**Kafka Project**

Gachon univ.

Ellie, Edgar, Hayeon, Wonjun

**SET\_A**

**<SetA-1>**

**Description:** Generate random data every second, plot that data and show it. Kafka producer send the data every second, then the Kafka consumer will draw the graph about that data.

**Technique:**

* Apache kafka 1.3.5
* Python 3.7
* Kafka-python 1.4.4

**Solving steps:**

1. Run the zookeeper server and Kafka server.
2. Create Kafka producer and Kafka consumer.
3. Kafka producer sends random data to Kafka consumer every second.
4. Kafka consumer receives that data and plots it using matplotlib library.

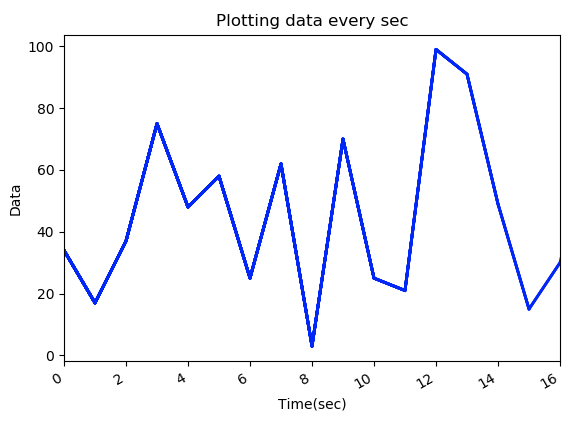
**Code**

<kafka\_consumer.py>

from kafka import KafkaConsumer  
from json import loads  
import matplotlib.pyplot as plt  
% matplotlib  
notebook  
  
# Consumer connection  
consumer = KafkaConsumer(  
 'kafka\_1',  
 bootstrap\_servers=['localhost:9092'],  
 auto\_offset\_reset='earliest',  
 enable\_auto\_commit=True,  
 group\_id='my\_group',  
 value\_deserializer=lambda x: loads(x.decode('utf-8')))  
  
# Draw graph  
plt.rcParams['animation.html'] = 'jshtml'  
fig = plt.figure()  
ax = fig.add\_subplot(111)  
fig.show()  
i = 0  
  
# x for xis, y for yxis  
  
x, y = [], []  
  
# Data from producer  
for message in consumer:  
 message = message.value  
 x.append(i)  
 y.append(message['data'])  
 ax.plot(x, y, color='b', linewidth=2, linestyle="-", label="Data per sec")  
 plt.gcf().autofmt\_xdate()  
 fig.canvas.draw()  
  
 ax.set\_xlim(left=max(0, i - 50), right=i)  
 ax.set\_title("Plotting data every sec")  
 ax.set\_xlabel("Time(sec)")  
 ax.set\_ylabel("Data")  
 i += 1  
  
 if len(x) > 50:  
 del (x[0])  
 del (y[0])

< kafka\_producer.py >

from time import sleep  
from json import dumps  
from kafka import KafkaProducer  
import random  
  
# Make a Kafaka producer  
producer = KafkaProducer(bootstrap\_servers=['localhost:9092'],  
 value\_serializer=lambda x: C  
 dumps(x).encode('utf-8'))  
  
# Generate data every second  
for e in range(100):  
 data = {'data' : random.randrange(0,100)}  
 producer.send('kafka\_1', value=data)  
 sleep(1)

**Screen shot**

**<SetA-2>**

**Description:** Plotting data from producer where producer generates data every sec. Plot the mean value of the five seconds

**Technique:**

* Apache kafka 1.3.5
* Python 3.7
* Kafka-python 1.4.4

**Solving steps:**

1. Run the zookeeper server and Kafka server.
2. Create Kafka producer and Kafka consumer.
3. Kafka producer sends random data to Kafka consumer every second.
4. Kafka consumer receives that data and plots the mean value of the five seconds using matplotlib library.

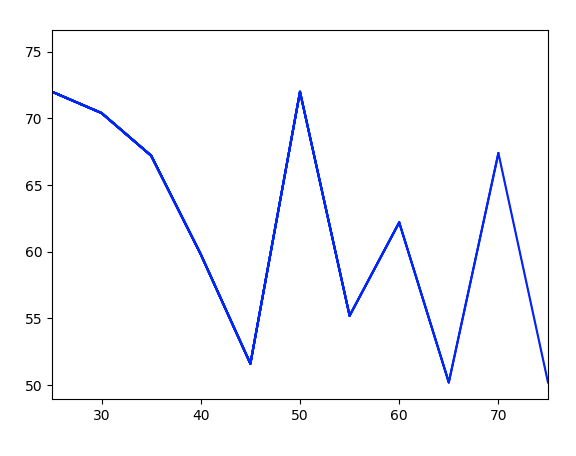
**Code**

<kafka\_consumer.py>

### Plotting data from producer where producer generates data every sec  
### Plot the mean value of the five seconds  
  
from kafka import KafkaConsumer  
from json import loads  
import matplotlib.pyplot as plt  
import statistics  
% matplotlib notebook  
  
