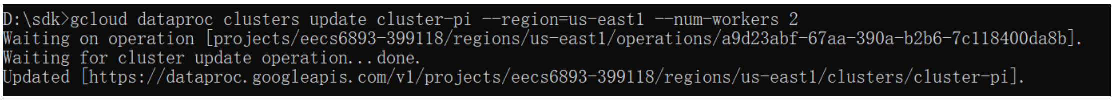
EECS6893 HW0

1. Warm-up exercises

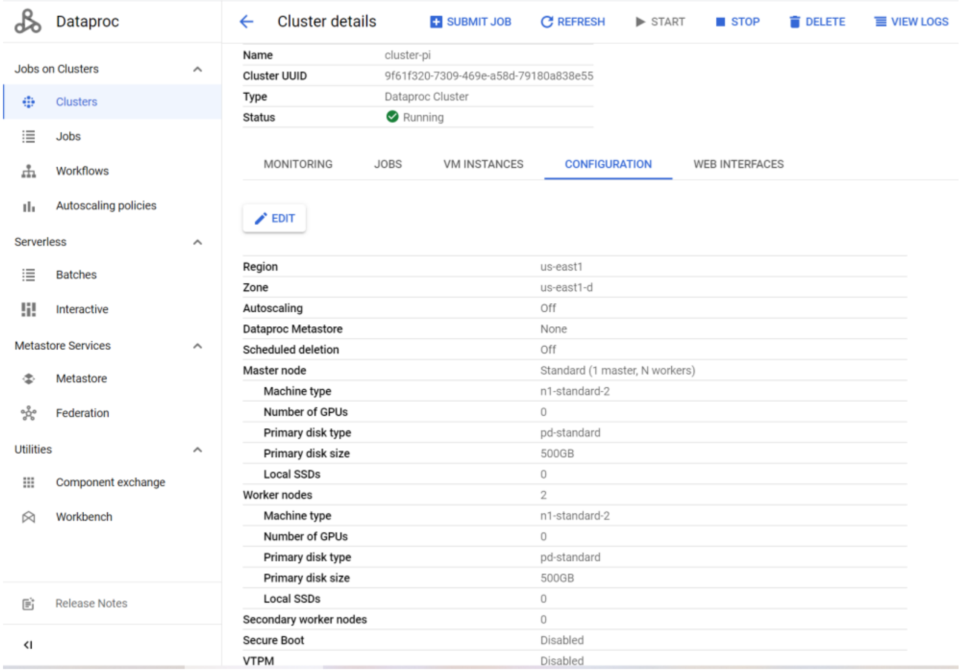
(1) Pi calculation

Create a computing cluster named “cluster-pi” and set the number of workers to 2.

The cluster is created through the API console, while the cluster is updated through the command-line tool.



```
D:\sdk>gcloud dataproc clusters update cluster-pi --region=us-east1 --num-workers 2
Waiting on operation [projects/eecs6893-399118/regions/us-east1/operations/a9d23abf-67aa-390a-b2b6-7c118400da8b].
Waiting for cluster update operation...done.
Updated [https://dataproc.googleapis.com/v1/projects/eecs6893-399118/regions/us-east1/clusters/cluster-pi].
```



Name	cluster-pi
Cluster UUID	9f61f320-7309-469e-a58d-79180a838e55
Type	Dataproc Cluster
Status	✓ Running

Region	us-east1
Zone	us-east1-d
Autoscaling	Off
Dataproc Metastore	None
Scheduled deletion	Off
Master node	Standard (1 master, N workers)
Machine type	n1-standard-2
Number of GPUs	0
Primary disk type	pd-standard
Primary disk size	500GB
Local SSDs	0
Worker nodes	2
Machine type	n1-standard-2
Number of GPUs	0
Primary disk type	pd-standard
Primary disk size	500GB
Local SSDs	0
Secondary worker nodes	0
Secure Boot	Disabled
VTPM	Disabled

n1-standrad-2 machines are used to create a cluster with 2 workers.

The java code of this program is:

```
/*
 * Licensed to the Apache Software Foundation (ASF) under one or more
 * contributor license agreements.  See the NOTICE file distributed with
 * this work for additional information regarding copyright ownership.
 * The ASF licenses this file to You under the Apache License, Version 2.0
```

```java
 * (the "License"); you may not use this file except in compliance with
 * the License.  You may obtain a copy of the License at
 *
 *    http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

package org.apache.spark.examples;

import org.apache.spark.api.java.JavaRDD;
import org.apache.spark.api.java.JavaSparkContext;
import org.apache.spark.sql.SparkSession;

import java.util.ArrayList;
import java.util.List;

/**
 * Computes an approximation to pi
 * Usage: JavaSparkPi [partitions]
 */
public final class JavaSparkPi {

  public static void main(String[] args) throws Exception {
    SparkSession spark = SparkSession
      .builder()
      .appName("JavaSparkPi")
      .getOrCreate();

    JavaSparkContext jsc = new JavaSparkContext(spark.sparkContext());

    int slices = (args.length == 1) ? Integer.parseInt(args[0]) : 2;
    int n = 100000 * slices;
    List<Integer> l = new ArrayList<>(n);
    for (int i = 0; i < n; i++) {
      l.add(i);
    }

    JavaRDD<Integer> dataSet = jsc.parallelize(l, slices);

    int count = dataSet.map(integer -> {
      double x = Math.random() * 2 - 1;
      double y = Math.random() * 2 - 1;
      return (x * x + y * y <= 1) ? 1 : 0;
    }).reduce((integer, integer2) -> integer + integer2);

    System.out.println("Pi is roughly " + 4.0 * count / n);

    spark.stop();
  }
}
```

In this program, the RDD action is:

reduce(func): this action aggregates the elements of the RDD.

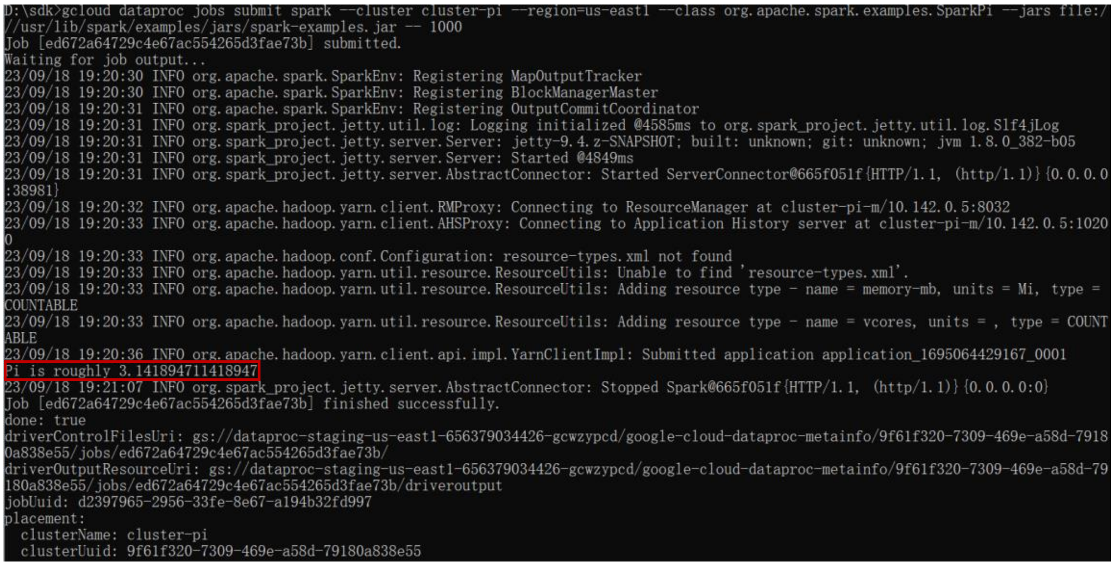
In this program, the ‘reduce’ action counts the number of the generated points located inside the circle.

