Universidad del Desarrollo
Facultad de Ingeniería

Hands-on II Big Data & Cloud Computing

MAYO - 2021

[redacted]

SR. Gustavo Adolfo Lobos Curamil

[redacted]

Universidad de Desarrollo
Facultad de Ingeniería

1. ¿Cuál fue el sitio web con mayor cantidad de eventos registrados el 6 de octubre del 2020?

   Al igual que en el primer Hands-on, las respuestas se repiten al utilizar Hive, Pig, Pyspark y Spark-SQL

   Spark-submit>

```
sshuser@hn0-pyspar:~$ cat question1.txt |tail -10
Row(mentionsourcename='reuters.com', cnt=7993)
Row(mentionsourcename='msn.com', cnt=7692)
Row(mentionsourcename='iheart.com', cnt=5390)
Row(mentionsourcename='yahoo.com', cnt=3575)
Row(mentionsourcename='dailymail.co.uk', cnt=2190)
Row(mentionsourcename='allafrica.com', cnt=2008)
Row(mentionsourcename='washingtontimes.com', cnt=1763)
Row(mentionsourcename='indiatimes.com', cnt=1632)
Row(mentionsourcename='wickedlocal.com', cnt=1558)
Row(mentionsourcename='chron.com', cnt=1321)
sshuser@hn0-pyspar:~$
```

   Script:

```
sshuser@hn0-pyspar:~$ cat spark1.py
from pyspark import SparkConf, SparkContext
from pyspark.sql import HiveContext
from pyspark.sql.functions import col,desc

conf = SparkConf().setAppName("test-magister")
sc = SparkContext(conf=conf)
hiveCtx = HiveContext(sc)
df = hiveCtx.sql("select * from mentions2")

query=
df.filter(df.mentiontimedate.like("%20201006%")).groupBy('mentionsourcename
').count().select('mentionsourcename',col('count').alias('cnt')).sort(desc(
'cnt')).limit(10)
for x in query.collect():
    print(x)
```

Utilizando Spark-SQL se llega a la misma conclusión que con Hive y PIG.

```
spark-sql> select mentionsourcename, count(*) from default.mentions2 where mentiontimedate like '20201006%' group by mentionsourcename order by 2 desc limit 10;
21/05/17 02:05:12 WARN SessionState [main]: METASTORE_FILTER_HOOK will be ignored, since hive.security.authorization.manager is set to instance of HiveAuthorizerFactory.
reuters.com     7993
msn.com 7692
iheart.com      5390
yahoo.com       3575
dailymail.co.uk 2190
allafrica.com   2008
washingtontimes.com     1763
indiatimes.com  1632
wickedlocal.com 1558
chron.com       1321
spark-sql>
```

2. ¿Cuál es la ciudad con mayor número de apariciones en los eventos registrados el 6 de octubre del 2020?

   El primer lugar, sigue siendo para los artículos sin nombre o lugar relacionado.
   Spark-submit>

```
sshuser@hn0-pyspar:~$ cat question2v2.txt
/usr/hdp/current/spark3-client/python/lib/pyspark.zip/pyspark/context.py:223: DeprecationWarning: Support for Python 2 and Python 3
 See also the plan for dropping Python 2 support at https://spark.apache.org/news/plan-for-dropping-python-2-support.html.
  DeprecationWarning)
Row(actor2geo_fullname='                                                                                                    ', cnt=37600)
Row(actor2geo_fullname='White House, District of Columbia, United States                                                    ', cnt=2221)
Row(actor2geo_fullname='United States                                                                                       ', cnt=1748)
Row(actor2geo_fullname='Washington, District of Columbia, United States                                                     ', cnt=1438)
Row(actor2geo_fullname='New York, United States                                                                             ', cnt=1434)
Row(actor2geo_fullname='United Kingdom                                                                                      ', cnt=1400)
Row(actor2geo_fullname='London, London, City of, United Kingdom                                                             ', cnt=1012)
Row(actor2geo_fullname='China                                                                                               ', cnt=988)
Row(actor2geo_fullname='Texas, United States                                                                                ', cnt=823)
Row(actor2geo_fullname='California, United States                                                                           ', cnt=817)
sshuser@hn0-pyspar:~$
```