# Consumer connection  
consumer = KafkaConsumer(  
 'kafka\_2',  
 bootstrap\_servers=['localhost:9092'],  
 auto\_offset\_reset='earliest',  
 enable\_auto\_commit=True,  
 group\_id='my\_group',  
 value\_deserializer=lambda x: loads(x.decode('utf-8')))  
  
# Draw graph  
plt.rcParams['animation.html'] = 'jshtml'  
fig = plt.figure()  
ax = fig.add\_subplot(111)  
fig.show()  
i = 0  
  
# x for xis, y for yxis  
x, y = [], []  
tmp\_data = []  
check = 0  
  
# Data from producer  
for message in consumer:  
 message = message.value  
  
 # Draw mean of data every 5 secs  
 if check == 5:  
 ax.set\_xlim(left=max(0, i - 50), right=i)  
 mean = statistics.mean(tmp\_data)  
 # print(mean)  
 x.append(i)  
 y.append(statistics.mean(tmp\_data))  
 tmp\_data.clear()  
 ax.plot(x, y, color='b')  
 fig.canvas.draw()  
 check = 0  
 i += 5  
  
 # Getting data every second and store into an array  
 tmp\_data.append(message['data'])  
  
 check += 1  
  
 if len(x) > 50:  
 del (x[0])  
 del (y[0])

<kafka\_producer.py>

from time import sleep  
from json import dumps  
from kafka import KafkaProducer  
import random  
  
# Make a Kafaka producer  
producer = KafkaProducer(bootstrap\_servers=['localhost:9092'],  
 value\_serializer=lambda x:  
 dumps(x).encode('utf-8'))  
  
  
# Generate data every sec  
for e in range(100):  
 data = {'data' : random.randrange(0,100)}  
 producer.send('kafka\_2', value=data)  
 sleep(1)

**Screen shot**

**<SetA-3>**

**Description:** Plotting random data from producer where producer generates data random sec.

**Technique:**

* Apache kafka 1.3.5
* Python 3.7
* Kafka-python 1.4.4

**Solving steps:**

1. Run the zookeeper server and Kafka server.
2. Create Kafka producer and Kafka consumer.
3. Kafka producer sends random data to Kafka consumer random interval seconds.
4. Kafka consumer receives that data and plots the data when it gets the data from producer.

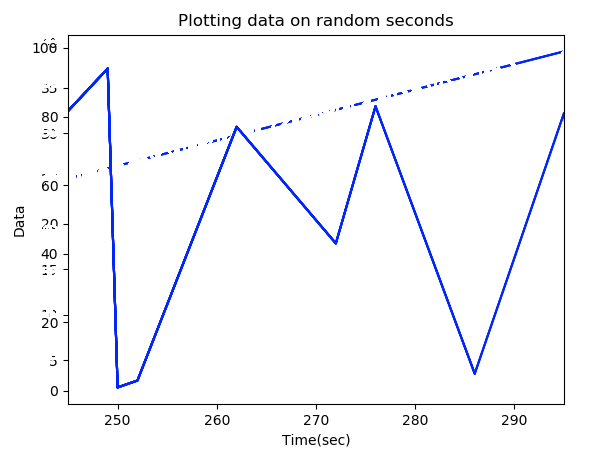
**Code**

<kafka\_consumer.py>

from kafka import KafkaConsumer  
from json import loads  
import matplotlib.pyplot as plt  
% matplotlib  
notebook  
  
# Consumer connection  
consumer = KafkaConsumer(  
 'kafka\_31',  
 bootstrap\_servers=['localhost:9092'],  
 auto\_offset\_reset='earliest',  
 enable\_auto\_commit=True,  
 group\_id='my\_group',  
 value\_deserializer=lambda x: loads(x.decode('utf-8')))  
  
# Draw graph  
plt.rcParams['animation.html'] = 'jshtml'  
fig = plt.figure()  
ax = fig.add\_subplot(111)  
fig.show()  
i = 0  
j = 0  
tmp\_time = []  
  
# x for xis, y for yxis  
x, y = [], []  
  
# Data from producer  
for message in consumer:  
 message = message.value  
 # tmp\_time = message['time']  
 tmp\_time.append(message['time'])  
 x.append(i)  
 y.append(message['data'])  
 ax.plot(x, y, color='b')  
 fig.canvas.draw()  
  
 ax.set\_xlim(left=max(0, i - 50), right=i)  
 ax.set\_title("Plotting data on random seconds")  
 ax.set\_xlabel("Time(sec)")  
 ax.set\_ylabel("Data")  
  
 i += tmp\_time[j]  
 j += 1  
 if len(x) > 50:  
 del (x[0])  
 del (y[0])

<kafka\_producer.py>

from time import sleep  
from json import dumps  
from kafka import KafkaProducer  
import random  
  
