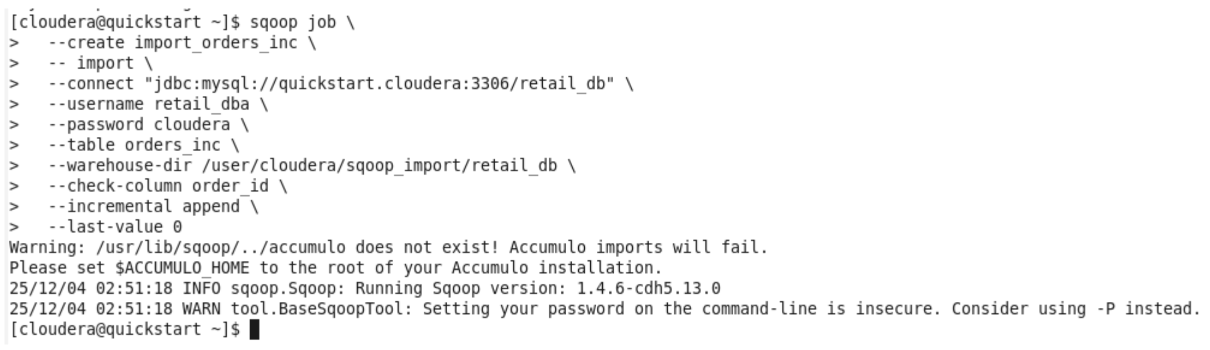
Name: Ilyas Hakkou

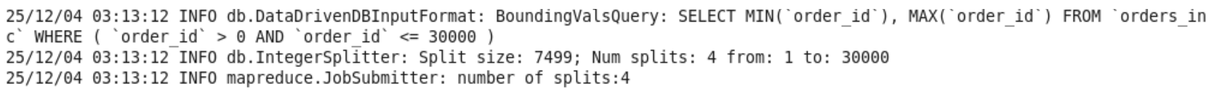
Lab Sqoop

1. Executing the job:



```
[cloudera@quickstart ~]$ sqoop job \
>   --create import_orders_inc \
>   -- import \
>   --connect "jdbc:mysql://quickstart.cloudera:3306/retail_db" \
>   --username retail_dba \
>   --password cloudera \
>   --table orders_inc \
>   --warehouse-dir /user/cloudera/sqoop_import/retail_db \
>   --check-column order_id \
>   --incremental append \
>   --last-value 0
Warning: /usr/lib/sqoop/../accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
25/12/04 02:51:18 INFO sqoop.Sqoop: Running Sqoop version: 1.4.6-cdh5.13.0
25/12/04 02:51:18 WARN tool.BaseSqoopTool: Setting your password on the command-line is insecure. Consider using -P instead.
[cloudera@quickstart ~]$ █
```

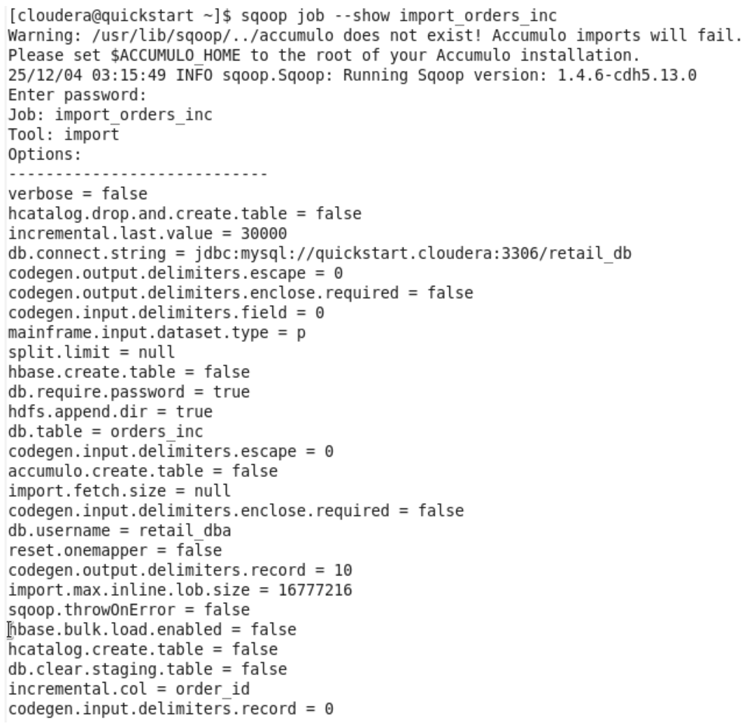
2. Inspecting the logs:



