PRÁCTICA Nº 2.10
Tratamiento de datos con diferentes herramientas Big Data

HIVE

Apache Pig

Apache Sqoop

- Nombre y apellidos: Alvaro Lucio-Villegas de Cea

Índice

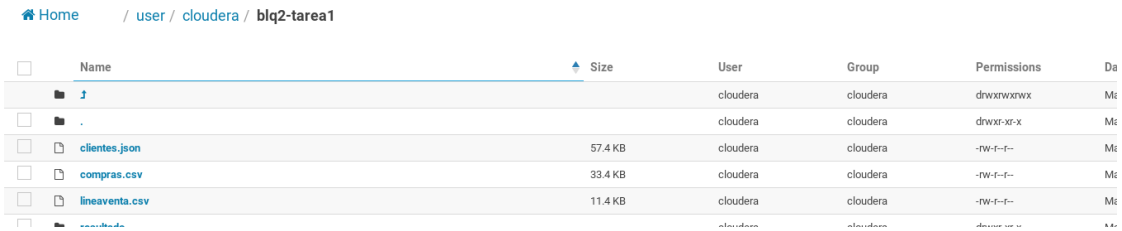
1. Definir origen de datos

Los datos se han generado utilizando https://www.mockaroo.com/

Los ficheros clientes.json , productos.sql y lineaventa.csv se encuentran en /home/cloudera/Desktop/BLQ2-Tarea1

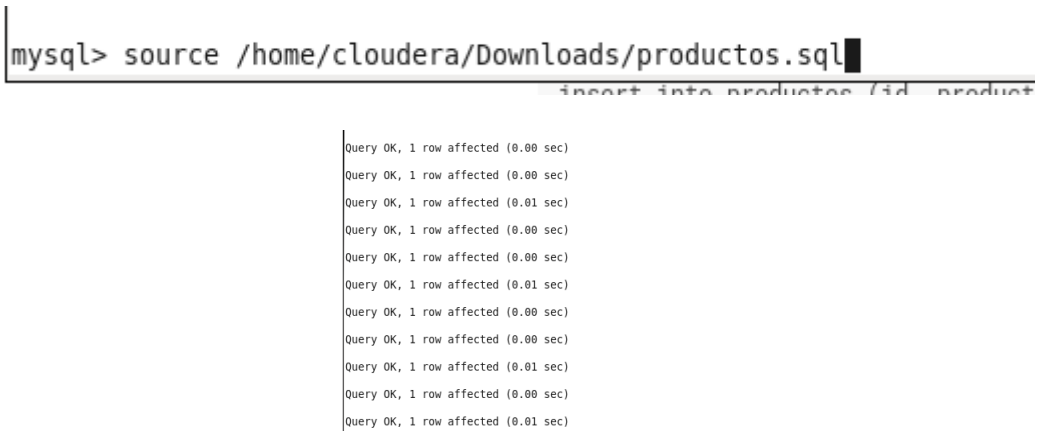
1.1. Subir compras.csv a HDFS

```
hdfs dfs -mkdir blq2-tarea1
hdfs dfs -put /home/cloudera/Desktop/blq2-tarea1/clientes.json
hdfs dfs -put /home/cloudera/Desktop/blq2-tarea1/lineaventa.csv
```



1.2. Crear tabla clientes y cargar datos

```
>mysql -u root -p (passwd cloudera)
>source /home/cloudera/Dowloads/productos.sql
```



2. Tratamiento Sqoop

Origen: tabla productos de la base de datos prueba

Destino: /user/cloudera/blq2-tarea1/tabla_productos

2.1. Definir sentencia Sqoop

```
sqoop import
--connect jdbc:mysql://localhost/mibd
--username root --password cloudera
 --table productos
--target-dir /user/cloudera/blq2-tarea1/tabla_productos -m 1
```

