IERG4330/IEMS5730 Spring 2022
Homework 3

Release date: Mar 11, 2022
Due date: 11:59:00 pm, Mar 28, 2022
The solution will be posted right after the deadline, so no late homework will be accepted!

Every Student MUST include the following statement, together with his/her signature in the submitted homework.

I declare that the assignment submitted on Elearning system is original except for source material explicitly acknowledged, and that the same or related material has not been previously submitted for another course. I also acknowledge that I am aware of University policy and regulations on honesty in academic work, and of the disciplinary guidelines and procedures applicable to breaches of such policy and regulations, as contained in the website http://www.cuhk.edu.hk/policy/academichonesty/.

Signed (Student Jesse) Date: 28 - 3 - 22

Name Chan Kai Yin SID 1155124983

Submission notice:

- Submit your homework via the elearning system

General homework policies:

A student may discuss the problems with others. However, the work a student turns in must be created COMPLETELY by oneself ALONE. A student may not share ANY written work or pictures, nor may one copy answers from any source other than one's own brain.

Each student **MUST LIST** on the homework paper the **name of every person he/she has discussed or worked with**. If the answer includes content from any other source, the student **MUST STATE THE SOURCE**. Failure to do so is cheating and will result in sanctions. Copying answers from someone else is cheating even if one lists their name(s) on the homework.

If there is information you need to solve a problem but the information is not stated in the problem, try to find the data somewhere. If you cannot find it, state what data you need, make a reasonable estimate of its value, and justify any assumptions you make. You will be graded not only on whether your answer is correct, but also on whether you have done an intelligent analysis.

Q1

a.)

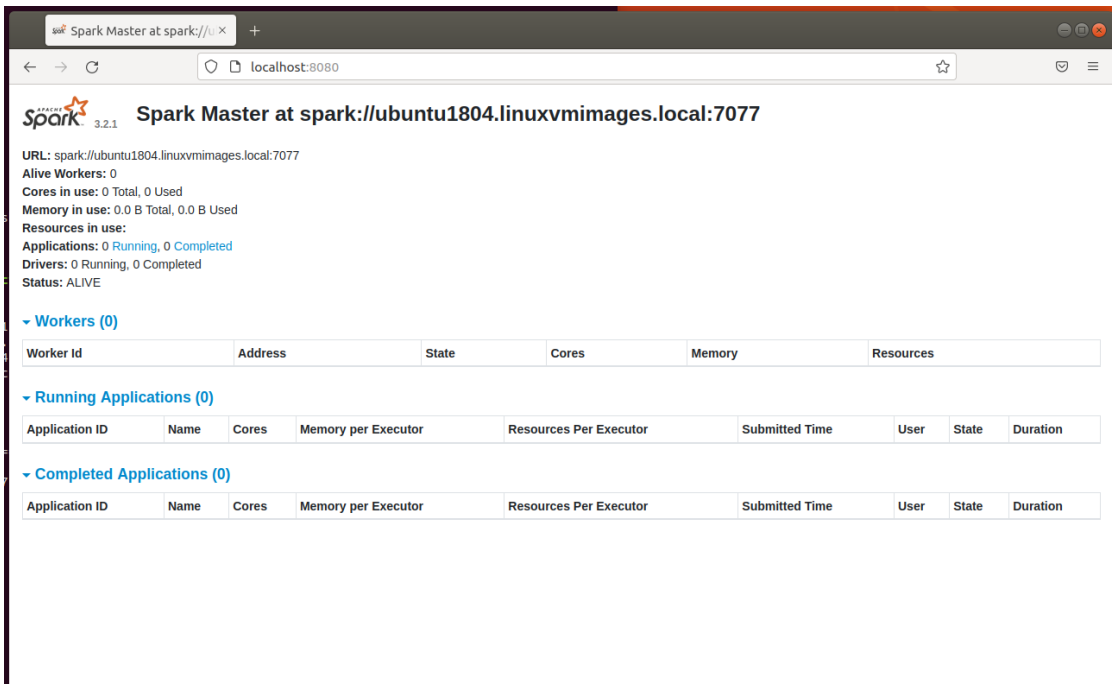
Download and unzip the Spark 3.2.1:

```
ubuntu@ubuntu1804:~/Downloads$ wget https://dlcdn.apache.org/spark/spark-3.2.1/spark-3.2.1-bin-hadoop3.2.tgz
--2022-03-23 23:06:22--  https://dlcdn.apache.org/spark/spark-3.2.1/spark-3.2.1-bin-hadoop3.2.tgz
Resolving dlcdn.apache.org (dlcdn.apache.org)... 151.101.2.132, 2a04:4e42::644
Connecting to dlcdn.apache.org (dlcdn.apache.org)|151.101.2.132|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 300971569 (287M) [application/x-gzip]
Saving to: 'spark-3.2.1-bin-hadoop3.2.tgz'

spark-3.2.1-bin-hadoop3.2.tgz          100%[===================================================================>] 287.03M  33.9MB/s    in 9.6s

2022-03-23 23:06:32 (29.9 MB/s) - 'spark-3.2.1-bin-hadoop3.2.tgz' saved [300971569/300971569]

ubuntu@ubuntu1804:~/Downloads$ tar xvf spark-3.2.1-bin-hadoop3.2.tgz
spark-3.2.1-bin-hadoop3.2/
spark-3.2.1-bin-hadoop3.2/LICENSE
spark-3.2.1-bin-hadoop3.2/NOTICE
spark-3.2.1-bin-hadoop3.2/R/
spark-3.2.1-bin-hadoop3.2/R/lib/
spark-3.2.1-bin-hadoop3.2/R/lib/SparkR/
spark-3.2.1-bin-hadoop3.2/R/lib/SparkR/DESCRIPTION
spark-3.2.1-bin-hadoop3.2/R/lib/SparkR/INDEX
spark-3.2.1-bin-hadoop3.2/R/lib/SparkR/Meta/
```