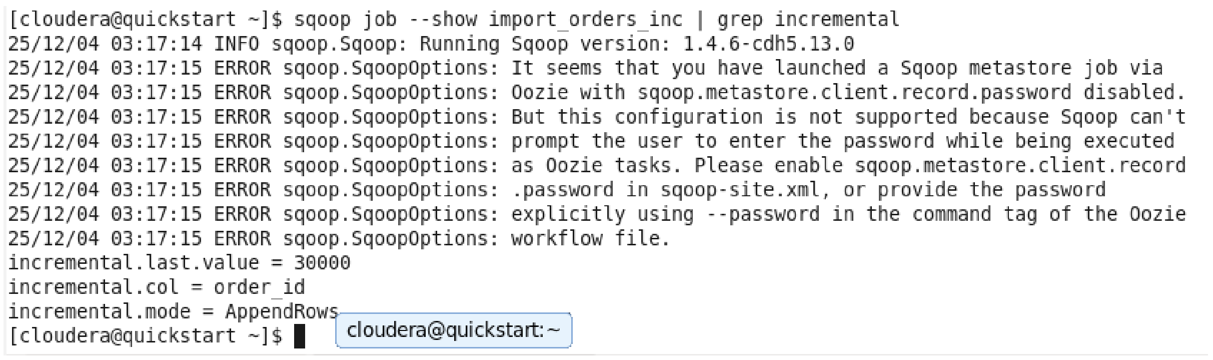
```
25/12/04 03:13:12 INFO db.DataDrivenDBInputFormat: BoundingValsQuery: SELECT MIN(`order_id`), MAX(`order_id`) FROM `orders_in
c` WHERE ( `order_id` > 0 AND `order_id` <= 30000 )
25/12/04 03:13:12 INFO db.IntegerSplitter: Split size: 7499; Num splits: 4 from: 1 to: 30000
25/12/04 03:13:12 INFO mapreduce.JobSubmitter: number of splits:4
```

We can see that in the request there is a limit, to start from id 0 to 30000.

