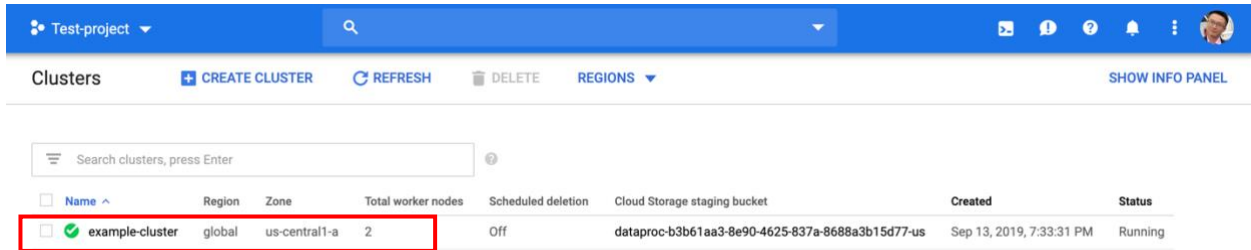


Homework0

1. Warmup

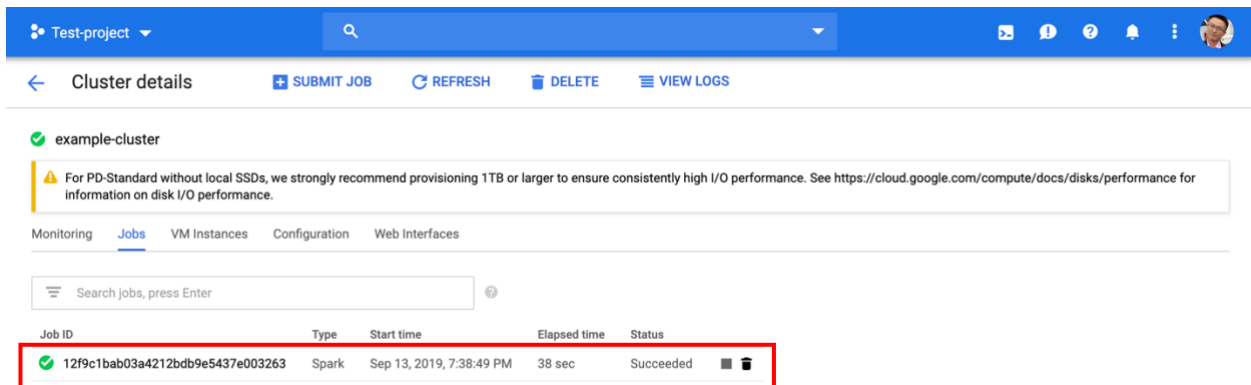
(1) pi calculation

- create cluster



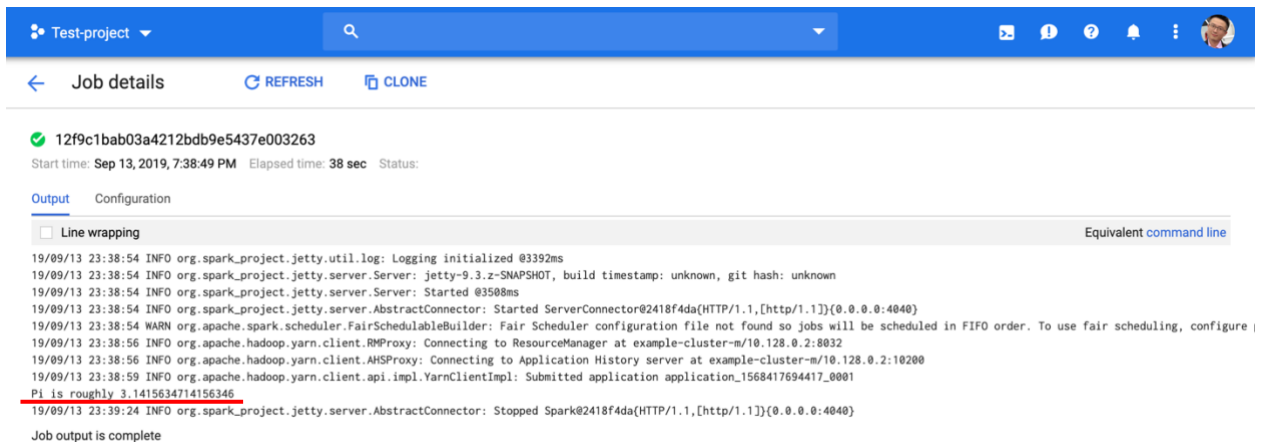
Name	Region	Zone	Total worker nodes	Scheduled deletion	Cloud Storage staging bucket	Created	Status
example-cluster	global	us-central1-a	2	Off	dataproc-b3b61aa3-8e90-4625-837a-8688a3b15d77-us	Sep 13, 2019, 7:33:31 PM	Running