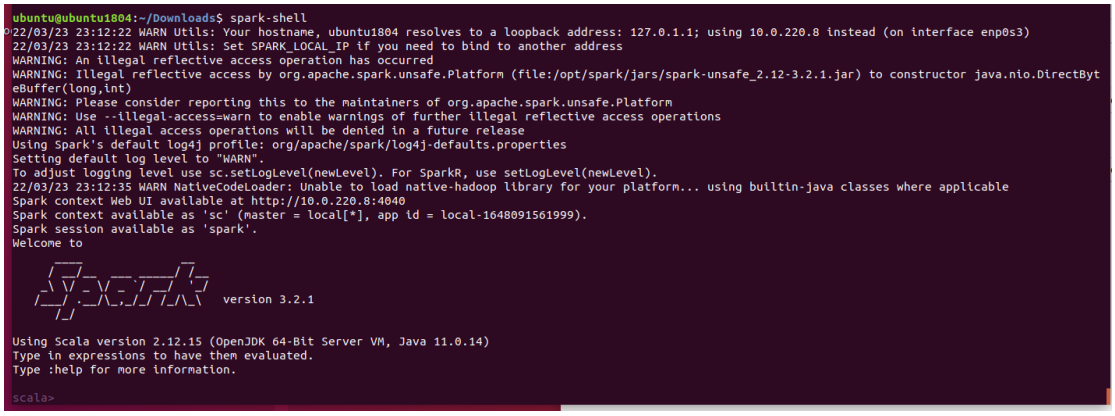
Set spark environment:

```
export SPARK_HOME=/opt/spark
export PATH=$PATH:$SPARK_HOME/bin:$SPARK_HOME/sbin
```

With command: start-master.sh, the Web UI with port 8080:



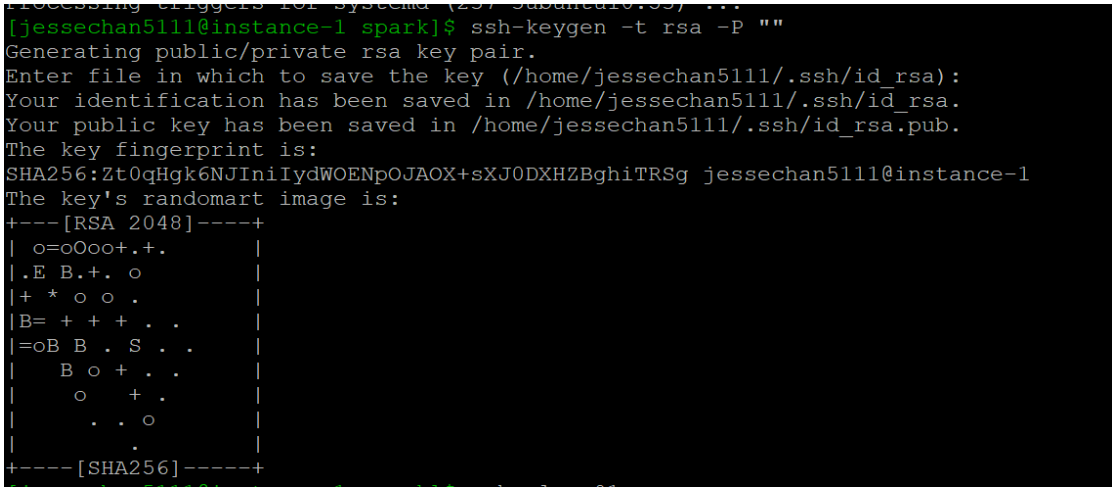
The spark shell:



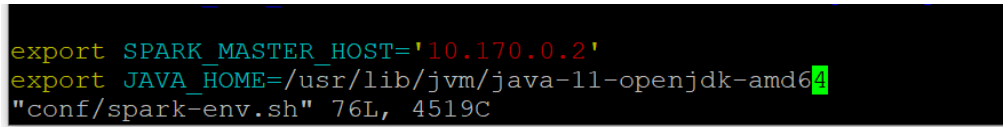
b.)

Install the spark same as part a for the master and slave node.