Final de la ejecución.

```
        Job Counters
                Launched map tasks=4
                Other local map tasks=4
                Total time spent by all maps in occupied slots (ms)=18964
                Total time spent by all reduces in occupied slots (ms)=0
                Total time spent by all map tasks (ms)=18964
                Total vcore-milliseconds taken by all map tasks=18964
                Total megabyte-milliseconds taken by all map tasks=19419136
        Map-Reduce Framework
                Map input records=1000
                Map output records=1000
                Input split bytes=408
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=606
                CPU time spent (ms)=3570
                Physical memory (bytes) snapshot=883064832
                Virtual memory (bytes) snapshot=6357393408
                Total committed heap usage (bytes)=1092616192
        File Input Format Counters
                Bytes Read=0
        File Output Format Counters
                Bytes Written=54433
23/05/03 10:36:41 INFO mapreduce.ImportJobBase: Transferred 53.1572 KB in 19.3736 seconds (2.7438 KB/sec)
23/05/03 10:36:41 INFO mapreduce.ImportJobBase: Retrieved 1000 records.
[cloudera@quickstart ~]$
```

Resultado en HDFS

3. Tratamiento Pig

Vamos a cargar las distintas tablas con la herramienta de Pig teniendo en cuenta que son ficheros con distintas extensiones.

3.1. Crear fichero importacion_lineaventa_clientes_pig.pig

```
lineaventa = LOAD 'blq2-tarea1/lineaventa.csv' USING PigStorage(',') AS (idlinea:int,cliente:int,producto:int);


STORE lineaventa INTO '/user/cloudera/blq2-tarea1/tabla_lineaventas' USING PigStorage(';');


clientes = LOAD 'blq2-tarea1/clientes.json' USING JsonLoader('id:int,cliente:chararray,fecha:chararray');


STORE clientes INTO '/user/cloudera/blq2-tarea1/tabla_clientes' USING PigStorage(';');
```

Final del comando de ejecución del script.

```
Successfully stored 758 records (6539 bytes) in: "/user/cloudera/blq2-tarea1/tabla_ventas"

Counters:
Total records written : 758
Total bytes written : 6539
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0

Job DAG:
job_1681898044415_0029


2023-05-03 10:52:43,238 [main] WARN  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Encountered Warning FIELD_DISCARDED_TYPE_CONVERSION_FAILED 194 time(s).
2023-05-03 10:52:43,238 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!
[cloudera@quickstart ~]$
```

Resultado en HDFS

Tabla línea de venta.

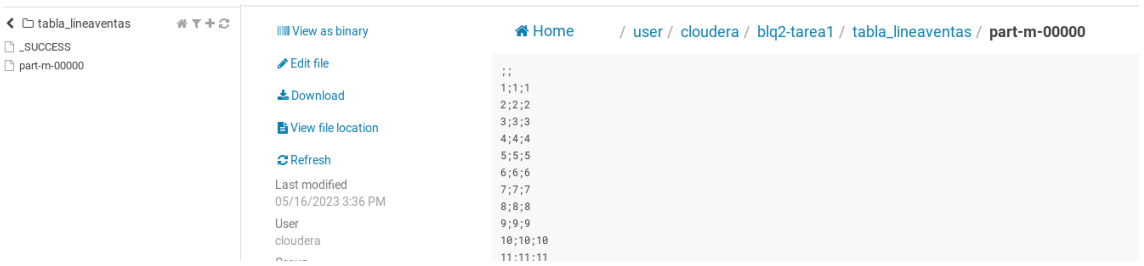
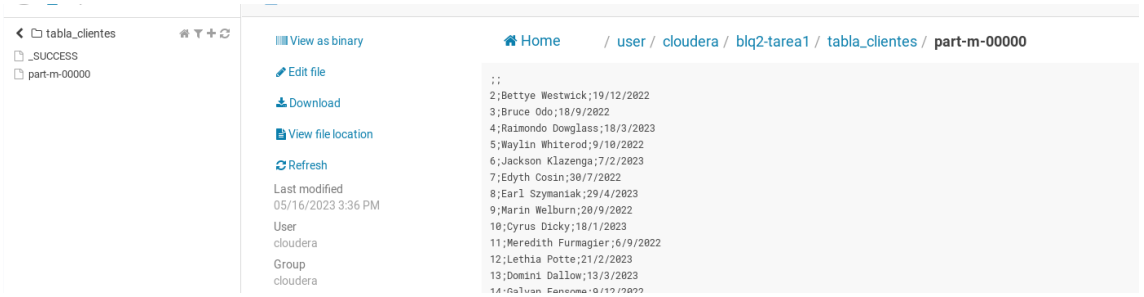


Tabla clientes.



