DataEng S24: PubSub

A. PubSub Tutorial

```
sridhamo@cloudshell:~ (data-engineering-420102)$ gcloud pubsub topics create gpubsub
Created topic [projects/data-engineering-420102/topics/gpubsub].
sridhamo@cloudshell:~ (data-engineering-420102)$ gcloud pubsub subscriptions create my-sub --topic gpubsub
Created subscription [projects/data-engineering-420102/subscriptions/my-sub].
```

```
sridhamo@cloudshell:~ (data-engineering-420102)$ python publisher.py
10931799994188555
10931903718253005
10931718122652412
10931778840629830
10932121510296402
10932010880547415
9472712957334475
9472685461137503
10931733395062903
Published messages to projects/data-engineering-420102/topics/gpubsub.
```

```
(env) sridhamo@pubsub-vm:~$ python reciever.py
Listening for messages on projects/data-engineering-420102/subscriptions/my-sub..

Received Message {
  data: b'Message number 3'
  ordering_key: ''
  attributes: {}
}.
Received Message {
  data: b'Message number 1'
  ordering_key: ''
  attributes: {}
}.
Received Message {
  data: b'Message number 2'
  ordering_key: ''
  attributes: {}
}.
Received Message {
  data: b'Message number 8'
  ordering_key: ''
  attributes: {}
}.
Received Message {
  data: b'Message number 6'
  ordering_key: ''
  attributes: {}
}.
Received Message {
  data: b'Message number 7'
  ordering_key: ''
  attributes: {}
}.
Received Message {
  data: b'Message number 4'
  ordering_key: ''
  attributes: {}
}.
Received Message {
  data: b'Message number 5'
  ordering_key: ''
  attributes: {}
}.
Received Message {
  data: b'Message number 9'
  ordering_key: ''
  attributes: {}
```

B. Create Sample Data

Publisher

```
sridhamo@cloudshell:~ (data-engineering-420102)$ python publisherb.py
The total number of messages sent:3936
```

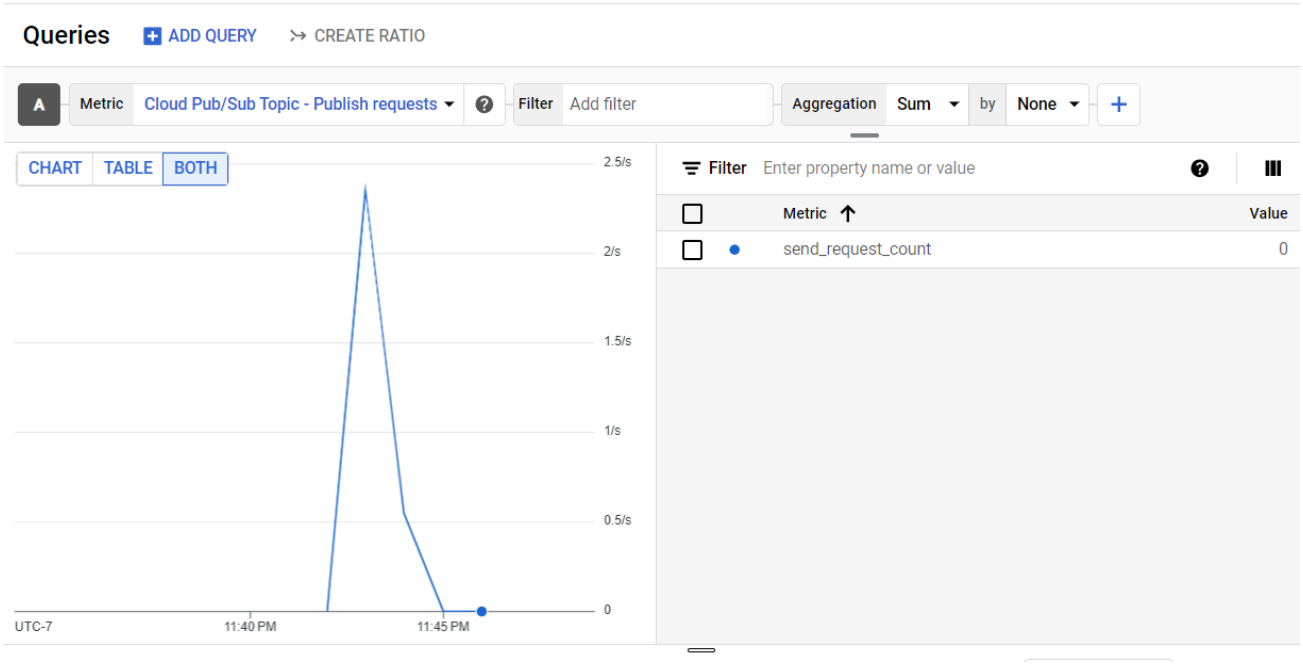
Receiver

```
(env) sridhamo@pubsub-vm:~$ python recieverb.py
Listening for messages on projects/data-engineering-420102/subscriptions/my-sub..

Received Message {
  data: b'{"EVENT_NO_TRIP": 221554918, "EVENT_NO_STOP": 2215...'
  ordering_key: ''
  attributes: {}
}.
Received Message {
  data: b'{"EVENT_NO_TRIP": 221554918, "EVENT_NO_STOP": 2215...'
  ordering_key: ''
  attributes: {}
}.
Received Message {
  data: b'{"EVENT_NO_TRIP": 221554918, "EVENT_NO_STOP": 2215...'
  ordering_key: ''
  attributes: {}
}.
Received Message {
  data: b'{"EVENT_NO_TRIP": 221554918, "EVENT_NO_STOP": 2215...'
  ordering_key: ''
  attributes: {}
}.
Received Message {
  data: b'{"EVENT_NO_TRIP": 221554918, "EVENT_NO_STOP": 2215...'
  ordering_key: ''
  attributes: {}
}.
Received Message {
  data: b'{"EVENT_NO_TRIP": 221554918, "EVENT_NO_STOP": 2215...'
  ordering_key: ''
  attributes: {}
}.
```