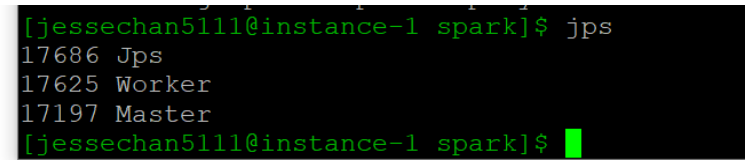
Setup SSH for the master and slave communication:



Setup the environment for spark:



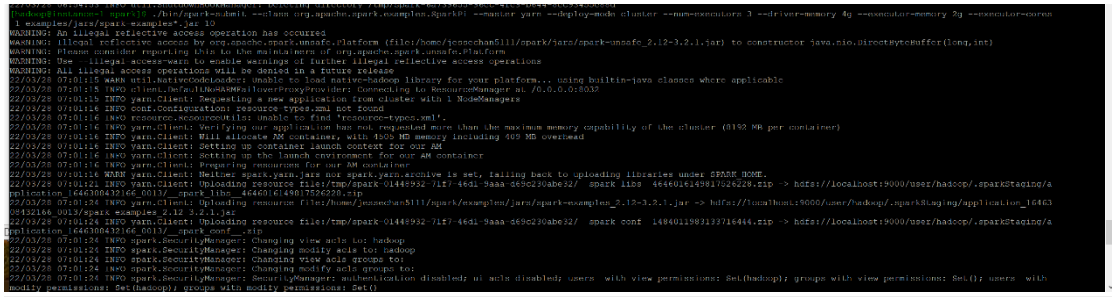
With start-all.sh to start, the services:



Spark shell:

```
[jessechan5111@instance-1 bin]$ ./spark-shell
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.spark.unsafe.Platform (file:/home/jessechan5111/spark/jars/spar
k-unsafe_2.12-3.2.1.jar) to constructor java.nio.DirectByteBuffer(long,int)
WARNING: Please consider reporting this to the maintainers of org.apache.spark.unsafe.Platform
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
22/03/24 04:43:54 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin
-java classes where applicable
Spark context Web UI available at http://master:4040
Spark context available as 'sc' (master = local[*], app id = local-1648097035825).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 3.2.1
      /_/

Using Scala version 2.12.15 (OpenJDK 64-Bit Server VM, Java 11.0.14)
Type in expressions to have them evaluated.
Type :help for more information.

scala>
```

Submitting job:



User:	hadoop
Name:	org.apache.spark.examples.SparkPi
Application Type:	SPARK
Application Tags:	
YarnApplicationState:	FINISHED
Queue:	default
FinalStatus Reported by AM:	SUCCEEDED
Started:	Mon Mar 28 07:01:24 +0000 2022
Elapsed:	41sec
Tracking URL:	History
Diagnostics:	

Q2

a.)

Code:

```
import re
import sys
from operator import add
from pyspark.sql import SparkSession
import pyspark

def computeContribs(urls, rank):
    num_urls = len(urls)
    for url in urls:
        yield (url, rank / num_urls)


def split_line(line):
    parts = re.split(r'\s+', line)
    return parts[0], parts[1]


if __name__ == "__main__":

    spark = SparkSession\
        .builder\
        .appName("PageRank")\
        .getOrCreate()

    lines = spark.read.text(sys.argv[1]).rdd.map(lambda r: r[0])
    links = lines.map(lambda urls: split_line(urls)).distinct().groupByKey()
    ranks = links.map(lambda url_neighbors: (url_neighbors[0], 1.0))

    for iteration in range(int(sys.argv[2])):
        contribs = links.join(ranks).flatMap(lambda url_urls_rank: computeContribs(
            url_urls_rank[1][0], url_urls_rank[1][1]
        ))

        ranks = contribs.reduceByKey(add).mapValues(lambda rank: rank * 0.85 + 0.15)

    top100 = ranks.top(100, key=lambda x: x[1])
    print(top100)
    spark.stop()
```

Output for the top100 for 10 iteration:

```
[(u'41909', 445.717785968565), (u'597621', 406.6283667503004), (u'504140', 399.0893087474903), (u'384666', 392.8258437305225), (
```

Time: 5 mins 1 sec:

Application Overview

User:	s1155124983
Name:	pagerank.py
Application Type:	SPARK
Application Tags:	
Application Priority:	0 (Higher Integer value indicates higher priority)
YarnApplicationState:	FINISHED
Queue:	default
FinalStatus Reported by AM:	SUCCEEDED
Started:	Mon Mar 28 12:37:17 +0800 2022
Elapsed:	5mins, 1sec
Tracking URI :	History
Log Aggregation Status	SUCCEEDED
Diagnostics:	
Unmanaged Application:	false
Application Node Label expression:	<Not set>
AM container Node Label expression:	<DEFAULT_PARTITION>

b.)

Code:

```
import re
import sys
from operator import add
from pyspark.sql import SparkSession
import pyspark

def computeContribs(urls, rank):
    num_urls = len(urls)
    for url in urls:
        yield (url, rank / num_urls)


def split_line(line):
    parts = re.split(r'\s+', line)
    return parts[0], parts[1]


if __name__ == "__main__":

    spark = SparkSession\
        .builder\
        .appName("PageRank")\
        .getOrCreate()

    lines = spark.read.text(sys.argv[1]).rdd.map(lambda r: r[0])
    links = lines.map(lambda urls: split_line(urls)).distinct().groupByKey().partitionBy(100).persist()
    ranks = links.map(lambda url_neighbors: (url_neighbors[0], 1.0))

    for iteration in range(int(sys.argv[2])):
        contribs = links.join(ranks).flatMap(lambda url_urls_rank: computeContribs(
            url_urls_rank[1][0], url_urls_rank[1][1]
        ))

        ranks = contribs.reduceByKey(add).mapValues(lambda rank: rank * 0.85 + 0.15)

    top100 = ranks.top(100, key=lambda x: x[1])
    print(top100)
    spark.stop()
~
~
~
```

