0', 'DEP_DELAY': '23.0', 'TAXI_OUT': '10.0', 'WHEELS_OFF': '1843.0', 'WHEELS_ON': '2051.0', 'TAXI_IN': '5.0', 'CRS_ARR_TIME': '2034', 'A
RR_TIME': '2056.0', 'ARR_DELAY': '22.0', 'CANCELLED': '0.0', 'CANCELLATION_CODE': '', 'DIVERTED': '0.0', 'CRS_ELAPSED_TIME': '144.0', 'ACTUAL
ELAPSED_TIME': '172.0', 'AIR_TIME': '128.0', 'DISTANCE': '901.0', 'CARRIER_DELAY': '2.0', 'WEATHER_DELAY': '0.0', 'NAS_DELAY': '0.0', 'SECURI
TY_DELAY': '0.0', 'LATE_AIRCRAFT_DELAY': '20.0', 'Unnamed': '' }
{ 'FL_DATE': '2018-06-30', 'OP_CARRIER': 'B6', 'OP_CARRIER_FL_NUM': '2951', 'ORIGIN': 'TPA', 'DEST': 'SJU', 'CRS_DEP_TIME': '820', 'DEP_TIME':
'812.0', 'DEP_DELAY': '-8.0', 'TAXI_OUT': '11.0', 'WHEELS_OFF': '823.0', 'WHEELS_ON': '1101.0', 'TAXI_IN': '3.0', 'CRS_ARR_TIME': '1114', 'ARR
TIME': '1104.0', 'ARR_DELAY': '-10.0', 'CANCELLED': '0.0', 'CANCELLATION_CODE': '', 'DIVERTED': '0.0', 'CRS_ELAPSED_TIME': '174.0', 'ACTUAL_E
LAPSED_TIME': '172.0', 'AIR_TIME': '158.0', 'DISTANCE': '1237.0', 'CARRIER_DELAY': '', 'WEATHER_DELAY': '', 'NAS_DELAY': '', 'SECURITY_DELAY':
'', 'LATE_AIRCRAFT_DELAY': '', 'Unnamed': '' }
{ 'FL_DATE': '2018-06-30', 'OP_CARRIER': 'B6', 'OP_CARRIER_FL_NUM': '2953', 'ORIGIN': 'FLL', 'DEST': 'SJU', 'CRS_DEP_TIME': '828', 'DEP_TIME':
'838.0', 'DEP_DELAY': '10.0', 'TAXI_OUT': '19.0', 'WHEELS_OFF': '857.0', 'WHEELS_ON': '1116.0', 'TAXI_IN': '4.0', 'CRS_ARR_TIME': '1100', 'ARR
TIME': '1120.0', 'ARR_DELAY': '20.0', 'CANCELLED': '0.0', 'CANCELLATION_CODE': '', 'DIVERTED': '0.0', 'CRS_ELAPSED_TIME': '152.0', 'ACTUAL_E
LAPSED_TIME': '162.0', 'AIR_TIME': '139.0', 'DISTANCE': '1046.0', 'CARRIER_DELAY': '10.0', 'WEATHER_DELAY': '0.0', 'NAS_DELAY': '10.0', 'SECURI
TY_DELAY': '0.0', 'LATE_AIRCRAFT_DELAY': '0.0', 'Unnamed': '' }
{ 'FL_DATE': '2018-06-30', 'OP_CARRIER': 'B6', 'OP_CARRIER_FL_NUM': '2954', 'ORIGIN': 'SJU', 'DEST': 'FLL', 'CRS_DEP_TIME': '2134', 'DEP_TIME':
'2130.0', 'DEP_DELAY': '-4.0', 'TAXI_OUT': '13.0', 'WHEELS_OFF': '2143.0', 'WHEELS_ON': '2356.0', 'TAXI_IN': '5.0', 'CRS_ARR_TIME': '9', 'ARR
TIME': '1.0', 'ARR_DELAY': '-8.0', 'CANCELLED': '0.0', 'CANCELLATION_CODE': '', 'DIVERTED': '0.0', 'CRS_ELAPSED_TIME': '155.0', 'ACTUAL_E
LAPSED_TIME': '151.0', 'AIR_TIME': '133.0', 'DISTANCE': '1046.0', 'CARRIER_DELAY': '', 'WEATHER_DELAY': '', 'NAS_DELAY': '', 'SECURITY_DELAY':
'', 'LATE_AIRCRAFT_DELAY': '', 'Unnamed': '' }
{ 'FL_DATE': '2018-06-30', 'OP_CARRIER': 'B6', 'OP_CARRIER_FL_NUM': '2989', 'ORIGIN': 'JFK', 'DEST': 'MCO', 'CRS_DEP_TIME': '2100', 'DEP_TIME':
'2100.0', 'DEP_DELAY': '0.0', 'TAXI_OUT': '32.0', 'WHEELS_OFF': '2132.0', 'WHEELS_ON': '2334.0', 'TAXI_IN': '36.0', 'CRS_ARR_TIME': '2358', 'A
RR_TIME': '10.0', 'ARR_DELAY': '12.0', 'CANCELLED': '0.0', 'CANCELLATION_CODE': '', 'DIVERTED': '0.0', 'CRS_ELAPSED_TIME': '170.0', 'ACTUAL_E
LAPSED_TIME': '190.0', 'AIR_TIME': '122.0', 'DISTANCE': '944.0', 'CARRIER_DELAY': '', 'WEATHER_DELAY': '', 'NAS_DELAY': '', 'SECURITY_DELAY':
'', 'LATE_AIRCRAFT_DELAY': '', 'Unnamed': '' }
root@79f4d4bad0c8: /var/www/html#
```

After executing the batch-data.py script with different data, two files batch-data.txt and batch-data1.txt are created which will be used in HDFS.

Now taking the above created files shared on the same bridge network as input files for hadoop and executing a map/reduce job. The map/reduce job calculates the average delay time minutes for each year. The data consists of canceled and delayed flights. Firstly, removing the canceled flights to find the correct delay time. Secondly, only considering the positive delay time as negative represents the flights were departed late. The map and reduce job files are shown below.

```
Terminal 17:16 3 جنوری
tazeen@tazeen-HP-15-Notebook-PC: ~

tazeen@tazeen-HP-15-Notebook-PC:~$ docker ps
CONTAINER ID        IMAGE                                     COMMAND                  CREATED            STATUS              PORTS
3145279367fe       sequencelq/hadoop-docker:2.7.0        "/etc/bootstrap.sh -d"  41 hours ago      Up 4 hours          2122/tcp, 8030-8033/tcp,
8040/tcp, 8042/tcp, 8088/tcp, 19888/tcp, 49707/tcp, 50010/tcp, 50020/tcp, 50070/tcp, 50075/tcp, 50090/tcp
79f4d4bad0c8       tazeen_py-mongo                        "docker-entrypoint.s..."  41 hours ago      Up 4 hours          0.0.0.0:27017->27017/tcp
                                     tazeen_py-mongo_1

tazeen@tazeen-HP-15-Notebook-PC:~$ docker exec -it 3145279367fe /bin/bash
bash-4.1# vi /usr/local/hadoop/share/hadoop/mapreduce/mapper.py
bash-4.1#
```

```
es Terminal 17:12 3 جنوری
tazeen@tazeen-HP-15-Notebook-PC: ~
#!/usr/bin/python
import sys
import json

try:
    #input from json files exported from database
    data_arr = list()
    for line in sys.stdin:
        data_arr = data_arr + line.strip().split(';')
    for obj in data_arr:
        #print 'obj' , obj
        json_obj = eval(obj)
        #all delayed flights which were not cancelled
        if json_obj['CANCELLED'] and json_obj['ARR_DELAY'] and int(float(json_obj['CANCELLED'])) == 0:
            fl_delay = float(json_obj['ARR_DELAY'])
            if fl_delay > 0:
                #write the results to standard output STDOUT
                print json_obj['FL_DATE'].split('-')[0], '\t', fl_delay
except Exception as e:
    print 'Error occured at mapper', e
    pass

