DataEng: Data Transport Activity

Yiming Lin
01/21/2021

[this lab activity references tutorials at confluence.com]

Make a copy of this document and use it to record your results. Store a PDF copy of the document in your git repository along with your code before submitting for this week. For your code, you create several producer/consumer programs or you might make various features within one program. There is no one single correct way to do it. Regardless, store your code in your repository.

The goal for this week is to gain experience and knowledge of using a streaming data transport system (Kafka). Complete as many of the following exercises as you can. Proceed at a pace that allows you to learn and understand the use of Kafka with python.

A. Initialization

1. Get your cloud.google.com account up and running
    a. Redeem your GCP coupon
    b. Login to your GCP console
    c. Create a new, separate VM instance
2. Follow the Kafka tutorial from project assignment #1
    a. Create a separate topic for this in-class activity
    b. Make it “small” as you will not want to use many resources for this activity. By “small” I mean that you should choose medium or minimal options when asked for any configuration decisions about the topic, cluster, partitions, storage, anything. GCP/Confluent will ask you to choose the configs, and because you are using a free account you should opt for limited resources where possible.
    c. Get a basic producer and consumer working with a Kafka topic as described in the tutorials.
3. Create a sample breadcrumb data file (named bcsample.json) consisting of a sample of 1000 breadcrumb records. These can be any records because we will not be concerned with the actual contents of the breadcrumb records during this assignment.
4. Update your producer to parse your sample.json file and send its contents, one record at a time, to the kafka topic.

5. Use your consumer.py program (from the tutorial) to consume your records.

B. Kafka Monitoring

1. Find the Kafka monitoring console for your topic. Briefly describe its contents. Do the measured values seem reasonable to you?
2. Use this monitoring feature as you do each of the following exercises.

C. Kafka Storage

1. Run the linux command “wc bcsample.json”. Record the output here so that we can verify that your sample data file is of reasonable size.

```
0  10281908 124760360 bcsample.json
```

2. What happens if you run your consumer multiple times while only running the producer once?
   (confluent-exercise) yl6@yml-dataeng-wk3:~/examples/clients/cloud/python$ ./consumer.py -f ./.confluent/librdkafka.config -t wk3-separate-topic
   Waiting for message or event/error in poll()
   Waiting for message or event/error in poll()
   Waiting for message or event/error in poll()
   Waiting for message or event/error in poll()
   Waiting for message or event/error in poll()
   Waiting for message or event/error in poll()
   Waiting for message or event/error in poll()

3. Before the consumer runs, where might the data go, where might it be stored?

4. Is there a way to determine how much data Kafka/Confluent is storing for your topic? Do the Confluent monitoring tools help with this?
   The Confluent webpage shows us how much data is in the storage over the time. Also, Confluent provides an API to obtain how many bytes are in the storage. For example, the following example tells that 473665 bytes is in the storage at 2021-01-19T11:33:00-08:00.

```
{
  "data": [
    {
      "value": 473665,
```

```
      "timestamp": "2021-01-19T11:33:00-08:00"
    },
  ]
}
```


5. Create a “topic_clean.py” consumer that reads and discards all records for a given topic. This type of program can be very useful during debugging.

   [See topic_clean.py]

D. Multiple Producers

1. Clear all data from the topic
2. Run two versions of your producer concurrently, have each of them send all 1000 of your sample records. When finished, run your consumer once. Describe the results.

(1) Create two producers (in two threads) and run producers as following:


```
(confluent-exercise) yl6@yml-dataeng-wk3:~/cs510-dataeng-inclass/wk3$ ./topic_clean.py -f ./.confluent/librdkafka.config -t wk3-separate-topic
No more message to consume, 0 messages are discarded
(confluent-exercise) yl6@yml-dataeng-wk3:~/cs510-dataeng-inclass/wk3$ ./producer.py -f ./.confluent/librdkafka.config -t wk3-separate-topic --nthread 2
INFO:root:Producing record: producer0   {"EVENT_NO_TRIP": "167589364", "EVENT_NO_STOP": "167589369", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METE
RS": "354", "ACT_TIME": "21756", "VELOCITY": "", "DIRECTION": "", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.604897", "GPS_LATITUDE": "45.63653", "GPS_S
ATELLITES": "9", "GPS_HDOP": "1", "SCHEDULE_DEVIATION": ""}
INFO:root:Producing record: producer1   {"EVENT_NO_TRIP": "167589364", "EVENT_NO_STOP": "167589369", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METE
RS": "354", "ACT_TIME": "21756", "VELOCITY": "", "DIRECTION": "", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.604897", "GPS_LATITUDE": "45.63653", "GPS_S
ATELLITES": "9", "GPS_HDOP": "1", "SCHEDULE_DEVIATION": ""}
INFO:root:Producing record: producer0   {"EVENT_NO_TRIP": "167589364", "EVENT_NO_STOP": "167589369", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METE
RS": "389", "ACT_TIME": "21761", "VELOCITY": "7", "DIRECTION": "109", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.60449", "GPS_LATITUDE": "45.636432", "G
PS_SATELLITES": "9", "GPS_HDOP": "0.9", "SCHEDULE_DEVIATION": ""}
INFO:root:Producing record: producer1   {"EVENT_NO_TRIP": "167589364", "EVENT_NO_STOP": "167589369", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METE
RS": "389", "ACT_TIME": "21761", "VELOCITY": "7", "DIRECTION": "109", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.60449", "GPS_LATITUDE": "45.636432", "G
PS_SATELLITES": "9", "GPS_HDOP": "0.9", "SCHEDULE_DEVIATION": ""}
INFO:root:Produced record to topic wk3-separate-topic partition [1] @ offset 10453
INFO:root:Produced record to topic wk3-separate-topic partition [1] @ offset 10454
INFO:root:Producing record: producer0   {"EVENT_NO_TRIP": "167589364", "EVENT_NO_STOP": "167589369", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METE
RS": "442", "ACT TIME": "21766", "VELOCITY": "10", "DIRECTION": "85", "RADIO QUALITY": "", "GPS LONGITUDE": "-122.60381", "GPS LATITUDE": "45.636475", "G
```