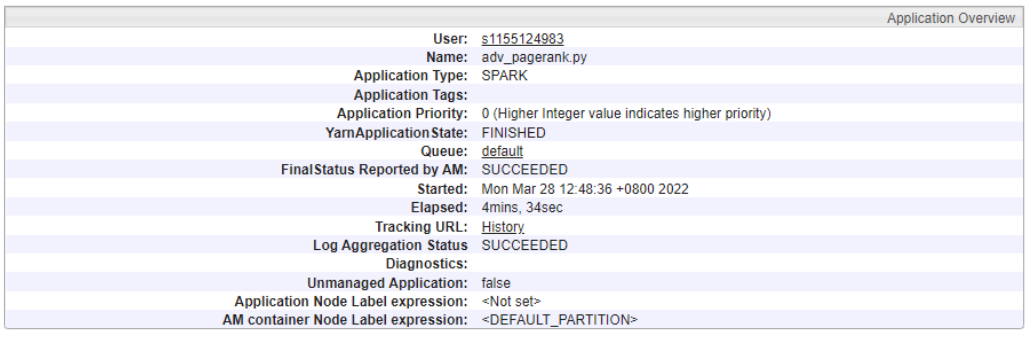
Change the partition number in partitionBy() function as comparison.

Compare the performance with part a in 10 iteration:

i.

Setting the partitions number for links RDD = 50:

Time: 4 mins 34 sec



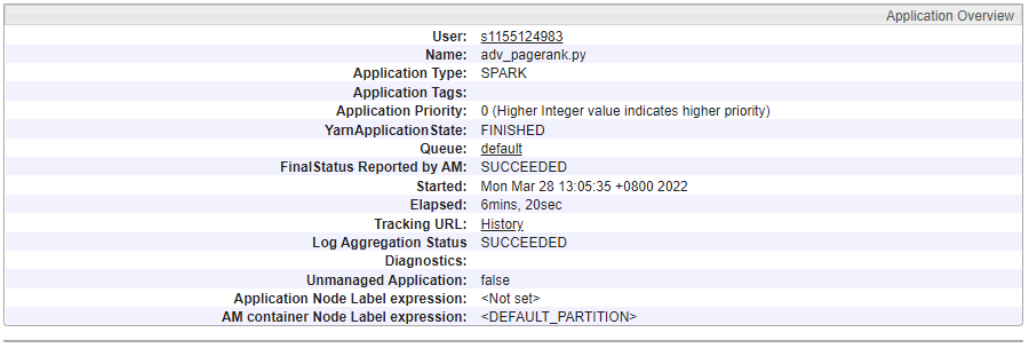
Application Overview

User:	s1155124983
Name:	adv_pagerank.py
Application Type:	SPARK
Application Tags:	
Application Priority:	0 (Higher Integer value indicates higher priority)
YarnApplicationState:	FINISHED
Queue:	default
FinalStatus Reported by AM:	SUCCEEDED
Started:	Mon Mar 28 12:48:36 +0800 2022
Elapsed:	4mins, 34sec
Tracking URL:	History
Log Aggregation Status	SUCCEEDED
Diagnostics:	
Unmanaged Application:	false
Application Node Label expression:	<Not set>
AM container Node Label expression:	<DEFAULT_PARTITION>

ii.

Setting the partitions number for links RDD = 100:

Time: 6 mins 20 sec



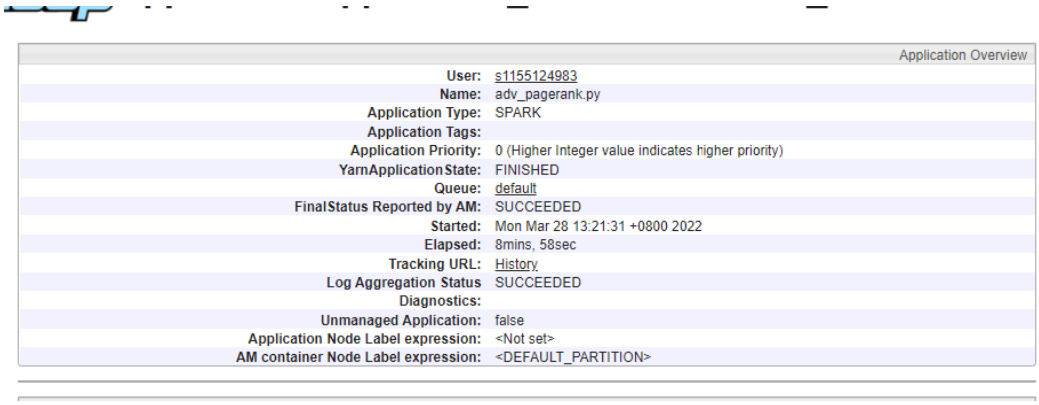
Application Overview

User:	s1155124983
Name:	adv_pagerank.py
Application Type:	SPARK
Application Tags:	
Application Priority:	0 (Higher Integer value indicates higher priority)
YarnApplicationState:	FINISHED
Queue:	default
FinalStatus Reported by AM:	SUCCEEDED
Started:	Mon Mar 28 13:05:35 +0800 2022
Elapsed:	6mins, 20sec
Tracking URL:	History
Log Aggregation Status	SUCCEEDED
Diagnostics:	
Unmanaged Application:	false
Application Node Label expression:	<Not set>
AM container Node Label expression:	<DEFAULT_PARTITION>

iii.