-- INSERT --
```

```
bash-4.1# vi /usr/local/hadoop/share/hadoop/mapreduce/reducer.py
```

```
es Terminal 17:15 3 جنوری
tazeen@tazeen-HP-15-Notebook-PC: ~
#!/usr/bin/python
import sys

count_of_delayed_flights = 0
total_delayed_time = 0
prev_year = None

try:
    for line in sys.stdin:
        line = line.strip()
        # parse the input we got from mapper.py
        year, fl_delay = line.split('\t', 1)
        #print year, fl_delay, type(fl_delay)
        # convert delay (currently a string) to float
        try:
            fl_delay = float(fl_delay)
        except ValueError:
            # delay was not a number, so silently
            # ignore/discard this line
            continue
        # this IF-switch only works because Hadoop sorts Map output
        # by key (here: year) before it is passed to the reducer
        if prev_year == year:
            count_of_delayed_flights += 1
            total_delayed_time += fl_delay
        else:
            if prev_year:
                # write result to STDOUT
                print 'Average Delayed time in year %s is %s minutes' % (prev_year, total_delayed_time/count_of_delayed_flights)
            prev_year = year
            total_delayed_time = fl_delay
            count_of_delayed_flights = 1
        #last year output
        if prev_year == year:
            print 'Average Delayed time in year %s is %s' % (year, total_delayed_time/count_of_delayed_flights)
except Exception as e:
    print 'Error occured at reducer', e
    pass
```

Now putting these batch data files (batch-data.txt and batch-data1.txt) in HDFS. A new folder is created as /user/input which will be for hadoop input data and output will be saved in /user/output.

hadoop RUNNING Applications

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
22	0	1	21	2	6 GB	8 GB	0 B	2	8	0	1	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation
Capacity Scheduler	(MEMORY)	<memory:1024, vCores:1>	<memory:8192, vCores:8>

Showing 1 to 1 of 1 entries

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Progress	Tracking UI
application_1609788795671_0023	root	streamjob3426571567828195357.jar	MAPREDUCE	default	Tue Jan 5 02:57:21 +0500 2021	N/A	RUNNING	UNDEFINED		ApplicationMaster

Showing 1 to 1 of 1 entries

hadoop Application application_1609788795671_0023

Kill Application

Application Overview

User: root
 Name: streamjob3426571567828195357.jar
 Application Type: MAPREDUCE
 Application Tags:
 YarnApplicationState: FINISHED
 FinalStatus Reported by AM: SUCCEEDED
 Started: Mon Jan 04 16:57:21 -0500 2021
 Elapsed: 2mins, 48sec
 Tracking URL: History
 Diagnostics:

Application Metrics

Total Resource Preempted: <memory:0, vCores:0>
 Total Number of Non-AM Containers Preempted: 0
 Total Number of AM Containers Preempted: 0
 Resource Preempted from Current Attempt: <memory:0, vCores:0>
 Number of Non-AM Containers Preempted from Current Attempt: 0
 Aggregate Resource Allocation: 1010793 MB-seconds, 330 vcore-seconds

Showing 1 to 1 of 1 entries

Attempt ID	Started	Node	Logs
appattempt_1609788795671_0023_000001	N/A	N/A	N/A

Showing 1 to 1 of 1 entries

III- Streaming Layer

Now starting the spark job with redis as streaming layer. The data will be read from mongodb and streams will be created in redis which will be used in spark for processing.

The dockerfile for spark is created as following:

```
tazeen@tazeen-HP-15-Notebook-PC:~$ sudo nano spark.Dockerfile
tazeen@tazeen-HP-15-Notebook-PC:~$
```



```
GNU nano 4.8 docker-compose.yml
version: "3.2"
services:
  hdfs:
    image: sequenceiq/hadoop-docker:2.7.0
    depends_on:
      - py-mongo
    networks:
      - default_bridge
    volumes:
      - ./mongo-app:/var/www/html
  py-mongo:
    # build the image from Dockerfile
    build:
      context: .
    volumes:
      - ./mongo-data:/data/db
      - ./mongo-app:/var/www/html
    ports:
      - "27017:27017"
    environment:
      - MONGO_INITDB_ROOT_USERNAME=root
      - MONGO_INITDB_ROOT_PASSWORD=1234
    networks:
      - default_bridge
  image: py_mongo
  spark-master:
    container_name: spark-master
    image: spark
    build:
      context: .
      dockerfile: spark.Dockerfile
    command: bin/spark-class org.apache.spark.deploy.master.Master -h spark-master
    hostname: spark-master
    environment:
```

```
environment:
  MASTER: spark://spark-master:7077
  SPARK_CONF_DIR: /conf
  SPARK_PUBLIC_DNS: localhost
expose:
  - 7001
  - 7002
  - 7003
  - 7004
  - 7005
  - 7006
  - 7077
  - 6066
ports:
  - 4040:4040
  - 6066:6066
  - 7077:7077
  - 8080:8080
volumes:
  - ./services/spark/dependencies:/master/lib
  - ./services/spark/py-scripts:/master/scripts
networks:
  - default_bridge
  spark-worker:
    image: spark
    container_name: spark-worker
    command: bin/spark-class org.apache.spark.deploy.worker.Worker spark://spark-master:7077
    hostname: spark-worker
    environment:
      SPARK_CONF_DIR: /conf
      SPARK_WORKER_CORES: 2
      SPARK_WORKER_MEMORY: 1g
      SPARK_WORKER_PORT: 8881
      SPARK_WORKER_WEBUI_PORT: 8081
```

```
GNU nano 4.8 docker-compose.yml
environment:
  SPARK_CONF_DIR: /conf
  SPARK_WORKER_CORES: 2
  SPARK_WORKER_MEMORY: 1g
  SPARK_WORKER_PORT: 8881
  SPARK_WORKER_WEBUI_PORT: 8081
  SPARK_PUBLIC_DNS: localhost
expose:
  - 7012
  - 7013
  - 7014
  - 7015
  - 7016
  - 8881
ports:
  - 8081:8081
links:
  - spark-master
depends_on:
  - spark-master
networks:
  - default_bridge
redis:
  image: redis:latest
  networks:
    - default_bridge
  volumes:
    - ./mongo-app:/var/www/html
networks:
  default_bridge:
    name: tazeen_bridge
volumes:
  mongo-app:
```

Spark is configured correctly and opening spark master terminal:

```
tazeen@tazeen-HP-15-Notebook-PC:~$ docker exec -it 2911d0b20f56 /bin/bash
root@spark-master:/usr/spark-2.4.7# spark-shell
21/01/10 21:31:59 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Spark context Web UI available at http://localhost:4040
Spark context available as 'sc' (master = spark://spark-master:7077, app id = app-20210110213219-0000).
Spark session available as 'spark'.
Welcome to

  ____      _
 / ___|    / \
| |  | |  / _ \
| |  | | / ___ \
| |  | |/ ___ \
| |  | || |_) |
| |  | || |_) |
| |  | || |_) |
|_|  |_| \____/

version 2.4.7

Using Scala version 2.11.12 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_131)
Type in expressions to have them evaluated.
Type :help for more information.

scala>

root@spark-master:/usr/spark-2.4.7# pyspark
Python 3.6.9 (default, Jul 10 2019, 03:14:56)
[GCC 4.9.2] on linux
Type "help", "copyright", "credits" or "license" for more information.
21/01/10 21:34:38 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Welcome to

  ____      _
 / ___|    / \
| |  | |  / _ \
| |  | | / ___ \
| |  | |/ ___ \
| |  | || |_) |
| |  | || |_) |
| |  | || |_) |
|_|  |_| \____/

version 2.4.7

Using Python version 3.6.9 (default, Jul 10 2019 03:14:56)
SparkSession available as 'spark'.
>>>
```

Navigate to localhost:8080 to see spark master is running.

Firefox Web Browser 04:22 10 جنوری

Spark Master at spark://spark-master:7077 - Mozilla Firefox

localhost:8080

Spark 2.4.7 Spark Master at spark://spark-master:7077

URL: spark://spark-master:7077
 Alive Workers: 1
 Cores in use: 2 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	State	Cores	Memory
worker-20210109205136-172.18.0.5-8881	172.18.0.5:8881	ALIVE	2 (0 Used)	1024.0 MB (0.0 B Used)

Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	----------------	------	-------	----------

Completed Applications (0)

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	----------------	------	-------	----------

Firefox Web Browser 04:22 10 جنوری

Spark Worker at 172.18.0.5:8881 - Mozilla Firefox

localhost:8081

Spark 2.4.7 Spark Worker at 172.18.0.5:8881

ID: worker-20210109205136-172.18.0.5-8881
 Master URL: spark://spark-master:7077
 Cores: 2 (0 Used)
 Memory: 1024.0 MB (0.0 B Used)
[Back to Master](#)

Running Executors (0)

ExecutorID	Cores	State	Memory	Job Details	Logs
------------	-------	-------	--------	-------------	------

To create streaming data, a text file is generated with redis streams from mongodb database. The data for November and December 2018. The mongodb script is saved as mongo-app/create-stream-data.py to generate data is below. I will only find the data for the flights which are not canceled, therefore, column 'CANCELLED' should be 0.