C.PubSub Monitoring

Publish requests chart



D. PubSub Storage

1. What happens if you run your receiver multiple times while only running the publisher once?

When the publisher runs once and the receiver runs multiple times, each instance of the receiver will attempt to read messages from the subscription. If a message has not been acknowledged by any of the receiver instances, it will be available for each receiver to read. If a receiver instance acknowledges a message, it is considered processed and will not be available for any other receiver instance to read. Therefore, if one receiver instance acknowledges a message, subsequent runs of the receiver will not be able to read the message.

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

Before the consumer runs, the data is typically stored in the message queue or topic within the PubSub system.The data is immediately written to storage by the forwarder in the PubSub system. The forwarder first writes the message to multiple clusters (N clusters, where N is an odd number) and considers the message persisted when it has been written to at least half of the clusters. These clusters are stored in the nearest cloud storage region.

3. Is there a way to determine how much data PubSub is storing for your topic? Do the PubSub monitoring tools help with this?

We can use PubSub monitoring to find out how much data is being stored for your topic. By selecting the topic and checking metrics like Average Message Size and Publish Message Count we can determine the amount of data being used by our topic.

E. Multiple Publishers

1. Run two versions of your publisher concurrently, have each of them send all of your sample records. When finished, run your receiver once. Describe the results.

Two versions of the publisher were run concurrently with different data . On executing the receiver once it can be observed that the receiver was able to read data from both the publishers.

F. Multiple Concurrent Publishers and Receivers

Describe the results.

Two publishers and two receivers were running concurrently, and their results were saved in different files. Upon checking the files, it was noticed that each receiver had a different total number of records. One receiver processed 984 lines, while the other processed 1870 lines. Both receivers processed fewer records than the total number published by the publishers. Additionally, both receiver files contained records from both publishers.

F. Multiple Subscriptions

Describe the results.

Two sets of publishers and subscribers were concurrently executed, with each subscriber assigned a different subscription ID. The data received by each subscriber was saved to individual files. One of the receivers processed 5946 lines, while the other processed 5929 lines. The total number of records processed by each receiver was almost equal to the total number of records published by the publisher. This indicated that both receivers were able to process a large portion of the messages sent by the publishers without interference from each other.