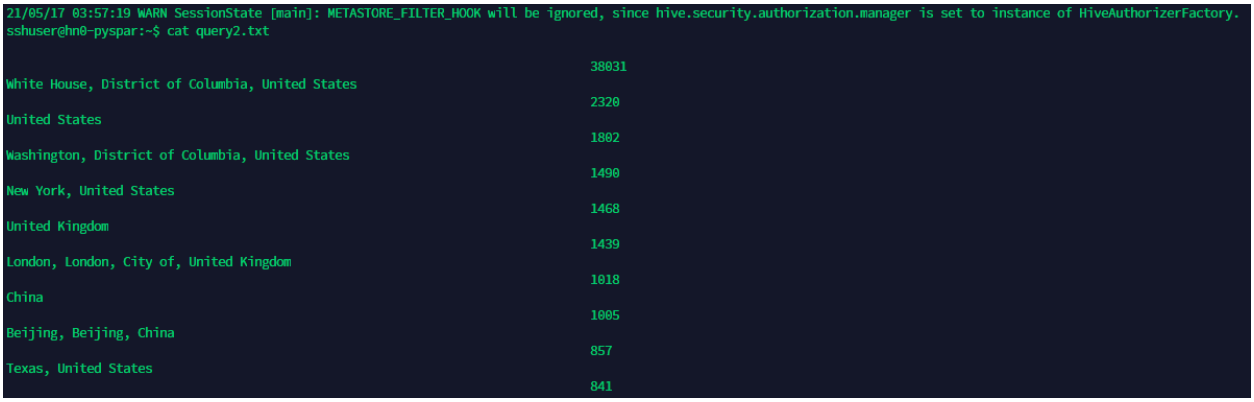
Script:

```
sshuser@hn0-pyspar:~$ cat spark2v2.py
from pyspark import SparkConf, SparkContext
from pyspark.sql import HiveContext
from pyspark.sql.functions import col,desc

conf = SparkConf().setAppName("test-magister")
sc = SparkContext(conf=conf)
hiveCtx = HiveContext(sc)
df = hiveCtx.sql("select * from exports1")
```

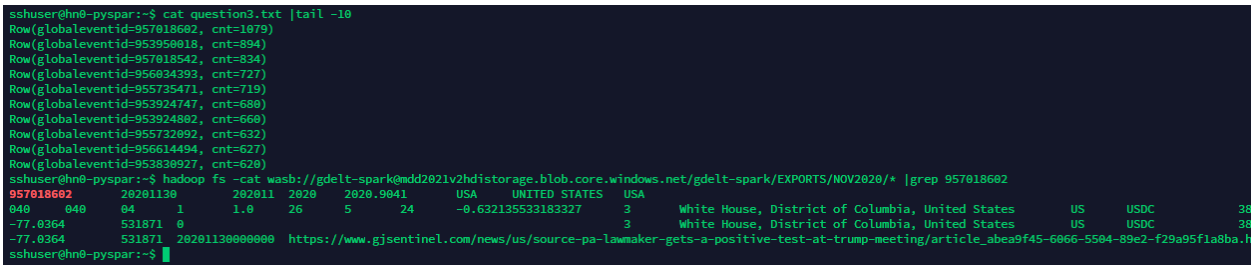
```
query=
df.filter(df.sqldate.like("20201006")).groupBy('actor2geo_fullname').
count().select('actor2geo_fullname',col('count').alias('cnt')).sort(d
esc('cnt')).limit(10)
for x in query.collect():
    print(x)
sshuser@hn0-pyspar:~$
```

Utilizando Spark-SQL pero ejecutando la query desde un archivo con la opción -f arroja lo siguiente:



3. ¿Cuál fue el evento generado durante noviembre del 2020 con mayor cantidad de menciones en los medios?

   Tanto como en Spark-SQL y Pyspark el ID con el evento con mayores menciones se repite y se llega a la misma conclusión que con el Hands-on I Spark-submit>




```
sshuser@hn0-pyspar:~$ cat spark3.py
from pyspark import SparkConf, SparkContext
from pyspark.sql import HiveContext
from pyspark.sql.functions import col,desc

conf = SparkConf().setAppName("test-magister")
sc = SparkContext(conf=conf)
hiveCtx = HiveContext(sc)
df = hiveCtx.sql("select * from mentions3")

query=df.filter(df.mentiontimedate.like("%202011%")).groupBy('globale
ventid').count().select('globaleventid',col('count').alias('cnt')).so
rt(desc('cnt')).limit(10)
for x in query.collect():
print(x)
```

Spark-SQL>

4. ¿Cuáles son los 10 sitios web que generaron en promedio la mayor cantidad de menciones poco confiables (<40% de confianza) durante noviembre del 2020?