The screenshot below shows the beginning offset of partition 5:

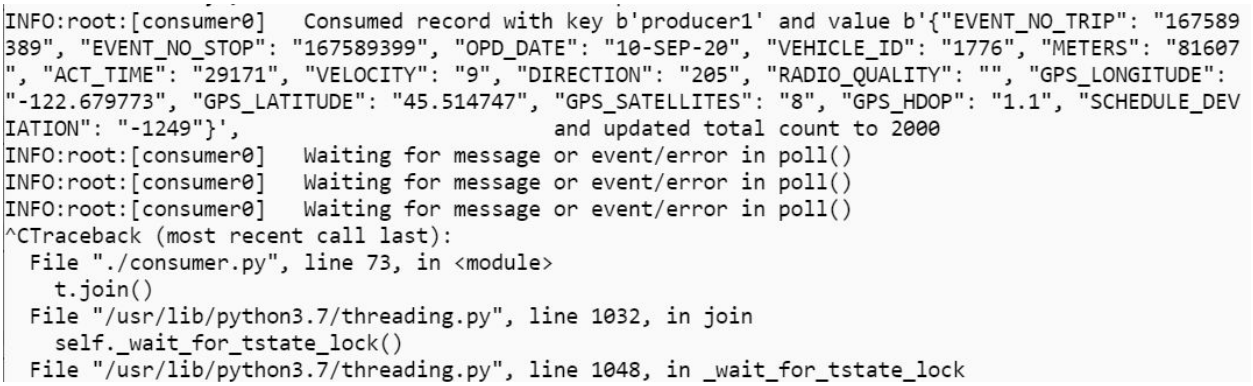

```
RS": "644", "ACT_TIME": "21781", "VELOCITY": "13", "DIRECTION": "84", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.6012", "GPS_LATITUDE": "45.636583", "GP
S_SATELLITES": "9", "GPS_HDOP": "0.9", "SCHEDULE_DEVIATION": ""}
INFO:root:Produced record to topic wk3-separate-topic partition [5] @ offset 42416
INFO:root:Produced record to topic wk3-separate-topic partition [5] @ offset 42417
```


Thus, the beginning offset of partition 1 is 10453, and the beginning offset of partition 5 is 42416.

(2) After producer finished, the screenshot below shows its ending offset:

The beginning offset of partition 1 is 11452, and the beginning offset of partition 5 is 43415. For partition 1, 11452 - 10453 + 1 = 1000; for partition 5, 43415 - 42416 + 1 = 1000. Thus, for each partition, each producer sends 1000 messages.

(3) Then I run a single consumer to consume all the messages sent from 2 producers.



As the counter indicated, all 2000 messages are consumed by the single consumer.

E. Multiple Concurrent Producers and Consumers

1. Clear all data from the topic
2. Update your Producer code to include a 250 msec sleep after each send of a message to the topic.
3. Run two or three concurrent producers and two concurrent consumers all at the same time.
4. Describe the results.

I experiments with multiple concurrent producers and consumers as following:

(1) Create two consumers (in two threads) that subscribe to the topic “wk3-separate-topic” before producers produce messages. The following screenshot shows that two consumers are waiting for messages.

```
(confluent-exercise) yl6@yml-dataeng-wk3:~/cs510-dataeng-inclass/wk3$ ./consumer.py -f ./.confluent
/librdkafka.config -t wk3-separate-topic --nthread 2
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer1]   Waiting for message or event/error in poll()
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer1]   Waiting for message or event/error in poll()
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer1]   Waiting for message or event/error in poll()
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer1]   Waiting for message or event/error in poll()
```