- submit job



Job ID	Type	Start time	Elapsed time	Status
12f9c1bab03a4212bdb9e5437e003263	Spark	Sep 13, 2019, 7:38:49 PM	38 sec	Succeeded

- get result



Job ID	Type	Start time	Elapsed time	Status
12f9c1bab03a4212bdb9e5437e003263	Spark	Sep 13, 2019, 7:38:49 PM	38 sec	Succeeded

Start time: Sep 13, 2019, 7:38:49 PM Elapsed time: 38 sec Status:

Output Configuration

☐ Line wrapping Equivalent command line

```

19/09/13 23:38:54 INFO org.spark_project.jetty.util.log: Logging initialized @3392ms
19/09/13 23:38:54 INFO org.spark_project.jetty.server.Server: jetty-9.3.z-SNAPSHOT, build timestamp: unknown, git hash: unknown
19/09/13 23:38:54 INFO org.spark_project.jetty.server.Server: Started @3508ms
19/09/13 23:38:54 INFO org.spark_project.jetty.server.AbstractConnector: Started ServerConnector@2418f4da(HTTP/1.1,[http/1.1]){0.0.0.0:4040}
19/09/13 23:38:54 WARN org.apache.spark.scheduler.FairSchedulableBuilder: Fair Scheduler configuration file not found so jobs will be scheduled in FIFO order. To use fair scheduling, configure
19/09/13 23:38:56 INFO org.apache.hadoop.yarn.client.RMProxy: Connecting to ResourceManager at example-cluster-m/10.128.0.2:8032
19/09/13 23:38:56 INFO org.apache.hadoop.yarn.client.AHSProxy: Connecting to Application History server at example-cluster-m/10.128.0.2:10200
19/09/13 23:38:59 INFO org.apache.hadoop.yarn.client.api.impl.YarnClientImpl: Submitted application application_1568417694417_0001
Pl is roughly 3.1415634714156346
19/09/13 23:39:24 INFO org.spark_project.jetty.server.AbstractConnector: Stopped Spark@2418f4da(HTTP/1.1,[http/1.1]){0.0.0.0:4040}
Job output is complete
  
```

Word count

- submit job

Test-project

Cluster details

example-cluster

For PD-Standard without local SSDs, we strongly recommend provisioning 1TB or larger to ensure consistently high I/O performance. See <https://cloud.google.com/compute/docs/disks/performance> for information on disk I/O performance.

Monitoring Jobs VM Instances Configuration Web Interfaces

Search jobs, press Enter

Job ID	Type	Start time	Elapsed time	Status
e955c4373aaa40d3a9cd4778d63b5ee2	PySpark	Sep 16, 2019, 3:48:22 PM	26 sec	Succeeded

Equivalent REST

- output result

```
wuguojing — -bash — 130x30
stateStartTime: '2019-09-16T19:48:22.895Z'
- state: SETUP_DONE
stateStartTime: '2019-09-16T19:48:22.946Z'
- details: Agent reported job success
state: RUNNING
stateStartTime: '2019-09-16T19:48:23.180Z'
yarnApplications:
  name: wordcount.py
  progress: 1.0
  state: FINISHED
  trackingUrl: http://example-cluster-m:8088/proxy/application_1568657424768_0002/
athens-70-56-81-ac-8f-43:~ wuguojing$ gsutil cat gs://big_data_hw/output/*
(u'a', 2)
(u'we', 1)
(u'would', 1)
(u'What's', 1)
(u'sweet.', 1)
(u'as', 1)
(u'call', 1)
(u'which', 1)
(u'smell', 1)
(u'name', 1)
(u'That', 1)
(u'rose', 1)
(u'By', 1)
(u'other', 1)
(u'in', 1)
(u'name?', 1)
(u'any', 1)
athens-70-56-81-ac-8f-43:~ wuguojing$
```