3. Running job show

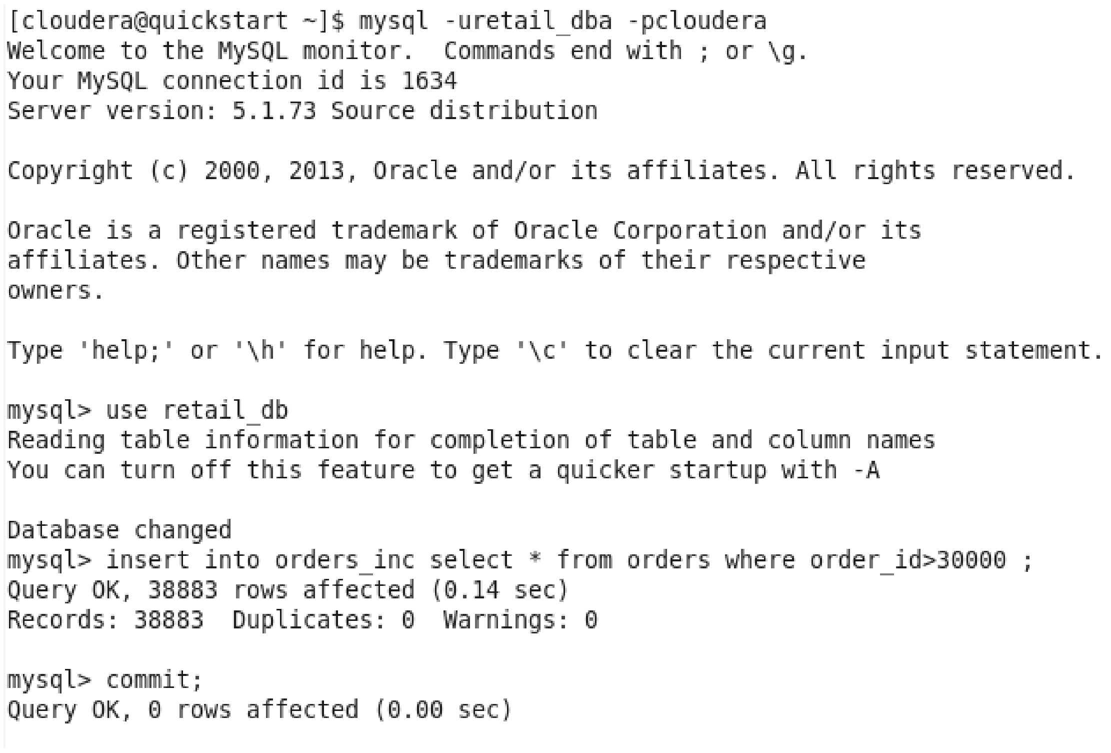


```
[cloudera@quickstart ~]$ sqoop job --show import_orders_inc
Warning: /usr/lib/sqoop/../accumulo does not exist! Accumulo imports will fail.
Please set $ACCUMULO_HOME to the root of your Accumulo installation.
25/12/04 03:15:49 INFO sqoop.Sqoop: Running Sqoop version: 1.4.6-cdh5.13.0
Enter password:
Job: import_orders_inc
Tool: import
Options:
----------------------------
verbose = false
hcatalog.drop.and.create.table = false
incremental.last.value = 30000
db.connect.string = jdbc:mysql://quickstart.cloudera:3306/retail_db
codegen.output.delimiters.escape = 0
codegen.output.delimiters.enclose.required = false
codegen.input.delimiters.field = 0
mainframe.input.dataset.type = p
split.limit = null
hbase.create.table = false
db.require.password = true
hdfs.append.dir = true
db.table = orders_inc
codegen.input.delimiters.escape = 0
accumulo.create.table = false
import.fetch.size = null
codegen.input.delimiters.enclose.required = false
db.username = retail_dba
reset.onemapper = false
codegen.output.delimiters.record = 10
import.max.inline.lob.size = 16777216
sqoop.throwOnError = false
hbase.bulk.load.enabled = false
hcatalog.create.table = false
db.clear.staging.table = false
incremental.col = order_id
codegen.input.delimiters.record = 0
```



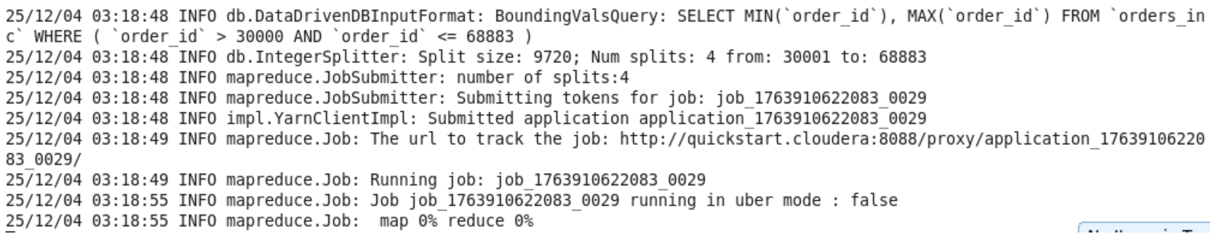
```
[cloudera@quickstart ~]$ sqoop job --show import_orders_inc | grep incremental
25/12/04 03:17:14 INFO sqoop.Sqoop: Running Sqoop version: 1.4.6-cdh5.13.0
25/12/04 03:17:15 ERROR sqoop.SqoopOptions: It seems that you have launched a Sqoop metastore job via
25/12/04 03:17:15 ERROR sqoop.SqoopOptions: Oozie with sqoop.metastore.client.record.password disabled.
25/12/04 03:17:15 ERROR sqoop.SqoopOptions: But this configuration is not supported because Sqoop can't
25/12/04 03:17:15 ERROR sqoop.SqoopOptions: prompt the user to enter the password while being executed
25/12/04 03:17:15 ERROR sqoop.SqoopOptions: as Oozie tasks. Please enable sqoop.metastore.client.record
25/12/04 03:17:15 ERROR sqoop.SqoopOptions: .password in sqoop-site.xml, or provide the password
25/12/04 03:17:15 ERROR sqoop.SqoopOptions: explicitly using --password in the command tag of the Oozie
25/12/04 03:17:15 ERROR sqoop.SqoopOptions: workflow file.
incremental.last.value = 30000
incremental.col = order_id
incremental.mode = AppendRows
[cloudera@quickstart ~]$
```