```
GNU nano 4.8 mongo-app/create-stream-data.py Modified
from pymongo import MongoClient, errors

# global variables for MongoDB host (default port is 27017)
DOMAIN = '172.18.0.3'
PORT = 27017

# use a try-except indentation to catch MongoClient() errors
try:
    # try to instantiate a client instance
    client = MongoClient(
        host = [ str(DOMAIN) + ":" + str(PORT) ],
        serverSelectionTimeoutMS = 3000, # 3 second timeout
        username = "root",
        password = "1234",
    )
    db = client.airline;
    batchData = db.flights.find({'$and':[{'FL_DATE':{'$gte': '2018-12-01'}},{'CANCELLED':{'$eq':'0.0'}},
    {'FL_DATE':{'$lte':'2018-12-31'}} ]}),
    {'ARR_DELAY':1, 'FL_DATE':1, '_id':0 });
    batchStr = str();
    for document in batchData:
        print (document);
        batchStr = batchStr + "XADD streams * FL_DATE " + document["FL_DATE"] + " ARR_DELAY " + document["ARR_DELAY"] + "\n";
        #batchJson.append(document);
    with open('stream-data1.txt', 'w') as outfile:
        outfile.write(batchStr)

except errors.ServerSelectionTimeoutError as err:
    # catch pymongo.errors.ServerSelectionTimeoutError
    print ("pymongo ERROR:", err)
```

Get Help Write Out Where Is Cut Text Justify Cur Pos Undo Mark Text To Bracket
Exit Read File Replace Paste Text To Spell Go To Line Redo Copy Text Where Was

```

GNU nano 4.8 mongo-app/create-stream-data.py Modified
from pymongo import MongoClient, errors

# global variables for MongoDB host (default port is 27017)
DOMAIN = '172.18.0.3'
PORT = 27017

# use a try-except indentation to catch MongoClient() errors
try:
    # try to instantiate a client instance
    client = MongoClient(
        host = [ str(DOMAIN) + ":" + str(PORT) ],
        serverSelectionTimeoutMS = 3000, # 3 second timeout
        username = "root",
        password = "1234",
    )
    db = client.airline;
    batchData = db.flights.find({'$and':[{"FL_DATE":{"$gte": "2018-11-01"}}, {"CANCELLED":{"$eq":"0.0"}}, {"FL_DATE":{"$lte":"2018-11-30"}} ]}, {"ARR_DELAY":1, "FL_DATE":1, "_id":0 });
    batchStr = str();
    for document in batchData:
        print (document);
        batchStr = batchStr + "XADD streams * FL_DATE " + document["FL_DATE"] + " ARR_DELAY " + document["ARR_DELAY"] + "\n";
    batchJson.append(document);
    with open('stream-data2.txt', 'w') as outfile:
        outfile.write(batchStr)
except errors.ServerSelectionTimeoutError as err:
    # catch pymongo.errors.ServerSelectionTimeoutError
    print ("pymongo ERROR:", err)

```

Now executing the script in mongo db docker container. It will create two stream data files as stream-data1.txt and stream-data2.txt. These files will be used by redis docker container to generate streams.

```

root@f818c348f361: /var/www/html
tazeen@tazeen-HP-15-Notebook-PC:~$ sudo nano mongo-app/create-stream-data.py
[sudo] password for tazeen:
tazeen@tazeen-HP-15-Notebook-PC:~$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED            STATUS              PORTS
914ac8b3ff5d       redis:latest        "docker-entrypoint.s..." 23 hours ago      Up 2 hours         6379/tcp
e7f3401af964       spark               "bin/spark-class org..." 27 hours ago      Up 2 hours         7012-7016/tcp, 8881/tcp,
0.0.0.0:8081->8081/tcp spark-worker
2911d0b20f56       spark               "bin/spark-class org..." 27 hours ago      Up 2 hours         0.0.0.0:4040->4040/tcp,
0.0.0.0:6066->6066/tcp, 0.0.0.0:7077->7077/tcp, 0.0.0.0:8080->8080/tcp, 7001-7006/tcp spark-master
340c48bb8220       sequenceiq/hadoop-docker:2.7.0 "/etc/bootstrap.sh -d"    2 days ago        Up 2 hours         2122/tcp, 8030-8033/tcp,
8040/tcp, 8042/tcp, 8088/tcp, 19888/tcp, 49707/tcp, 50010/tcp, 50020/tcp, 50070/tcp, 50075/tcp, 50090/tcp tazeen_hdfs_1
f818c348f361       py_mongo            "docker-entrypoint.s..." 2 days ago        Up 2 hours         0.0.0.0:27017->27017/tcp tazeen_py-mongo_1

tazeen@tazeen-HP-15-Notebook-PC:~$ docker exec -it f818c348f361 /bin/bash
root@f818c348f361:/# cd /var/www/html
root@f818c348f361:/var/www/html# python3 create-stream-data.py

```

Let's execute the script to execute streaming data from redis to spark master. The dependencies required for spark is mounted on /master/lib folder in spark master. These dependencies consists of jar files required to read redis streams in pyspark.

Copying jar files to ./services/spark/dependencies folder which is mounted as /master/lib in spark master.

```
Terminal 04:38 10 جنوری
tazeen@tazeen-HP-15-Notebook-PC: ~/services/spark/dependencies

tazeen@tazeen-HP-15-Notebook-PC:~$ cd services/spark/dependencies/
tazeen@tazeen-HP-15-Notebook-PC:~/services/spark/dependencies$ cp /home/tazeen/spark-
spark-base.Dockerfile spark-redis-2.4.0.jar spark-stream.py
spark-master.Dockerfile spark-redis-dependencies/ spark-worker.Dockerfile
tazeen@tazeen-HP-15-Notebook-PC:~/services/spark/dependencies$ cp /home/tazeen/spark-redis-dependencies/spark-redis-2.4.1.jar
cp: missing destination file operand after '/home/tazeen/spark-redis-dependencies/spark-redis-2.4.1.jar'
Try 'cp --help' for more information.
tazeen@tazeen-HP-15-Notebook-PC:~/services/spark/dependencies$ cp /home/tazeen/spark-redis-dependencies/spark-redis-2.4.1.jar .
cp: cannot create regular file './spark-redis-2.4.1.jar': Permission denied
tazeen@tazeen-HP-15-Notebook-PC:~/services/spark/dependencies$ sudo cp /home/tazeen/spark-redis-dependencies/spark-redis-2.4.1.jar .
tazeen@tazeen-HP-15-Notebook-PC:~/services/spark/dependencies$ sudo cp /home/tazeen/spark-redis-dependencies/jedis-3.2.0.jar .
tazeen@tazeen-HP-15-Notebook-PC:~/services/spark/dependencies$ sudo cp /home/tazeen/spark-redis-dependencies/commons-pool2-2.0.jar .
tazeen@tazeen-HP-15-Notebook-PC:~/services/spark/dependencies$ ls
commons-pool2-2.0.jar jedis-3.2.0.jar spark-redis-2.4.1.jar
tazeen@tazeen-HP-15-Notebook-PC:~/services/spark/dependencies$
```

The python script to execute streaming reads streaming data from redis host, it's host ip can be seen from bridge network. For redis-docker IP address is : 172.18.0.4

```
es Terminal 04:46 10 جنوری
tazeen@tazeen-HP-15-Notebook-PC: ~
tazeen@tazeen-HP-15-Notebook-PC:~$ docker network inspect tazeen_bridge
[
  {
    "Name": "tazeen_bridge",
    "Id": "dedd0bcc7794a920abff474b5a60420925dc41a4b1009e8abf56c9980ecbf727",
    "Created": "2021-01-04T16:03:48.017315154+05:00",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": null,
      "Config": [
        {
          "Subnet": "172.18.0.0/16",
          "Gateway": "172.18.0.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": true,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {
      "2911d0b20f56af3aec48fd0c310390d97332a13348d260f45e0833da0ce473ec": {
        "Name": "spark-master",
        "EndpointID": "679a0bdccc207e58a33d3b2abf0d2485cbea955a33293edf27061f6f1699f03f",
        "MacAddress": "02:42:ac:12:00:02",
        "IPv4Address": "172.18.0.2/16",
        "IPv6Address": ""
      },
      "340c48bb8220ec3fddf0a0c6449105aafcc92f9baf9957ca6792752a60bb979e": {
        "Name": "tazeen_hdfs_1",
        "EndpointID": "60c7c504525b53a272541979538bcb17cb95bde0eb0ef382789f9153bd007e5a",
        "MacAddress": "02:42:ac:12:00:06",
        "IPv4Address": "172.18.0.6/16"
      }
    }
  }
]
```

