CS 4843 Cloud Computing

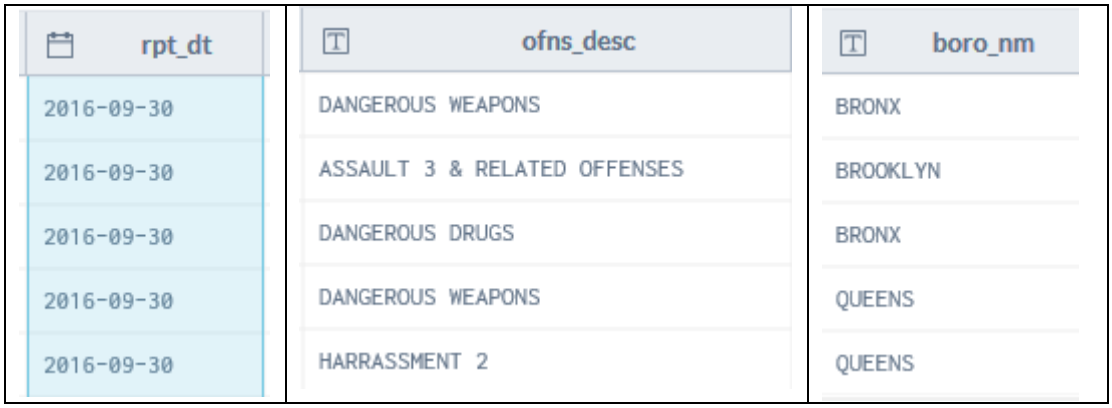
Assignment 2: Hadoop Cluster Setup and MapReduce Programming

1. Install and Setup Hadoop on your existing VMs as described in the lecture notes.

2. Download the New York City Crime Data from 2016.

   ```
   wget http://cs.utsa.edu/~plama/CS4843/NYPD_Complaint_Data_Current_YTD.csv
   ```

   The file contains 361,740 rows and 25 columns, total size 89.48 MB. The columns relevant to this assignment are shown below:



rpt_dt	ofns_desc	boro_nm
2016-09-30	DANGEROUS WEAPONS	BRONX
2016-09-30	ASSAULT 3 & RELATED OFFENSES	BROOKLYN
2016-09-30	DANGEROUS DRUGS	BRONX
2016-09-30	DANGEROUS WEAPONS	QUEENS
2016-09-30	HARRASSMENT 2	QUEENS

3. Create an input directory in HDFS, and copy the downloaded New York City Crime data file to HDFS as follows:

   ```
   cd $HADOOP_PREFIX
   bin/hadoop fs -mkdir /hw2-input
   bin/hadoop fs -put ~/NYPD_Complaint_Data_Current_YTD.csv /hw2-input/
   ```

4. Write a MapReduce program (in Python) that will answer the following ..
   (based on New York City Crime Data 2016). Run the program with only one reduce task.

   Where is most of the crime happening in New York?
   What is the total number of crimes reported in that location ?
   What types of crime are happening in that location (show unique crime types) ?

Reading CSV file:

The NYPD police report is a CSV file. Please note that some of the comma separated values in this file have commas embedded inside double quotes. Therefore, a simple split(“,”) function will incorrectly split those special values. In order to avoid this issue, you need to import and use Python’s CSV module as follows:

```
#!/usr/bin/env python2.7
from csv import reader
import sys

# skip first line (the header)
next(sys.stdin)

for line in reader(sys.stdin):
     boro, crime = (line[13].strip(), line[7].strip())
     if not boro or not crime:
       continue

     # rest of the code
```

Program Execution:

(a) Test your python programs locally on your master VM before running it in the Hadoop cluster.

To use Python2.7

cat NYPD_Complaint_Data_Current_YTD.csv | python2.7 hw2-mapper.py | sort | python2.7 hw2-reducer.py

To use Python3

cat NYPD_Complaint_Data_Current_YTD.csv | python hw2-mapper.py | sort | python hw2-reducer.py

(b) After uploading the crime data to HDFS, run your program in Hadoop cluster as follows:

To use Python2.7 (assuming that you have **`#!/usr/bin/env python2.7`** in the first line of your mapper and reducer programs).

```
$HADOOP_PREFIX/bin/hadoop  jar $HADOOP_PREFIX/contrib/streaming/hadoop-streaming-*.jar \
   -input /hw2-input \
   -output /hw2-output \
   -file /home/cc/hw2-mapper.py \
   -mapper /home/cc/hw2-mapper.py \
   -file /home/cc/hw2-reducer.py \
   -reducer /home/cc/hw2-reducer.py
```

To use Python3 (assuming that you have **`#!/usr/bin/env python`** in the first line of your mapper and reducer programs).

```
$HADOOP_PREFIX/bin/hadoop  jar $HADOOP_PREFIX/contrib/streaming/hadoop-streaming-*.jar \
   -input /hw2-input \
   -output /hw2-output \
   -file /home/cc/hw2-mapper.py \
   -mapper /home/cc/hw2-mapper.py \
   -file /home/cc/hw2-reducer.py \
   -reducer /home/cc/hw2-reducer.py \
   -cmdenv LC_CTYPE=en_GB.UTF-8
```

Troubleshooting Tips:

(c) If a job is long-running, and you want to free the shell for other activities, you can let the job run in the background by using:

```
Ctrl-C
```

(d) If a job hangs (making no progress), you can fetch the job id and kill the job as follows:

```
bin/hadoop job –list

bin/hadoop job –kill job_2014----
```

(e) To troubleshoot the task that took too long or failed, take the following steps:

- Check which tasks have failed on which worker nodes by using the following command

```
bin/hadoop job –history <output-directory>
```

for example:

```
FAILED task attempts by nodes
Hostname        FailedTasks
================================
hadoop-worker-3 task_201810011740_0012_m_000000, task_201810011740_0012_m_000001,
hadoop-worker-2 task_201810011740_0012_m_000000, task_201810011740_0012_m_000001,
hadoop-worker-1 task_201810011740_0012_m_000000, task_201810011740_0012_m_000001,
```

- Login to one of the worker nodes where task(s) have failed and check the corresponding “stderr” file under the directory,

```
/usr/local/hadoop-1.2.1/logs/userlogs
```

Submission Policy and Deliverables

Only one submission per group is required. Submission must include:

1. MapReduce programs (python files)

2. The resulting output files of your MapReduce programs should be downloaded from HDFS and submitted as a deliverable.

3. A PDF report that includes:
   a. Representative Screenshots of the console output when you execute your programs.
   b. Describe how the work was divided among your group members.