# Make a Kafaka producer  
producer = KafkaProducer(bootstrap\_servers=['localhost:9092'],  
 value\_serializer=lambda x:  
 dumps(x).encode('utf-8'))  
  
# Generate data on random time  
for e in range(100):  
 tmp\_time = random.randint(1,10)  
 data = {'data' : random.randrange(0,100), 'time' : tmp\_time}  
 producer.send('kafka\_31', value=data)  
 sleep(tmp\_time)

**Screen shot**

**<SetA-4>**

**Description:** Generate data in random time, random amount of data in random range from 1-100. Plotting data from producer where producer generates random amount of data in random time

**Technique:**

* Apache kafka 1.3.5
* Python 3.7
* Kafka-python 1.4.4

**Solving steps:**

1. Run the zookeeper server and Kafka server.
2. Create Kafka producer and Kafka consumer.
3. Kafka producer sends random data and random amount of data to Kafka consumer random interval seconds.
4. Kafka consumer receives that data and plots(scatters) the data when it gets the data from producer.

**Code**

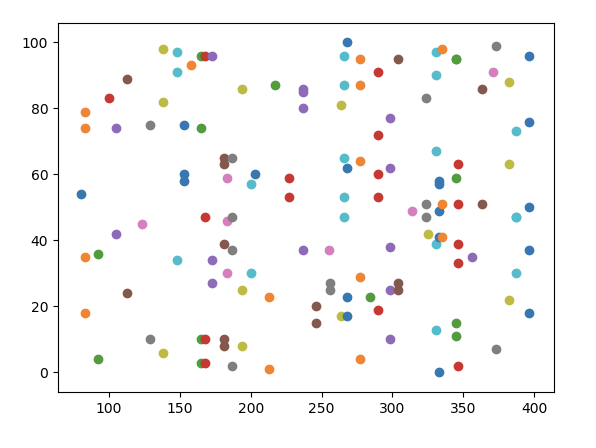
<kafka\_consumer.py>

from kafka import KafkaConsumer  
from json import loads  
import matplotlib.pyplot as plt  
% matplotlib  
notebook  
  
# Consumer connection  
consumer = KafkaConsumer(  
 'kafka\_4',  
 bootstrap\_servers=['localhost:9092'],  
 auto\_offset\_reset='earliest',  
 enable\_auto\_commit=True,  
 group\_id='my\_group',  
 value\_deserializer=lambda x: loads(x.decode('utf-8')))  
  
plt.rcParams['animation.html'] = 'jshtml'  
fig = plt.figure()  
ax = fig.add\_subplot(111)  
fig.show()  
i = 0  
j = 0  
tmp\_time = []  
  
# x for xAxis, y for yAxis  
x, y = [], []  
  
# Data from producer  
for message in consumer:  
 message = message.value  
  
 # Range checking  
 if len(x) > 50:  
 del (x[0])  
 del (y[0])  
  
 tmp\_time.append(message['time'])  
  
 # value of xAxis and yAxis with max value of yAxis  
 x.append(i)  
 y.append(message['data'])  
  
 ax.clear()  
 for xe, ye in zip(x, y):  
 plt.scatter([xe] \* len(ye), ye, label="Data from producer")  
 fig.canvas.draw()  
  
 ax.set\_xlim(left=max(0, i - 50), right=i)  
  
 i += tmp\_time[j]  
 j += 1

<kafka\_producer.py>

from time import sleep  
from json import dumps  
from kafka import KafkaProducer  
import random  
  
# Make a Kafaka producer  
producer = KafkaProducer(bootstrap\_servers=['localhost:9092'],  
 value\_serializer=lambda x:  
 dumps(x).encode('utf-8'))  
  
tmp\_data = []  
  
# Generate data every random sec  
for e in range(100):  
 getsu = random.randint(1, 5)  
 tmp\_time = random.randint(1, 10)  
 for i in range(0, getsu):  
 tmp\_data.append(random.randint(0, 100))  
 data = {'data': tmp\_data, 'time': tmp\_time}  
 producer.send('kafka\_4', value=data)  
 sleep(tmp\_time)  
 tmp\_data.clear()

**Screen shot**



**<SetA-5>**

**Description:** Data has a range. So we know the range of the value will always in between. But if the data has no boundary, so we don’t know the boundary, but the consumer will draw the graph anyways. Additional, when we draw the graph, we show maximum, minimum, current value.