```
Terminal 04:46 10 جنوري
tazeen@tazeen-HP-15-Notebook-PC: ~

"340c48bb8220ec3fddf0a0c6449105aafcc92f9baf9957ca6792752a60bb979e": {
  "Name": "tazeen_hdfs_1",
  "EndpointID": "06c7c504525b53a272541979538bcb17cb95bde0eb0ef382789f9153bd007e5a",
  "MacAddress": "02:42:ac:12:00:06",
  "IPv4Address": "172.18.0.6/16",
  "IPv6Address": ""
},
"914ac8b3ff5d583e35eca5ef9354b37dfb788fef739b2fdbbbd6d1a5c73a6967": {
  "Name": "tazeen_redis_1",
  "EndpointID": "b5c97a7b4d48b4db722aa9e2c0e645ec4c3036baa8f231cac8c93058042af433",
  "MacAddress": "02:42:ac:12:00:04",
  "IPv4Address": "172.18.0.4/16",
  "IPv6Address": ""
},
"e7f3401af9649c91f1fd07e134d477b7a8c66b475560827b1bbcef903cfe3da": {
  "Name": "spark-worker",
  "EndpointID": "80b071c8c3a69b0610a9932ce6cf5e0acb4c77aeb5a18d28f0700f8eb4c63169",
  "MacAddress": "02:42:ac:12:00:05",
  "IPv4Address": "172.18.0.5/16",
  "IPv6Address": ""
},
"f818c348f3611cf8859c5bde16421b748f06ab9814e1d4590836d0d604a805c1": {
  "Name": "tazeen_py-mongo_1",
  "EndpointID": "15ae562a43de1c70253a0c31f5a480d83b6497abf68d1442617cb96abcae5ec7",
  "MacAddress": "02:42:ac:12:00:03",
  "IPv4Address": "172.18.0.3/16",
  "IPv6Address": ""
}
}
"Options": {},
"Labels": {
  "com.docker.compose.network": "tazeen_bridge",
  "com.docker.compose.project": "tazeen",
  "com.docker.compose.version": "1.27.4"
}
}
tazeen@tazeen-HP-15-Notebook-PC:~$
```

The python script is summing the delay time for positive flight delays, the number of rows provided in each stream and the month of flight date. The script is following:

```
Terminal 17:52 10 جنوري
tazeen@tazeen-HP-15-Notebook-PC: ~

GNU nano 4.8 services/spark/py-scripts/spark-stream.py Modified
from pyspark.sql import SparkSession, SQLContext, DataFrame
from pyspark.sql.types import *
from pyspark.sql.functions import col, sum as _sum, first, concat_ws, split, count as _count

spark = SparkSession \
    .builder \
    .master("local[*]") \
    .config("spark.redis.host", "172.18.0.4") \
    .config("spark.redis.port", "6379") \
    .getOrCreate()

sensors = spark \
    .readStream \
    .format("redis") \
    .option("stream.keys", "streams") \
    .schema(StructType([ \
        StructField("FL_DATE", StringType()), \
        StructField("ARR_DELAY", FloatType()) \
    ])) \
    .load()

def process_row(row, id):
    # Process row
    df_filtered = row.filter(col("ARR_DELAY") >= 0)
    df_stats = df_filtered.select(_sum(col("ARR_DELAY")).alias('sum'), _count(col("ARR_DELAY")).alias('count'), \
        concat_ws('-', split(first(col('FL_DATE')), '[-]')[1], \
        split(first(col('FL_DATE')), '[-]')[0]).alias('mon-year')).collect()
    print(df_stats)
    month = df_stats[0]['mon-year']
    #store results in redis table
    sc = spark.sparkContext
    myJson = sc.parallelize([{"sum_of_delay":df_stats[0]['sum'], "count": df_stats[0]['count'], "month": month}])
    myDf = spark.read.json(myJson).write.format("org.apache.spark.sql.redis") \
        .option("table", "avgDelay") \
        .mode("append") \

G Get Help ^O Write Out ^M Where Is ^K Cut Text ^J Justify ^C Cur Pos ^U Undo ^A Mark Text ^_ To Bracket
X Exit ^R Read File ^\ Replace ^U Paste Text ^T To Spell ^_ Go To Line ^E Redo ^G Copy Text ^Q Where Was
```

```
GNU nano 4.8 services/spark/py-scripts/spark-stream.py Modified
.option("stream.keys", "streams") \
.schema(StructType([ \
    StructField("FL_DATE", StringType()), \
    StructField("ARR_DELAY", FloatType()) \
])) \
.load()
def process_row(row, id):
    # Process row
    df_filtered = row.filter(col("ARR_DELAY") >=0)
    df_stats = df_filtered.select(_sum(col("ARR_DELAY")).alias('sum'), _count(col("ARR_DELAY")).alias('count'), \
    concat_ws('-', split(first(col('FL_DATE')), '[-]')[1], \
    split(first(col('FL_DATE')), '[-]')[0]).alias('mon-year')).collect()
    print(df_stats)
    month = df_stats[0]['mon-year']
    #store results in redis table
    sc = spark.sparkContext
    myJson = sc.parallelize([{"sum_of_delay":df_stats[0]['sum'], "count": df_stats[0]['count'], "month": month}])
    myDf = spark.read.json(myJson).write.format("org.apache.spark.sql.redis") \
    .option("table", "avgDelay") \
    .mode("append") \
    .save()
    pass
query = sensors \
    .writeStream \
    .outputMode("update") \
    .foreachBatch(process_row) \
    .start()

try:
    query.awaitTermination()
except Exception as error:
    print('Streaming query exception', error)
[]

^G Get Help ^O Write Out ^N Where Is ^C Cut Text ^J Justify ^C Cur Pos ^M-U Undo ^M-A Mark Text ^M-] To Bracket
^X Exit ^R Read File ^A\ Replace ^U Paste Text ^T To Spell ^_ Go To Line ^M-E Redo ^M-C Copy Text ^M-_ Where Was
```

Now copying the script in ./services/spark/py-scripts/ folders which is mounted on /master/scripts folder in spark master.

```
18:21 8 جنوری
tazeen@tazeen-HP-15-Notebook-PC: ~/services/spark/py-scripts

tazeen@tazeen-HP-15-Notebook-PC:~$ cd services/spark/
tazeen@tazeen-HP-15-Notebook-PC:~/services/spark$ sudo mkdir py-scripts
tazeen@tazeen-HP-15-Notebook-PC:~/services/spark$ cd py-scripts/
tazeen@tazeen-HP-15-Notebook-PC:~/services/spark/py-scripts$ sudo nano spark-stream.py
tazeen@tazeen-HP-15-Notebook-PC:~/services/spark/py-scripts$
```

Connecting to the spark master terminal and executing the script for streaming.

```

s Terminal 04:27 10 جنوری
tazeen@tazeen-HP-15-Notebook-PC: ~
CONTAINER ID    IMAGE    COMMAND    CREATED    STATUS    PORTS
914ac8b3ff5d    redis:latest    "docker-entrypoint.s..."    24 hours ago    Up 3 hours    6379/tcp
e7f3401af964    spark    "bin/spark-class org..."    28 hours ago    Up 3 hours    7012-7016/tcp, 8881/tcp,
0.0.0.0:8081->8081/tcp
2911d0b20f56    spark    "bin/spark-class org..."    28 hours ago    Up 3 hours    0.0.0.0:4040->4040/tcp,
0.0.0.0:6066->6066/tcp, 0.0.0.0:7077->7077/tcp, 0.0.0.0:8080->8080/tcp, 7001-7006/tcp
340c48bb8220    sequenceiq/hadoop-docker:2.7.0    "/etc/bootstrap.sh -d"    2 days ago    Up 3 hours    2122/tcp, 8030-8033/tcp,
8040/tcp, 8042/tcp, 8088/tcp, 19888/tcp, 49707/tcp, 50010/tcp, 50020/tcp, 50070/tcp, 50075/tcp, 50090/tcp
f818c348f361    py_mongo    "docker-entrypoint.s..."    2 days ago    Up 3 hours    0.0.0.0:27017->27017/tcp
tazeen_py-mongo_1

