

CS4830 - BIG DATA LABORATORY

LAB 3 - ASSIGNMENT 1

VISHAL RISHI MK - CH18B013

1. Dataflow job to count the number of lines in a file:

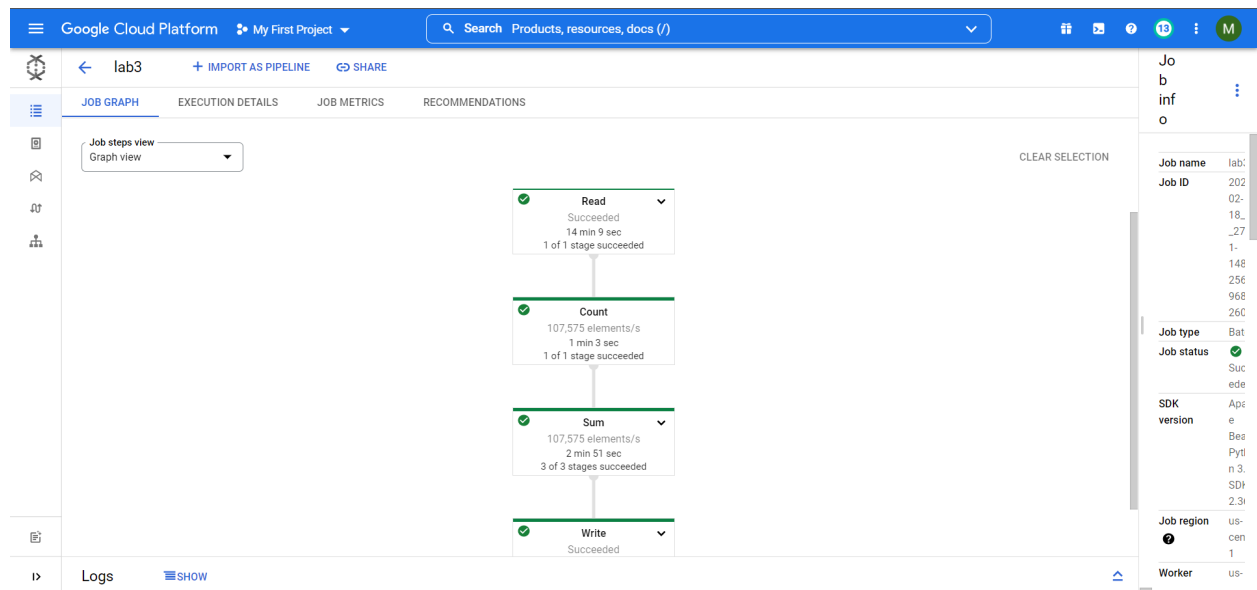
The code is given in the *dataflow_2.py* file. The screenshot of the output file is shown below.

The screenshot displays the Google Cloud Platform interface for a Cloud Storage bucket named 'ch18b013'. The 'OBJECTS' tab is active, showing a list of objects. A single object is visible: 'Output1.txt-00000-of-00001' with a size of 9 B, type 'text/plain', and created on Feb 19, 2020. The interface includes a sidebar with navigation options like 'Browser', 'Monitoring', and 'Settings'. The top bar shows the project name 'My First Project' and a search bar.

Name	Size	Type	Created	Storage class	Last modified	Public access	Version history	Encryption	Retent
Output1.txt-00000-of-00001	9 B	text/plain	Feb 19, 20...	Standard	Feb 19, 20...	Not public	—	Google-managed key	—

The number of lines in the file turns out to be 51791868.

The execution graph for the job is shown below:



2. Dataflow job to compute the average number of words in a line of a file:

The code is given in the `dataflow_3.py` file. The screenshot of the output file is shown below.