**Technique:**

* Apache Kafka 1.3.5
* Python 3.7
* Kafka-python 1.4.4

**Solving steps:**

1. Run the zookeeper server and Kafka server.
2. Create Kafka producer and Kafka consumer.
3. Kafka producer sends random data to Kafka consumer every second.
4. Kafka consumer receives that data and plots the data of maximum, minimum for 10 seconds, and current data using annotation of matplotlib

**Code**

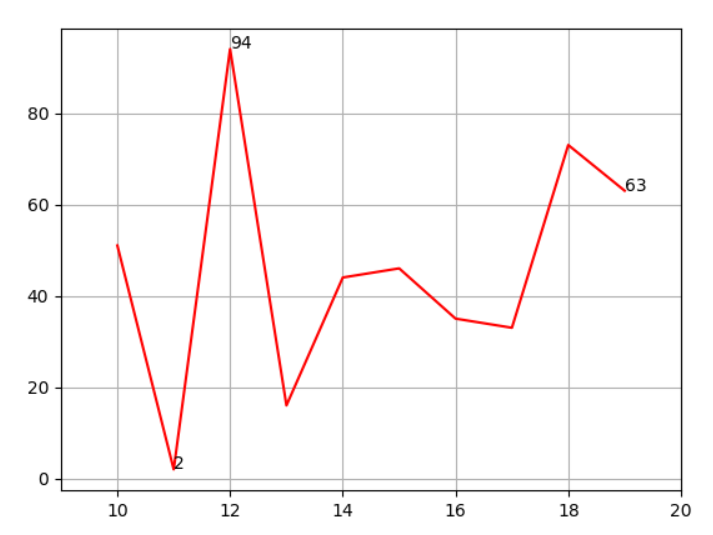
<kafka\_consumer.py>

from kafka import KafkaConsumer  
from json import loads  
import matplotlib.pyplot as plt  
  
# Make a Kafaka consumer  
consumer = KafkaConsumer(  
 'pro\_1',  
 bootstrap\_servers=['localhost:9092'],  
 auto\_offset\_reset='earliest',  
 enable\_auto\_commit=True,  
 group\_id='m\_g',  
 value\_deserializer=lambda x: loads(x.decode('utf-8')))  
  
# Make a plot graph  
plt.rcParams['animation.html'] = 'jshtml'  
plt.rcParams['axes.grid'] = True  
fig = plt.figure()  
ax = fig.add\_subplot(111)  
fig.show()  
i = 0  
  
# x is a second, y is a data from producer  
x, y = [], []  
  
# Get message from the producer via kafka  
for message in consumer:  
 message = message.value  
 if len(x) > 9:  
 del (x[0])  
 del (y[0])  
 # z is for min, max and current value in last ten second  
 z = []  
 x.append(i)  
 y.append(message['number'])  
 z.append(min(y))  
 z.append(max(y))  
 z.append(message['number'])  
 ax.clear()  
 ax.plot(x, y, color='r')  
 for n, txt in enumerate(z):  
 ax.annotate(txt, (i - 9 + y.index(z[n]), z[n]))  
 ax.set\_xlim(left=max(0, i - 10), right=i + 1)  
 fig.canvas.draw()  
 i += 1

<kafka\_producer.py>

from time import sleep  
from json import dumps  
from kafka import KafkaProducer  
import random  
  
# Make a Kafaka producer  
producer = KafkaProducer(bootstrap\_servers=['localhost:9092'],  
 value\_serializer=lambda x:  
 dumps(x).encode('utf-8'))  
  
# Generate data every second  
for e in range(100):  
 data = {'number': random.randrange(0, 100)}  
 producer.send('pro\_1', value=data)  
 sleep(1)

**Screen shot**



**<SetA-6>**

**Description:** Show the average of current value with plotting. And we get series of data and we wait for 10 secs. For example, At the 11 secs, plot the average between 1~10secs. Plotting data every second, with average data of last 10 secs. Additional, show the original data.

**Technique:**

* Apache Kafka 1.3.5
* Python 3.7
* Kafka-python 1.4.4
* statistics

**Solving steps:**

1. Run the zookeeper server and Kafka server.
2. Create Kafka producer and Kafka consumer.
3. Kafka producer sends random data to Kafka consumer every second.
4. Range checking for axis.
5. Kafka consumer receives that data and calculate the average for 1-10secs.
6. Then, consumer draws graph about above average and original data.

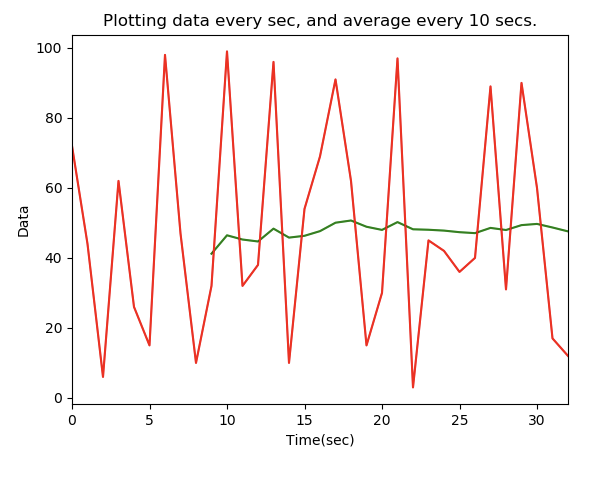
**Code**

<kafka\_consumer.py>

from kafka import KafkaConsumer  
from json import loads  
import matplotlib.pyplot as plt  
import statistics  
% matplotlib  
notebook  
  