Setting the partitions number for links RDD = 150:

Time: 8 mins 58 sec



Application Overview

User:	s1155124983
Name:	adv_pagerank.py
Application Type:	SPARK
Application Tags:	
Application Priority:	0 (Higher Integer value indicates higher priority)
YarnApplicationState:	FINISHED
Queue:	default
FinalStatus Reported by AM:	SUCCEEDED
Started:	Mon Mar 28 13:21:31 +0800 2022
Elapsed:	8mins, 58sec
Tracking URL:	History
Log Aggregation Status	SUCCEEDED
Diagnostics:	
Unmanaged Application:	false
Application Node Label expression:	<Not set>
AM container Node Label expression:	<DEFAULT_PARTITION>

The links RDD is unchanged after creating it while the ranks need to update in every iteration. Links RDD can be partitioned in first to reduce the network shuffle and network communication overhead problem.

Run time regarding to partitions number: 50 < without partitions setting < 100 < 150

The fastest is the one with partitions number = 50, faster than part a) by 30 seconds. And with partitions number = 100 and 150, the speed will much slower. Extra time may be caused by the process of mapping a lot of partitions to perform the join operation.

Q3

a.)

Code:

```
from pyspark.sql import SparkSession
from pyspark.context import SparkContext

sc = SparkSession.builder.appName("q3a").config ("spark.sql.shuffle.partitions", "50").config("spark.driver.max
ResultSize","5g").config ("spark.sql.execution.arrow.enabled", "true").getOrCreate()

df = sc.read.csv('hdfs:///user/sl155124983/Crime_Incidents_in_2013.csv',header=True)
df_drop_empCol = df.drop("OCTO_RECORD_ID")
df_drop_cmpCol = df_drop_empCol.na.drop(how = 'any')
df_tarCol = df_drop_empCol.select("CCN", "REPORT_DAT", "OFFENSE", "METHOD", "END_DATE", "DISTRICT")

df_tarCol.write.csv('hdfs:/user/sl155124983/hw3')

~
~
~
~
~
~
~
~
```

Dropping the “OCTO_RECORD_ID” column because it is empty for all.

Output:

```
>>> df_tarCol.show()
+--------+-------------------+--------------------+------+-------------------+--------+
|     CCN|          REPORT_DAT|             OFFENSE|METHOD|           END_DATE|DISTRICT|
+--------+-------------------+--------------------+------+-------------------+--------+
|04104147|2013/04/16 04:00:...|           HOMICIDE| KNIFE|2004/07/28 00:30:...|       1|
|10028985|2013/02/27 05:00:...|         SEX ABUSE|OTHERS|2010/03/07 07:45:...|       5|
|10033521|2013/10/10 04:00:...|         SEX ABUSE|OTHERS|2008/12/30 01:00:...|       6|
|10124918|2013/04/09 04:00:...|         SEX ABUSE|OTHERS|               null|       7|
|10124918|2013/04/09 04:00:...|         SEX ABUSE|OTHERS|               null|       7|
|11010107|2013/07/31 04:00:...|           HOMICIDE|OTHERS|               null|       5|
|11045512|2013/01/31 05:00:...|           HOMICIDE|   GUN|2011/04/06 02:32:...|       6|
|11250281|2013/07/08 04:00:...|         SEX ABUSE|OTHERS|2011/05/13 03:15:...|       6|
|12003591|2013/01/09 05:59:...|       THEFT/OTHER|OTHERS|2013/01/09 05:59:...|       5|
|12139462|2013/11/13 05:00:...|         SEX ABUSE|OTHERS|2012/10/01 08:00:...|       1|
|12182426|2013/01/01 05:15:...|            ROBBERY|   GUN|2013/01/01 05:15:...|       6|
|12182466|2013/01/01 07:10:...|      THEFT F/AUTO|OTHERS|2013/01/01 06:15:...|       7|
|12182502|2013/01/01 08:00:...|       THEFT/OTHER|OTHERS|2013/01/01 06:30:...|       1|
|12182505|2013/01/01 07:29:...|            ROBBERY|   GUN|2013/01/01 07:26:...|       4|
|12182530|2013/01/01 08:10:...|ASSAULT W/DANGERO...| KNIFE|2013/01/01 08:10:...|       7|
|12182534|2013/01/01 08:12:...|ASSAULT W/DANGERO...|   GUN|2013/01/01 08:12:...|       7|
|12182544|2013/01/01 08:31:...|       THEFT/OTHER|OTHERS|2013/01/01 08:14:...|       4|
|12182550|2013/01/01 08:45:...|      THEFT F/AUTO|OTHERS|2013/01/01 08:30:...|       5|
|12182554|2013/01/01 08:38:...|            ROBBERY|OTHERS|2013/01/01 08:38:...|       3|
|12182577|2013/01/01 09:23:...|ASSAULT W/DANGERO...|   GUN|2013/01/01 09:23:...|       7|
+--------+-------------------+--------------------+------+-------------------+--------+
only showing top 20 rows
```

b.)

i. Number of each type offenses)