The RDD transformation is:

map(func): this applys a function to every element of an RDD and return a new result RDD

In this program, the ‘map’ transformation is used to randomly generate points and decide whether the points are inside the circle.

Submit the job and runs the Spark program:

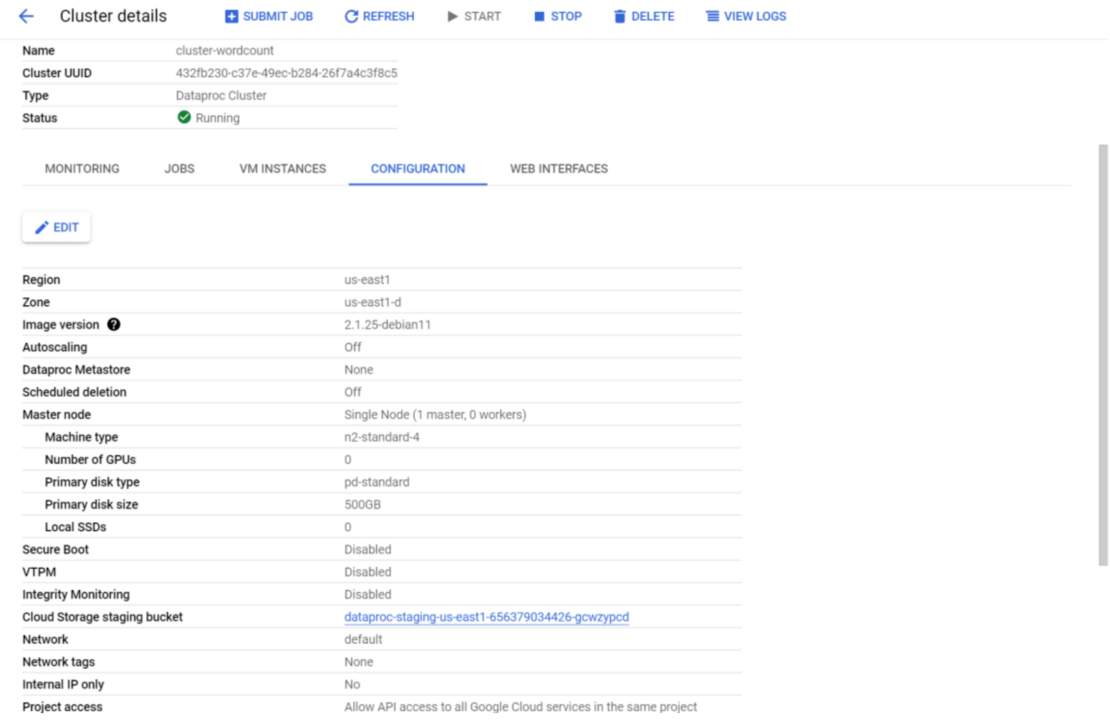


```
D:\sdk>gcloud dataproc jobs submit spark --cluster cluster-pi --region=us-east1 --class org.apache.spark.examples.SparkPi --jars file:/
//usr/lib/spark/examples/jars/spark-examples.jar -- 1000
Job [ed672a64729c4e67ac554265d3fae73b] submitted.
Waiting for job output...
23/09/18 19:20:30 INFO org.apache.spark.SparkEnv: Registering MapOutputTracker
23/09/18 19:20:30 INFO org.apache.spark.SparkEnv: Registering BlockManagerMaster
23/09/18 19:20:31 INFO org.apache.spark.SparkEnv: Registering OutputCommitCoordinator
23/09/18 19:20:31 INFO org.spark_project.jetty.util.log: Logging initialized @4585ms to org.spark_project.jetty.util.log.Slf4jLog
23/09/18 19:20:31 INFO org.spark_project.jetty.server.Server: jetty-9.4.z-SNAPSHOT; built: unknown; git: unknown; jvm 1.8.0_382-b05
23/09/18 19:20:31 INFO org.spark_project.jetty.server.Server: Started @4849ms
23/09/18 19:20:31 INFO org.spark_project.jetty.server.AbstractConnector: Started ServerConnector@665f051f{HTTP/1.1, (http/1.1)}{0.0.0.0
:38981}
23/09/18 19:20:32 INFO org.apache.hadoop.yarn.client.RMProxy: Connecting to ResourceManager at cluster-pi-m/10.142.0.5:8032
23/09/18 19:20:33 INFO org.apache.hadoop.yarn.client.AHSProxy: Connecting to Application History server at cluster-pi-m/10.142.0.5:1020
0
23/09/18 19:20:33 INFO org.apache.hadoop.conf.Configuration: resource-types.xml not found
23/09/18 19:20:33 INFO org.apache.hadoop.yarn.util.resource.ResourceUtils: Unable to find 'resource-types.xml'.
23/09/18 19:20:33 INFO org.apache.hadoop.yarn.util.resource.ResourceUtils: Adding resource type - name = memory-mb, units = Mi, type =
COUNTABLE
23/09/18 19:20:33 INFO org.apache.hadoop.yarn.util.resource.ResourceUtils: Adding resource type - name = vcores, units = , type = COUNT
ABLE
23/09/18 19:20:36 INFO org.apache.hadoop.yarn.client.api.impl.YarnClientImpl: Submitted application application_1695064429167_0001
Pi is roughly 3.141894711418947
23/09/18 19:21:07 INFO org.spark_project.jetty.server.AbstractConnector: Stopped Spark@665f051f{HTTP/1.1, (http/1.1)}{0.0.0.0:0}
Job [ed672a64729c4e67ac554265d3fae73b] finished successfully.
done: true
driverControlFilesUri: gs://dataproc-staging-us-east1-656379034426-gcwzypcd/google-cloud-dataproc-metainfo/9f61f320-7309-469e-a58d-7918
0a838e55/jobs/ed672a64729c4e67ac554265d3fae73b/
driverOutputResourceUri: gs://dataproc-staging-us-east1-656379034426-gcwzypcd/google-cloud-dataproc-metainfo/9f61f320-7309-469e-a58d-79
180a838e55/jobs/ed672a64729c4e67ac554265d3fae73b/driveroutput
jobUuid: d2397965-2956-33fe-8e67-a194b32fd997
placement:
  clusterName: cluster-pi
  clusterUuid: 9f61f320-7309-469e-a58d-79180a838e55
```

The program shows the result:

Pi is roughly 3.141894711418947

(2) Word count



Name	cluster-wordcount
Cluster UUID	432fb230-c37e-49ec-b284-26f7a4c3f8c5
Type	Dataproc Cluster
Status	Running

Region	us-east1
Zone	us-east1-d
Image version	2.1.25-debian11
Autoscaling	Off
Dataproc Metastore	None
Scheduled deletion	Off
Master node	Single Node (1 master, 0 workers)
Machine type	n2-standard-4
Number of GPUs	0
Primary disk type	pd-standard
Primary disk size	500GB
Local SSDs	0
Secure Boot	Disabled
VTPM	Disabled
Integrity Monitoring	Disabled
Cloud Storage staging bucket	dataproc-staging-us-east1-656379034426-gcwzypcd
Network	default
Network tags	None
Internal IP only	No
Project access	Allow API access to all Google Cloud services in the same project

A single-node cluster is used in this part (and problem 3).