4. Tratamiento Hive

Origen de información:

/user/cloudera/blq2-tarea1/tabla_productos

/user/cloudera/blq2-tarea1/tabla_lineaventas

/user/cloudera/blq2-tarea1/tabla_clientes

Vamos a crear una consulta para que me muestre y me guarde el resultado de la consulta en un fichero en este caso la consulta trata de buscar al cliente que ha comprado la cosa más cara.

4.1. Script hive

```
create database if not exists blq2tarea1;
use blq2tarea1;

drop table if exists lineaventa;
create external table lineaventa (idlinea int,cliente int,producto int) row format
delimited fields terminated by '\073' STORED AS TEXTFILE LOCATION
'/user/cloudera/blq2-tarea1/tabla_lineaventas';

drop table if exists producto;
create external table producto (id int, producto string, precio int) row format
delimited fields terminated by ',' STORED AS TEXTFILE LOCATION
'/user/cloudera/blq2-tarea1/tabla_productos';

drop table if exists clientes;
create external table clientes (id int,cliente string,fecha string) row format delimited
fields terminated by '\073' STORED AS TEXTFILE LOCATION
'/user/cloudera/blq2-tarea1/tabla_clientes';

INSERT OVERWRITE DIRECTORY '/user/cloudera/blq2-tarea1/resultado' row format
delimited fields terminated by '\073' select max(precio) as
precio_maximo,p.producto , c.cliente , c.fecha from lineaventa as lv , clientes as c ,
producto as p where lv.cliente = c.id and lv.producto = p.id group by c.cliente ,
c.fecha, p.producto , p.precio order by precio_maximo DESC limit 1;
```

Ejecución en HUE del Script


```
create database if not exists blq2tarea1;
use blq2tarea1;

drop table if exists lineaventa;
create external table lineaventa (idlinea int,cliente int,producto int) row format delimited fields terminated by '\073' STORED AS TEXTFILE LOCATION '/user/cloudera/blq2-tarea1/tabla_lin

drop table if exists producto;
create external table producto (id int, producto string, precio int) row format delimited fields terminated by ',' STORED AS TEXTFILE LOCATION '/user/cloudera/blq2-tarea1/tabla_productos

drop table if exists clientes;
create external table clientes (id int,cliente string,fecha string) row format delimited fields terminated by '\073' STORED AS TEXTFILE LOCATION '/user/cloudera/blq2-tarea1/tabla_cliente

INSERT OVERWRITE DIRECTORY '/user/cloudera/blq2-tarea1/resultado' row format delimited fields terminated by '\073'
select max(precio) as precio_maximo , c.nombre , c.fecha from lineaventa as lv , clientes as c , producto as p
where lv.cliente = c.id and lv.producto = p.id
group by c.cliente , c.fecha, p.producto , p.precio
order by precio_maximo DESC limit 1;
```

