Spark SQL

Run spark (scala):

```
/opt/mapr/spark/spark-2.3.2/bin/spark-shell --master local
/opt/mapr/spark/spark-2.3.2/bin/spark-shell --master yarn --deploy-mode client
```

python:

```
/opt/mapr/spark/spark-2.3.2/bin/pyspark --master local
```

Load data from hive:

```
val df = sql("SELECT␣*␣from␣transfers");
```

Load data from csv:

```
import org.apache.spark.sql.types._

val schema = StructType(Array(StructField("src", StringType, true), StructField("dst", StringType, true), StructField("amount", IntegerType, true), StructField("date"

val df = spark.read.format("csv").schema(schema).load("/path/to/csv")

// or
val df = spark.read.format("csv").load("/path/to/csv").toDF("src", "dst", "amount", "date")
```

Load data from binary format:

```
val df = spark.read.format("orc").load("/path/to/orc/table")
```

Filtering the data:

```
df.where("amount␣<␣5000")
df.filter(\$"amount" < 5000)
df.filter(col("amount") < 5000)
df.filter(r => r.getInt(2) < 5000)
```

Mapping the data:

```
df.map(row => (row.getInt(2) * 2, row.getString(0))).toDF("amount␣times␣2", "src")
```

Sorting:

```
df.orderBy(\$"count".desc)
```

Save data to hive:

```
df.createOrReplaceTempView("tempDfView")
sql("create␣table␣table_name␣as␣select␣*␣from␣tempDfView");
```

Save data to file:

```
df.write.csv("/user/xyz/dir")
df.write.orc("/user/xyz/dir")
```

Word count:

```
sc.textFile("/user/xyz/loremipsum").flatMap(line => line.split("␣")).map(word => (word, 1)).reduceByKey((a, b) => a + b).toDF("word", "count").show()
sc.textFile("/user/xyz/loremipsum").flatMap(line => line.split("␣")).map(word => (word, 1)).reduceByKey((a, b) => a + b).collect()
```

Tasks

1. count letters in loremipsum
2. Count how many incoming transfers were there for each account.
3. find number of unique accounts