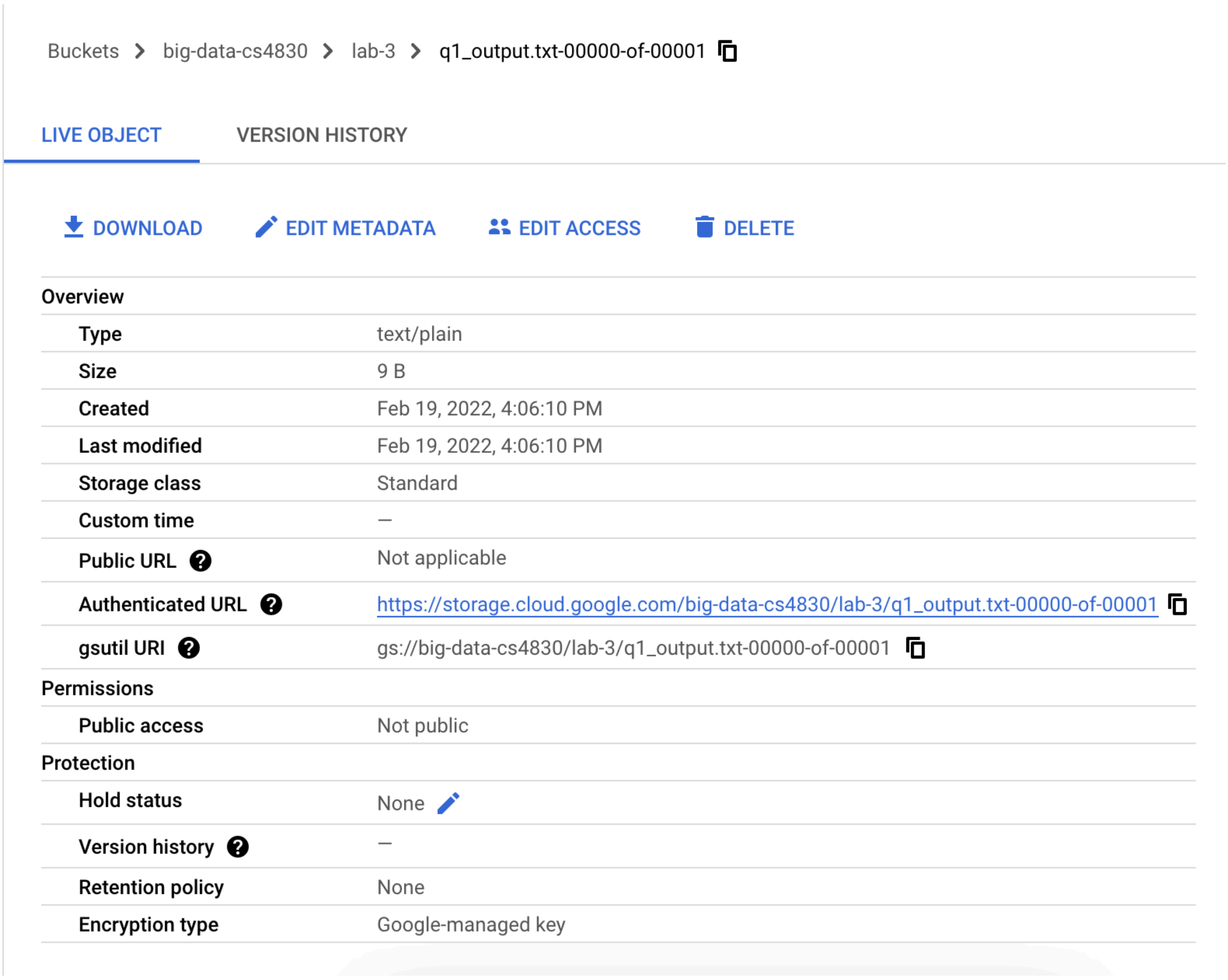
Big Data Lab-3 Assignment

Vasudev Gupta (ME18B182)

PLEASE FOLLOW THE DETAILED INSTRUCTIONS IN THE README.MD FOR RUNNING THE CODE

1) By running **python3 me18b182.py**, we get **q1_output.txt-00000-of-00001** in the GCS bucket.