Code:

```
from pyspark.sql import SparkSession
from pyspark.context import SparkContext

sc = SparkSession.builder.appName("q3a").config ("spark.sql.shuffle.partitions", "50").config("spark.driver.maxResultS
ize","5g").config ("spark.sql.execution.arrow.enabled", "true").getOrCreate()

df = sc.read.csv('hdfs:///user/s1155124983/Crime_Incidents_in_2013.csv',header=True)
df_drop_empCol = df.drop("OCTO_RECORD_ID")
df_tarCol = df_drop_empCol.select("CCN", "REPORT_DAT", "OFFENSE", "METHOD", "END_DATE", "DISTRICT")

#df_tarCol.write.csv('hdfs:/user/s1155124983/hw3')

df_gpbyoffcount = df_tarCol.groupBy("OFFENSE").count().orderBy("count", ascending = False)
```

Output:

```
>>> df_gpbyoffcount.show()
+--------------------+-----+
|            OFFENSE|count|
+--------------------+-----+
|         THEFT/OTHER|12891|
|        THEFT F/AUTO|10183|
|             ROBBERY| 3991|
|            BURGLARY| 3356|
| MOTOR VEHICLE THEFT| 2667|
|ASSAULT W/DANGERO...| 2401|
|           SEX ABUSE|  299|
|            HOMICIDE|  104|
|               ARSON|   35|
+--------------------+-----+

>>>
```

ii. which time-slot (shift) did the most crimes occur?)

Assume the report date is the same as the time that crimes occur and the time shift is in per one hour.

Code:

```
from pyspark.sql import SparkSession
from pyspark.context import SparkContext

sc = SparkSession.builder.appName("q3a").config ("spark.sql.shuffle.partitions", "50").config("spark.driver.maxR
esultSize","5g").config ("spark.sql.execution.arrow.enabled", "true").getOrCreate()

df = sc.read.csv('hdfs:///user/s1155124983/Crime_Incidents_in_2013.csv',header=True)
df_drop_empCol = df.drop("OCTO_RECORD_ID")
df_drop_cmpCol = df_drop_empCol.na.drop(how = 'any')
df_tarCol = df_drop_empCol.select("CCN", "REPORT_DAT", "OFFENSE", "METHOD", "END_DATE", "DISTRICT","SHIFT")

#df_tarCol.write.csv('hdfs:/user/s1155124983/hw3')

df_gpbyoffcount = df_tarCol.groupBy("OFFENSE").count().orderBy("count", ascending = False)

from pyspark.sql.functions import split
from pyspark.sql.functions import col

#df_Time = df_tarCol.select("REPORT_DAT").withColumn("Re_Time", split(split(col("REPORT_DAT")," ").getItem(1), "
:").getItem(0))
df_time_count = df_tarCol.groupBy("SHIFT").count().orderBy("count", ascending = False)
~
~
~
~
~
```

Output:

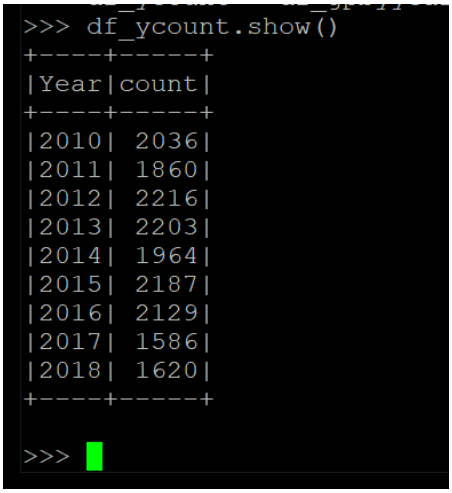
```
>>> df_time_count.show()
+--------+-----+
|   SHIFT|count|
+--------+-----+
| EVENING|15082|
|     DAY|14318|
|MIDNIGHT| 6527|
+--------+-----+
```

Time shift evening has most crimes occur.

c.)

Code:

Output:



According to news report, President Obama unveiled a new strategy to curb gun violence in the United States in 2016. The gun offense dropped to 1586 in 2017 from 2129. The case number decrease by about 25% and I would regard it a successful policy.

Q4

a.)

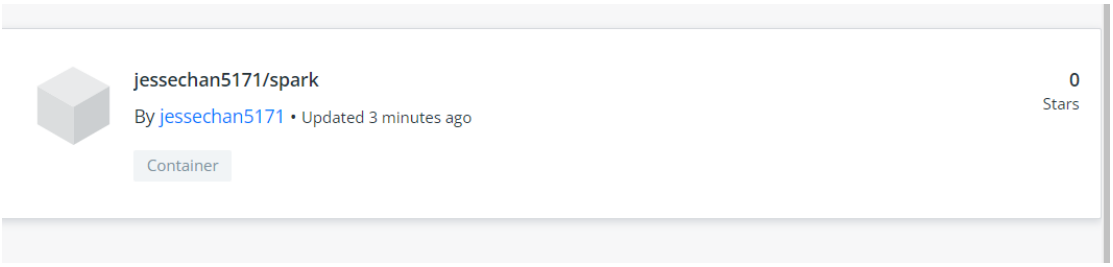
Build the Docker image:

```
PS C:\Spark\spark-3.2.1-bin-hadoop3.2> docker images
REPOSITORY            TAG       IMAGE ID       CREATED             SIZE
jessechan5171/spark   v3.2.1    a7992b67bcd0   About an hour ago   615MB
PS C:\Spark\spark-3.2.1-bin-hadoop3.2>
```

b.)