The Python code of 'word count' is:

```python
import pyspark
import sys
```

```

if len(sys.argv) != 3:
  raise Exception("Exactly 2 arguments are required: <inputUri> <outputUri>")

inputUri = sys.argv[1]
outputUri = sys.argv[2]

sc = pyspark.SparkContext()
lines = sc.textFile(sys.argv[1])
words = lines.flatMap(lambda line: line.split())
wordCounts = words.map(lambda word: (word, 1)).reduceByKey(lambda count1, count2: count1 + count2)
wordCounts.saveAsTextFile(sys.argv[2])
```

In this program, the RDD transformations are:

1) map(func): Return a new distributed dataset formed by passing each element of the source through a function func.

   In this program, ‘map’ maps each word to a key-value pair, where the word is the key and the value is 1.

2) faltMap(func): Similar to map, but each input item can be mapped to 0 or more output items (so func should return a Seq rather than a single item).

   In this program, ‘flatMap’ split the lines to words.

3) reduceByKey(func, [numPartitions]): When called on a dataset of (K, V) pairs, returns a dataset of (K, V) pairs where the values for each key are aggregated using the given reduce function func, which must be of type (V,V) => V. Like in groupByKey, the number of reduce tasks is configurable through an optional second argument.

   In this program, ‘reduceByKey’ counts the number of occurrences of each word.

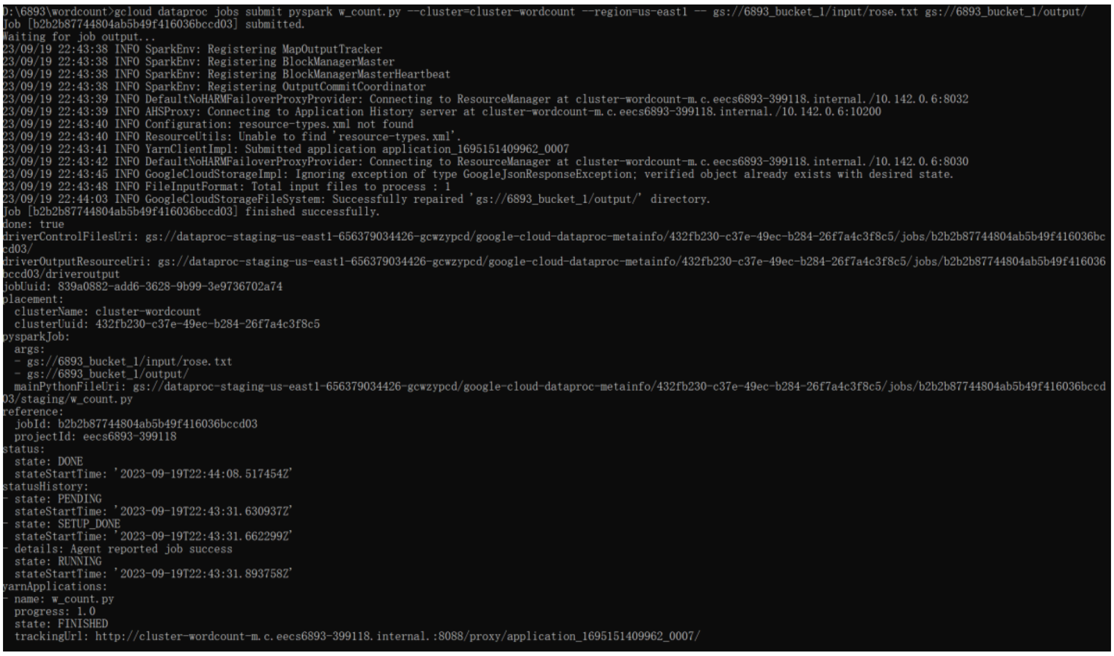
In this program, the RDD actions are:

1) saveAsTextFile(path): Write the elements of the dataset as a text file (or set of text files) in a given directory in the local filesystem, HDFS or any other Hadoop-

supported file system. Spark will call toString on each element to convert it to a line of text in the file.

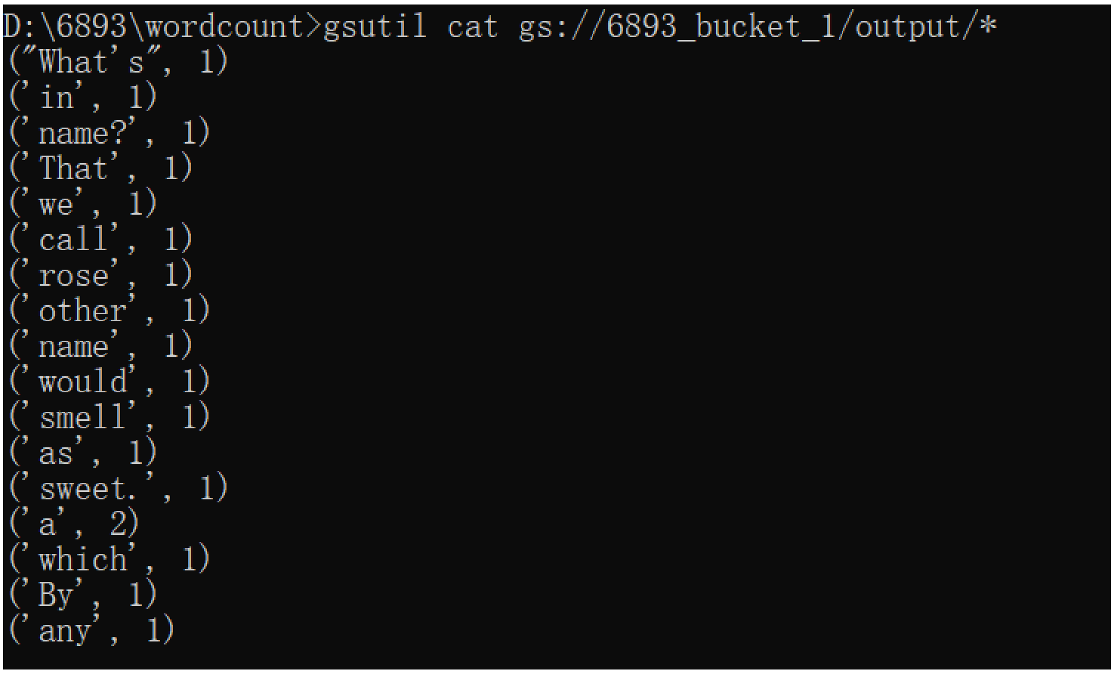
In this program, ‘saveAsTextFile’ saves the words and the number of them in the output address entered in the command.