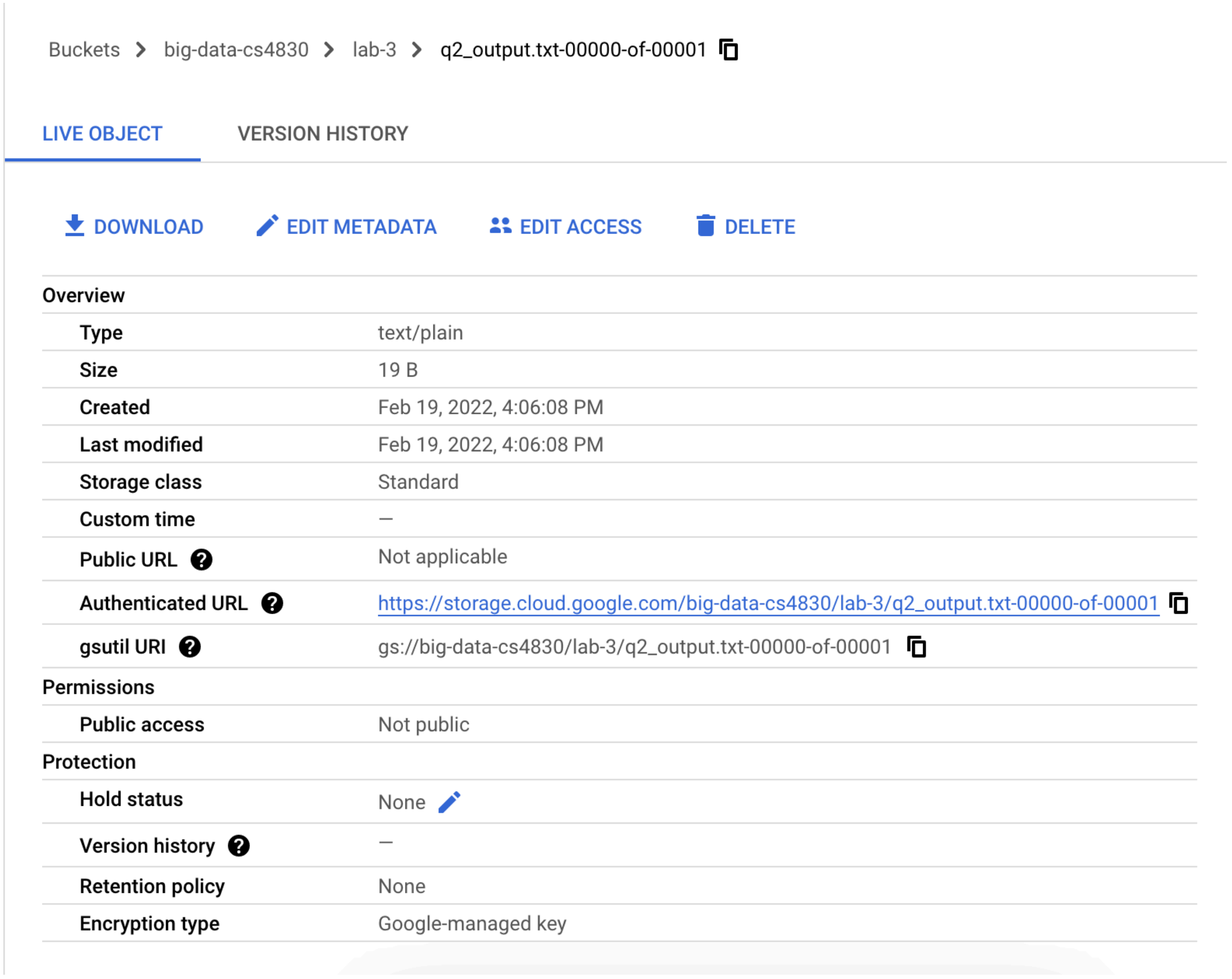
Output: 51791868


Buckets > big-data-cs4830 > lab-3 > q1_output.txt-00000-of-00001
LIVE OBJECT
VERSION HISTORY
DOWNLOAD
EDIT METADATA
EDIT ACCESS
DELETE

Overview	
Type	text/plain
Size	9 B
Created	Feb 19, 2022, 4:06:10 PM
Last modified	Feb 19, 2022, 4:06:10 PM
Storage class	Standard
Custom time	—
Public URL	Not applicable
Authenticated URL	https://storage.cloud.google.com/big-data-cs4830/lab-3/q1_output.txt-00000-of-00001
gsutil URI	gs://big-data-cs4830/lab-3/q1_output.txt-00000-of-00001
Permissions	
Public access	Not public
Protection	
Hold status	None
Version history	—
Retention policy	None
Encryption type	Google-managed key

2) By running the command mentioned in above question, we also generate **q2_output.txt-00000-of-00001** in the same GCS bucket