Resultado en HDFS

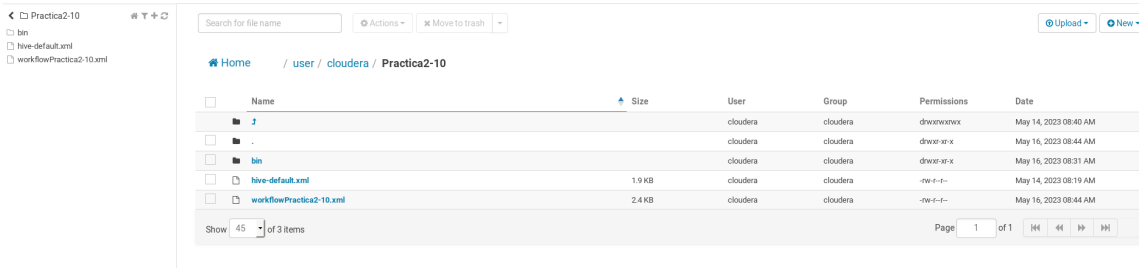
5. Unir todo con Oozie

5.1. Crear una carpeta en LOCAL y en HDFS (se replicará la información)

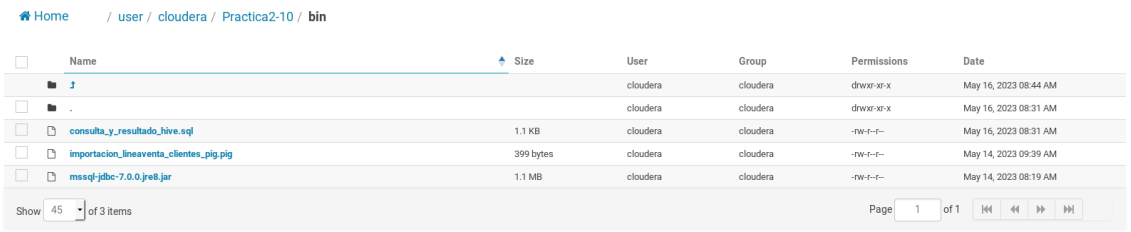
5.2. Unir todos los scripts anteriores en subcarpeta bin

Será necesario crear un directorio en HDFS que llamaremos “Practica2-10” y pondremos el “hive-default.xml” a la misma altura que el xml.

*Este archivo le deberemos de cambiar el nombre de hive-site.xml con el nombre hive-dafault.xml.Este fichero se encuentra en en “/etc/hive/conf.dist/hive-site.xml”



Dentro del directorio /bin tendremos los scripts que creamos anteriormente.



5.3. Definir un workflow

```
<workflow-app name='practica_2-10' xmlns="uri:oozie:workflow:0.1">
        <start to="forking" />
        <fork name="forking">
                <path start="importacion_productos_sqoop" />
                <path start="importacion_lineaventa_clientes_pig" />
        </fork>
        <action name="importacion_productos_sqoop">
                <sqoop xmlns="uri:oozie:sqoop-action:0.2">
                        <job-tracker>${jobTracker}</job-tracker>
                        <name-node>${nameNode}</name-node>
                        <prepare>
                                <delete
path="${nameNode}/user/cloudera/blq2-tarea1/tabla_productos" />
                        </prepare>
                        <configuration>
                                <property>

<name>mapred.job.queue.name</name>
                                        <value>${queueName}</value>
                                </property>
                        </configuration>
                        <command>import --connect
jdbc:mysql://localhost/mibd --username root --password cloudera --table
productos --target-dir /user/cloudera/blq2-tarea1/tabla_productos -m
1</command>
                </sqoop>
                <ok to="joining" />
                <error to="kill" />
        </action>
        <action name="importacion_lineaventa_clientes_pig">
                <pig>
                        <job-tracker>${jobTracker}</job-tracker>
                        <name-node>${nameNode}</name-node>
                        <prepare>
                                <delete
path="${nameNode}/user/cloudera/blq2-tarea1/tabla_lineaventas" />
                        <delete
path="${nameNode}/user/cloudera/blq2-tarea1/tabla_clientes" />
                        </prepare>
                        <configuration>
                                <property>
```

```
<name>mapred.job.queue.name</name>
                                        <value>root.default</value>
                                    </property>
                            </configuration>
                            <script>
                                    bin/importacion_lineaventa_clientes_pig.pig
                            </script>
                    </pig>
                    <ok to="joining" />
                    <error to="kill" />
            </action>
            <join name="joining" to="consulta_y_resultado_hive" />
            <action name="consulta_y_resultado_hive">
                    <hive xmlns="uri:oozie:hive-action:0.2">
                            <job-tracker>${jobTracker}</job-tracker>
                            <name-node>${nameNode}</name-node>
                            <configuration>
                                    <property>

<name>mapred.job.queue.name</name>
                                        <value>${queueName}</value>
                                    </property>
                        <property>
                            <name>ozzie.hive.default</name>

<value>${nameNode}/user/cloudera/Practica2-10/hive-default.xml</value>
                        </property>
                            </configuration>
                            <script>
                                    bin/consulta_y_resultado_hive.sql
                            </script>
                    </hive>
                    <ok to='end' />
                    <error to='kill' />
            </action>
            <kill name='kill'>
                    <message>
                            Se rompio, mensaje de error
[${wf:errorMessage(wf:lastErrorNode())}]
                    </message>
            </kill>
            <end name='end' />
</workflow-app>
```