# Consumer connection  
consumer = KafkaConsumer(  
 'kafka\_6',  
 bootstrap\_servers=['localhost:9092'],  
 auto\_offset\_reset='earliest',  
 enable\_auto\_commit=True,  
 group\_id='my\_group',  
 value\_deserializer=lambda x: loads(x.decode('utf-8')))  
  
plt.rcParams['animation.html'] = 'jshtml'  
fig = plt.figure()  
ax = fig.add\_subplot(111)  
fig.show()  
i = 0  
check = 0  
  
x, y, x\_, avg = [], [], [], []  
  
# Data from producer  
for message in consumer:  
 message = message.value  
  
 # Range checking  
 if len(x) > 49:  
 del (x[0])  
 del (y[0])  
 if len(avg) > 49:  
 del (x\_[0])  
 del (avg[0])  
  
 x.append(i)  
 # original data  
 y.append(message['data'])  
 ax.clear()  
  
 check += 1  
 # For last 10 secs, plotting average.  
 if check > 9:  
 avg.append(statistics.mean(y))  
 x\_.append(i)  
 ax.plot(x\_, avg, color='g', label="Average")  
  
 ax.plot(x, y, color='r', label="Data per sec")  
  
 ax.set\_xlim(left=max(0, i - 50), right=i)  
 ax.set\_title("Plotting data every sec, and average every 10 secs.")  
 ax.set\_xlabel("Time(sec)")  
 ax.set\_ylabel("Data")  
 fig.canvas.draw()  
 i += 1

<kafka\_producer.py>

from time import sleep  
from json import dumps  
from kafka import KafkaProducer  
import random  
  
# Make a Kafaka producer  
producer = KafkaProducer(bootstrap\_servers=['localhost:9092'],  
 value\_serializer=lambda x:  
 dumps(x).encode('utf-8'))  
  
# Generate data every sec  
for e in range(100):  
 data = {'data': random.randrange(0, 100)}  
 producer.send('kafka\_6', value=data)  
 sleep(1)

**Screen shot**

**<SetA-7>**

**Description:** From three producer, plot each data from each producer and plot average data. So, total number of graphs is 4.

**Technique:**

* Apache Kafka 1.3.5
* Python 3.7
* Kafka-python 1.4.4
* statistics

**Solving steps:**

1. Run the zookeeper server and Kafka server.
2. Create 3-Kafka producer and Kafka consumer.
3. Kafka 3-producer send random data to Kafka consumer every second.
4. Kafka consumer receives that data and calculate the average of all producer’s data.
5. Then, consumer draws 4 graphs about all producer’s data and average calculated.

**Code**

<kafka\_consumer.py>

from kafka import KafkaConsumer  
from json import loads  
import statistics  
import time  
import matplotlib.pyplot as plt  
% matplotlib  
notebook  
  
# Consumer connection  
consumer = KafkaConsumer(  
 'kafka\_7',  
 bootstrap\_servers=['localhost:9092'],  
 auto\_offset\_reset='earliest',  
 enable\_auto\_commit=True,  
 group\_id='my\_group',  
 value\_deserializer=lambda x: loads(x.decode('utf-8')))  
  
# Draw 4 graphs  
plt.rcParams['animation.html'] = 'jshtml'  
fig = plt.figure(figsize=(6, 9))  
ax1 = fig.add\_subplot(811)  
ax2 = fig.add\_subplot(813)  
ax3 = fig.add\_subplot(815)  
ax4 = fig.add\_subplot(817)  
fig.show()  
  
# x for xis, y for yxis, temp for average  
x1, y1 = [], []  
x2, y2 = [], []  
x3, y3 = [], []  
ave\_x, ave\_y = [], []  
temp = []  
tmp\_time1, tmp\_time2, tmp\_time3, tmp\_time4 = 0, 0, 0, 0  
  
i = [0, 0, 0, 0]  
count = 0  
# Data from producer  
for message in consumer:  
 count += 1  
 message = message.value  
  