4. Adding the remaining rows:



```
[cloudera@quickstart ~]$ mysql -uretail_dba -pcloudera
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1634
Server version: 5.1.73 Source distribution

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use retail_db
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> insert into orders_inc select * from orders where order_id>30000 ;
Query OK, 38883 rows affected (0.14 sec)
Records: 38883  Duplicates: 0  Warnings: 0

mysql> commit;
Query OK, 0 rows affected (0.00 sec)
```

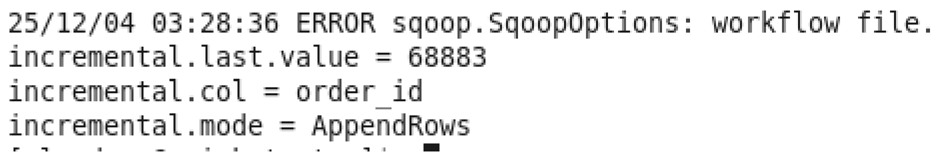
5. Re-executing the job:



```
25/12/04 03:18:48 INFO db.DataDrivenDBInputFormat: BoundingValsQuery: SELECT MIN(`order_id`), MAX(`order_id`) FROM `orders_in
c` WHERE ( `order_id` > 30000 AND `order_id` <= 68883 )
25/12/04 03:18:48 INFO db.IntegerSplitter: Split size: 9720; Num splits: 4 from: 30001 to: 68883
25/12/04 03:18:48 INFO mapreduce.JobSubmitter: number of splits:4
25/12/04 03:18:48 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1763910622083_0029
25/12/04 03:18:48 INFO impl.YarnClientImpl: Submitted application application_1763910622083_0029
25/12/04 03:18:49 INFO mapreduce.Job: The url to track the job: http://quickstart.cloudera:8088/proxy/application_17639106220
83_0029/
25/12/04 03:18:49 INFO mapreduce.Job: Running job: job_1763910622083_0029
25/12/04 03:18:55 INFO mapreduce.Job: Job job_1763910622083_0029 running in uber mode : false
25/12/04 03:18:55 INFO mapreduce.Job:  map 0% reduce 0%
```

We can see that the request has limits 30000 to 68883.

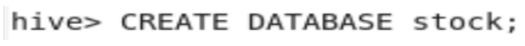
6. Running job show again, we can see that the last value has changed from 30000 to 68883:



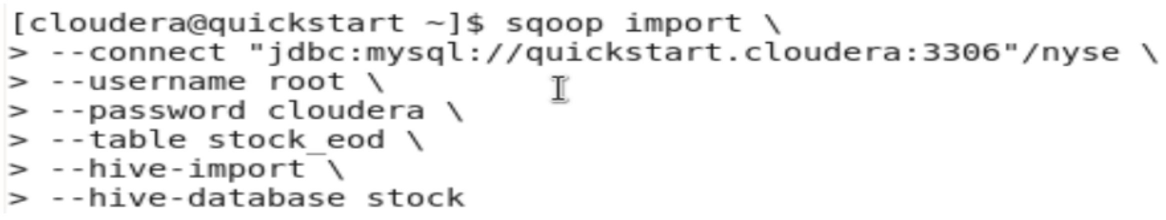
```
25/12/04 03:28:36 ERROR sqoop.SqoopOptions: workflow file.
incremental.last.value = 68883
incremental.col = order_id
incremental.mode = AppendRows
```

