Final Project

Stack Exchange Analysis

Akanksha Sinha

9177

1. Importing libraries

```
val sqlContext = new org.apache.spark.sql.SQLContext(sc);
import sqlContext.implicits._
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.functions._
Import org.apache.spark.sql.Column
Import org.apache.spark.sql.types._
import org.apache.spark.sql.types.{IntegerType, StringType, StructField,
StructType,LongType};
```

2. Creating Structure for dataset

```
val schema = StructType(Array(StructField("rownum",IntegerType,true),

StructField("qid",IntegerType,true),

StructField ("userid", IntegerType,true),

StructField("qscore",IntegerType,true),

StructField("qtime",LongType,true),

StructField("tag",StringType,true),

StructField("qvc",IntegerType,true),

StructField("qac",IntegerType,true),

StructField("aid",IntegerType,true),
```

StructField("uida",IntegerType,true),

StructField("as",IntegerType,true),

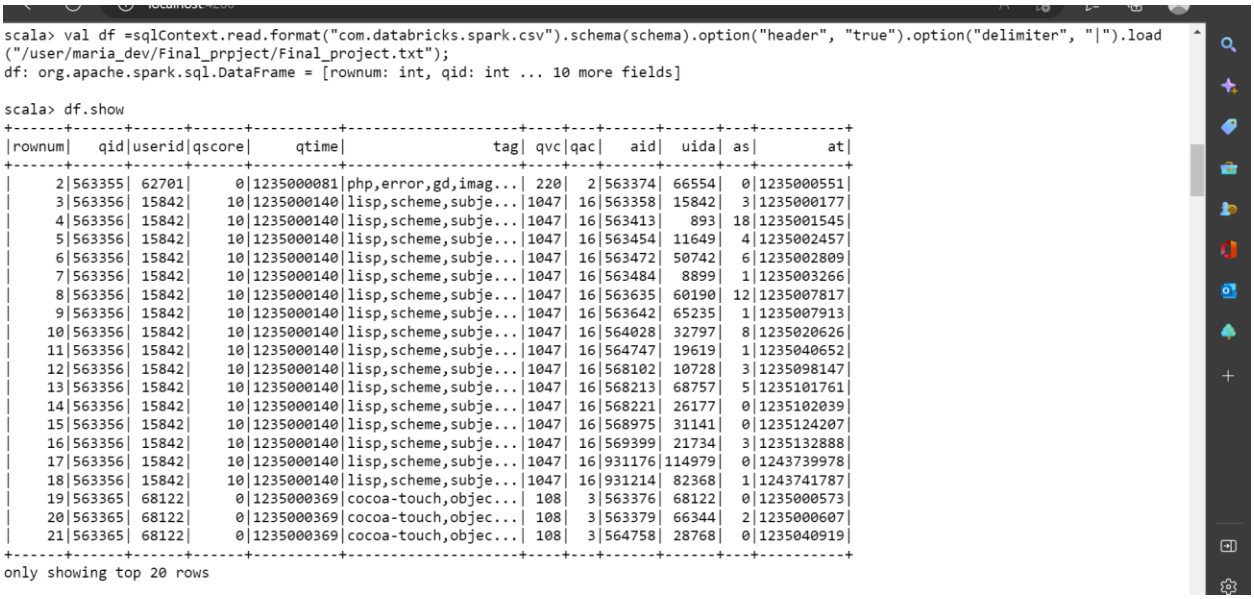
StructField("at",LongType,true)

))

3. Loading dataset

val df = sqlContext.read.format("com.databricks.spark.csv").schema(schema).option("header", "true").option("delimiter", "|").load("/user/maria_dev/Final_prpject/Final_project.txt");