5.4. Definir jobPractica2-10.properties

```
nameNode=hdfs://localhost:8020

jobTracker=localhost:8032

queueName=default

oozie.use.system.libpath=true

oozie.wf.application.path=${nameNode}/user/${user.name}/Practica2-10/workflowPractica2-10.xml
```

5.5. Lanzar Oozie

```
export OOZIE_URL="http://localhost:11000/oozie"

oozie job --oozie http://localhost:11000/oozie -config jobPractica2-10.properties -run
```

```
job: 0000001-230508124210163-oozie-oozi-W
[cloudera@quickstart map-reduce]$ oozie job --oozie http://localhost:11000/oozie -config jobPractica2-10.properties -run
job: 0000002-230508124210163-oozie-oozi-W
[cloudera@quickstart map-reduce]$
```

Posibles problemas y soluciones

Fichero SQL

En este fichero me he encontrado el error de codificación ya que el workflow no identifica el carácter ";" y se debe de reemplazar por los caracteres "\073" que es el mismo carácter pero con la codificación correspondiente.

Fichero Workflow.xml

En este fichero debemos agregar al workflow una propiedad en la configuración de la cola de los procesos para que no se sature y funcione correctamente.

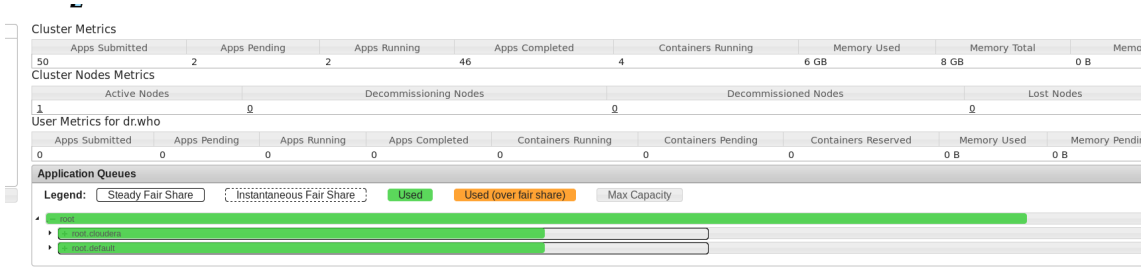
Para ello agregaremos el siguiente comando ya sea en la parte del workflow de pig como de Hive.Pero solo en una de ellas.

Lo que estamos haciendo es mandar ese job a otra cola diferente al job anterior, en este caso lo mandamos a la cola del usuario root.

```
<configuration>
        <property>
                <name>mapred.job.queue.name</name>
                <value>root.default</value>
        </property>
</configuration>
```

```
</action>
<action name="importacion_ventas_pig">
        <pig>
                <job-tracker>${jobTracker}</job-tracker>
                <name-node>${nameNode}</name-node>
                <prepare>
                        <delete path="${nameNode}/user/cloudera/blq2-tarea1/tabla_ventas" />
                </prepare>
                <configuration>
                        <property>
                                <name>mapred.job.queue.name</name>
                                <value>root.default</value>
                        </property>
                </configuration>
                <script>
                        bin/importacion_ventas_pig.pig
                </script>
```

Podemos observar cómo se están realizando los dos trabajos de forma paralela en el panel de control de yarm.