Lab Hive

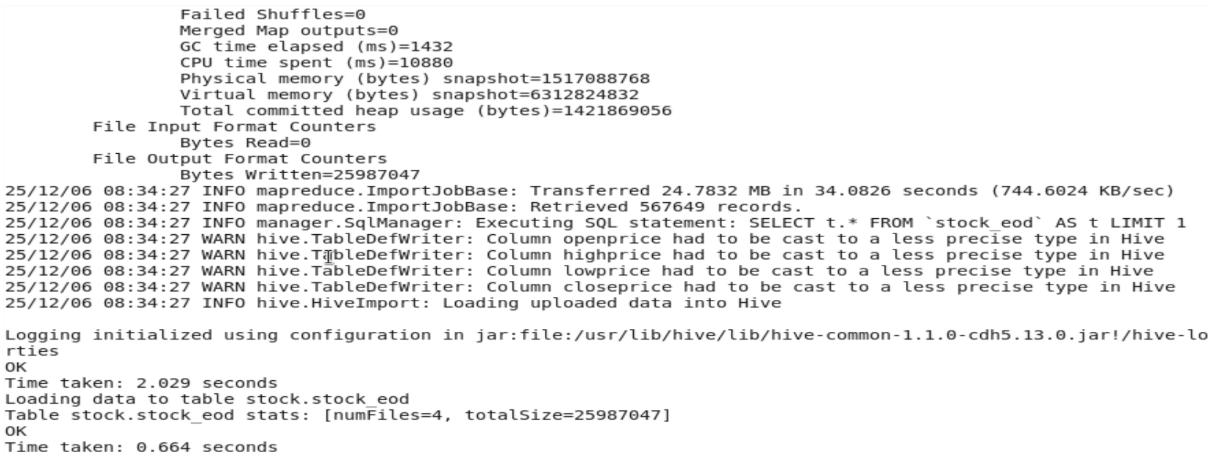
1. Importing the table **“stock_eod”** to Hive:



```
hive> CREATE DATABASE stock;
```

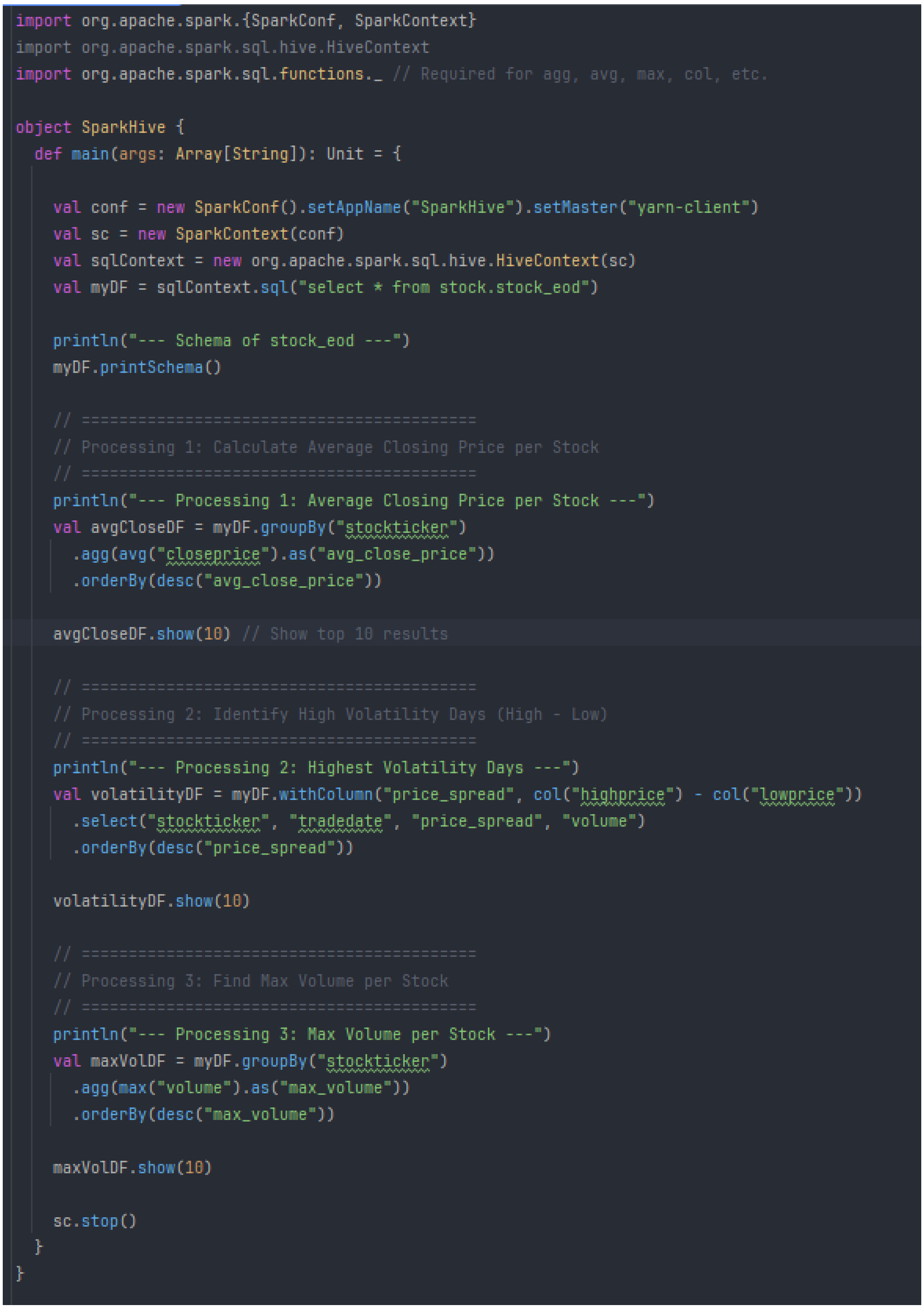


```
[cloudera@quickstart ~]$ sqoop import \
> --connect "jdbc:mysql://quickstart.cloudera:3306"/nyse \
> --username root \
> --password cloudera \
> --table stock_eod \
> --hive-import \
> --hive-database stock
```