Push the Docker image:

https://hub.docker.com/u/jessechan5171



c.)

The command:

Extract from the log file:

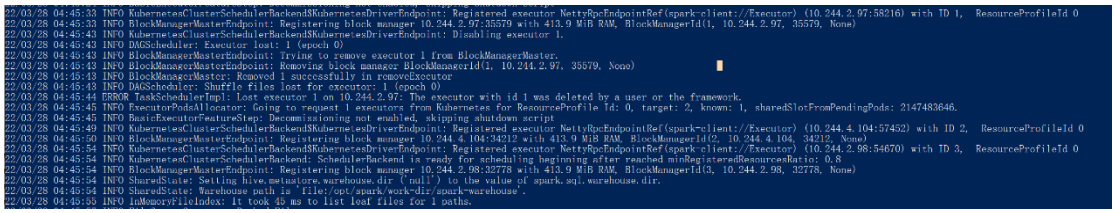
```
winners: 1
exultation": 1
bough: 1
Paulina!": 1
question'd,: 1
mind--to: 1
justified": 1
directing,": 1
leisurely": 1
dissever'd:: 1
22/03/27 18:15:11 INFO SparkUI: Stopped Spark web UI at http://spark-wordcount-f9355d7fcc95e284-driver-svc.s1155124983.svc:4040
22/03/27 18:15:11 INFO KubernetesClusterSchedulerBackend: Shutting down all executors
22/03/27 18:15:11 INFO KubernetesClusterSchedulerBackend$KubernetesDriverEndpoint: Asking each executor to shut down
22/03/27 18:15:11 WARN ExecutorPodsWatchSnapshotSource: Kubernetes client has been closed.
22/03/27 18:15:11 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
22/03/27 18:15:11 INFO MemoryStore: MemoryStore cleared
```

Q5

a.)

```
PS C:\WINDOWS\system32> kubectl get pods -n s1155124983
NAME                                   READY   STATUS              RESTARTS   AGE
pythonwordcount-4563097fced7cced-exec-1   1/1     Running             0          7s
pythonwordcount-4563097fced7cced-exec-2   0/1     ContainerCreating   0          7s
spark-pi-89b2d27fcc45dd3e-driver       0/1     Completed           0          11h
spark-wordcount-33441d7fced2e037-driver   0/1     Completed           0          5m29s
spark-wordcount-a4db027fced4dfd2-driver   0/1     Completed           0          3m19s
spark-wordcount-c64c797fced7ac30-driver   1/1     Running             0          15s
spark-wordcount-f9355d7fcc95e284-driver   0/1     Completed           0          10h
PS C:\WINDOWS\system32> kubectl delete pod pythonwordcount-4563097fced7cced-exec-1 -n s1155124983
pod "pythonwordcount-4563097fced7cced-exec-1" deleted
PS C:\WINDOWS\system32> kubectl get pods -n s1155124983
NAME                                   READY   STATUS      RESTARTS   AGE
pythonwordcount-4563097fced7cced-exec-2   1/1     Running     0          33s
pythonwordcount-4563097fced7cced-exec-3   1/1     Running     0          12s
spark-pi-89b2d27fcc45dd3e-driver       0/1     Completed   0          11h
spark-wordcount-33441d7fced2e037-driver   0/1     Completed   0          5m55s
spark-wordcount-a4db027fced4dfd2-driver   0/1     Completed   0          3m45s
spark-wordcount-c64c797fced7ac30-driver   1/1     Running     0          41s
spark-wordcount-f9355d7fcc95e284-driver   0/1     Completed   0          10h
PS C:\WINDOWS\system32> kubectl get pods -n s1155124983
NAME                                   READY   STATUS      RESTARTS   AGE
pythonwordcount-4563097fced7cced-exec-2   1/1     Running     0          39s
pythonwordcount-4563097fced7cced-exec-3   1/1     Running     0          18s
spark-pi-89b2d27fcc45dd3e-driver       0/1     Completed   0          11h
spark-wordcount-33441d7fced2e037-driver   0/1     Completed   0          6m1s
spark-wordcount-a4db027fced4dfd2-driver   0/1     Completed   0          3m51s
spark-wordcount-c64c797fced7ac30-driver   1/1     Running     0          47s
spark-wordcount-f9355d7fcc95e284-driver   0/1     Completed   0          10h
PS C:\WINDOWS\system32>
```

Log file:



At first the application has 2 executor pod (exec-1, 2). I used the command line “kubectl delete pod” to kill the exec-1 pod in the middle of its execution. The scheduler detect that the exec-1 pod is killed, then it creates another executor pod (exec-3) to replace the exec-1 pod and rerun its task.