(2) In another cloud shell, create three consumers (in three threads) that produce 1000 messages with key “producer0”, “producer1”, and “producer2”.

```
(confluent-exercise) yl6@yml-dataeng-wk3:~/cs510-dataeng-inclass/wk3$ ./producer.py -f ./.conflu
ent/librdkafka.config -t wk3-separate-topic --nthread 3
INFO:root:Producing record: producer0   {"EVENT_NO_TRIP": "167589364", "EVENT_NO_STOP": "1675893
69", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "354", "ACT_TIME": "21756", "VELOC
ITY": "", "DIRECTION": "", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.604897", "GPS_LATITUDE":
"45.63653", "GPS_SATELLITES": "9", "GPS_HDOP": "1", "SCHEDULE_DEVIATION": ""}
INFO:root:Producing record: producer1   {"EVENT_NO_TRIP": "167589364", "EVENT_NO_STOP": "1675893
69", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "354", "ACT_TIME": "21756", "VELOC
ITY": "", "DIRECTION": "", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.604897", "GPS_LATITUDE":
"45.63653", "GPS_SATELLITES": "9", "GPS_HDOP": "1", "SCHEDULE_DEVIATION": ""}
INFO:root:Producing record: producer2   {"EVENT_NO_TRIP": "167589364", "EVENT_NO_STOP": "1675893
69", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "354", "ACT_TIME": "21756", "VELOC
ITY": "", "DIRECTION": "", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.604897", "GPS_LATITUDE":
"45.63653", "GPS_SATELLITES": "9", "GPS_HDOP": "1", "SCHEDULE_DEVIATION": ""}
INFO:root:Producing record: producer0   {"EVENT_NO_TRIP": "167589364", "EVENT_NO_STOP": "1675893
69", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "389", "ACT_TIME": "21761", "VELOC
ITY": "7", "DIRECTION": "109", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.60449", "GPS_LATITUDE
": "45.636432", "GPS_SATELLITES": "9", "GPS_HDOP": "0.9", "SCHEDULE_DEVIATION": ""}
```

(3) Then observed that consumers begin to consume messages.

```
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer1]   Waiting for message or event/error in poll()
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer1]   Waiting for message or event/error in poll()
INFO:root:[consumer1]   Consumed record with key b'producer0' and value b'{"EVENT_NO_TRIP": "167589
364", "EVENT_NO_STOP": "167589369", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "354",
 "ACT_TIME": "21756", "VELOCITY": "", "DIRECTION": "", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.
604897", "GPS_LATITUDE": "45.63653", "GPS_SATELLITES": "9", "GPS_HDOP": "1", "SCHEDULE_DEVIATION":
""}',                          and updated total count to 1
INFO:root:[consumer1]   Consumed record with key b'producer0' and value b'{"EVENT_NO_TRIP": "167589
364", "EVENT_NO_STOP": "167589369", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "389",
 "ACT_TIME": "21761", "VELOCITY": "7", "DIRECTION": "109", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-
122.60449", "GPS_LATITUDE": "45.636432", "GPS_SATELLITES": "9", "GPS_HDOP": "0.9", "SCHEDULE_DEVIAT
ION": ""}',                            and updated total count to 2
INFO:root:[consumer1]   Consumed record with key b'producer0' and value b'{"EVENT_NO_TRIP": "167589
364", "EVENT_NO_STOP": "167589369", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "442",
 "ACT_TIME": "21766", "VELOCITY": "10", "DIRECTION": "85", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-
122.60381", "GPS_LATITUDE": "45.636475", "GPS_SATELLITES": "9", "GPS_HDOP": "0.9", "SCHEDULE_DEVIAT
ION": ""}',                            and updated total count to 3
INFO:root:[consumer1]   Consumed record with key b'producer0' and value b'{"EVENT_NO_TRIP": "167589
364", "EVENT_NO_STOP": "167589369", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "509",
 "ACT_TIME": "21771", "VELOCITY": "13", "DIRECTION": "88", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-
122.602948", "GPS_LATITUDE": "45.636493", "GPS_SATELLITES": "9", "GPS_HDOP": "0.9", "SCHEDULE_DEVIA
TION": ""}',                            and updated total count to 4
INFO:root:[consumer0]   Consumed record with key b'producer1' and value b'{"EVENT_NO_TRIP": "167589
364", "EVENT_NO_STOP": "167589369", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "354",
```

