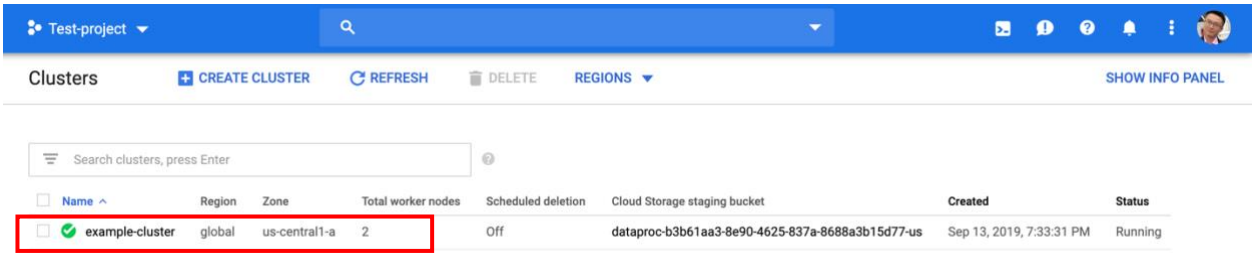
Homework0

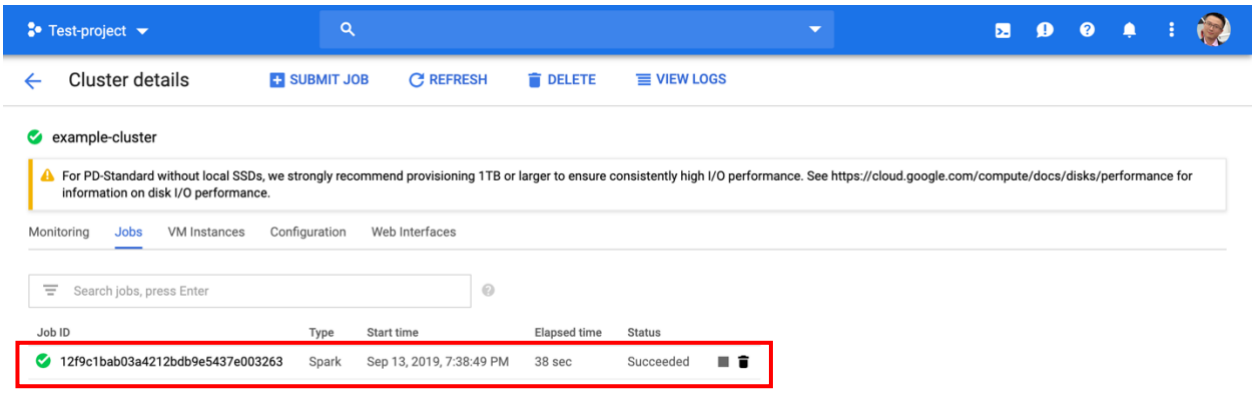
1. Warmup

(1) pi calculation

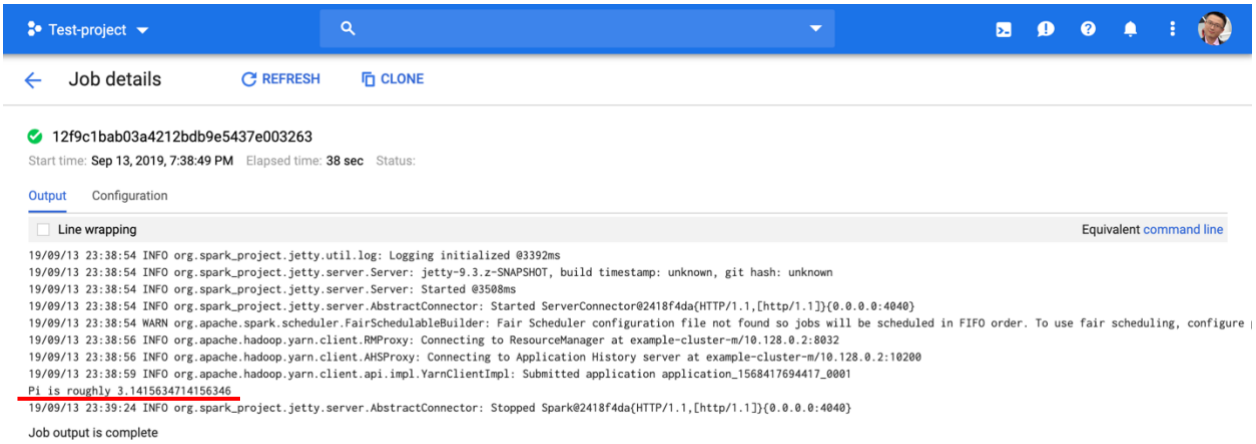
- create cluster



- submit job

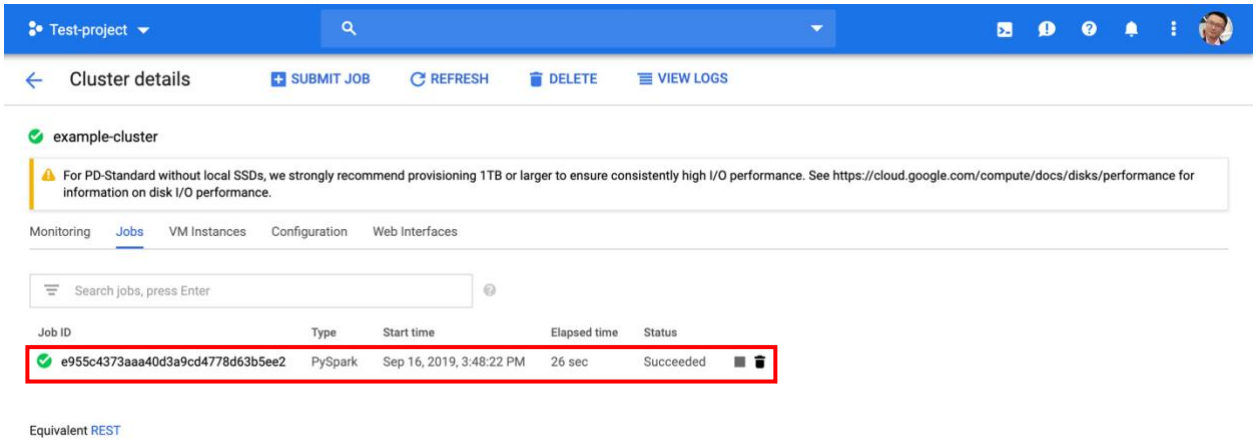


- get result

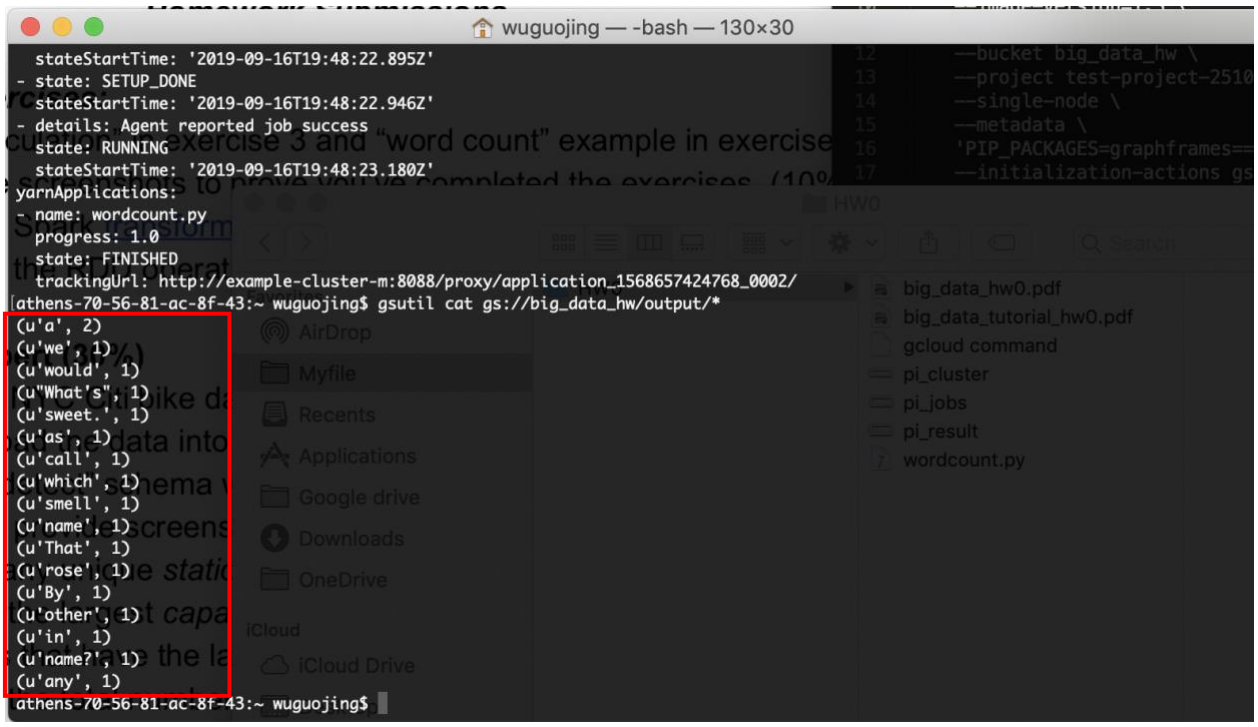


Word count

- submit job



- output result



(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

```
wuguojings-MacBook-Pro:~ wuguojing$ 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}

```
wuguojings-MacBook-Pro:~ wuguojing$ 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 |
+-----+
wuguojings-MacBook-Pro:~ wuguojing$ 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

```
wuguojings-MacBook-Pro:~ wuguojing$ 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:

```
import pyspark
import nltk
nltk.download('punkt')
nltk.download('stopwords')
from nltk.corpus import stopwords
stop_words=set(stopwords.words('english'))
import string
list_punct=list(string.punctuation)

def word_tokenize1(x):
    lowerW = x.lower()
    return nltk.word_tokenize(lowerW)

list_punct=list(string.punctuation)

lines = sc.textFile("shakes.txt")
words = lines.flatMap(word_tokenize1)\
.filter(lambda word : word not in stop_words and word != '')\
.filter(lambda punct : punct not in list_punct and punct[0] not in list_punct and punct[-1] not in list_punct)

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)

print(wordCounts)
```