Submit the job:



```
D:\6893\wordcount>gcloud dataproc jobs submit pyspark w_count.py --cluster=cluster-wordcount --region=us-east1 -- gs://6893_bucket_1/input/rose.txt gs://6893_bucket_1/output/
Job [b2b2b87744804ab5b49f416036bccd03] submitted.
Waiting for job output...
23/09/19 22:43:38 INFO SparkEnv: Registering MapOutputTracker
23/09/19 22:43:38 INFO SparkEnv: Registering BlockManagerMaster
23/09/19 22:43:38 INFO SparkEnv: Registering BlockManagerMasterHeartbeat
23/09/19 22:43:38 INFO SparkEnv: Registering OutputCommitCoordinator
23/09/19 22:43:39 INFO DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at cluster-wordcount-m.c.eecs6893-399118.internal./10.142.0.6:8032
23/09/19 22:43:39 INFO AHSProxy: Connecting to Application History server at cluster-wordcount-m.c.eecs6893-399118.internal./10.142.0.6:10200
23/09/19 22:43:40 INFO Configuration: resource-types.xml not found
23/09/19 22:43:40 INFO ResourceUtils: Unable to find 'resource-types.xml'.
23/09/19 22:43:41 INFO YarnClientImpl: Submitted application application_1695151409962_0007
23/09/19 22:43:42 INFO DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at cluster-wordcount-m.c.eecs6893-399118.internal./10.142.0.6:8030
23/09/19 22:43:45 INFO GoogleCloudStorageImpl: Ignoring exception of type GoogleJsonResponseException; verified object already exists with desired state.
23/09/19 22:43:48 INFO FileInputFormat: Total input files to process : 1
23/09/19 22:44:03 INFO GoogleCloudStorageFileSystem: Successfully repaired 'gs://6893_bucket_1/output/' directory.
Job [b2b2b87744804ab5b49f416036bccd03] finished successfully.
done: true
driverControlFilesUri: gs://dataproc-staging-us-east1-656379034426-gcwzypcd/google-cloud-dataproc-metainfo/432fb230-c37e-49ec-b284-26f7a4c3f8c5/jobs/b2b2b87744804ab5b49f416036bc
cd03/
driverOutputResourceUri: gs://dataproc-staging-us-east1-656379034426-gcwzypcd/google-cloud-dataproc-metainfo/432fb230-c37e-49ec-b284-26f7a4c3f8c5/jobs/b2b2b87744804ab5b49f416036
bccd03/driveroutput
jobUuid: 839a0882-add6-3628-9b99-3e9736702a74
placement:
  clusterName: cluster-wordcount
  clusterUuid: 432fb230-c37e-49ec-b284-26f7a4c3f8c5
pysparkJob:
  args:
  - gs://6893_bucket_1/input/rose.txt
  - gs://6893_bucket_1/output/
  mainPythonFileUri: gs://dataproc-staging-us-east1-656379034426-gcwzypcd/google-cloud-dataproc-metainfo/432fb230-c37e-49ec-b284-26f7a4c3f8c5/jobs/b2b2b87744804ab5b49f416036bccd
03/staging/w_count.py
reference:
  jobId: b2b2b87744804ab5b49f416036bccd03
  projectId: eecs6893-399118
status:
  state: DONE
  stateStartTime: '2023-09-19T22:44:08.517454Z'
statusHistory:
- state: PENDING
  stateStartTime: '2023-09-19T22:43:31.630937Z'
- state: SETUP_DONE
  stateStartTime: '2023-09-19T22:43:31.662299Z'
- details: Agent reported job success
  state: RUNNING
  stateStartTime: '2023-09-19T22:43:31.893758Z'
yarnApplications:
- name: w_count.py
  progress: 1.0
  state: FINISHED
  trackingUrl: http://cluster-wordcount-m.c.eecs6893-399118.internal.:8088/proxy/application_1695151409962_0007/
```

Check the output:

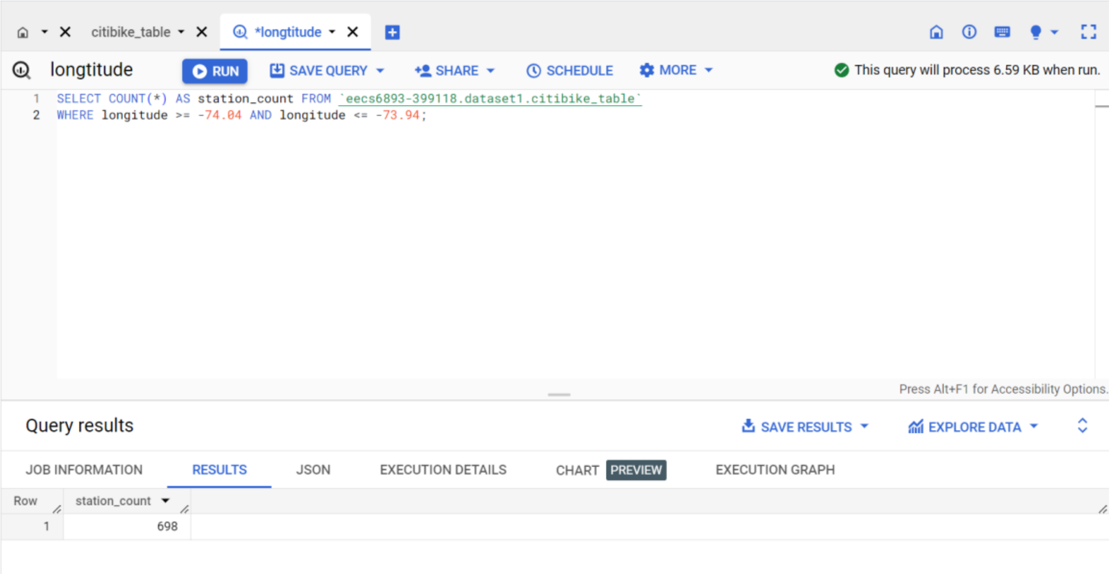


```
D:\6893\wordcount>gsutil cat gs://6893_bucket_1/output/*
("What's", 1)
('in', 1)
('name?', 1)
('That', 1)
('we', 1)
('call', 1)
('rose', 1)
('other', 1)
('name', 1)
('would', 1)
('smell', 1)
('as', 1)
('sweet.', 1)
('a', 2)
('which', 1)
('By', 1)
('any', 1)
```

2. NYC Bike expert

(1) Get the number of stations with longitude between -73.94 and -74.04

The following query is used:



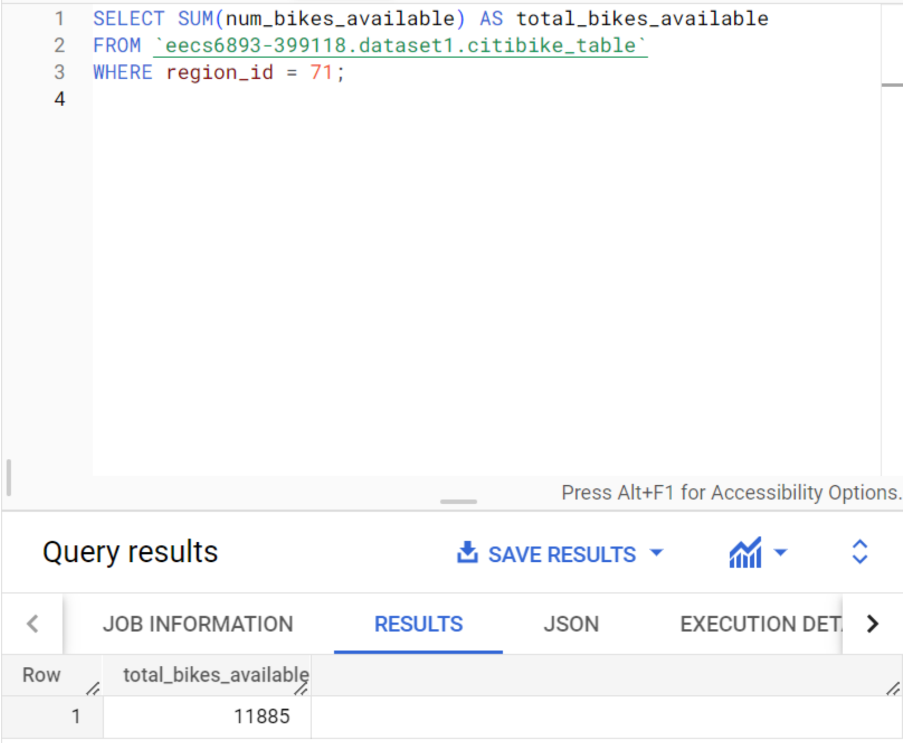
```sql
SELECT COUNT(*) AS station_count FROM `eecs6893-399118.dataset1.citibike_table`
WHERE longitude >= -74.04 AND longitude <= -73.94;
```

Row	station_count
1	698

The number of stations with longitude between -73.94 and -74.04 is 698.

(2) Total number of bikes available in region_id 71

The following query is used:



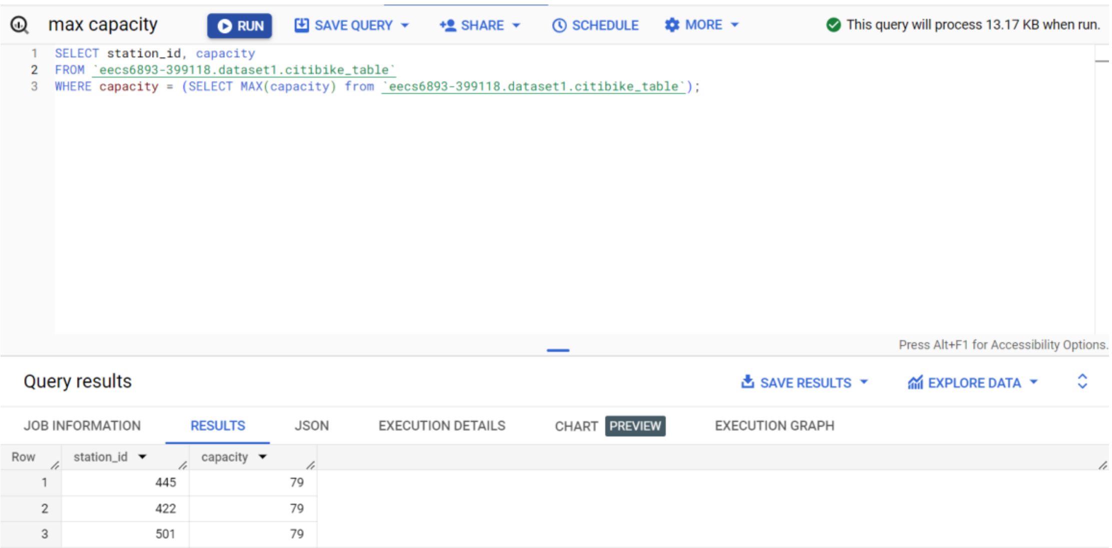
```sql
SELECT SUM(num_bikes_available) AS total_bikes_available
FROM `eecs6893-399118.dataset1.citibike_table`
WHERE region_id = 71;

```

Row	total_bikes_available
1	11885

Total number of bikes available in region_id 71 is 11884.

(3) List all the station_id of the stations that have the largest capacity

The following query is used:



```sql
SELECT station_id, capacity
FROM `eecs6893-399118.dataset1.citibike_table`
WHERE capacity = (SELECT MAX(capacity) from `eecs6893-399118.dataset1.citibike_table`);
```

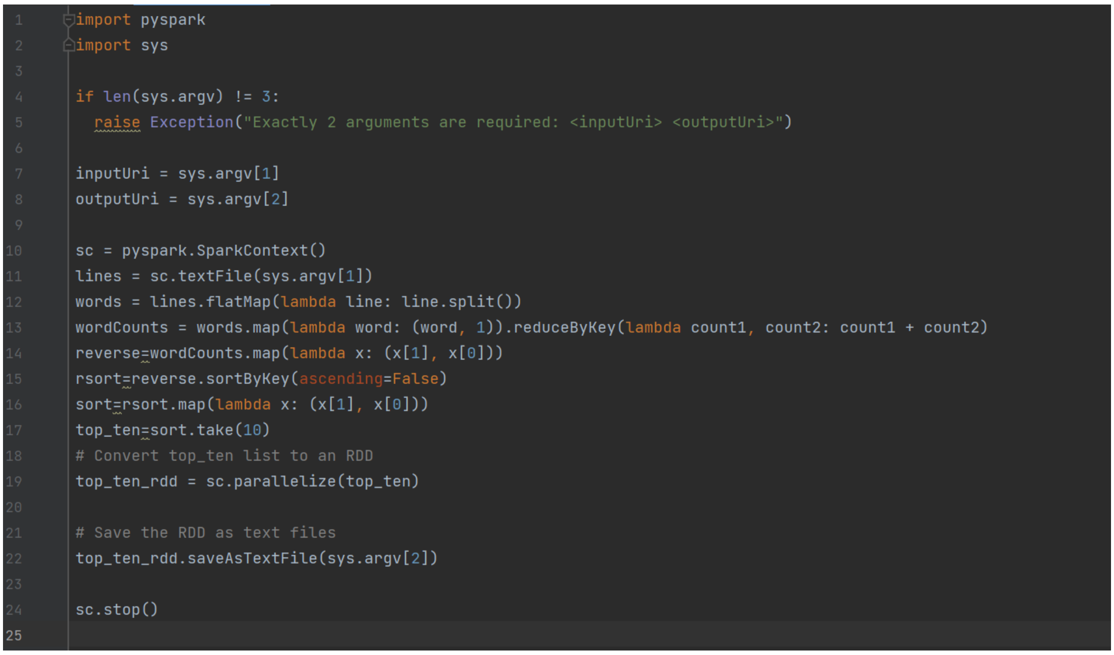
Row	station_id	capacity
1	445	79
2	422	79
3	501	79

Stations 445, 422 and 501 have the max capacity of 79.