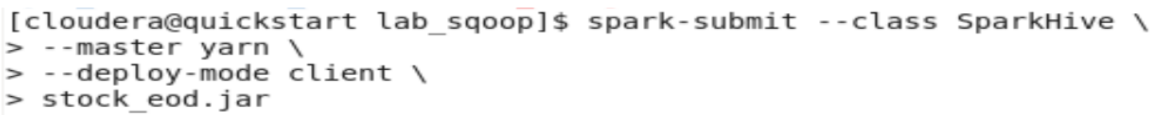
```
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=1432
                CPU time spent (ms)=10880
                Physical memory (bytes) snapshot=1517088768
                Virtual memory (bytes) snapshot=6312824832
                Total committed heap usage (bytes)=1421869056
        File Input Format Counters
                Bytes Read=0
        File Output Format Counters
                Bytes Written=25987047
25/12/06 08:34:27 INFO mapreduce.ImportJobBase: Transferred 24.7832 MB in 34.0826 seconds (744.6024 KB/sec)
25/12/06 08:34:27 INFO mapreduce.ImportJobBase: Retrieved 567649 records.
25/12/06 08:34:27 INFO manager.SqlManager: Executing SQL statement: SELECT t.* FROM `stock_eod` AS t LIMIT 1
25/12/06 08:34:27 WARN hive.TableDefWriter: Column openprice had to be cast to a less precise type in Hive
25/12/06 08:34:27 WARN hive.TableDefWriter: Column highprice had to be cast to a less precise type in Hive
25/12/06 08:34:27 WARN hive.TableDefWriter: Column lowprice had to be cast to a less precise type in Hive
25/12/06 08:34:27 WARN hive.TableDefWriter: Column closeprice had to be cast to a less precise type in Hive
25/12/06 08:34:27 INFO hive.HiveImport: Loading uploaded data into Hive

Logging initialized using configuration in jar:file:/usr/lib/hive/lib/hive-common-1.1.0-cdh5.13.0.jar!/hive-lo
rties
OK
Time taken: 2.029 seconds
Loading data to table stock.stock_eod
Table stock.stock_eod stats: [numFiles=4, totalSize=25987047]
OK
Time taken: 0.664 seconds
```