b.)

```
spark-wordcount-f9355d7fcc95e284-driver   0/1     Completed   0          10h
PS C:\WINDOWS\system32> kubectl get pods -n s1155124983
NAME                                      READY   STATUS              RESTARTS   AGE
spark-pi-89b2d27fcc45dd3e-driver          0/1     Completed           0          12h
spark-wordcount-00c5797fcef1b6ea-driver   0/1     ContainerCreating   0          3s
spark-wordcount-33441d7fced2e037-driver   0/1     Completed           0          33m
spark-wordcount-a4db027fced4dfd2-driver   0/1     Completed           0          31m
spark-wordcount-c64c797fced7ac30-driver   0/1     Completed           0          28m
spark-wordcount-f9355d7fcc95e284-driver   0/1     Completed           0          10h
PS C:\WINDOWS\system32> kubectl get pods -n 1155124983
Error from server (Forbidden): pods is forbidden: User "s1155124983" cannot list resource "pods" in API group "" in the
namespace "1155124983"
PS C:\WINDOWS\system32> kubectl get pods -n s1155124983
NAME                                      READY   STATUS              RESTARTS   AGE
pythonwordcount-d5d06b7fcef1d88e-exec-1   0/1     ContainerCreating   0          3s
pythonwordcount-d5d06b7fcef1d88e-exec-2   0/1     ContainerCreating   0          3s
spark-pi-89b2d27fcc45dd3e-driver          0/1     Completed           0          12h
spark-wordcount-00c5797fcef1b6ea-driver   1/1     Running             0          12s
spark-wordcount-33441d7fced2e037-driver   0/1     Completed           0          33m
spark-wordcount-a4db027fced4dfd2-driver   0/1     Completed           0          31m
spark-wordcount-c64c797fced7ac30-driver   0/1     Completed           0          28m
spark-wordcount-f9355d7fcc95e284-driver   0/1     Completed           0          10h
PS C:\WINDOWS\system32> kubectl delete pod spark-wordcount-00c5797fcef1b6ea-driver -n s1155124983
pod "spark-wordcount-00c5797fcef1b6ea-driver" deleted
PS C:\WINDOWS\system32> kubectl get pods -n s1155124983
NAME                                      READY   STATUS      RESTARTS   AGE
spark-pi-89b2d27fcc45dd3e-driver          0/1     Completed   0          12h
spark-wordcount-33441d7fced2e037-driver   0/1     Completed   0          37m
spark-wordcount-a4db027fced4dfd2-driver   0/1     Completed   0          34m
spark-wordcount-c64c797fced7ac30-driver   0/1     Completed   0          31m
spark-wordcount-f9355d7fcc95e284-driver   0/1     Completed   0          11h
PS C:\WINDOWS\system32>
```

```
22/03/28 13:14:05 INFO LoggingPodStatusWatcherImpl: State changed, new state:
        pod name: spark-wordcount-00c5797fcef1b6ea-driver
        namespace: s1155124983
        labels: spark-app-selector -> spark-88577bcc02d342b7a88b5e6e35adfea1, spark-role -> driver
        pod uid: 30e314d6-9bb2-4e21-a685-e62616af398c
        creation time: 2022-03-28T05:13:42Z
        service account name: spark
        volumes: spark-local-dir-1, spark-conf-volume-driver, spark-token-f8mw5
        node name: dicvm2.ie.cuhk.edu.hk
        start time: 2022-03-28T05:13:42Z
        phase: Running
        container status:
                container name: spark-kubernetes-driver
                container image: jessechan5171/spark-py:v3.2.1
                container state: running
                container started at: 2022-03-28T05:13:46Z
22/03/28 13:14:06 INFO LoggingPodStatusWatcherImpl: Application status for spark-88577bcc02d342b7a88b5e6e35adfea1 (phase: Running)
22/03/28 13:14:07 INFO LoggingPodStatusWatcherImpl: State changed, new state:
        pod name: spark-wordcount-00c5797fcef1b6ea-driver
        namespace: s1155124983
        labels: spark-app-selector -> spark-88577bcc02d342b7a88b5e6e35adfea1, spark-role -> driver
        pod uid: 30e314d6-9bb2-4e21-a685-e62616af398c
        creation time: 2022-03-28T05:13:42Z
        service account name: spark
        volumes: spark-local-dir-1, spark-conf-volume-driver, spark-token-f8mw5
        node name: dicvm2.ie.cuhk.edu.hk
        start time: 2022-03-28T05:13:42Z
        phase: Failed
        container status:
                container name: spark-kubernetes-driver
                container image: jessechan5171/spark-py:v3.2.1
                container state: terminated
                container started at: 2022-03-28T05:13:46Z
                container finished at: 2022-03-28T05:14:05Z
                exit code: 143
                termination reason: Error
22/03/28 13:14:07 INFO LoggingPodStatusWatcherImpl: Application status for spark-88577bcc02d342b7a88b5e6e35adfea1 (phase: Failed)
22/03/28 13:14:07 INFO LoggingPodStatusWatcherImpl: Container final statuses:


        container name: spark-kubernetes-driver
        container image: jessechan5171/spark-py:v3.2.1
        container state: terminated
        container started at: 2022-03-28T05:13:46Z
        container finished at: 2022-03-28T05:14:05Z
        exit code: 143
        termination reason: Error
22/03/28 13:14:07 INFO LoggingPodStatusWatcherImpl: Application spark-wordcount with submission ID s1155124983:spark-wordcount-00c5797fcef1b6ea-driver finished
22/03/28 13:14:07 INFO ShutdownHookManager: Shutdown hook called
22/03/28 13:14:07 INFO ShutdownHookManager: Deleting directory C:\Users\Not For Playing\AppData\Local\Temp\spark-79cf3fcf-5d57-411c-9f17-59d3e7557907
PS C:\Spark\spark-3.2.1-bin-hadoop3.2>
```

Driver is a single point of failure of a Spark application. The driver will not restart when it dies and the pods and the task will terminate.