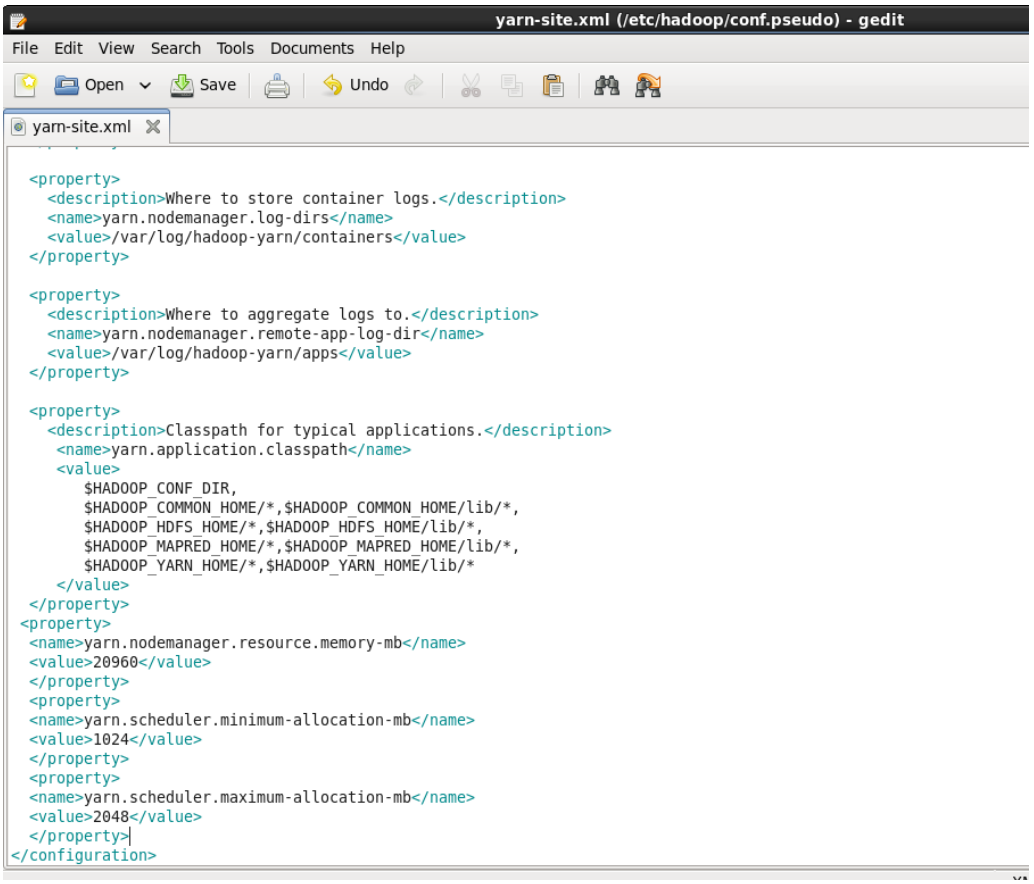


También nos sucederá que el proceso se queda procesando continuamente y esto sucede por la memoria asignada a los procesos. Para solucionar este problema tendremos que dirigirnos al fichero de configuración de yarn, para asignarle una nueva configuración.

El fichero de configuración se encuentra en :"etc/hadoop/conf.pseudo/yarn-site.xml"