 # if 'data1' & 'time1' in message:  
 if 'data1' in message:  
 now1 = time.gmtime(time.time())  
 tmp\_sec1 = now1.tm\_sec  
 if tmp\_sec1 < 10:  
 tmp\_sec1 \*= 0.01  
 if tmp\_sec1 > 10:  
 tmp\_sec1 \*= 0.01  
 tmp\_time1 = str(now1.tm\_min + tmp\_sec1)  
 x1.append(float(tmp\_time1))  
 i[0] = float(tmp\_time1)  
 y1.append(message['data1'])  
 temp.append(message['data1'])  
 if 'data2' in message:  
 now2 = time.gmtime(time.time())  
 tmp\_sec2 = now2.tm\_sec  
 if tmp\_sec2 < 10:  
 tmp\_sec2 \*= 0.01  
 if tmp\_sec2 > 10:  
 tmp\_sec2 \*= 0.01  
 tmp\_time2 = str(now2.tm\_min + tmp\_sec2)  
 print(tmp\_time2)  
 x2.append(float(tmp\_time2))  
 i[1] = float(tmp\_time2)  
 y2.append(message['data2'])  
 temp.append(message['data2'])  
 if 'data3' in message:  
 now3 = time.gmtime(time.time())  
 tmp\_sec3 = now3.tm\_sec  
 if tmp\_sec3 < 10:  
 tmp\_sec3 \*= 0.01  
 if tmp\_sec3 > 10:  
 tmp\_sec3 \*= 0.01  
 tmp\_time3 = str(now3.tm\_min + tmp\_sec3)  
 x3.append(float(tmp\_time3))  
 i[2] = float(tmp\_time3)  
 y3.append(message['data3'])  
 temp.append(message['data3'])  
  
 now4 = time.gmtime(time.time())  
 tmp\_time4 = str(now4.tm\_min) + '.' + str(now4.tm\_sec)  
 i[3] = float(tmp\_time3)  
 print(i[3])  
 ave\_x.append(float(tmp\_time3))  
 ave\_y.append(statistics.mean(temp))  
  
 # Range checking  
 if count > 49:  
 del (x1[0])  
 del (y1[0])  
 del (x2[0])  
 del (y2[0])  
 del (x3[0])  
 del (y3[0])  
 del (ave\_x[0])  
 del (ave\_y[0])  
 count = 0  
  
 # Draw graph  
 ax1.clear()  
 ax2.clear()  
 ax3.clear()  
 ax4.clear()  
  
 ax1.plot(x1, y1, color='b')  
 ax2.plot(x2, y2, color='b')  
 ax3.plot(x3, y3, color='b')  
 ax4.plot(ave\_x, ave\_y, color='b')  
  
 ax1.set\_xlim(left=max(0, int(float(tmp\_time1))), right=int(float(tmp\_time1)) + 0.6)  
 ax1.set\_title("Plotting data every sec")  
 ax1.set\_xlabel("Time(sec)")  
 ax1.set\_ylabel("Data")  
 fig.canvas.draw()  
  
 ax2.set\_xlim(left=max(0, int(float(tmp\_time2))), right=int(float(tmp\_time2)) + 0.6)  
 # ax2.set\_title("Plotting data every sec \n from producer\_2")  
 ax2.set\_xlabel("Time(sec)")  
 ax2.set\_ylabel("Data")  
 fig.canvas.draw()  
  
 ax3.set\_xlim(left=int(float(tmp\_time3)) - 0.6, right=int(float(tmp\_time3)) + 0.6)  
 # ax3.set\_title("Plotting data every sec \n from producer\_3")  
 ax3.set\_xlabel("Time(sec)")  
 ax3.set\_ylabel("Data")  
 fig.canvas.draw()  
  
 ax4.set\_xlim(left=int(float(tmp\_time3)) - 0.6, right=int(float(tmp\_time3)) + 0.6)  
 # ax4.set\_title("Plotting data every sec \n from producer\_4")  
 ax4.set\_xlabel("Time(sec)")  
 ax4.set\_ylabel("Data")  
 fig.canvas.draw()  
  
# Close plot  
plt.close('all')

<kafka\_producer1.py>

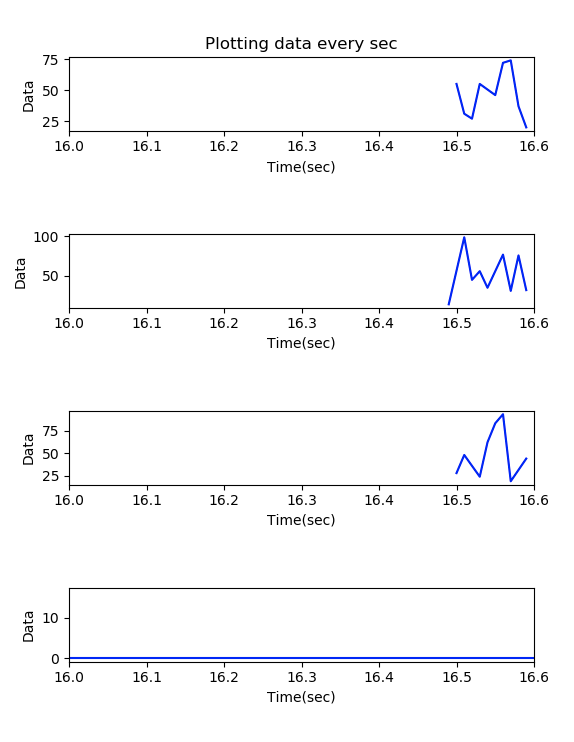
from time import sleep  
from json import dumps  
from kafka import KafkaProducer  
import random  
  
# Make a Kafaka producer  
producer = KafkaProducer(bootstrap\_servers=['localhost:9092'],  
 value\_serializer=lambda x:  
 dumps(x).encode('utf-8'))  
  