(4) I also recorded **the beginning offset of partition 1 is at 9453, and partition 5 is at 40416** (will be used in the rest of the context).

```
INFO:root:Produced record to topic wk3-separate-topic partition [1] @ offset 9453
INFO:root:Produced record to topic wk3-separate-topic partition [1] @ offset 9454
INFO:root:Produced record to topic wk3-separate-topic partition [1] @ offset 9455
INFO:root:Producing record: producer0   {"EVENT_NO_TRIP": "167589364", "EVENT_NO_STOP": "1675893
69", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "509", "ACT_TIME": "21771", "VELOC
ITY": "13", "DIRECTION": "88", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.602948", "GPS_LATITUD
E": "45.636493", "GPS_SATELLITES": "9", "GPS_HDOP": "0.9", "SCHEDULE_DEVIATION": ""}
INFO:root:Producing record: producer2   {"EVENT_NO_TRIP": "167589364", "EVENT_NO_STOP": "1675893
69", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "578", "ACT_TIME": "21776", "VELOC
ITY": "13", "DIRECTION": "87", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.602053", "GPS_LATITUD
E": "45.636525", "GPS_SATELLITES": "9", "GPS_HDOP": "0.9", "SCHEDULE_DEVIATION": ""}
INFO:root:Produced record to topic wk3-separate-topic partition [5] @ offset 40416
INFO:root:Produced record to topic wk3-separate-topic partition [5] @ offset 40417
INFO:root:Produced record to topic wk3-separate-topic partition [5] @ offset 40418
INFO:root:Produced record to topic wk3-separate-topic partition [5] @ offset 40419
INFO:root:Producing record: producer1   {"EVENT_NO_TRIP": "167589364", "EVENT_NO_STOP": "1675893
69", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "578", "ACT_TIME": "21776", "VELOC
ITY": "13", "DIRECTION": "87", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.602053", "GPS_LATITUD
```

(5) After all 3000 messages are produced, **the offset of partition 1 is at 10452, and the partition 5 is at 42415**.

```
OCITY": "9", "DIRECTION": "205", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.679773", "GPS_LATIT
UDE": "45.514747", "GPS_SATELLITES": "8", "GPS_HDOP": "1.1", "SCHEDULE_DEVIATION": "-1249"}
INFO:root:Produced record to topic wk3-separate-topic partition [5] @ offset 42414
INFO:root:Produced record to topic wk3-separate-topic partition [1] @ offset 10452
INFO:root:Produced record to topic wk3-separate-topic partition [5] @ offset 42415
(confluent-exercise) yl6@yml-dataeng-wk3:~/cs510-dataeng-inclass/wk3$
```

So for partition 1: 10452 - 9453 + 1 = 1000
Partition 5: 42415 - 40416 + 1 = 2000

(6) Then I observed that **“consumer 1” totally consumed 1000 messages,** and **“consumer 0” totally consumed 2000 messages** as the screenshot below shows.

```
INFO:root:[consumer1]   Consumed record with key b'producer0' and value b'{"EVENT_NO_TRIP": "167589
389", "EVENT_NO_STOP": "167589399", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "81607
", "ACT_TIME": "29171", "VELOCITY": "9", "DIRECTION": "205", "RADIO_QUALITY": "", "GPS_LONGITUDE":
"-122.679773", "GPS_LATITUDE": "45.514747", "GPS_SATELLITES": "8", "GPS_HDOP": "1.1", "SCHEDULE_DEV
IATION": "-1249"}',                         and updated total count to 1000
INFO:root:[consumer0]   Consumed record with key b'producer1' and value b'{"EVENT_NO_TRIP": "167589
389", "EVENT_NO_STOP": "167589399", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "81607
", "ACT_TIME": "29171", "VELOCITY": "9", "DIRECTION": "205", "RADIO_QUALITY": "", "GPS_LONGITUDE":
"-122.679773", "GPS_LATITUDE": "45.514747", "GPS_SATELLITES": "8", "GPS_HDOP": "1.1", "SCHEDULE_DEV
IATION": "-1249"}',                         and updated total count to 1999
INFO:root:[consumer0]   Consumed record with key b'producer2' and value b'{"EVENT_NO_TRIP": "167589
389", "EVENT_NO_STOP": "167589399", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "81607
", "ACT_TIME": "29171", "VELOCITY": "9", "DIRECTION": "205", "RADIO_QUALITY": "", "GPS_LONGITUDE":
"-122.679773", "GPS_LATITUDE": "45.514747", "GPS_SATELLITES": "8", "GPS_HDOP": "1.1", "SCHEDULE_DEV
IATION": "-1249"}',                         and updated total count to 2000
INFO:root:[consumer1]   Waiting for message or event/error in poll()
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer1]   Waiting for message or event/error in poll()
INFO:root:[consumer0]   Waiting for message or event/error in poll()
```

Is that a coincidence? Then I repeated the above 6 steps multiple times, all of them gave me the same result. Thus, I conjecture that “consumer 0” consumed messages from partition 2; and “consumer 1” consumed messages from partition 1.

F. Varying Keys

1. Clear all data from the topic

So far you have kept the “key” value constant for each record sent on a topic. But keys can be very useful to choose specific records from a stream.

2. Update your producer code to choose a random number between 1 and 5 for each record’s key.
3. Modify your consumer to consume only records with a specific key (or subset of keys).
4. Attempt to consume records with a key that does not exist. E.g., consume records with key value of “100”. Describe the results
5. Can you create a consumer that only consumes specific keys? If you run this consumer multiple times with varying keys then does it allow you to consume messages out of order while maintaining order within each key?