(2) pi calculation

- transformation: filter()
- action: count()

```
def inside(p):
    x, y = random.random(), random.random()
    return x*x + y*y < 1

count = sc.parallelize(xrange(0, NUM_SAMPLES)) \
    transformation .filter(inside).count() Action
print "Pi is roughly %f" % (4.0 * count / NUM_SAMPLES)
```

Word count:

- transformation: map(), flatMap(), reduceByKey()
- action: saveAsTextFile()

```
#!/usr/bin/env 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]) transformation
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])
```

action

2. NYC Bike Expert

(1) unique station_ids

n = 843

```
wuguojiangs-MacBook-Pro:~ wuguojiang$ bq query --nouse_legacy_sql \
> 'SELECT
>   COUNT(DISTINCT station_id)
> FROM
> `test-project-251000`.my_dataset.nyc_citi_bike'
Waiting on bqjob_r5cb722b5f0376342_0000016d4c518db9_1 ... (0s) Current status: DONE
+-----+
| f0_ |
+-----+
| 843 |
+-----+
```

(2) largest capacity and corresponding station_id

MAX = 79

Station_id = {422, 445, 501}

```
wuguoings-MacBook-Pro:~ wuguoijing$ bq query --nouse_legacy_sql \
> 'SELECT
>   MAX(capacity)
> FROM
[>   `test-project-251000`.my_dataset.nyc_citi_bike'
Waiting on bqjob_r30cb39cbf563e6d5_0000016d4c58e6b7_1 ... (0s) Current status: DONE
+-----+
| f0_ |
+-----+
| 79 |
+-----+
wuguoings-MacBook-Pro:~ wuguoijing$ bq query --nouse_legacy_sql \
> 'SELECT
>   station_id
> FROM
>   `test-project-251000`.my_dataset.nyc_citi_bike
> WHERE
[>   capacity = 79'
Waiting on bqjob_ra7302e4cf53238c_0000016d4c59b7fa_1 ... (0s) Current status: DONE
+-----+
| station_id |
+-----+
|      445 |
|      422 |
|      501 |
+-----+
```

(3) total number of bikes available in region_id 70

n = 394

```
wuguoings-MacBook-Pro:~ wuguoijing$ bq query --nouse_legacy_sql \
> 'SELECT
>   SUM(num_bikes_available)
> FROM
>   `test-project-251000`.my_dataset.nyc_citi_bike
> WHERE
[>   region_id = 70'
Waiting on bqjob_r4e7d8ea017100dbe_0000016d4c5b9b7f_1 ... (0s) Current status: DONE
+-----+
| f0_ |
+-----+
| 394 |
+-----+
```

3. Understanding William Shakespeare

(1) without preprocess

```
620: the
427: and
396: of
367: to
326: I
```

Code:

```
#!/usr/bin/env python

import pyspark
import sys

if len(sys.argv) != 2:
    raise Exception("Exactly 1 arguments are required: <inputUri>")

inputUri=sys.argv[1]

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)\
    .map(lambda kv: (kv[1], kv[0]))\
    .sortByKey(ascending=False)\
    .take(5)

for (count, word) in wordCounts:
    print "%i: %s" % (count, word)
```

(2) NLTK preprocessed

Here we removed stop words, punctuations and words like 's, 'd, 'm

```
[(137, 'macb'), (122, 'haue'), (87, 'thou'), (81, 'enter'), (68, 'shall')]
```

Code:

```
1 import pyspark
2 import nltk
3 nltk.download('punkt')
4 nltk.download('stopwords')
5 from nltk.corpus import stopwords
6 stop_words=set(stopwords.words('english'))
7 import string
8 list_punct=list(string.punctuation)
9
10 def word_tokenizer(x):
11     lowerW = x.lower()
12     return nltk.word_tokenize(lowerW)
13
14 list_punct=list(string.punctuation)
15
16 lines = sc.textFile("shakes.txt")
17 words = lines.flatMap(word_tokenizer)\
18     .filter(lambda word : word not in stop_words and word != '')\
19     .filter(lambda punct : punct not in list_punct and punct[0] not in list_punct and punct[-1] not in list_punct)
20
21 wordCounts = words.map(lambda word: (word, 1))\
22     .reduceByKey(lambda count1, count2: count1 + count2)\
23     .map(lambda kv: (kv[1], kv[0]))\
24     .sortByKey(ascending=False)\
25     .take(5)
26
27 print(wordCounts)
```