tazeen@tazeen-HP-15-Notebook-PC:~$ docker exec -it 2911d0b20f56 /bin/bash
root@spark-master:/usr/spark-2.4.7# ls /master/lib
commons-pool2-2.0.jar jedis-3.2.0.jar spark-redis-2.4.0.jar spark-redis-2.4.1.jar
root@spark-master:/usr/spark-2.4.7# ls /master/scripts/
spark-stream.py spark-stream.py.save
root@spark-master:/usr/spark-2.4.7# ./bin/spark-submit --jars /master/lib/spark-redis-2.4.1.jar,/master/lib/commons-pool2-2.0.jar,/master/lib/
jedis-3.2.0.jar --class GenericObjectPoolConfig /master/scripts/spark-stream.py
21/01/09 23:27:21 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
21/01/09 23:27:24 INFO SparkContext: Running Spark version 2.4.7
21/01/09 23:27:24 INFO SparkContext: Submitted application: GenericObjectPoolConfig
21/01/09 23:27:24 INFO SecurityManager: Changing view acls to: root
21/01/09 23:27:24 INFO SecurityManager: Changing modify acls to: root
21/01/09 23:27:24 INFO SecurityManager: Changing view acls groups to:
21/01/09 23:27:24 INFO SecurityManager: Changing modify acls groups to:
21/01/09 23:27:24 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view permissions: Set(root); g
roups with view permissions: Set(); users with modify permissions: Set(root); groups with modify permissions: Set()
21/01/09 23:27:25 INFO Utils: Successfully started service 'sparkDriver' on port 41215.
21/01/09 23:27:25 INFO SparkEnv: Registering MapOutputTracker
21/01/09 23:27:25 INFO SparkEnv: Registering BlockManagerMaster
21/01/09 23:27:25 INFO BlockManagerMasterEndpoint: Using org.apache.spark.storage.DefaultTopologyMapper for getting topology information
21/01/09 23:27:25 INFO BlockManagerMasterEndpoint: BlockManagerMasterEndpoint up
21/01/09 23:27:26 INFO DiskBlockManager: Created local directory at /tmp/blockmgr-beb1607d-89f3-4c82-aeae-d106eef04635
21/01/09 23:27:26 INFO MemoryStore: MemoryStore started with capacity 366.3 MB
21/01/09 23:27:26 INFO SparkEnv: Registering OutputCommitCoordinator

```

Now connecting to the redis docker container to send streams to spark. It consists of stream-data1.txt and stream-data2.txt as the redis streams for November 2018 and December 2018. The data is mounted as /var/www/html in redis docker.

```
Terminal 17:47 10 جنوری
tazeen@tazeen-HP-15-Notebook-PC: ~
tazeen@tazeen-HP-15-Notebook-PC:~$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS
914ac8b3ff5d        redis:latest        "docker-entrypoint.s..." 37 hours ago        Up 5 hours          6379/tcp
e7f3401af964        spark               "bin/spark-class org..." 41 hours ago        Up 5 hours          7012-7016/tcp, 8881/tcp,
0.0.0.0:8081->8081/tcp
2911d0b20f56        spark               "bin/spark-class org..." 41 hours ago        Up 5 hours          0.0.0.0:4040->4040/tcp,
0.0.0.0:6066->6066/tcp, 0.0.0.0:7077->7077/tcp, 0.0.0.0:8080->8080/tcp, 7001-7006/tcp
340c48bb8220        sequenceiq/hadoop-docker:2.7.0 "/etc/bootstrap.sh -d" 2 days ago          Up 5 hours          2122/tcp, 8030-8033/tcp,
8040/tcp, 8042/tcp, 8088/tcp, 19888/tcp, 49707/tcp, 50010/tcp, 50020/tcp, 50070/tcp, 50075/tcp, 50090/tcp
f818c348f361        py_mongo            "docker-entrypoint.s..." 2 days ago          Up 5 hours          0.0.0.0:27017->27017/tcp
tazeen@tazeen-HP-15-Notebook-PC:~$ docker exec -it 914ac8b3ff5d /bin/bash
root@914ac8b3ff5d:/data# ls /var/www/html/
batch-data.py  batch-data.txt  batch-data1.txt  create-stream-data.py  output.txt  stream-data.txt  stream-data1.txt  stream-data2.txt
root@914ac8b3ff5d:/data# redis-cli < /var/www/html/stream-data1.txt
```

```
Terminal 17:50 10 جنوری
tazeen@tazeen-HP-15-Notebook-PC: ~
"1610283002444-1"
"1610283002444-2"
"1610283002444-3"
"1610283002444-4"
"1610283002444-5"
"1610283002444-6"
"1610283002444-7"
"1610283002445-0"
"1610283002445-1"
"1610283002445-2"
"1610283002445-3"
"1610283002445-4"
"1610283002445-5"
"1610283002445-6"
"1610283002445-7"
"1610283002445-8"
"1610283002446-0"
"1610283002446-1"
"1610283002446-2"
"1610283002446-3"
"1610283002446-4"
"1610283002446-5"
"1610283002446-6"
"1610283002446-7"
"1610283002446-8"
"1610283002447-0"
"1610283002447-1"
"1610283002447-2"
"1610283002447-3"
"1610283002447-4"
"1610283002447-5"
"1610283002447-6"
"1610283002447-7"
"1610283002447-8"
"1610283002448-0"
"1610283002448-1"
"1610283002448-2"
root@914ac8b3ff5d:/data# redis-cli < /var/www/html/stream-data2.txt
```

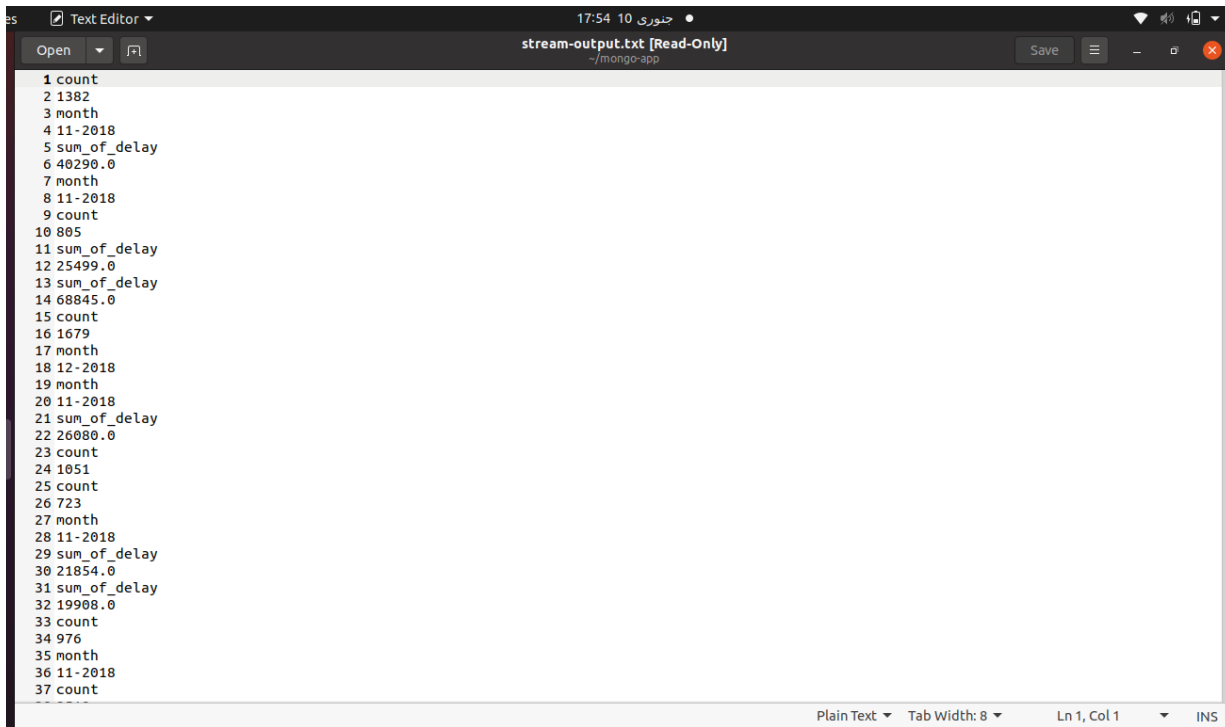
Using the cli to check the potput saved in redis table. Keys * lists all the keys created in redis. All keys starting with aygAverage are created by streaming. So saving all these results in stream-output.txt file. The output file will be saved in mongo-app folder in disk as /var/www/html is mounted for it.