There are two places to modify the program.
In consumer.py: I assign consumer a particular partition so that consumer only consumes messages in that partition.

```
def consume_msg(consumer, topic, consumer_name, partition=-1):
    # Subscribe to topic
    consumer.subscribe([topic])

    if partition > 0:
        consumer.assign([TopicPartition(topic, partition)])
```

In producer.py: I make the producer to send message to partition such that partition id is same as record key, so consumer can consume such a partition that has the same value with key.

```
        else:
            # If random_key is enabled, in order to let consumer consume message
            # with specific key, we manually assign partition that is exactly
            # the key value
            producer.produce(
                topic, key=str(record_key), value=record_value, on_delivery=acked,
                partition=record_key)
```

I experimented with running one producer that produces 1000 messages with keys randomly assigned to each record between 1 and 5. Then I ran a consumer that only

consumed the specified key passed by argument. The following screenshot shows the number of consumed messages with key = 1, 2, 3, 4, and 5.

Key = 1, 215 messages.

```
(confluent-exercise) yl6@yml-dataeng-wk3:~/cs510-dataeng-inclass/wk3$ ./consumer.py -f ./.confluent
/librdkafka.config -t wk3-separate-topic --nthread 1 --random --key 1
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer0]   Consumed record with key b'1' and value b'{"EVENT_NO_TRIP": "167589364", "E
```

```
7503", "GPS_LATITUDE": "45.519027", "GPS_SATELLITES": "9", "GPS_HDOP": "1", "SCHEDULE_DEVIATION": "
-1239"}',                          and updated total count to 214
INFO:root:[consumer0]   Consumed record with key b'1' and value b'{"EVENT_NO_TRIP": "167589389", "E
VENT_NO_STOP": "167589398", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "81263", "ACT_
TIME": "29101", "VELOCITY": "4", "DIRECTION": "198", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.67
8215", "GPS_LATITUDE": "45.517635", "GPS_SATELLITES": "8", "GPS_HDOP": "1.1", "SCHEDULE_DEVIATION":
 "-1248"}',                         and updated total count to 215
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer0]   Waiting for message or event/error in poll()
```

Key = 2, 187 messages

```
(confluent-exercise) yl6@yml-dataeng-wk3:~/cs510-dataeng-inclass/wk3$ ./consumer.py -f ./.confluent
/librdkafka.config -t wk3-separate-topic --nthread 1 --random --key 2
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer0]   Consumed record with key b'2' and value b'{"EVENT_NO_TRIP": "167589364", "E
```

```
INFO:root:[consumer0]   Consumed record with key b'2' and value b'{"EVENT_NO_TRIP": "167589389", "E
VENT_NO_STOP": "167589399", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "81514", "ACT_
TIME": "29161", "VELOCITY": "9", "DIRECTION": "199", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.67
93", "GPS_LATITUDE": "45.51552", "GPS_SATELLITES": "9", "GPS_HDOP": "2", "SCHEDULE_DEVIATION": "-12
40"}',                        and updated total count to 186
INFO:root:[consumer0]   Consumed record with key b'2' and value b'{"EVENT_NO_TRIP": "167589389", "E
VENT_NO_STOP": "167589399", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "81562", "ACT_
TIME": "29166", "VELOCITY": "9", "DIRECTION": "201", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.67
9523", "GPS_LATITUDE": "45.515118", "GPS_SATELLITES": "8", "GPS_HDOP": "1.4", "SCHEDULE_DEVIATION":
 "-1245"}',                         and updated total count to 187
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer0]   Waiting for message or event/error in poll()
```

Key = 3, 200 messages

```
(confluent-exercise) yl6@yml-dataeng-wk3:~/cs510-dataeng-inclass/wk3$ ./consumer.py -f ./.confluent
/librdkafka.config -t wk3-separate-topic --nthread 1 --random --key 3
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer0]   Consumed record with key b'3' and value b'{"EVENT_NO_TRIP": "167589364", "E
```

```
1249"}',                           and updated total count to 198
INFO:root:[consumer0]   Consumed record with key b'3' and value b'{"EVENT_NO_TRIP": "167589389", "E
VENT_NO_STOP": "167589398", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "81343", "ACT_
TIME": "29116", "VELOCITY": "5", "DIRECTION": "199", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.67
8575", "GPS_LATITUDE": "45.516958", "GPS_SATELLITES": "10", "GPS_HDOP": "0.8", "SCHEDULE_DEVIATION"
: "-1249"}',                        and updated total count to 199
INFO:root:[consumer0]   Consumed record with key b'3' and value b'{"EVENT_NO_TRIP": "167589389", "E
VENT_NO_STOP": "167589399", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "81607", "ACT_
TIME": "29171", "VELOCITY": "9", "DIRECTION": "205", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.67
9773", "GPS_LATITUDE": "45.514747", "GPS_SATELLITES": "8", "GPS_HDOP": "1.1", "SCHEDULE_DEVIATION":
 "-1249"}',                         and updated total count to 200
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer0]   Waiting for message or event/error in poll()
```