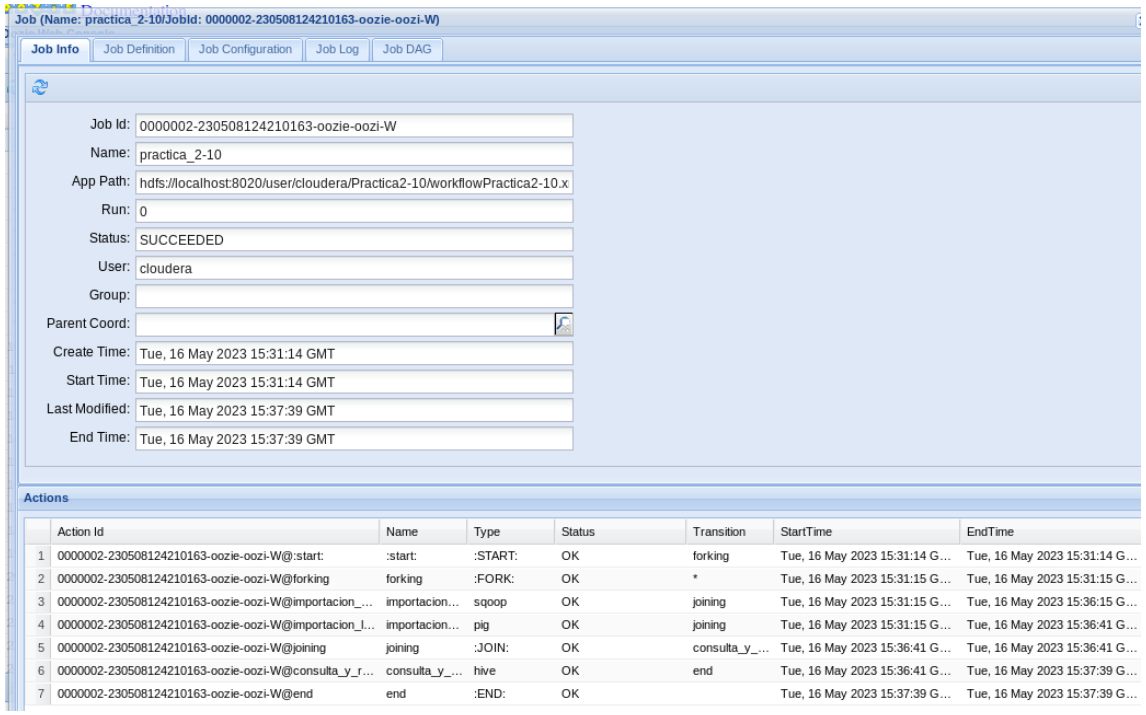
Una vez localizado el fichero será necesario agregar estas líneas de comando.

```
<property>
  <name>yarn.nodemanager.resource.memory-mb</name>
  <value>20960</value>
</property>
<property>
  <name>yarn.scheduler.minimum-allocation-mb</name>
  <value>1024</value>
</property>
<property>
  <name>yarn.scheduler.maximum-allocation-mb</name>
  <value>2048</value>
</property>
```


yarn-site.xml (/etc/hadoop/conf.pseudo) - gedit
File Edit View Search Tools Documents Help
Open
Save
Undo
yarn-site.xml
<property>
<description>Where to store container logs.</description>
<name>yarn.nodemanager.log-dirs</name>
<value>/var/log/hadoop-yarn/containers</value>
</property>
<property>
<description>Where to aggregate logs to.</description>
<name>yarn.nodemanager.remote-app-log-dir</name>
<value>/var/log/hadoop-yarn/apps</value>
</property>
<property>
<description>Classpath for typical applications.</description>
<name>yarn.application.classpath</name>
<value>
$HADOOP_CONF_DIR,
$HADOOP_COMMON_HOME/*,$HADOOP_COMMON_HOME/lib/*,
$HADOOP_HDFS_HOME/*,$HADOOP_HDFS_HOME/lib/*,
$HADOOP_MAPRED_HOME/*,$HADOOP_MAPRED_HOME/lib/*,
$HADOOP_YARN_HOME/*,$HADOOP_YARN_HOME/lib/*
</value>
</property>
<property>
<name>yarn.nodemanager.resource.memory-mb</name>
<value>20960</value>
</property>
<property>
<name>yarn.scheduler.minimum-allocation-mb</name>
<value>1024</value>
</property>
<property>
<name>yarn.scheduler.maximum-allocation-mb</name>
<value>2048</value>
</property>
</configuration>

Resultado Final

Observamos el job y los subprocesos que genera con sus respectivos estados.



Nos dirigimos a la carpeta “blq2-tarea1/resultado/0000_0” en este archivo podemos observar el resultado final del workflow.