3. Understanding William Shakespeare

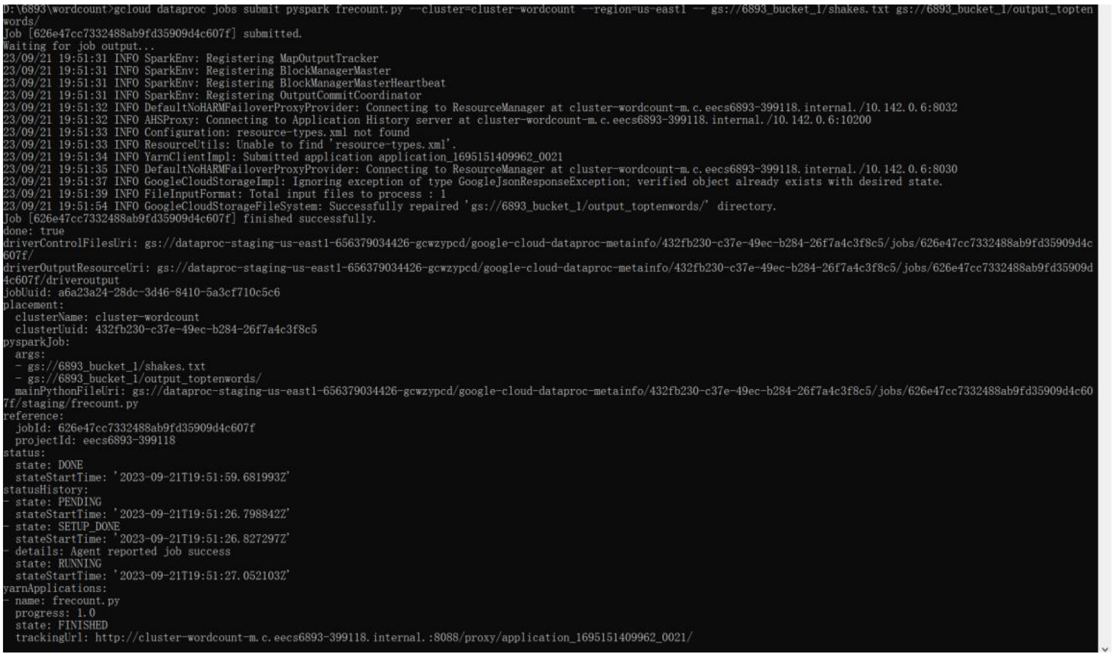
(1) Find top 10 frequent words without any text preprocessing.

The python code is shown as follows:



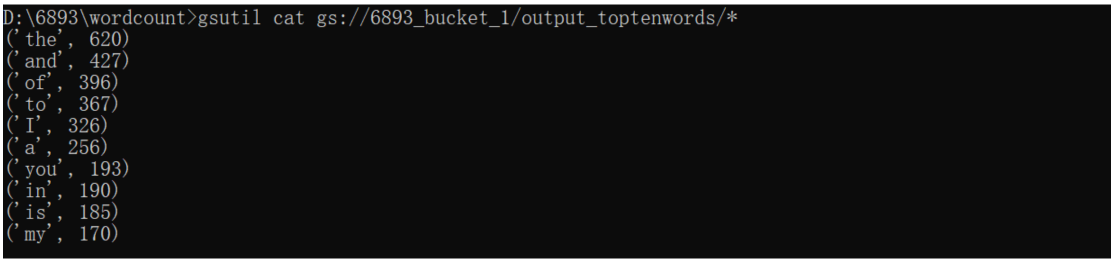
```python
import pyspark
import sys

if len(sys.argv) != 3:
  raise Exception("Exactly 2 arguments are required: <inputUri> <outputUri>")

inputUri = sys.argv[1]
outputUri = sys.argv[2]

sc = pyspark.SparkContext()
lines = sc.textFile(sys.argv[1])
words = lines.flatMap(lambda line: line.split())
wordCounts = words.map(lambda word: (word, 1)).reduceByKey(lambda count1, count2: count1 + count2)
reverse=wordCounts.map(lambda x: (x[1], x[0]))
rsort=reverse.sortByKey(ascending=False)
sort=rsort.map(lambda x: (x[1], x[0]))
top_ten=sort.take(10)
# Convert top_ten list to an RDD
top_ten_rdd = sc.parallelize(top_ten)

# Save the RDD as text files
top_ten_rdd.saveAsTextFile(sys.argv[2])

sc.stop()
```

Submit the job and store the output to the bucket.



```
D:\6893\wordcount>gcloud dataproc jobs submit pyspark frecount.py --cluster=cluster-wordcount --region=us-east1 -- gs://6893_bucket_1/shakes.txt gs://6893_bucket_1/output_topten
words/
Job [626e47cc7332488ab9fd35909d4c607f] submitted.
Waiting for job output...
23/09/21 19:51:31 INFO SparkEnv: Registering MapOutputTracker
23/09/21 19:51:31 INFO SparkEnv: Registering BlockManagerMaster
23/09/21 19:51:31 INFO SparkEnv: Registering BlockManagerMasterHeartbeat
23/09/21 19:51:31 INFO SparkEnv: Registering OutputCommitCoordinator
23/09/21 19:51:32 INFO DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at cluster-wordcount-m.c.eecs6893-399118.internal./10.142.0.6:8032
23/09/21 19:51:32 INFO AHSProxy: Connecting to Application History server at cluster-wordcount-m.c.eecs6893-399118.internal./10.142.0.6:10200
23/09/21 19:51:33 INFO Configuration: resource-types.xml not found
23/09/21 19:51:33 INFO ResourceUtils: Unable to find 'resource-types.xml'.
23/09/21 19:51:34 INFO YarnClientImpl: Submitted application application_1695151409962_0021
23/09/21 19:51:35 INFO DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at cluster-wordcount-m.c.eecs6893-399118.internal./10.142.0.6:8030
23/09/21 19:51:37 INFO GoogleCloudStorageImpl: Ignoring exception of type GoogleJsonResponseException; verified object already exists with desired state.
23/09/21 19:51:39 INFO FileInputFormat: Total input files to process : 1
23/09/21 19:51:54 INFO GoogleCloudStorageFileSystem: Successfully repaired 'gs://6893_bucket_1/output_toptenwords/' directory.
Job [626e47cc7332488ab9fd35909d4c607f] finished successfully.
done: true
driverControlFilesUri: gs://dataproc-staging-us-east1-656379034426-gcwzypcd/google-cloud-dataproc-metainfo/432fb230-c37e-49ec-b284-26f7a4c3f8c5/jobs/626e47cc7332488ab9fd35909d4c
607f/
driverOutputResourceUri: gs://dataproc-staging-us-east1-656379034426-gcwzypcd/google-cloud-dataproc-metainfo/432fb230-c37e-49ec-b284-26f7a4c3f8c5/jobs/626e47cc7332488ab9fd35909d
4c607f/driveroutput
jobUuid: a6a23a24-28dc-3d46-8410-5a3cf710c5c6
placement:
  clusterName: cluster-wordcount
  clusterUuid: 432fb230-c37e-49ec-b284-26f7a4c3f8c5
pysparkJob:
  args:
  - gs://6893_bucket_1/shakes.txt
  - gs://6893_bucket_1/output_toptenwords/
  mainPythonFileUri: gs://dataproc-staging-us-east1-656379034426-gcwzypcd/google-cloud-dataproc-metainfo/432fb230-c37e-49ec-b284-26f7a4c3f8c5/jobs/626e47cc7332488ab9fd35909d4c60
7f/staging/frecount.py
reference:
  jobId: 626e47cc7332488ab9fd35909d4c607f
  projectId: eecs6893-399118
status:
  state: DONE
  stateStartTime: '2023-09-21T19:51:59.681993Z'
statusHistory:
- state: PENDING
  stateStartTime: '2023-09-21T19:51:26.798842Z'
- state: SETUP_DONE
  stateStartTime: '2023-09-21T19:51:26.827297Z'
- details: Agent reported job success
  state: RUNNING
  stateStartTime: '2023-09-21T19:51:27.052103Z'
yarnApplications:
- name: frecount.py
  progress: 1.0
  state: FINISHED
  trackingUrl: http://cluster-wordcount-m.c.eecs6893-399118.internal.:8088/proxy/application_1695151409962_0021/
```