Key = 4, 192 messages

```
(confluent-exercise) yl6@yml-dataeng-wk3:~/cs510-dataeng-inclass/wk3$ ./consumer.py -f ./.confluent
/librdkafka.config -t wk3-separate-topic --nthread 1 --random --key 4
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer0]   Consumed record with key b'4' and value b'{"EVENT_NO_TRIP": "167589389", "E
VENT_NO_STOP": "167589397", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "80938", "ACT_
TIME": "29011", "VELOCITY": "4", "DIRECTION": "289", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.67
6425", "GPS_LATITUDE": "45.520005", "GPS_SATELLITES": "9", "GPS_HDOP": "0.9", "SCHEDULE_DEVIATION":
 "-1277"}',                          and updated total count to 191
INFO:root:[consumer0]   Consumed record with key b'4' and value b'{"EVENT_NO_TRIP": "167589389", "E
VENT_NO_STOP": "167589397", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "80946", "ACT_
TIME": "29016", "VELOCITY": "1", "DIRECTION": "289", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.67
6543", "GPS_LATITUDE": "45.520033", "GPS_SATELLITES": "9", "GPS_HDOP": "1.5", "SCHEDULE_DEVIATION":
 "-1273"}',                          and updated total count to 192
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer0]   Waiting for message or event/error in poll()
```

Key = 5, 206 messages

```
(confluent-exercise) yl6@yml-dataeng-wk3:~/cs510-dataeng-inclass/wk3$ ./consumer.py -f ./.confluent
/librdkafka.config -t wk3-separate-topic --nthread 1 --random --key 5
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer0]   Consumed record with key b'5' and value b'{"EVENT_NO_TRIP": "167589364", "E
INFO:root:[consumer0]   Consumed record with key b'5' and value b'{"EVENT_NO_TRIP": "167589389", "E
VENT_NO_STOP": "167589398", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "81413", "ACT_
TIME": "29126", "VELOCITY": "6", "DIRECTION": "203", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.67
894", "GPS_LATITUDE": "45.516377", "GPS_SATELLITES": "9", "GPS_HDOP": "0.9", "SCHEDULE_DEVIATION":
"-1254"}',                           and updated total count to 205
INFO:root:[consumer0]   Consumed record with key b'5' and value b'{"EVENT_NO_TRIP": "167589389", "E
VENT_NO_STOP": "167589399", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "81466", "ACT_
TIME": "29156", "VELOCITY": "1", "DIRECTION": "194", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.67
91", "GPS_LATITUDE": "45.515928", "GPS_SATELLITES": "9", "GPS_HDOP": "1.5", "SCHEDULE_DEVIATION": "
-1235"}',                           and updated total count to 206
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer0]   Waiting for message or event/error in poll()
```

215 + 187 + 200 + 192 + 206 = 1000 messages. There are too much message log during running consumer 5 times , I've inspected that for each running, all the key values are the same, also equal to the specified key.

G. Producer Flush

The provided tutorial producer program calls "producer.flush()" at the very end, and presumably your new producer also calls producer.flush().

1. What does Producer.flush() do?

   Flush() is blocked until all the messages in the producer message queue are delivered to broker so that the callback function passed into produce() (for each produced record) will be called.

2. What happens if you do not call producer.flush()?

If I don’t call flush, in the same time I don’t poll, then callback function will never be called.

3. What happens if you call producer.flush() after sending each record?

The callback function passed into produce() will be called one by one. Since each time the producer produces a record, flush() will be blocked until it is delivered to the broker, then the callback function passed in to corresponding produce() will be called.

4. What happens if you wait for 2 seconds after every 5th record send, and you call flush only after every 15 record sends, and you have a consumer running concurrently? Specifically, does the consumer **receive** each message immediately? only after a flush? Something else?

After I implement this functionality, I run one consumer and one producer concurrently. The following screenshots show the pattern of output:

For producer, for each 15 produced records, the callback function returns. For consumer, for each 5 message consumed, it has 1 - 3 consumer.poll() that returns None, which means consumer is waiting for message to be sent to broker.

Thus, I conclude that consumer POLL (I cannot use the word “receive:”) each message immediately after each message reach the broker, which is independent of the calling of flush().