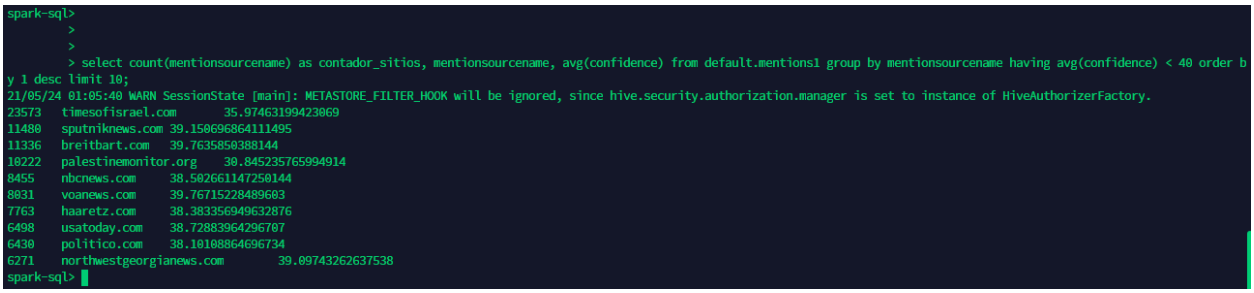
   Spark-submit>

```
sshuser@hn0-mddspa:~$ cat salida
/usr/hdp/current/spark3-client/python/lib/pyspark.zip/pyspark/context.py:223:
ersion 3.6 is deprecated as of Spark 3.0. See also the plan for dropping Pytho
ython-2-support.html.
  DeprecationWarning)
+-------------------+-----------------+-------------------+-----+
|   mentionsourcename|   avg(confidence)|   mentionsourcename|count|
+-------------------+-----------------+-------------------+-----+
|   timesofisrael.com| 35.97463199423069|   timesofisrael.com|23573|
|     sputniknews.com|39.150696864111495|     sputniknews.com|11480|
|       breitbart.com|  39.7635850388144|       breitbart.com|11336|
|palestinemonitor.org|30.845235765994914|palestinemonitor.org|10222|
|         nbcnews.com|38.502661147250144|         nbcnews.com| 8455|
|         voanews.com| 39.76715228489603|         voanews.com| 8031|
|         haaretz.com|38.383356949632876|         haaretz.com| 7763|
|        usatoday.com| 38.72883964296707|        usatoday.com| 6498|
|        politico.com| 38.10108864696734|        politico.com| 6430|
|northwestgeorgian...| 39.09743262637538|northwestgeorgian...| 6271|
|middleeastmonitor...|39.817358978503314|middleeastmonitor...| 6187|
|         madison.com| 39.76015663240333|         madison.com| 6129|
|        albawaba.com|39.629564913506904|        albawaba.com| 5723|
|   middleeasteye.net|35.367729831144466|   middleeasteye.net| 5330|
|         cnsnews.com|  38.8968315301391|         cnsnews.com| 5176|
|      al-monitor.com|38.562538892345984|      al-monitor.com| 4821|
|         qctimes.com| 39.77743835915339|         qctimes.com| 4583|
|      algemeiner.com|  36.3888169816205|      algemeiner.com| 3863|
|    eagletribune.com| 39.96756756756757|    eagletribune.com| 3700|
|           rferl.org|  39.1976911976912|           rferl.org| 3465|
+-------------------+-----------------+-------------------+-----+
only showing top 20 rows
```

```
from pyspark.sql.functions import when, sum, avg, col
from pyspark import SparkConf, SparkContext
from pyspark.sql import HiveContext
from pyspark.sql.functions import col,desc
from pyspark.sql.types import IntegerType
from pyspark.sql.window import Window
```

```
        conf = SparkConf().setAppName("test-magister")
        sc = SparkContext(conf=conf)
        hiveCtx = HiveContext(sc)
        df = hiveCtx.sql("select * from mentions1")
        columnas = df.groupby('mentionsourcename').avg('confidence')
        columnas2 = df.groupby('mentionsourcename').count()
        resultado = columnas.join(columnas2, columnas.mentionsourcename ==
        columnas2.mentionsourcename)
        resultado2 = resultado.where(resultado['avg(confidence)'] <
        40).sort(desc('count'))
resultado2.show()
```

Spark-SQL>




```
select count(mentionsourcename) as contador_sitios, mentionsourcename,
avg(confidence) from default.mentions1 group by mentionsourcename having
avg(confidence) < 40 order by 1 desc limit 10;
```