```
Terminal 17:53 10 جنوری
tazeen@tazeen-HP-15-Notebook-PC: ~

root@914ac8b3ff5d:/data# redis-cli
127.0.0.1:6379> keys *
 1) "avgDelay:32a14818738d4c30abdeb29ecf0126e9"
 2) "avgDelay:552f8584eb6d4e97a7e5b0723fe11b97"
 3) "avgDelay:13ab1e135bb849ef990250ad22051afd"
 4) "avgDelay:cf396dcbb4a046f2aaaaadea80ebcc223"
 5) "avgDelay:071f858ef03c4bea9719e9e27014bb87"
 6) "avgDelay:b7f7e927a27d42ebbe39338e535bef9a"
 7) "avgDelay:d05f504f2cf941debe2643cecdad008b"
 8) "avgDelay:976f3e87cf26483baa9f644800d22a3b"
 9) "avgDelay:dcaf63cf593fa53897369d76183891f"
10) "avgDelay:0a34323b6fc44cec8502c6a652daf5fe"
11) "avgDelay:50507c56e05c4c2aa671af610bbeede0"
12) "avgDelay:326279eda5dd43cda83d9861cb81dd80"
13) "avgDelay:202c091906dd479d839724c30f2722cf"
14) "avgDelay:b9c4378e36064ce698e3e9b93a6b6386"
15) "avgDelay:60e66ae9681f4f959fb9e9ec4055e59a"
16) "avgDelay:6d7ff531755c4fe687b0a189c633e9f9"
17) "avgDelay:4916537331e14087b4c663e77842fd7f"
18) "avgDelay:669db4dbc16049d69db692edaa94258f"
19) "avgDelay:e2ae2866cd274fc797377d858a666621"
20) "avgDelay:bab7291fa0441e093da598f06a7a163"
21) "avgDelay:2c1462bd1aa3460f970dd4e1d658f5cb"
22) "avgDelay:2ddad59d09814996986a34c366127b46"
23) "avgDelay:048c5164ba804f24bebd6d4e3df58be"
24) "avgDelay:70770a65120e468e8e25cf94b4350b80"
25) "avgDelay:7d3e6f9780564a97be7ff94b01083055"
26) "avgDelay:f259cd4385e642d28ceb4321e366d883"
27) "avgDelay:e5b44f9fa5da4ed1a1a335252b5eefbe"
28) "avgDelay:32a67d301d784b67bd8e567d7dd11c91"
29) "avgDelay:0537aeb651e4e6c80f8e2f628701317"
30) "avgDelay:72aabc758d3545ebae4f32131c500489"
31) "avgDelay:8040a55e584947a9bda85930ec86b5c"
32) "avgDelay:3bf82bacab4a4671af95b733c8dfab1b"
33) "avgDelay:5dd093d1bab649338a21023f114ee1da"
34) "avgDelay:8764cca168d4031b1f0ed87bb58bcd0"
35) "avgDelay:ad805a71634342afa5cb21872aff34b"
36) "avgDelay:cbc4df722ff64965a0ad5baa8f8e094d"
37) "avgDelay:073f5f00045b049c50a055c085b022600"
```

```
Terminal 17:53 10 جنوری
tazeen@tazeen-HP-15-Notebook-PC: ~

279) "avgDelay:13de0aae4d284e6491d366f726363b22"
280) "avgDelay:9d0b6bb30203405a99deda371d933ad0"
281) "avgDelay:8b0c1367ed244038a831bb649de9a53"
282) "avgDelay:6b1b816b922f4feb80bbc36bfa68b981"
283) "avgDelay:de9e546f43854096877b8ed0dea052a5"
284) "avgDelay:0b36f42182f4430bb3aa6bcd3be0a44e"
285) "avgDelay:cd9230ff5c604a2fb27ce30f709ae979"
286) "avgDelay:15a38c2d9e7e4f078b94806e35c55a4f"
287) "avgDelay:1b31e26dfbca4feb8c99a02a98bd315d"
288) "avgDelay:dc757adb6610493ca681049ac176a748"
289) "avgDelay:93cbd52cde794d82a8945d928f324224"
290) "avgDelay:e9769fc33cd742659625d51ee580bd32"
291) "avgDelay:a56a40f7b50c4850aa5ed23442b70350"
292) "avgDelay:99171796bdbb450e8b399b4271752f5f"
293) "avgDelay:51410f3ed8554923bd11d5667fe56c9f"
294) "avgDelay:432b16913b6d45a892059b0fbee5b9f2"
295) "avgDelay:9996809389844695a82b2a4e1f057d44"
296) "avgDelay:36bd058676b24c43a878a8c284956e71"
297) "avgDelay:7b4009f5aaa1408c9cea1a32cac483fe"
298) "avgDelay:61e20d4923cb48ae827405ef4f8f38d5"
299) "avgDelay:ccc8766ba98e4caabc615eaa0e6392e4"
300) "avgDelay:a1a715ead9e4406893e5fb2a6d4d77bb"
301) "avgDelay:9de73f69e8f541b58474e5a37c320383"
302) "avgDelay:34984b6b30664ec0958f30cabb7efeb0"
303) "avgDelay:b0db3dce28304c6f985ef5bbe2690418"
304) "avgDelay:cf0392aa92e64d5099e190074f6368d1"
305) "avgDelay:9f09a046cad4bae9b2b50ae02e47505"
127.0.0.1:6379> hgetall avgDelay:9f09a046cad4bae9b2b50ae02e47505
1) "count"
2) "1799"
3) "month"
4) "12-2018"
5) "sum_of_delay"
6) "90202.0"
127.0.0.1:6379> exit
root@914ac8b3ff5d:/data# redis-cli --raw keys avgDelay* | awk '{printf "hgetall %s\n", $1}' | redis-cli --raw > /var/www/html/stream-output.txt
root@914ac8b3ff5d:/data#
```



The screenshot shows a text editor window titled "stream-output.txt [Read-Only]" with a path of "~/mongo-app". The editor contains a list of 37 numbered lines of data. The data alternates between counts and sums of delays for different months. The status bar at the bottom indicates "Plain Text", "Tab Width: 8", "Ln 1, Col 1", and "INS".

```
1 count
2 1382
3 month
4 11-2018
5 sum_of_delay
6 40290.0
7 month
8 11-2018
9 count
10 805
11 sum_of_delay
12 25499.0
13 sum_of_delay
14 68845.0
15 count
16 1679
17 month
18 12-2018
19 month
20 11-2018
21 sum_of_delay
22 26080.0
23 count
24 1051
25 count
26 723
27 month
28 11-2018
29 sum_of_delay
30 21854.0
31 sum_of_delay
32 19908.0
33 count
34 976
35 month
36 11-2018
37 count
```

IV- Serving Layer

The last step is to create the serving layer which will show the results from batch and stream layers. To create a serving layer, jupyter notebook image is used in container. The updated docker-compose.yml is adds the jupyter image as below:

```
Terminal 20:19 10 جنوری
tazeen@tazeen-HP-15-Notebook-PC: ~
GNU nano 4.8 docker-compose.yml
- 7013
- 7014
- 7015
- 7016
- 8881
ports:
- 8081:8081
links:
- spark-master
depends_on:
- spark-master
networks:
- default_bridge
redis:
image: redis:latest
networks:
- default_bridge
volumes:
- ./mongo-app:/var/www/html
datascience-notebook:
image: jupyter/datascience-notebook
volumes:
- /mongo-app:/home/tazeen/jup-notebook
ports:
- 8888:8888
container_name: datascience-notebook-container
networks:
- default_bridge
networks:
default_bridge:
name: tazeen_bridge
volumes:
mongo-app:
[]
Get Help Write Out Where Is Cut Text Justify Cur Pos M-U Undo M-A Mark Text M-] To Bracket
Exit Read File Replace Paste Text To Spell Go To Line M-E Redo M-C Copy Text M-Q Where Was
```

Now executing docker-compose -build

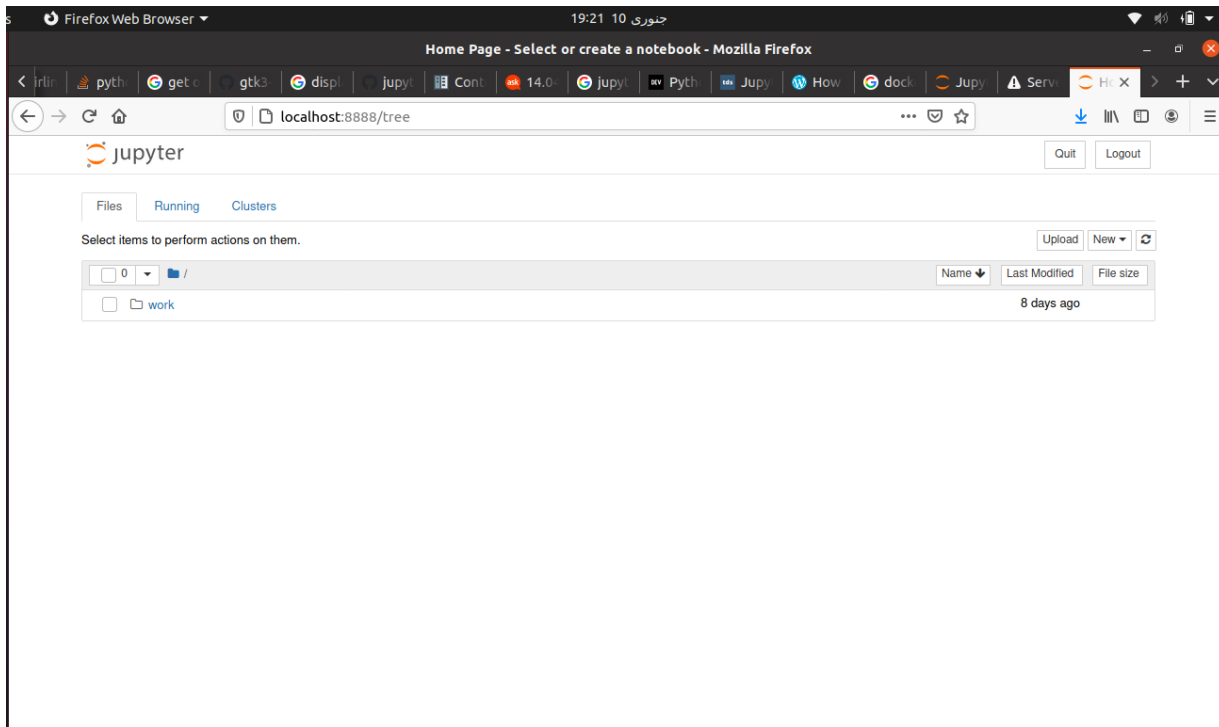
```
Terminal 18:19 10 جنوری
tazeen@tazeen-HP-15-Notebook-PC: ~
--> Using cache
--> e41e1a3c1be4
Step 22/22 : CMD ["bin/spark-class", "org.apache.spark.deploy.master.Master"]
--> Using cache
--> f0732961debf