2. Performing 3 processings of my choice (on the table stock_eod):



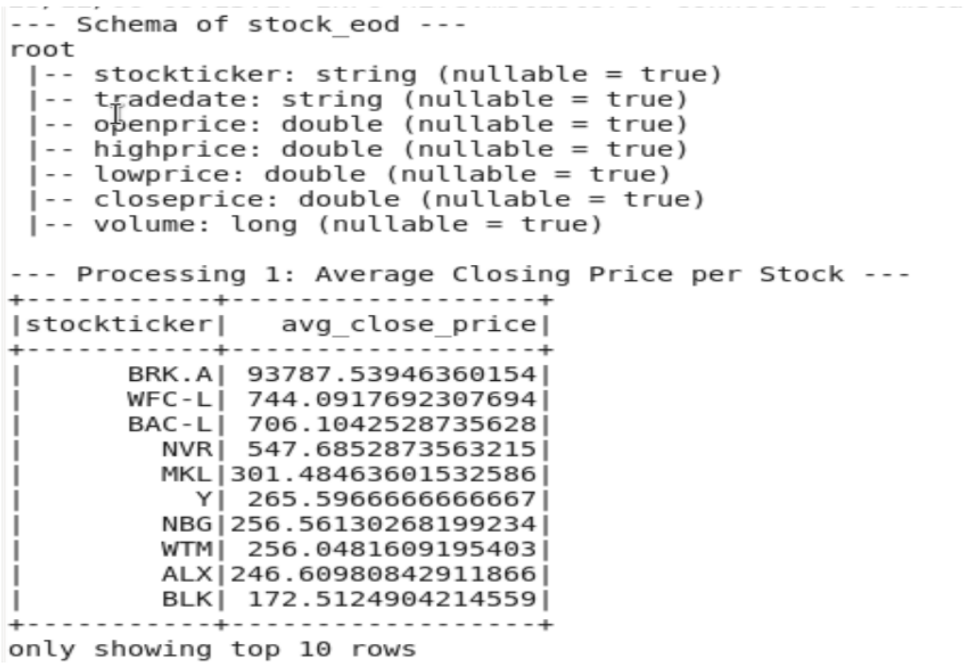
```scala
import org.apache.spark.{SparkConf, SparkContext}
import org.apache.spark.sql.hive.HiveContext
import org.apache.spark.sql.functions._ // Required for agg, avg, max, col, etc.

object SparkHive {
  def main(args: Array[String]): Unit = {

    val conf = new SparkConf().setAppName("SparkHive").setMaster("yarn-client")
    val sc = new SparkContext(conf)
    val sqlContext = new org.apache.spark.sql.hive.HiveContext(sc)
    val myDF = sqlContext.sql("select * from stock.stock_eod")

    println("--- Schema of stock_eod ---")
    myDF.printSchema()

    // ==========================================
    // Processing 1: Calculate Average Closing Price per Stock
    // ==========================================
    println("--- Processing 1: Average Closing Price per Stock ---")
    val avgCloseDF = myDF.groupBy("stockticker")
      .agg(avg("closeprice").as("avg_close_price"))
      .orderBy(desc("avg_close_price"))

    avgCloseDF.show(10) // Show top 10 results

    // ==========================================
    // Processing 2: Identify High Volatility Days (High - Low)
    // ==========================================
    println("--- Processing 2: Highest Volatility Days ---")
    val volatilityDF = myDF.withColumn("price_spread", col("highprice") - col("lowprice"))
      .select("stockticker", "tradedate", "price_spread", "volume")
      .orderBy(desc("price_spread"))

    volatilityDF.show(10)

    // ==========================================
    // Processing 3: Find Max Volume per Stock
    // ==========================================
    println("--- Processing 3: Max Volume per Stock ---")
    val maxVolDF = myDF.groupBy("stockticker")
      .agg(max("volume").as("max_volume"))
      .orderBy(desc("max_volume"))

    maxVolDF.show(10)

    sc.stop()
  }
}
```

Executing the jar on the cluster using spark-submit:

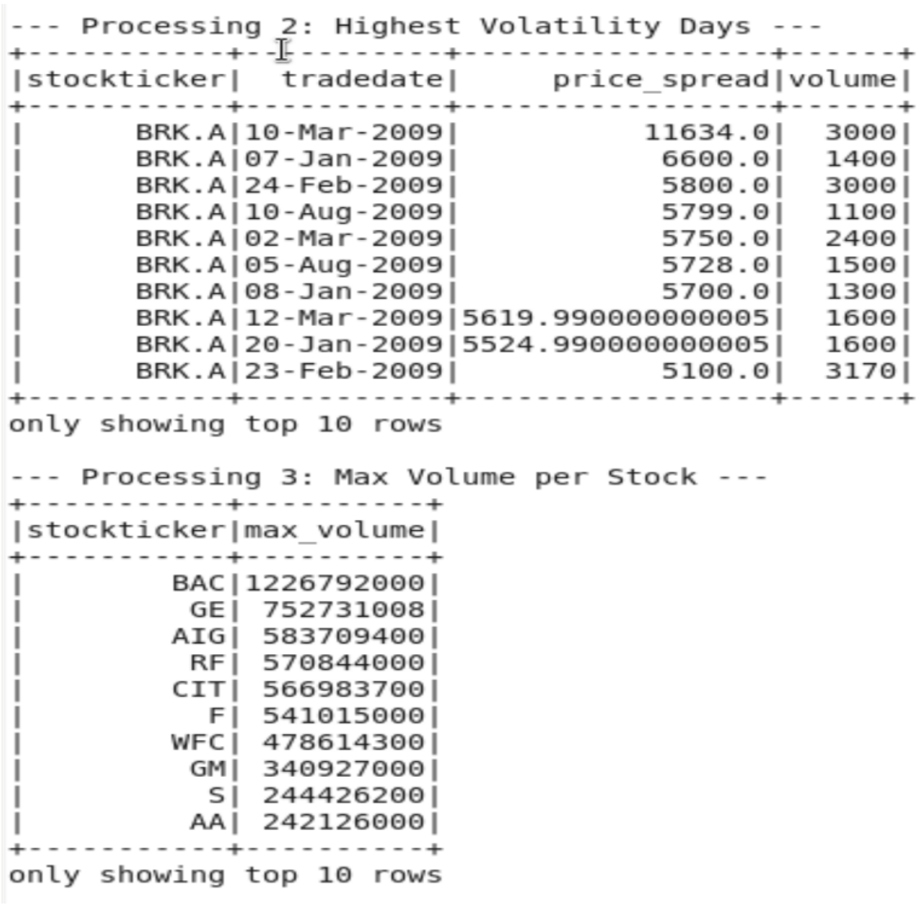


```
[cloudera@quickstart lab_sqoop]$ spark-submit --class SparkHive \
> --master yarn \
> --deploy-mode client \
> stock_eod.jar
```

Outputs:



```
--- Schema of stock_eod ---
root
 |-- stockticker: string (nullable = true)
 |-- tradedate: string (nullable = true)
 |-- openprice: double (nullable = true)
 |-- highprice: double (nullable = true)
 |-- lowprice: double (nullable = true)
 |-- closeprice: double (nullable = true)
 |-- volume: long (nullable = true)

--- Processing 1: Average Closing Price per Stock ---
+-----------+-----------------+
|stockticker|   avg_close_price|
+-----------+-----------------+
|      BRK.A| 93787.53946360154|
|      WFC-L| 744.0917692307694|
|      BAC-L| 706.1042528735628|
|        NVR| 547.6852873563215|
|        MKL|301.48463601532586|
|          Y| 265.5966666666667|
|        NBG|256.56130268199234|
|        WTM| 256.0481609195403|
|        ALX|246.60980842911866|
|        BLK| 172.5124904214559|
+-----------+-----------------+
only showing top 10 rows
```



```
--- Processing 2: Highest Volatility Days ---
+-----------+-----------+-----------------+------+
|stockticker|  tradedate|     price_spread|volume|
+-----------+-----------+-----------------+------+
|      BRK.A|10-Mar-2009|          11634.0|  3000|
|      BRK.A|07-Jan-2009|           6600.0|  1400|
|      BRK.A|24-Feb-2009|           5800.0|  3000|
|      BRK.A|10-Aug-2009|           5799.0|  1100|
|      BRK.A|02-Mar-2009|           5750.0|  2400|
|      BRK.A|05-Aug-2009|           5728.0|  1500|
|      BRK.A|08-Jan-2009|           5700.0|  1300|
|      BRK.A|12-Mar-2009|5619.990000000005|  1600|
|      BRK.A|20-Jan-2009|5524.990000000005|  1600|
|      BRK.A|23-Feb-2009|           5100.0|  3170|
+-----------+-----------+-----------------+------+
only showing top 10 rows

--- Processing 3: Max Volume per Stock ---
+-----------+----------+
|stockticker|max_volume|
+-----------+----------+
|        BAC|1226792000|
|         GE| 752731008|
|        AIG| 583709400|
|         RF| 570844000|
|        CIT| 566983700|
|          F| 541015000|
|        WFC| 478614300|
|         GM| 340927000|
|          S| 244426200|
|         AA| 242126000|
+-----------+----------+
only showing top 10 rows
```