# Generate data on random time  
for e in range(100):  
 data = {'data1' : random.randrange(0,100)}  
 producer.send('kafka\_7', value=data)  
 sleep(1)

<kafka\_producer2.py>

from time import sleep  
from json import dumps  
from kafka import KafkaProducer  
import random  
import time  
  
# Make a Kafaka producer  
producer = KafkaProducer(bootstrap\_servers=['localhost:9092'],  
 value\_serializer=lambda x:   
 dumps(x).encode('utf-8'))  
  
# Generate data on random time  
for e in range(100):  
 data = {'data2' : random.randrange(0,100)}  
 producer.send('kafka\_7', value=data)  
 sleep(1)

<kafka\_producer3.py>

from time import sleep  
from json import dumps  
from kafka import KafkaProducer  
import random  
  
# Make a Kafaka producer  
producer = KafkaProducer(bootstrap\_servers=['localhost:9092'],  
 value\_serializer=lambda x:  
 dumps(x).encode('utf-8'))  
  
# Generate data on random time  
for e in range(100):  
 data = {'data3' : random.randrange(0,100)}  
 producer.send('kafka\_7', value=data)  
 sleep(1)

**Screen shot**

**SET\_B**

**<SetB-1>**

**Description:** Generate data every second, consumer stores that data in mongoDB.

**Technique:**

* Apache Kafka 1.3.5
* Python 3.7
* Kafka-python 1.4.4
* Pymongo 3.7.2

**Solving steps:**

1. Run the zookeeper server and Kafka server.
2. Create 3-Kafka producer and Kafka consumer.
3. Connect the mongoDB (command mongod)
4. Kafka producer sends data to Kafka consumer every second.
5. Kafka consumer receives that data and store in mongo DB.

**Code**

<kafka\_consumer.py>

from kafka import KafkaConsumer  
from pymongo import MongoClient  
from json import loads  
  
consumer = KafkaConsumer(  
 'numtest1',  
 bootstrap\_servers=['localhost:9092'],  
 auto\_offset\_reset='earliest',  
 enable\_auto\_commit=True,  
 group\_id='my-group',  
 value\_deserializer=lambda x: loads(x.decode('utf-8')))  
  
# connect mongoDB  
client = MongoClient('localhost:27017')  
  
# make database and collection  
collection = client.numtest.numtest  
  
# receive the data from producer, and store that data in mongoDB  
for message in consumer:  
 message = message.value  
 collection.insert\_one(message)  
 print('{} added to {}'.format(message, collection))

<kafka\_producer.py>

from time import sleep  
from json import dumps  
from kafka import KafkaProducer  
  
producer = KafkaProducer(bootstrap\_servers=['localhost:9092'],  
 value\_serializer=lambda x:  
 dumps(x).encode('utf-8'))  
  
for e in range(100):  
 data = {'number': e}  
 producer.send('numtest1', value=data)  
 sleep(1)

**Screen shot**

**<SetB-2>**

**Description:** Generate data every second from Kafka producer, Kafka consumer has to check whether it has null value. If it has null value, that data represent (ex. “\*”) in mongoDB table.

**Technique:**

* Apache Kafka 1.3.5
* Python 3.7
* Kafka-python 1.4.4
* Pymongo 3.7.2

**Solving steps:**

1. Run the zookeeper server and Kafka server.
2. Create 3-Kafka producer and Kafka consumer.
3. Connect the mongoDB (command mongod).
4. Kafka consumer checks that value exists, and valid range.
5. Then, insert data into mongoDB.
6. After checking, If no value, represent “\*”. DB update.

**Code**

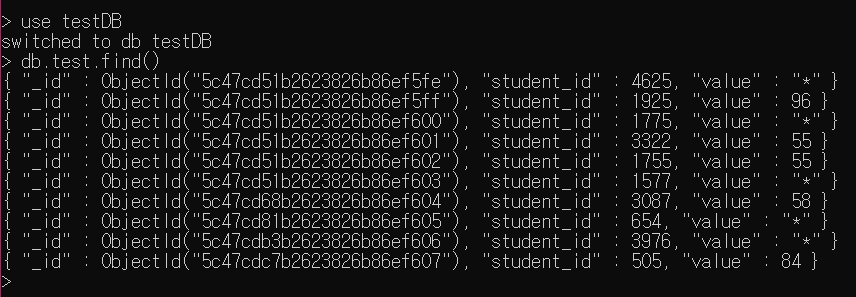
<kafka\_consumer.py>

from kafka import KafkaConsumer  
from pymongo import MongoClient  
from json import loads  
  
consumer = KafkaConsumer(  
 'mongotest',  
 bootstrap\_servers=['localhost:9092'],  
 auto\_offset\_reset='earliest',  
 enable\_auto\_commit=True,  
 group\_id='my-group',  
 value\_deserializer=lambda x: loads(x.decode('utf-8')))  
  
