Spark RDD

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
```

Create rdd:

```
sc.parallelize(Array(1, 2, 3, 4, 5))
```

Read text file:

```
val rdd = sc.textFile("/user/xyz/loremipsum")
```

Transformations:

```
rdd.map(row => row + "␣suffix")
rdd.filter(x => x < 5)
```

Documentation

Actions:

```
rdd.count()
rdd.take(5)
rdd.collect()
rdd.reduce((a, b) => a + b)
```

Documentation

Shuffle operations:

```
rdd.sortBy(i => -i)
rdd.reduceByKey((a, b) => a + b)
```

Save to maprfs / hdfs:

```
rdd.saveAstTextFile("/user/xyz/result")
```

Word count:

```
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

Extra Task

1. join owners with transfers. Load owners csv into memory. Use bradcasting for this. Join the data with transfers by replacing account id with owner name.