** LOG ANALYZER **

Calculates the total number of hits for each hour of the day for a specific day. Hour which has the least number of hits is perfect for the downtime for a deployment execution. If there are multiple minimum hours in a day, pick the latest

```
import org.apache.spark.sql.SQLContext
import org.apache.spark.SparkContext
import org.apache.spark.sql.{Dataset, DataFrame, SparkSession}
import org.apache.spark.sql.functions._
import org.apache.spark.sql._

val sparkSession = SparkSession.builder.
  appName("SBTB").
  master("local[*]").getOrCreate()
```

```
import org.apache.spark.sql.SQLContext
import org.apache.spark.SparkContext
import org.apache.spark.sql.{Dataset, DataFrame, SparkSession}
import org.apache.spark.sql.functions._
import org.apache.spark.sql._
sparkSession: org.apache.spark.sql.SparkSession = org.apache.spark.sql.SparkS
ession@1eca601f
```

Took: 483 milliseconds, at 2017-6-21 12:28

```
val logFile = sc.textFile("/Users/Marius/Desktop/ApacheLog")
```

```
logFile: org.apache.spark.rdd.RDD[String] = /Users/Marius/Desktop/ApacheLog M
apPartitionsRDD[20] at textFile at <console>:103
```

Took: 502 milliseconds, at 2017-6-21 12:28

```
logFile.count
```

```
res76: Long = 6
```

6

Took: 634 milliseconds, at 2017-6-21 12:28

```
logFile.first
```

```
res78: String = 64.242.88.10,[07/Mar/2004:16:05:49 -0800],"GET /twiki/bin/edi
t/Main/Double_bounce_sender?topicparent=Main.ConfigurationVariables HTTP/1.1"
401 12846
```

64.242.88.10,[07/Mar/2004:16:05:49 -0800],"GET /twiki/bin/edit/Main/Double_bounce_sender?topicparent=Main.ConfigurationVariables HTTP/1.1" 401 12846

Took: 623 milliseconds, at 2017-6-21 12:28

```
case class LogEntry (clientIp: String, hour: String, clientAction: Option[String])
defined class LogEntry
```

Took: 418 milliseconds, at 2017-6-21 12:28

```
def converter(line: String): LogEntry = {
  val a = line.split(",").map(_.trim)

  LogEntry(
     clientIp = a(0),
     hour = a(1).substring(13,15),
     clientAction = Some(a(2))
  )
}
converter: (line: String)LogEntry
```

Took: 471 milliseconds, at 2017-6-21 12:28

```
val logFileConverted = logFile.map(line => converter(line)) // or map(converter(_))
logFileConverted: org.apache.spark.rdd.RDD[LogEntry] = MapPartitionsRDD[21] a
t map at <console>:109
```

Took: 513 milliseconds, at 2017-6-21 12:28

```
logFileConverted.first
res83: LogEntry = LogEntry(64.242.88.10,16,Some("GET /twiki/bin/edit/Main/Dou
ble_bounce_sender?topicparent=Main.ConfigurationVariables HTTP/1.1" 401 12846
))
```

LogEntry(64.242.88.10,16,Some("GET /twiki/bin/edit/Main/Double_bounce_sender?topicparent=Main.ConfigurationVariables HTTP/1.1″ 401 12846))

Took: 540 milliseconds, at 2017-6-21 12:28

```
val result = logFileConverted.map(logEntry => (logEntry.hour, 1))
result: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[22] at map
at <console>:111
```

Took: 518 milliseconds, at 2017-6-21 12:28

```
result foreach println
(18,1)
(16,1)
(16,1)
(16,1)
(16,1)
(17,1)
```

Took: 644 milliseconds, at 2017-6-21 12:28

```
//val finalResult = result.reduceByKey(_ + _).sortBy({
```

```
// case (hour, hitsCount) => hitsCount
//}, ascending = false)

val finalResult = result.reduceByKey(_ + _).sortBy(_._2, true)

finalResult.first

//only foreach not display sorted properly, need take
finalResult take 3 foreach println
(17,1)
(18,1)
(16,4)
finalResult: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[230]
at sortBy at <console>:119
Took: 849 milliseconds, at 2017-6-21 13:26
//better coding practice

val finalResult = result.reduceByKey(_ + _).groupBy {
  case(hour, hitsCount) => hitsCount
}.map{
  case (hitsCount, sequence) => (hitsCount, sequence.to[Vector].map(pair => pair._1).sortBy(hour => hour))
}.sortBy({
   case (hitsCount, vector) => hitsCount

}, ascending = true).map {
  case (hitsCount, vector) => (hitsCount, vector(vector.length-1))
}.first._2

println(s"$finalResult")

//finalResult take 3 foreach println
18
finalResult: String = 18
Took: 855 milliseconds, at 2017-6-21 13:52
result take 6 foreach println
println(">>>>>>> reduceByKey")

val fr1 = result.reduceByKey(_ + _)

fr1 take 3 foreach println
println(">>>>>>> groupBy hitsCount")

val fr2 = fr1.groupBy  {
  case(hour, hitsCount) => hitsCount
}
```

```
fr2 take 3 foreach println
println(">>>>>>> map from Buffer to Vector and sortBy Vector")

val fr3 = fr2.map{
  case (hitsCount, sequence) => (hitsCount, sequence.to[Vector].map(pair => pair._1).sortBy(hour => hour))
}

fr3 take 3 foreach println
println(">>>>>>> sortBy hitsCount")

val fr4 = fr3.sortBy({
   case (hitsCount, vector) => hitsCount

}, ascending = true)

fr4 take 3 foreach println
println(">>>>>>> map")

val fr5 = fr4.first._2(fr4.first._2.length -1)

println(s">>>>>>>>>> final ${fr4.first._2.length -1}")
println(s">>>>>>>>>> final $fr5")
```

```
(16,1)
(16,1)
(16,1)
(17,1)
(18,1)
(16,1)
>>>>>>> reduceByKey
(17,1)
(18,1)
(16,4)
>>>>>>> groupBy hitsCount
(4,CompactBuffer((16,4)))
(1,CompactBuffer((17,1), (18,1)))
>>>>>>> map from Buffer to Vector and sortBy Vector
(4,Vector(16))
(1,Vector(17, 18))
>>>>>>> sortBy hitsCount
(1,Vector(17, 18))
(4,Vector(16))
```

```
>>>>>>> map
>>>>>>>>>> final 1
>>>>>>>>>> final 18
fr1: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[564] at reduceByKe
y at <console>:122
fr2: org.apache.spark.rdd.RDD[(Int, Iterable[(String, Int)])] = ShuffledRDD[5
66] at groupBy at <console>:127
fr3: org.apache.spark.rdd.RDD[(Int, scala.collection.immutable.Vector[String]
)] = MapPartitionsRDD[567] at map at <console>:134
fr4: org.apache.spark.rdd.RDD[(Int, scala.collection.immutable.Vector[String]
)] = MapPartitionsRDD[572] at sortBy at <console>:141
fr5: String = 18
```

Took: 1 second 360 milliseconds, at 2017-6-21 14:32