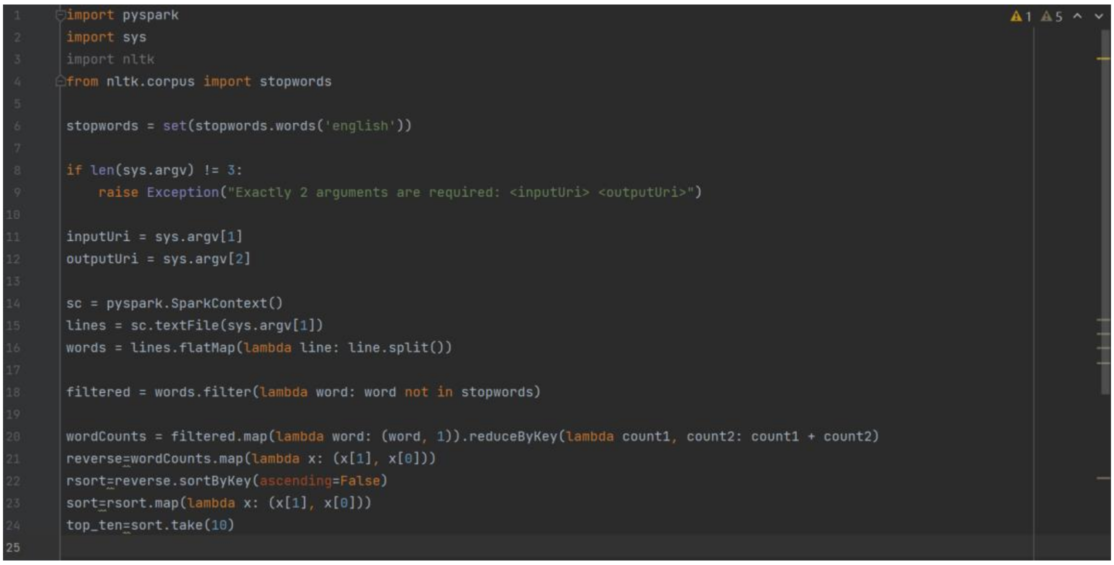
Print the output:



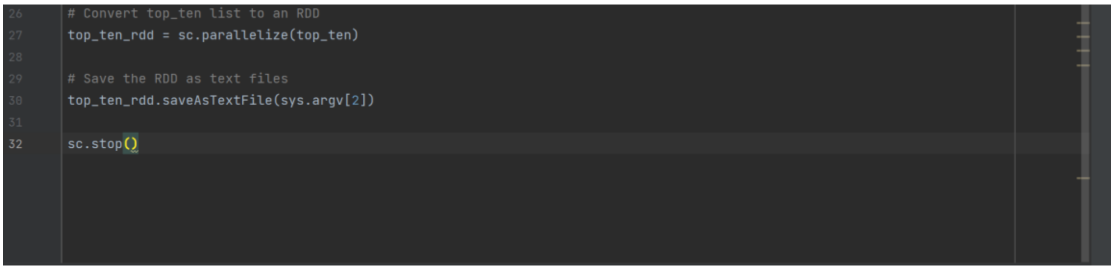
```
D:\6893\wordcount>gsutil cat gs://6893_bucket_1/output_toptenwords/*
('the', 620)
('and', 427)
('of', 396)
('to', 367)
('I', 326)
('a', 256)
('you', 193)
('in', 190)
('is', 185)
('my', 170)
```

(2)) Find top 10 frequent words by first filtering out stop words provided byNLTK package. The Natural Language Toolkit, or more commonly NLTK, is a suite of libraries and programs to conduct natural language processing in Python.

The python code is shown as follows:

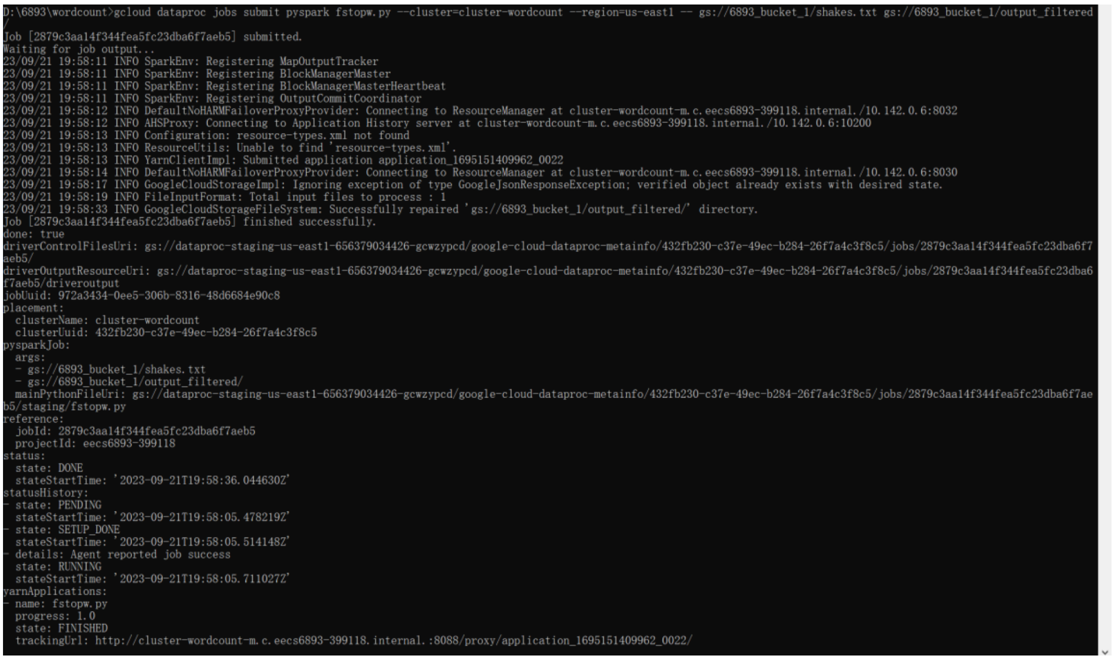


```python
import pyspark
import sys
import nltk
from nltk.corpus import stopwords

stopwords = set(stopwords.words('english'))

if len(sys.argv) != 3:
    raise Exception("Exactly 2 arguments are required: <inputUri> <outputUri>")

inputUri = sys.argv[1]
outputUri = sys.argv[2]

sc = pyspark.SparkContext()
lines = sc.textFile(sys.argv[1])
words = lines.flatMap(lambda line: line.split())

filtered = words.filter(lambda word: word not in stopwords)

wordCounts = filtered.map(lambda word: (word, 1)).reduceByKey(lambda count1, count2: count1 + count2)
reverse=wordCounts.map(lambda x: (x[1], x[0]))
rsort=reverse.sortByKey(ascending=False)
sort=rsort.map(lambda x: (x[1], x[0]))
top_ten=sort.take(10)