5. ¿Cuáles son los 10 sitios web que generaron en promedio la mayor cantidad de menciones muy confiables (>80% de confianza) durante septiembre, octubre, noviembre y diciembre del 2020? (promedio de todas las apariciones durante los 4 meses)
   Pyspark-submit>

```
sshuser@hn0-mddspa:~$ cat salida2
/usr/hdp/current/spark3-client/python/lib/pyspark.zip/pyspark/context
ecated as of Spark 3.0. See also the plan for dropping Python 2 suppor
  DeprecationWarning)
+--------------------+-----------------+--------------------+-----+
|   mentionsourcename|  avg(confidence)|   mentionsourcename|count|
+--------------------+-----------------+--------------------+-----+
|            meed.com|88.49744245524296|            meed.com| 3128|
|            live.com|99.97919916796671|            live.com| 1923|
|business2communit...|84.02045633359559|business2communit...| 1271|
|   lewistownnews.com|89.87676056338029|   lewistownnews.com| 1136|
|      coolsocial.net|97.59024390243903|      coolsocial.net| 1025|
|      alleywatch.com|92.25581395348837|      alleywatch.com|  860|
|androidheadlines.com|80.08578431372548|androidheadlines.com|  816|
|     justjaredjr.com|80.14492753623189|     justjaredjr.com|  759|
|        energyfm.net|96.93989071038251|        energyfm.net|  732|
|   androidpolice.com| 82.3013698630137|   androidpolice.com|  730|
|marketbusinessnew...|85.76223776223776|marketbusinessnew...|  715|
|              bgr.in| 80.1015228426396|              bgr.in|  591|
|           rexmd.com|            100.0|           rexmd.com|  518|
|seminolesentinel.com|99.43775100401606|seminolesentinel.com|  498|
|    rustonleader.com| 93.9323467230444|    rustonleader.com|  473|
|      chinavitae.com|            100.0|      chinavitae.com|  470|
|          heartfm.ca|82.11267605633803|          heartfm.ca|  426|
| themobileindian.com|80.43147208121827| themobileindian.com|  394|
|androidcommunity.com|80.77127659574468|androidcommunity.com|  376|
|stamfordmercury.c...|89.73190348525469|stamfordmercury.c...|  373|
+--------------------+-----------------+--------------------+-----+
only showing top 20 rows

sshuser@hn0-mddspa:~$
```

```
        from pyspark.sql.functions import when, sum, avg, col
        from pyspark import SparkConf, SparkContext
        from pyspark.sql import HiveContext
        from pyspark.sql.functions import col,desc
        from pyspark.sql.types import IntegerType
        from pyspark.sql.window import Window
        from pyspark.sql.functions import broadcast
        from pyspark.sql import SparkSession

        spark = SparkSession.builder.appName("test-
        magister").config("spark.sql.broadcastTimeout", 36000)
        conf = SparkConf().setAppName("test-magister")
        sc = SparkContext(conf=conf)
        hiveCtx = HiveContext(sc)
        df = hiveCtx.sql("select * from mentions3")
        columnas = df.groupby('mentionsourcename').avg('confidence')
        columnas2 = df.groupby('mentionsourcename').count()
        resultado = columnas.join(columnas2, columnas.mentionsourcename ==
        columnas2.mentionsourcename).persist()
        resultado2 = resultado.where(resultado['avg(confidence)'] >
        80).sort(desc('count'))
resultado2.show()
```

Spark-SQL>

```
spark-sql> select count(mentionsourcename) as contador_sitios, mentionsourcename, avg(confidence) from default.mentions1 group by mentionsourcename
y 1 desc limit 10;
758     meed.com        85.44854881266491
244     live.com        100.0
235     alleywatch.com  97.87234042553192
222     wrestlingnewssource.com 81.08108108108108
211     lewistownnews.com       88.67298578199052
209     highlandradio.com       83.11004784688996
182     coolsocial.net  98.07692307692308
172     business2community.com  87.73255813953489
162     energyfm.net    96.54320987654322
161     androidcentral.com      80.6832298136646
spark-sql> client_loop: send disconnect: Connection reset by peer
```

```
select count(mentionsourcename) as contador_sitios, mentionsourcename,
avg(confidence) from default.mentions1 group by mentionsourcename having
avg(confidence) > 80 order by 1 desc limit 10;
```

6. Diseñe una consulta que le parezca relevante y que involucre cruzar los datasets export y mentions usando la llave GlobalEventID (primeros campos en ambos).
    a. ¿Cuáles fueron los sitios chilenos que generaron mayor cantidad de eventos con un nivel de confianza superior 80% durante el mes de diciembre en 2020?