Successfully built f0732961debf
Successfully tagged spark:latest
Pulling datascience-notebook (jupyter/datascience-notebook)...
latest: Pulling from jupyter/datascience-notebook
da7391352a9b: Already exists
14428a6d4bcd: Already exists
2c2d948710f2: Already exists
ca78f2d1592f: Pull complete
9e7a2fbed339: Pull complete
d8b4c85d06bd: Pull complete
773273f9a979: Pull complete
73968a1cce87: Extracting [=====] 5.368kB/5.368kB
b9bc7c3f9e37: Download complete
5c47e33bf3e4: Downloading [==>] 2.127MB/62.9MB
0f60d494db95: Downloading [==>] 3.233MB/96.36MB
62111c680dc9: Download complete
1165e39cb1e8: Download complete
0e36996f6dd3: Waiting
f9c15206a9d3: Waiting
f79fdd95f721: Waiting
ec79857d12e8: Waiting
556c3a8978ae: Waiting
1696bccf31c7: Waiting
729adcb479ab: Waiting
2c1bda956783: Waiting
821bb95dab07: Waiting
340dabe791df: Waiting
2f0ef3706b64: Waiting
eabb550054b1: Waiting
38cb299b6e9d: Waiting
d85280d20762: Waiting
[]
```

The jupyter docker container is up and running on the url provided below:

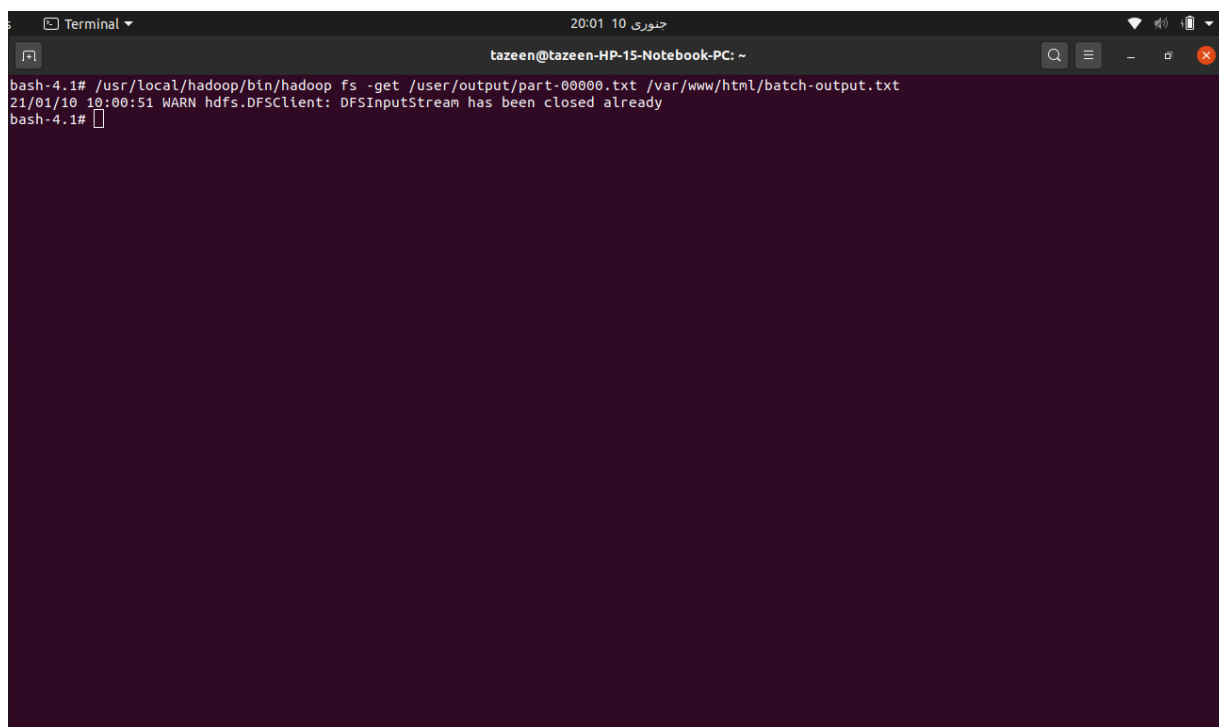
```
me/notebook_cookie_secret
datascience-notebook-container | [I 14:07:36.853 NotebookApp] JupyterLab extension loaded from /opt/conda/lib/python3.8/site-packages/jupyterlab
datascience-notebook-container | [I 14:07:36.853 NotebookApp] JupyterLab application directory is /opt/conda/share/jupyter/lab
datascience-notebook-container | [I 14:07:36.857 NotebookApp] Serving notebooks from local directory: /home/jovyan
datascience-notebook-container | [I 14:07:36.857 NotebookApp] Jupyter Notebook 6.1.6 is running at:
datascience-notebook-container | [I 14:07:36.858 NotebookApp] http://e587c9a78ffc:8888/?token=00dc82fbda7910a4692abcb6f67abbbe150440c49517c5f1
datascience-notebook-container | [I 14:07:36.858 NotebookApp] or http://127.0.0.1:8888/?token=00dc82fbda7910a4692abcb6f67abbbe150440c49517c5f1
datascience-notebook-container | [I 14:07:36.858 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
datascience-notebook-container | [C 14:07:36.876 NotebookApp]
datascience-notebook-container | To access the notebook, open this file in a browser:
datascience-notebook-container | file:///home/jovyan/.local/share/jupyter/runtime/nbserver-7-open.html
datascience-notebook-container | Or copy and paste one of these URLs:
datascience-notebook-container | http://e587c9a78ffc:8888/?token=00dc82fbda7910a4692abcb6f67abbbe150440c49517c5f1
datascience-notebook-container | or http://127.0.0.1:8888/?token=00dc82fbda7910a4692abcb6f67abbbe150440c49517c5f1
spark-master | Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
spark-worker | Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
spark-master | 21/01/10 14:07:39 INFO Master: Started daemon with process name: 1@spark-master
spark-worker | 21/01/10 14:07:39 INFO Worker: Started daemon with process name: 1@spark-worker
spark-master | 21/01/10 14:07:39 INFO SignalUtils: Registered signal handler for TERM
spark-worker | 21/01/10 14:07:39 INFO SignalUtils: Registered signal handler for TERM
spark-master | 21/01/10 14:07:39 INFO SignalUtils: Registered signal handler for HUP
spark-worker | 21/01/10 14:07:39 INFO SignalUtils: Registered signal handler for HUP
spark-master | 21/01/10 14:07:39 INFO SignalUtils: Registered signal handler for INT
spark-worker | 21/01/10 14:07:39 INFO SignalUtils: Registered signal handler for INT
spark-master | 21/01/10 14:07:43 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-jav
spark-worker | 21/01/10 14:07:43 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-jav
a classes where applicable
spark-master | 21/01/10 14:07:44 INFO SecurityManager: Changing view acls to: root
spark-worker | 21/01/10 14:07:44 INFO SecurityManager: Changing view acls to: root
spark-master | 21/01/10 14:07:44 INFO SecurityManager: Changing modify acls to: root
spark-worker | 21/01/10 14:07:44 INFO SecurityManager: Changing modify acls to: root
spark-master | 21/01/10 14:07:44 INFO SecurityManager: Changing view acls groups to:
spark-worker | 21/01/10 14:07:44 INFO SecurityManager: Changing view acls groups to:
spark-master | 21/01/10 14:07:44 INFO SecurityManager: Changing modify acls groups to:
```

Navigating to the jupyter notebook on the url provided.

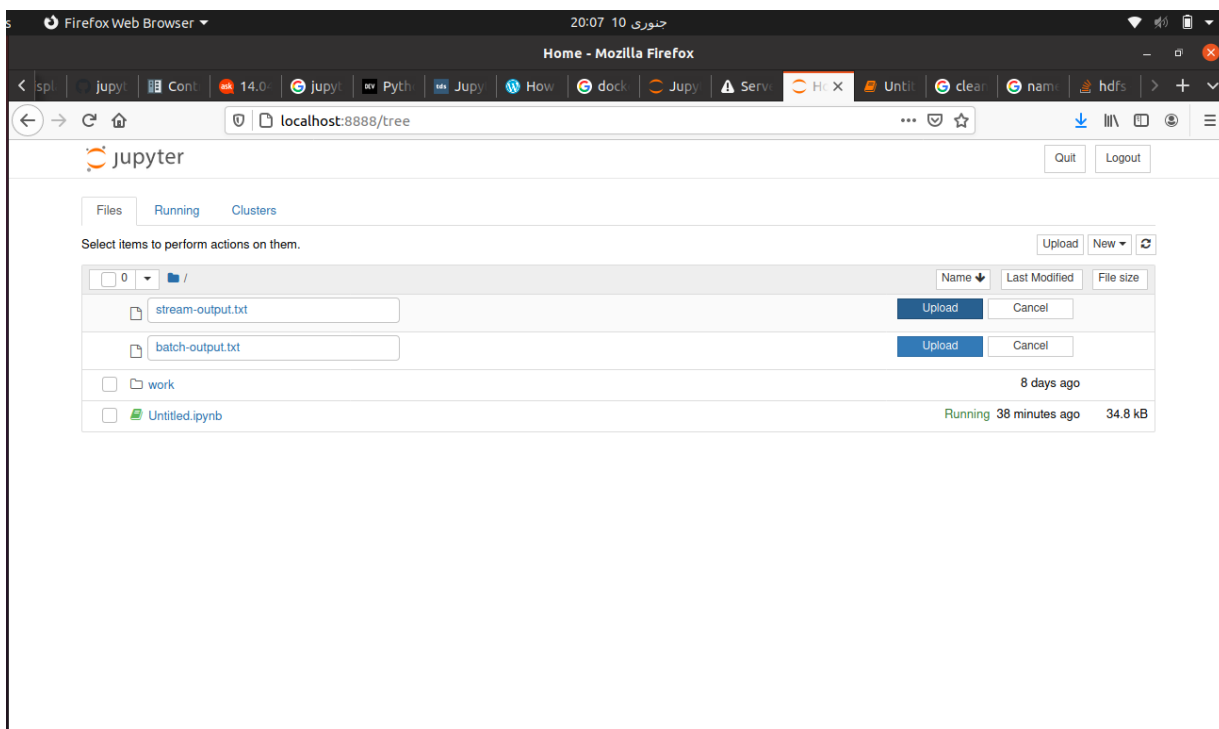
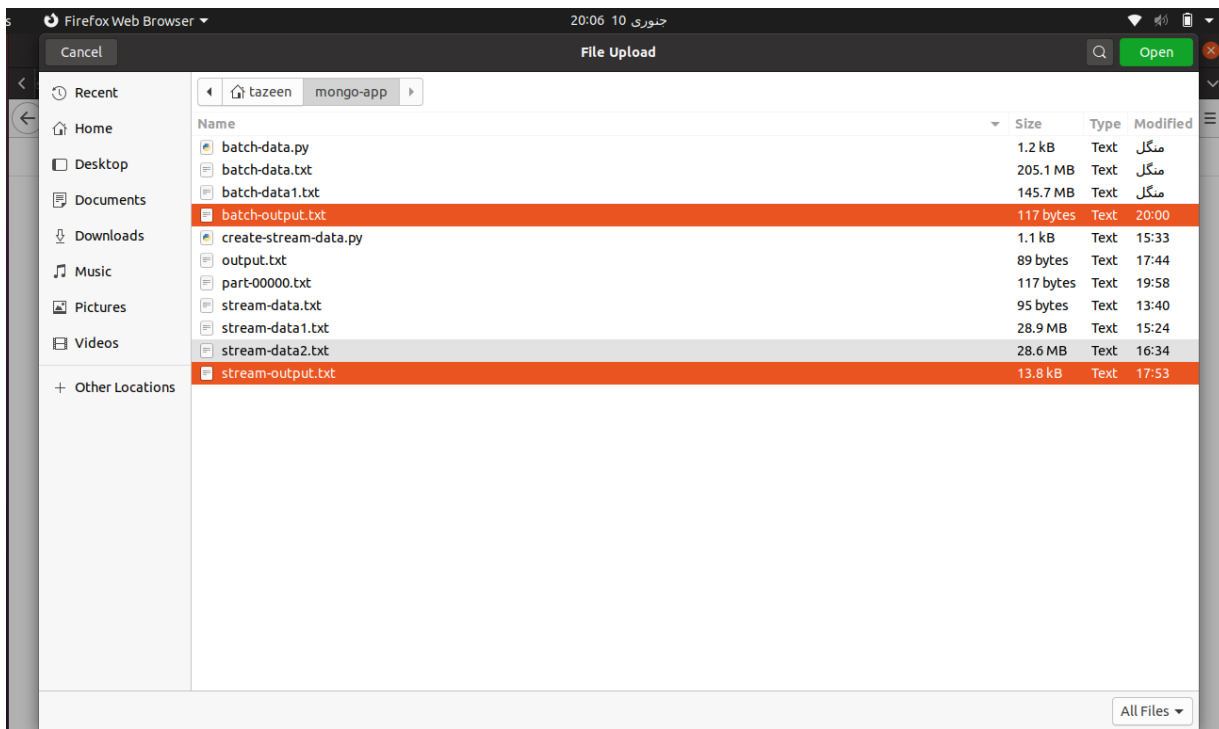


Now uploading output files in jupyter. These files are save in mongo-app folder which is mounted in the jupyter container.

First saving HDFS batch output file in mongo-app folder by:



The stream-output file in already saved in mongo-app folder as previously shown. Now uploading these files in jupyter notebook.



Finally creating new notebook to display results from the batch and stream layers.