Producer:

```
INFO:root:Producing record: producer0   {"EVENT_NO_TRIP": "167589389", "EVENT_NO_STOP": "1675
89398", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "81263", "ACT_TIME": "29101"
, "VELOCITY": "4", "DIRECTION": "198", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.678215", "
GPS_LATITUDE": "45.517635", "GPS_SATELLITES": "8", "GPS_HDOP": "1.1", "SCHEDULE_DEVIATION": "
-1248"}
INFO:root:Produced record to topic wk3-separate-topic partition [1] @ offset 20137
INFO:root:Produced record to topic wk3-separate-topic partition [1] @ offset 20138
INFO:root:Produced record to topic wk3-separate-topic partition [1] @ offset 20139
INFO:root:Produced record to topic wk3-separate-topic partition [1] @ offset 20140
INFO:root:Produced record to topic wk3-separate-topic partition [1] @ offset 20141
INFO:root:Produced record to topic wk3-separate-topic partition [1] @ offset 20142
INFO:root:Produced record to topic wk3-separate-topic partition [1] @ offset 20143
INFO:root:Produced record to topic wk3-separate-topic partition [1] @ offset 20144
INFO:root:Produced record to topic wk3-separate-topic partition [1] @ offset 20145
INFO:root:Produced record to topic wk3-separate-topic partition [1] @ offset 20146
INFO:root:Produced record to topic wk3-separate-topic partition [1] @ offset 20147
INFO:root:Produced record to topic wk3-separate-topic partition [1] @ offset 20148
INFO:root:Produced record to topic wk3-separate-topic partition [1] @ offset 20149
INFO:root:Produced record to topic wk3-separate-topic partition [1] @ offset 20150
INFO:root:Produced record to topic wk3-separate-topic partition [1] @ offset 20151
INFO:root:Producing record: producer0   {"EVENT_NO_TRIP": "167589389", "EVENT_NO_STOP": "1675
89398", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "81283", "ACT_TIME": "29106"
, "VELOCITY": "4", "DIRECTION": "201", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.678307", "
GPS_LATITUDE": "45.51747", "GPS_SATELLITES": "8", "GPS_HDOP": "1", "SCHEDULE_DEVIATION": "-12
47"}
INFO:root:Producing record: producer0   {"EVENT_NO_TRIP": "167589389", "EVENT_NO_STOP": "1675
89398", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "81316", "ACT_TIME": "29111"
, "VELOCITY": "6", "DIRECTION": "201", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.67846", "G
PS_LATITUDE": "45.517192", "GPS_SATELLITES": "9", "GPS_HDOP": "1", "SCHEDULE_DEVIATION": "-12
49"}
INFO:root:Producing record: producer0   {"EVENT_NO_TRIP": "167589389", "EVENT_NO_STOP": "1675
89398", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "81343", "ACT_TIME": "29116"
, "VELOCITY": "5", "DIRECTION": "199", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.678575", "
GPS_LATITUDE": "45.516958", "GPS_SATELLITES": "10", "GPS_HDOP": "0.8", "SCHEDULE_DEVIATION":
"-1249"}
INFO:root:Producing record: producer0   {"EVENT_NO_TRIP": "167589389", "EVENT_NO_STOP": "1675
89398", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "81380", "ACT_TIME": "29121"
, "VELOCITY": "7", "DIRECTION": "204", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.678772", "
GPS_LATITUDE": "45.516653", "GPS_SATELLITES": "10", "GPS_HDOP": "1.4", "SCHEDULE_DEVIATION":
"-1252"}
INFO:root:Producing record: producer0   {"EVENT_NO_TRIP": "167589389", "EVENT_NO_STOP": "1675
89398", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "81413", "ACT_TIME": "29126"
, "VELOCITY": "6", "DIRECTION": "203", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.67894", "G
PS_LATITUDE": "45.516377", "GPS_SATELLITES": "9", "GPS_HDOP": "0.9", "SCHEDULE_DEVIATION": "-
1254"}
INFO:root:Producing record: producer0   {"EVENT_NO_TRIP": "167589389", "EVENT_NO_STOP": "1675
89399", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "81466", "ACT_TIME": "29156"
, "VELOCITY": "1", "DIRECTION": "194", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.6791", "GP
S_LATITUDE": "45.515928", "GPS_SATELLITES": "9", "GPS_HDOP": "1.5", "SCHEDULE_DEVIATION": "-1
235"}
INFO:root:Producing record: producer0   {"EVENT_NO_TRIP": "167589389", "EVENT_NO_STOP": "1675
89399", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "81514", "ACT_TIME": "29161"
, "VELOCITY": "9", "DIRECTION": "199", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122.6793", "GP
S_LATITUDE": "45.51552", "GPS_SATELLITES": "9", "GPS_HDOP": "2", "SCHEDULE_DEVIATION": "-1240
"}
INFO:root:Producing record: producer0   {"EVENT_NO_TRIP": "167589389", "EVENT_NO_STOP": "1675
```