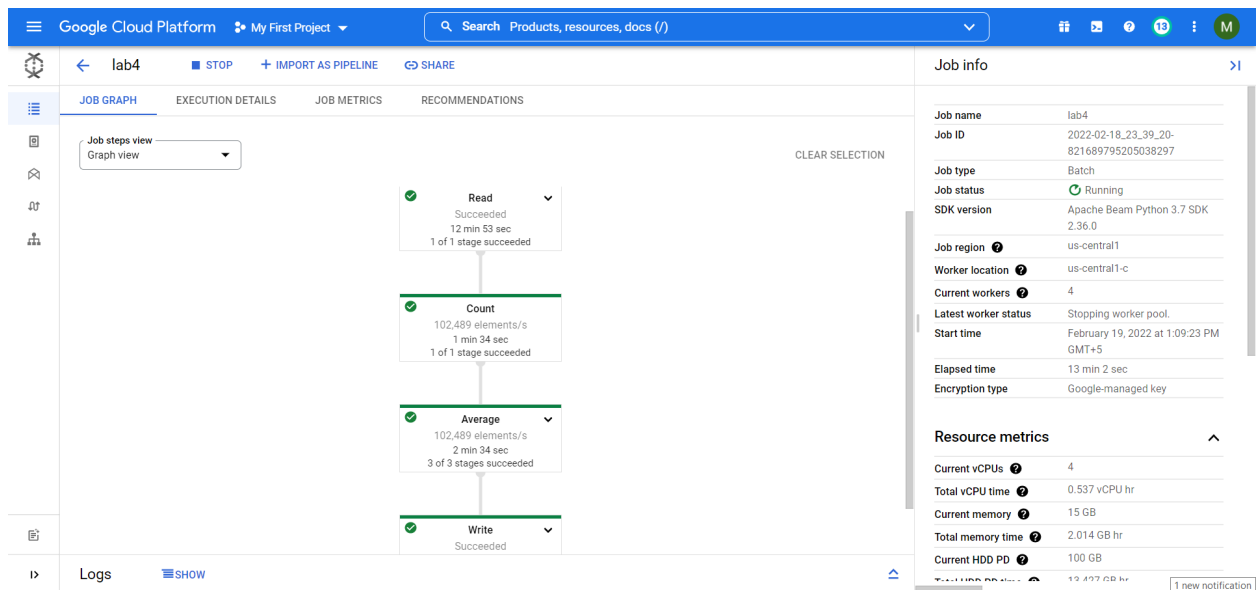
Google Cloud Platform console showing the details of a Cloud Storage bucket named 'ch18b013'. The bucket is located in 'us (multiple regions in United States)' with a 'Standard' storage class, 'Not public' access, and 'None' protection. The 'OBJECTS' tab is selected, showing a list of files. The list includes 'Output1.txt-00000-of-00001' and 'Output2.txt-00000-of-00001', both of size 9 B and 19 B respectively, created on Feb 19, 2020. The table also shows columns for Name, Size, Type, Created, Storage class, Last modified, Public access, Version history, Encryption, and Retention.

Name	Size	Type	Created	Storage class	Last modified	Public access	Version history	Encryption	Retention
Output1.txt-00000-of-00001	9 B	text/plain	Feb 19, 20...	Standard	Feb 19, 20...	Not public	—	Google-managed key	—
Output2.txt-00000-of-00001	19 B	text/plain	Feb 19, 20...	Standard	Feb 19, 20...	Not public	—	Google-managed key	—

1.9996232613197114

The average number of words in a line turns out to be 1.996232613197114.

The execution graph for the job is shown below:



3. The execution graphs for both the dataflow jobs created in the background are already presented.

4.

Pipeline for counting the number of lines:

- From the text file, we read the contents and store it in a Pcollection. Each line of the text file is stored as a string in the Pcollection.
- The Pcollection undergoes a ParDo transformation in which each element (string) of the Pcollection is transformed into count 1. The resulting Pcollection is stored in another variable.
- The new Pcollection undergoes a Combine transformation where we sum all the elements (count 1) and store the sum in a new Pcollection.
- The final Pcollection is written in an output file.

Pipeline for computing the average words in a line:

- From the text file, we read the contents and store it in a Pcollection. Each line of the text file is stored as a string in the Pcollection.
- The Pcollection undergoes a ParDo transformation in which each element (string) of the Pcollection is transformed into a count stating the number of words in the line. The resulting Pcollection is stored in another variable.
- The new Pcollection undergoes a Combine transformation where we compute the mean of all the elements (count) and store the mean in a new Pcollection.
- The final Pcollection is written in an output file.

Issues faced while designing the pipelines:

- While creating the Combine transformation for computing the mean, passing a simple function (like mean()) did not work. Since mean computation is a much more complex transformation than computing the sum, the issue was resolved by using an in-built combiner for calculating the mean (**MeanCombineFn**).
- Apart from the above issue, things were clear after going through the Apache Beam documentation. Particularly, I got to know about the various transformations and the programming model in detail.