Parte II

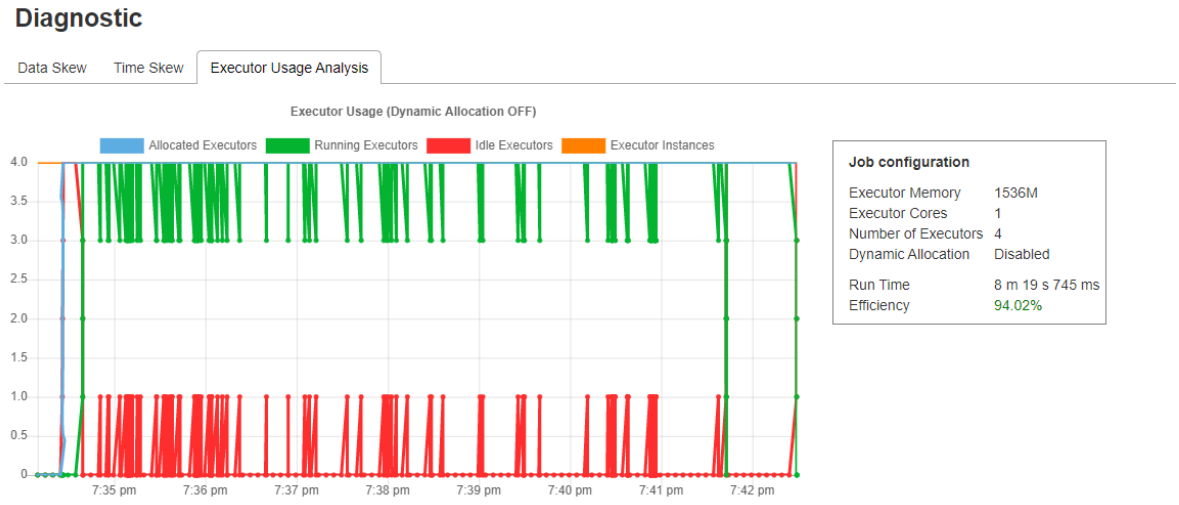
1. Elija una de las consultas de la parte I y realice un análisis de performance usando Spark UI y el cálculo de eficiencia de la ejecución (3 puntos).

   Para esta parte del Hands-On se ha escogido la consulta Nro. 5, que, al contemplar la totalidad de archivos de menciones de todos meses, ejecuta cálculos de la media para los niveles de confiabilidad para finalmente ordenar la salida para agrupar los sitios con mayor conteo de mediciones y un 'confidence' superior al 80%. Esto en términos de performance es la que mayor tiempo demanda, por lo cual al revisar el Spark UI la task duró más de 10 minutos.

2. Modifique los parámetros de ejecución para mejorar el performance, ¿cuál fue la configuración que escogió y cuál puede ser la explicación de esto? (3 puntos)

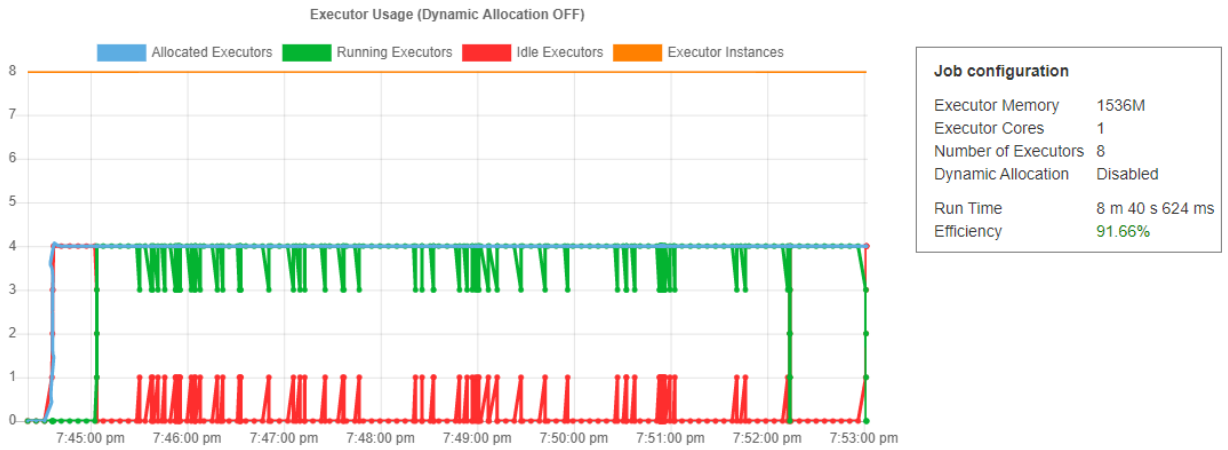
   En la primera prueba se comenzó a usar el cluster completo y, además, se elevó el numero de executors pasando de 2 a 4, esto tuvo un impacto directo en el tiempo y un leve aumento en la eficiencia de 92.79% a 94.02%.

```
# spark-submit --master yarn --num-executors 4 spark5.py
> performance
```