# connect with mongoDB  
connection = MongoClient("localhost", 27017)  
  
# make database  
db = connection.testDB  
  
# make collection  
collection = db.test  
  
for message in consumer:  
 message = message.value  
  
 # if value exists, check range  
 if 'value' in message:  
 if message['value'] > 100:  
 print("the number is bigger than 100")  
 continue  
 # insert data into collection  
 collection.insert\_one(message)  
  
 # if no value, represent "\*"  
 collection.find\_one\_and\_update(  
 {"value": {"$exists": False}}, {"$set": {"value": "\*"}}, new=True)

<kafka\_producer.py>

from time import sleep  
from json import dumps  
from kafka import KafkaProducer  
import random  
  
producer = KafkaProducer(bootstrap\_servers=['localhost:9092'],  
 value\_serializer=lambda x:  
 dumps(x).encode('utf-8'))  
  
for e in range(100):  
 if e % 10 != 0:  
 data = {'student\_id': random.randrange(0, 5000),  
 'value': random.randrange(0, 1000)  
 }  
 # for 10 interval, randomly send no data  
 else:  
 data = {'student\_id': random.randrange(0, 5000)  
 }  
  
 producer.send('mongotest', value=data)  
 sleep(5)  
 e += 1

**Screen shot**



**<SetB-3>**

**Description:** The problem is to send random values between 0 and 100 from two producers and then plot two received values onto a single graph. Finally save to mongodb.

**Technique:**

* Apache Kafka 1.3.5
* Python 3.7
* Kafka-python 1.4.4
* Pymongo 3.7.2

**Solving steps:**

1. Run the zookeeper server and Kafka server.
2. Create 2-Kafka producer and Kafka consumer.
3. Connect the mongoDB (command mongod).
4. Plots data from two producers in different colors.
5. Then insert the data into mongoDB

**Code:**

<SetB\_P3\_Consumer.py>

from kafka import KafkaConsumer  
from pymongo import MongoClient  
import matplotlib  
matplotlib.use('TkAgg')  
import matplotlib.pyplot as plt  
from json import loads  
  
#% matplotlib notebook  
  
# Consumer connection  
consumer = KafkaConsumer(  
 'SetB\_Problem3\_1',  
 bootstrap\_servers=['localhost:9092'],  
 auto\_offset\_reset='earliest',  
 enable\_auto\_commit=True,  
 group\_id='my\_group',  
 value\_deserializer=lambda x: loads(x.decode('utf-8')))  
  
client = MongoClient('localhost:27017')  
collection = client.p3.p3  
  
fig = plt.figure()  
ax = fig.add\_subplot(111)  
# Draw graph  
plt.rcParams['animation.html'] = 'jshtml'  
i = 0  
  
# x1 for axis, y1 for y axis  
x1, y1 = [], []  
# x2 for x axis y2 for another y axis  
x2, y2 = [], []  
  
fig.show()  
# Data from producer  
for message in consumer:  
 message = message.value  
   
 collection.insert\_one(message)  
 if len(x1) > 9:  
 del (x1[0])  
 del (y1[0])  
 if len(x2) > 9:  
 del (x2[0])  
 del (y2[0])  
  
 # print('{} added to {}'.format(message,collection))  
  
 if 'data1' in message:  
 x1.append(i)  
 y1.append(message['data1'])  
  
 if 'data2' in message:  
 x2.append(i)  
 y2.append(message['data2'])  
  
 i += 1  
 plt.title('comparable graph')  
 ax.clear()  
 # print(x1, y1)  
 # print(x2, y2)  
 ax.plot(x1, y1, 'r', label='data1')  
 ax.plot(x2, y2, 'b', label='data2')  
 ax.set\_xlim(left=max(0, i - 10), right=i + 1)  
 fig.canvas.draw()  
# plt.show()  
  
# Close plot  
plt.close('all')

<SetB\_P3\_Producer1.py>

from time import sleep  
from json import dumps  
from kafka import KafkaProducer  
import random  
  
# Make a Kafka producer  
producer = KafkaProducer(bootstrap\_servers=['localhost:9092'],  
 value\_serializer=lambda x:  
 dumps(x).encode('utf-8'))  
  
# Generate data every second  
for e in range(100):  
 data = {'data1': random.randrange(0, 100)}  
 producer.send('SetB\_Problem3\_1', value=data)  
 sleep(1)

<SetB\_P3\_Producer2.py>

from time import sleep  
from json import dumps  
from kafka import KafkaProducer  
import random  
  
# Make a Kafka producer  
producer = KafkaProducer(bootstrap\_servers=['localhost:9092'],  
 value\_serializer=lambda x:  
 dumps(x).encode('utf-8'))  
  
# Generate data every second  
for e in range(100):  
 data = {'data2': random.randrange(0, 100)}  
 producer.send('SetB\_Problem3\_1', value=data)  
 sleep(1)

**Screen shot**