```



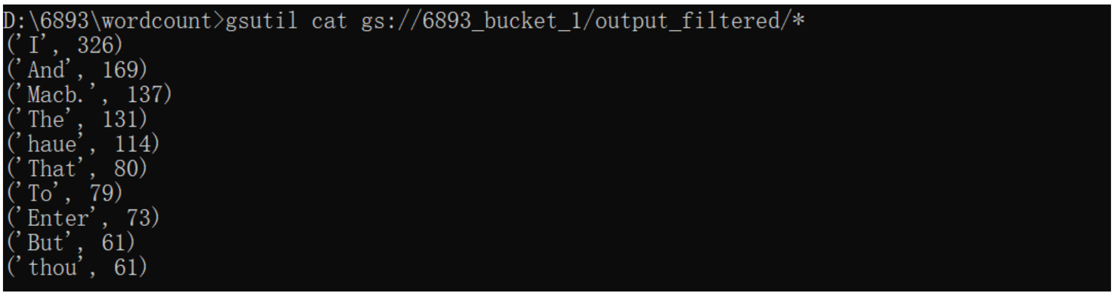
```python
# Convert top_ten list to an RDD
top_ten_rdd = sc.parallelize(top_ten)

# Save the RDD as text files
top_ten_rdd.saveAsTextFile(sys.argv[2])

sc.stop()
```

Submit the job and store the output to the bucket:



```
D:\6893\wordcount>gcloud dataproc jobs submit pyspark fstopw.py --cluster=cluster-wordcount --region=us-east1 -- gs://6893_bucket_1/shakes.txt gs://6893_bucket_1/output_filtered
/
Job [2879c3aa14f344fea5fc23dba6f7aeb5] submitted.
Waiting for job output...
23/09/21 19:58:11 INFO SparkEnv: Registering MapOutputTracker
23/09/21 19:58:11 INFO SparkEnv: Registering BlockManagerMaster
23/09/21 19:58:11 INFO SparkEnv: Registering BlockManagerMasterHeartbeat
23/09/21 19:58:11 INFO SparkEnv: Registering OutputCommitCoordinator
23/09/21 19:58:12 INFO DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at cluster-wordcount-m.c.eecs6893-399118.internal./10.142.0.6:8032
23/09/21 19:58:12 INFO AHSProxy: Connecting to Application History server at cluster-wordcount-m.c.eecs6893-399118.internal./10.142.0.6:10200
23/09/21 19:58:13 INFO Configuration: resource-types.xml not found
23/09/21 19:58:13 INFO ResourceUtils: Unable to find 'resource-types.xml'.
23/09/21 19:58:13 INFO YarnClientImpl: Submitted application application_1695151409962_0022
23/09/21 19:58:14 INFO DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at cluster-wordcount-m.c.eecs6893-399118.internal./10.142.0.6:8030
23/09/21 19:58:17 INFO GoogleCloudStorageImpl: Ignoring exception of type GoogleJsonResponseException; verified object already exists with desired state.
23/09/21 19:58:19 INFO FileInputFormat: Total input files to process : 1
23/09/21 19:58:33 INFO GoogleCloudStorageFileSystem: Successfully repaired 'gs://6893_bucket_1/output_filtered/' directory.
Job [2879c3aa14f344fea5fc23dba6f7aeb5] finished successfully.
done: true
driverControlFilesUri: gs://dataproc-staging-us-east1-656379034426-gcwzypcd/google-cloud-dataproc-metainfo/432fb230-c37e-49ec-b284-26f7a4c3f8c5/jobs/2879c3aa14f344fea5fc23dba6f7
aeb5/
driverOutputResourceUri: gs://dataproc-staging-us-east1-656379034426-gcwzypcd/google-cloud-dataproc-metainfo/432fb230-c37e-49ec-b284-26f7a4c3f8c5/jobs/2879c3aa14f344fea5fc23dba6
f7aeb5/driveroutput
jobUuid: 972a3434-0ee5-306b-8316-48d6684e90c8
placement:
  clusterName: cluster-wordcount
  clusterUuid: 432fb230-c37e-49ec-b284-26f7a4c3f8c5
pysparkJob:
  args:
  - gs://6893_bucket_1/shakes.txt
  - gs://6893_bucket_1/output_filtered/
  mainPythonFileUri: gs://dataproc-staging-us-east1-656379034426-gcwzypcd/google-cloud-dataproc-metainfo/432fb230-c37e-49ec-b284-26f7a4c3f8c5/jobs/2879c3aa14f344fea5fc23dba6f7ae
b5/staging/fstopw.py
reference:
  jobId: 2879c3aa14f344fea5fc23dba6f7aeb5
  projectId: eecs6893-399118
status:
  state: DONE
  stateStartTime: '2023-09-21T19:58:36.044630Z'
statusHistory:
- state: PENDING
  stateStartTime: '2023-09-21T19:58:05.478219Z'
- state: SETUP_DONE
  stateStartTime: '2023-09-21T19:58:05.514148Z'
- details: Agent reported job success
  state: RUNNING
  stateStartTime: '2023-09-21T19:58:05.711027Z'
yarnApplications:
- name: fstopw.py
  progress: 1.0
  state: FINISHED
  trackingUrl: http://cluster-wordcount-m.c.eecs6893-399118.internal.:8088/proxy/application_1695151409962_0022/
```

Print the output:



```
D:\6893\wordcount>gsutil cat gs://6893_bucket_1/output_filtered/*
('I', 326)
('And', 169)
('Macb.', 137)
('The', 131)
('haue', 114)
('That', 80)
('To', 79)
('Enter', 73)
('But', 61)
('thou', 61)
```