Consumer:

```
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer0]   Consumed record with key b'producer0' and value b'{"EVENT_NO_TRIP": "16758938
9", "EVENT_NO_STOP": "167589396", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "79141", "
ACT_TIME": "28856", "VELOCITY": "17", "DIRECTION": "182", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122
.665408", "GPS_LATITUDE": "45.526612", "GPS_SATELLITES": "7", "GPS_HDOP": "1.4", "SCHEDULE_DEVIATION"
: "-1227"}',                          and updated total count to 947
INFO:root:[consumer0]   Consumed record with key b'producer0' and value b'{"EVENT_NO_TRIP": "16758938
9", "EVENT_NO_STOP": "167589396", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "79234", "
ACT_TIME": "28861", "VELOCITY": "18", "DIRECTION": "182", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122
.665452", "GPS_LATITUDE": "45.525763", "GPS_SATELLITES": "7", "GPS_HDOP": "1.4", "SCHEDULE_DEVIATION"
: "-1229"}',                          and updated total count to 948
INFO:root:[consumer0]   Consumed record with key b'producer0' and value b'{"EVENT_NO_TRIP": "16758938
9", "EVENT_NO_STOP": "167589396", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "79325", "
ACT_TIME": "28866", "VELOCITY": "18", "DIRECTION": "185", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122
.665557", "GPS_LATITUDE": "45.524945", "GPS_SATELLITES": "7", "GPS_HDOP": "1.4", "SCHEDULE_DEVIATION"
: "-1231"}',                          and updated total count to 949
INFO:root:[consumer0]   Consumed record with key b'producer0' and value b'{"EVENT_NO_TRIP": "16758938
9", "EVENT_NO_STOP": "167589396", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "79418", "
ACT_TIME": "28871", "VELOCITY": "18", "DIRECTION": "189", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122
.665743", "GPS_LATITUDE": "45.524108", "GPS_SATELLITES": "7", "GPS_HDOP": "1.4", "SCHEDULE_DEVIATION"
: "-1233"}',                          and updated total count to 950
INFO:root:[consumer0]   Consumed record with key b'producer0' and value b'{"EVENT_NO_TRIP": "16758938
9", "EVENT_NO_STOP": "167589396", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "79512", "
ACT_TIME": "28876", "VELOCITY": "18", "DIRECTION": "183", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122
.665798", "GPS_LATITUDE": "45.523253", "GPS_SATELLITES": "7", "GPS_HDOP": "1.4", "SCHEDULE_DEVIATION"
: "-1235"}',                          and updated total count to 951
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer0]   Waiting for message or event/error in poll()
INFO:root:[consumer0]   Consumed record with key b'producer0' and value b'{"EVENT_NO_TRIP": "16758938
9", "EVENT_NO_STOP": "167589396", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "79603", "
ACT_TIME": "28881", "VELOCITY": "18", "DIRECTION": "182", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122
.665843", "GPS_LATITUDE": "45.52242", "GPS_SATELLITES": "7", "GPS_HDOP": "1.4", "SCHEDULE_DEVIATION":
 "-1237"}',                          and updated total count to 952
INFO:root:[consumer0]   Consumed record with key b'producer0' and value b'{"EVENT_NO_TRIP": "16758938
9", "EVENT_NO_STOP": "167589397", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "79686", "
ACT_TIME": "28886", "VELOCITY": "16", "DIRECTION": "187", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122
.66597", "GPS_LATITUDE": "45.52167", "GPS_SATELLITES": "7", "GPS_HDOP": "1.4", "SCHEDULE_DEVIATION":
"-1241"}',                          and updated total count to 953
INFO:root:[consumer0]   Consumed record with key b'producer0' and value b'{"EVENT_NO_TRIP": "16758938
9", "EVENT_NO_STOP": "167589397", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "79770", "
ACT_TIME": "28891", "VELOCITY": "16", "DIRECTION": "192", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122
.666188", "GPS_LATITUDE": "45.520927", "GPS_SATELLITES": "7", "GPS_HDOP": "1.4", "SCHEDULE_DEVIATION"
: "-1247"}',                          and updated total count to 954
INFO:root:[consumer0]   Consumed record with key b'producer0' and value b'{"EVENT_NO_TRIP": "16758938
9", "EVENT_NO_STOP": "167589397", "OPD_DATE": "10-SEP-20", "VEHICLE_ID": "1776", "METERS": "79850", "
ACT_TIME": "28896", "VELOCITY": "16", "DIRECTION": "194", "RADIO_QUALITY": "", "GPS_LONGITUDE": "-122
.666438", "GPS_LATITUDE": "45.52021", "GPS_SATELLITES": "7", "GPS_HDOP": "1.4", "SCHEDULE_DEVIATION":
 "-1252"}',                          and updated total count to 955
```

H. Consumer Groups

1. Create two consumer groups with one consumer program instance in each group.
2. Run the producer and have it produce all 1000 messages from your sample file.
3. Run each of the consumers and verify that each consumer consumes all of the 50 messages.

4. Create a second consumer within one of the groups so that you now have three consumers total.
5. Rerun the producer and consumers. Verify that each consumer group consumes the full set of messages but that each consumer within a consumer group only consumes a portion of the messages sent to the topic.

I. Kafka Transactions

6. Create a new producer, similar to the previous producer, that uses transactions.
7. The producer should begin a transaction, send 4 records in the transactions, then wait for 2 seconds, then choose True/False randomly with equal probability. If True then finish the transaction successfully with a commit. If False is picked then cancel the transaction.
8. Create a new transaction-aware consumer. The consumer should consume the data. It should also use the Confluent/Kaka transaction API with a “read_committed” isolation level. (I can’t find evidence of other isolation levels).
9. Transaction across multiple topics. Create a second topic and modify your producer to send two records to the first topic and two records to the second topic before randomly committing or canceling the transaction. Modify the consumer to consume from the two queues. Verify that it only consumes committed data and not uncommitted or canceled data.