df.show()



```
scala> val df =sqlContext.read.format("com.databricks.spark.csv").schema(schema).option("header", "true").option("delimiter", "|").load
("/user/maria_dev/Final_prpject/Final_project.txt");
df: org.apache.spark.sql.DataFrame = [rownum: int, qid: int ... 10 more fields]

scala> df.show
+------+------+------+------+----------+--------------------+----+---+------+------+---+----------+
|rownum|   qid|userid|qscore|     qtime|                 tag| qvc|qac|   aid|  uida| as|        at|
+------+------+------+------+----------+--------------------+----+---+------+------+---+----------+
|     2|563355| 62701|     0|1235000081|php,error,gd,imag...| 220|  2|563374| 66554|  0|1235000551|
|     3|563356| 15842|    10|1235000140|lisp,scheme,subje...|1047| 16|563358| 15842|  3|1235000177|
|     4|563356| 15842|    10|1235000140|lisp,scheme,subje...|1047| 16|563413|   893| 18|1235001545|
|     5|563356| 15842|    10|1235000140|lisp,scheme,subje...|1047| 16|563454| 11649|  4|1235002457|
|     6|563356| 15842|    10|1235000140|lisp,scheme,subje...|1047| 16|563472| 50742|  6|1235002809|
|     7|563356| 15842|    10|1235000140|lisp,scheme,subje...|1047| 16|563484|  8899|  1|1235003266|
|     8|563356| 15842|    10|1235000140|lisp,scheme,subje...|1047| 16|563635| 60190| 12|1235007817|
|     9|563356| 15842|    10|1235000140|lisp,scheme,subje...|1047| 16|563642| 65235|  1|1235007913|
|    10|563356| 15842|    10|1235000140|lisp,scheme,subje...|1047| 16|564028| 32797|  8|1235020626|
|    11|563356| 15842|    10|1235000140|lisp,scheme,subje...|1047| 16|564747| 19619|  1|1235040652|
|    12|563356| 15842|    10|1235000140|lisp,scheme,subje...|1047| 16|568102| 10728|  3|1235098147|
|    13|563356| 15842|    10|1235000140|lisp,scheme,subje...|1047| 16|568213| 68757|  5|1235101761|
|    14|563356| 15842|    10|1235000140|lisp,scheme,subje...|1047| 16|568221| 26177|  0|1235102039|
|    15|563356| 15842|    10|1235000140|lisp,scheme,subje...|1047| 16|568975| 31141|  0|1235124207|
|    16|563356| 15842|    10|1235000140|lisp,scheme,subje...|1047| 16|569399| 21734|  3|1235132888|
|    17|563356| 15842|    10|1235000140|lisp,scheme,subje...|1047| 16|931176|114979|  0|1243739978|
|    18|563356| 15842|    10|1235000140|lisp,scheme,subje...|1047| 16|931214| 82368|  1|1243741787|
|    19|563365| 68122|     0|1235000369|cocoa-touch,objec...| 108|  3|563376| 68122|  0|1235000573|
|    20|563365| 68122|     0|1235000369|cocoa-touch,objec...| 108|  3|563379| 66344|  2|1235000607|
|    21|563365| 68122|     0|1235000369|cocoa-touch,objec...| 108|  3|564758| 28768|  0|1235040919|
+------+------+------+------+----------+--------------------+----+---+------+------+---+----------+
only showing top 20 rows
```

Queries:-

Question 1:
Top 10 most commonly used tags in this data set.

df.select(explode(split(col("tag"),","))).groupBy("col").count().orderBy(col("count").desc).show(10)

```
scala> df.select(explode(split(col("tag"),","))).groupBy("col").count().orderBy(col("count").desc).show(10)
+----------+-----+
|       col|count|
+----------+-----+
|        c#|38399|
|      java|20003|
|      ûnet|19509|
|       c++|17445|
|   aspûnet|14525|
|       php|12909|
|javascript|12589|
|subjective|12416|
|    python|10028|
|       sql| 9695|
+----------+-----+
only showing top 10 rows
```

Question 2:

Average time to answer questions.

df.registerTempTable("social")

val sql1=spark.sql("select avg(timetaken)/3600 from (select (at - qtime) as timetaken from social)")

```
scala> val sql1=spark.sql("select avg(timetaken)/3600 from (select (at - qtime) as timetaken from social)")
sql1: org.apache.spark.sql.DataFrame = [(avg(timetaken) / CAST(3600 AS DOUBLE)): double]

scala> sql1.show()
+---------------------------------------+
|(avg(timetaken) / CAST(3600 AS DOUBLE))|
+---------------------------------------+
|                     37.157327866505945|
+---------------------------------------+
```

Question 3.

Number of questions which got answered within 1 hour

val sql2=spark.sql("select count(qid) from (select (at - qtime) as timetaken,* from social) where timetaken <= 3600")

sql2.show()

```
scala> val sql2=spark.sql("select count(qid) from (select (at - qtime) as timetaken,* from social) where timetaken <= 3600")
sql2: org.apache.spark.sql.DataFrame = [count(qid): bigint]

scala> sql2.show()
+----------+
|count(qid)|
+----------+
|    174608|
+----------+
```

Question 4.

Tags of questions which got answered within 1 hour.

val sql3 = spark.sql("select tag from (select (at - qtime) as timetaken,* from social) where timetaken <= 3600")

```
scala> val sql3 = spark.sql("select tag from (select (at - qtime) as timetaken,* from social) where timetaken <= 3600")
sql3: org.apache.spark.sql.DataFrame = [tag: string]

scala> sql3.show
+--------------------+
|                 tag|
+--------------------+
|php,error,gd,imag...|
|lisp,scheme,subje...|
|lisp,scheme,subje...|
|lisp,scheme,subje...|
|lisp,scheme,subje...|
|lisp,scheme,subje...|
|cocoa-touch,objec...|
|cocoa-touch,objec...|
|      core-animation|
|             aspûnet|
|jquery,javascript...|
|winforms,gridview...|
|         python,http|
|         python,http|
|         python,http|
|         python,http|
|         python,http|
|        visualstudio|
|        visualstudio|
|        visualstudio|
+--------------------+
only showing top 20 rows
```