```
Results from Hadoop file System after batch data analysis

In [2]: myfile = open("batch-output.txt")
        txt = myfile.read()
        print(txt)
        myfile.close()

Average Delayed time in year 2017 is 37.647597526 minutes
Average Delayed time in year 2018 is 36.3042229856 minutes

Results from Spark after stream data analysis

In [4]: myfile = open("stream-output.txt")
        txt = myfile.read()
        print(txt)
        myfile.close()

month
11-2018
count
1677
sum_of_delay
75396.0
month
```

5. Results

I was able to complete the lambda architecture and list few commands for redis lab. Multiple containers on the same network through bridge network. A custom bridge network-tazeen_bridge is created for communication. The results of lambda architecture are displayed in jupyter lab. It can be seen by the map/reduce job that average delayed flights time is around 36 minutes for year 2018 and around 37 minutes for year 2017. There is not a lot of delayed time in flights in both years. More such analysis can be done for more years. Streaming job has been quite fast and results are saved in redis quicker.

Problems faced:

It was difficult project because of different layers and communication. Each step was challenge and working alone increased the difficulty. Understanding communication with hadoop and mongo database, then creating a map/ reduce job, communication with redis and spark then showing results in jupyter lab. I had difficulty in understanding each step and had to search a lot. Spark SQL and queries are quite difficult and new to me. I had spent a lot of time to understand each process and execute it. There were also problem due system resources, a lot of time was required for huge data. Script execution would stuck and need to start again for large data. Overall it was a lot of learning but very difficult to implement. It took a lot of efforts and time to complete it.