Output: 1.9372018016419104

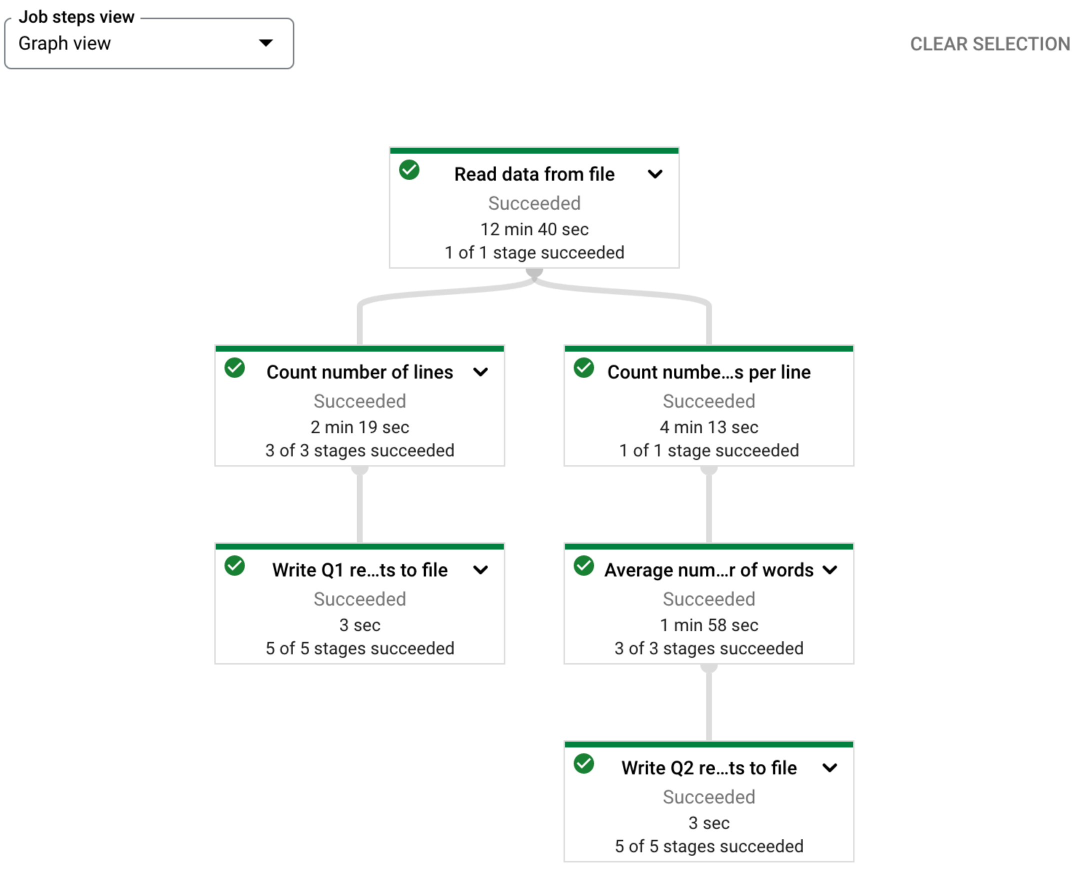

Buckets > big-data-cs4830 > lab-3 > q2_output.txt-00000-of-00001

LIVE OBJECT | VERSION HISTORY

DOWNLOAD | EDIT METADATA | EDIT ACCESS | DELETE

Overview	
Type	text/plain
Size	19 B
Created	Feb 19, 2022, 4:06:08 PM
Last modified	Feb 19, 2022, 4:06:08 PM
Storage class	Standard
Custom time	—
Public URL	Not applicable
Authenticated URL	https://storage.cloud.google.com/big-data-cs4830/lab-3/q2_output.txt-00000-of-00001
gsutil URI	gs://big-data-cs4830/lab-3/q2_output.txt-00000-of-00001
Permissions	
Public access	Not public
Protection	
Hold status	None
Version history	—
Retention policy	None
Encryption type	Google-managed key

3) Below is the screenshot of the dataflow generate in question-1 & question-2



4) Following is the explanation of the pipelines implemented in above questions:

Pipeline for Question 1

We first read the input file line by line using the beam.io.ReadFromText and generates the Pcollection object. Then we used the beam.combiners.Count.Globally to count the total number of samples in the output of previous step. Finally, count is written to the text file using the beam.io.WriteToText.

Pipeline for Question 2

We first read the input file line by line using the beam.io.ReadFromText and generates the Pcollection object. Then we count the number of words in each line using python function & beam.Map. Further, we use beam.combiners.Mean.Globally for calculating the mean. Finally, the mean is written to the text file using the beam.io.WriteToText.

Issues & Solutions

1) It took me some time to figure out the how beam.Map is exactly working.
2) It took me some time to figure out installing apace beam inside my GCP VM.
3) I was unable to run the dataflow using python 3.9 and had to shift to python 3.8 for running it.

5) In this question, I wrote a google cloud function for triggering the dataflow whenever user pushes text file to the GCS bucket. This dataflow counts the total number of lines and the average number of words per line in the document. Code for running bonus question is available in the **gcf/** directory. Please refer to the **README.md** for detailed instructions for running this question.