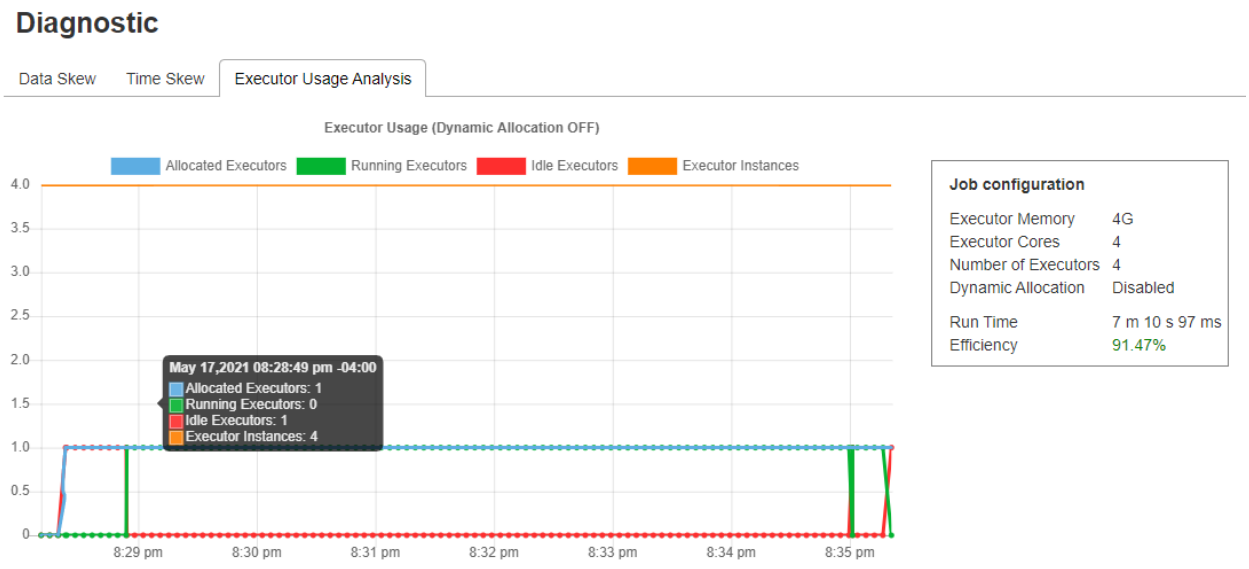
Se realizó una tercera prueba utilizando 8 executors, sin embargo, esto no tuvo el efecto deseado ya que el cluster HDInsight armado no dispone de más de 4 executors en el entorno de Spark. Junto con esto, la performance empeoró y el tiempo no se vio alterado.

```
# spark-submit --master yarn --num-executors 8 spark5.py
> performance
```



En última instancia, en busca una performance óptima, se ejecutó nuevamente el script pero en esta oportunidad se elevó la memoria, se corrigió el número de executors y se aumentó el número de núcleos utilizados.
Esto se tradujo en que el tiempo bajó, pero la performance a pesar de ser aceptable se redujo también.

```
#spark-submit --master yarn --num-executors 4 --
executor-cores 4 --executor-memory 4G spark5.py >
performance
```



Conclusiones:

Para la pregunta Nro. 5 en específico la mejor performance se encontró utilizando el cluster entero y aumentando la cantidad de executors, pero sin exceder la cantidad máxima de estos disponibles. Setear la memoria al máximo posible tampoco parece ayudar, esto entorpece el funcionamiento del cluster y degrada la performance.

Para tiempos de procesamiento alto, pero con tareas breves, lo mejor es disponer de varios executores, ya que disponer de memoria en tareas donde no se utiliza toda, degrada la performance.