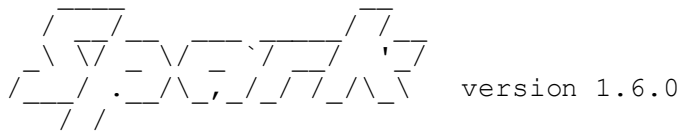


```
[lakshyajha@1994 ~]$ spark-shell
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel).
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/zookeeper/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/flume-ng/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/parquet/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/avro/avro-tools-1.7.6-cdh5.12.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
Welcome to
```



```
Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.7.0_67)
Type in expressions to have them evaluated.
Type :help for more information.
18/04/20 13:01:27 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Spark context available as sc (master = local[*], app id = local-1524254497745).
18/04/20 13:01:44 WARN shortcircuit.DomainSocketFactory: The short-circuit local reads feature cannot be used because libhadoop cannot be loaded.
SQL context available as sqlContext.
```

```
scala> import org.apache.spark.sql._
import org.apache.spark.sql._
```

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

```
scala> import sqlContext.implicits._
import sqlContext.implicits._
```

```
scala> val data = sc.textFile("file:///home/bankanalysis/Desktop/bank-full.csv").map(x => x.split("(?=(^[^\\"])*\\\".*\\\"|[^\"]+)$",-1))
data: org.apache.spark.rdd.RDD[Array[String]] = MapPartitionsRDD[2] at map at <console>:36
```

```
scala> val header = data.first()
header: Array[String] = Array("age", "job", "marital", "education", "default", "balance", "housing", "loan", "contact", "day", "month", "duration", "campaign", "pdays", "previous", "poutcome", "y")
```

```
scala> val filtered = data.filter(x => x(0) != header(0))
filtered: org.apache.spark.rdd.RDD[Array[String]] = MapPartitionsRDD[3]
at filter at <console>:40
```

```
scala> val rdds = filtered.map(x => Row(x(0).toInt, x(1), x(2), x(3), x(4),
x(5).toInt, x(6), x(7), x(8), x(9).toInt, x(10), x(11).toInt, x(12).toInt,
x(13).toInt, x(14).toInt, x(15), x(16) ))
rdds: org.apache.spark.rdd.RDD[org.apache.spark.sql.Row] =
MapPartitionsRDD[4] at map at <console>:42
```

```
scala> val schema = StructType( List(StructField("age", IntegerType,
true), StructField("job", StringType, true), StructField("marital",
StringType, true), StructField("education", StringType, true)
, StructField("default", StringType, true), StructField("balance",
IntegerType, true), StructField("housing", StringType, true)
, StructField("loan", StringType, true), StructField("contact",
StringType, true), StructField("day", IntegerType, true)
, StructField("month", StringType, true), StructField("duration",
IntegerType, true), StructField("campaign", IntegerType, true)
, StructField("pdays", IntegerType, true), StructField("previous",
IntegerType, true), StructField("poutcome", StringType, true)
, StructField("y", StringType, true)) )
schema: org.apache.spark.sql.types.StructType =
StructType(StructField(age,IntegerType,true),
StructField(job,StringType,true), StructField(marital,StringType,true),
StructField(education,StringType,true),
StructField(default,StringType,true),
StructField(balance,IntegerType,true),
StructField(housing,StringType,true), StructField(loan,StringType,true),
StructField(contact,StringType,true), StructField(day,IntegerType,true),
StructField(month,StringType,true),
StructField(duration,IntegerType,true),
StructField(campaign,IntegerType,true),
StructField(pdays,IntegerType,true),
StructField(previous,IntegerType,true),
StructField(poutcome,StringType,true), StructField(y,StringType,true))
```

```
scala> val df = sqlContext.createDataFrame(rdds, schema)
df: org.apache.spark.sql.DataFrame = [age: int, job: string, marital:
string, education: string, default: string, balance: int, housing:
string, loan: string, contact: string, day: int, month: string, duration:
int, campaign: int, pdays: int, previous: int, poutcome: string, y:
string]
```

```
scala> val success_rate = (df.filter($"y" === "\"yes\"").count).toDouble
/ (df.count).toDouble
success_rate: Double = 0.11698480458295547
```

```
scala> df.select(max($"age"), min($"age") , mean($"age")).show
+-----+-----+-----+
|max(age)|min(age)|      avg(age)|
+-----+-----+-----+
|      95|      18|40.93621021432837|
```

```
+-----+-----+-----+-----+
```

```
scala> df.registerTempTable("df")
```

```
scala> sqlContext.sql("select percentile(age, 0.50) from df").show
18/04/20 13:14:40 WARN metastore.ObjectStore: Version information not
found in metastore. hive.metastore.schema.validation is not enabled so
recording the schema version 1.1.0-cdh5.12.0
18/04/20 13:14:40 WARN metastore.ObjectStore: Failed to get database
default, returning NoSuchObjectException
```

```
+-----+
| _c0 |
+-----+
| 39.0 |
+-----+
```

```
scala> sqlContext.sql("select max(age), min(age), avg(age) ,
percentile(age, 0.50) from df").show
```

```
+---+---+-----+---+
|_c0|_c1|          _c2|_c3|
+---+---+-----+---+
| 95| 18|40.93621021432837|39.0|
+---+---+-----+---+
```

```
scala> sqlContext.sql("select avg(balance), percentile(balance, 0.50)
from df").show
```

```
+-----+-----+
|          _c0|_c1|
+-----+-----+
|1362.2720576850766|448.0|
+-----+-----+
```

```
scala> df.groupBy("y").agg(avg($"age")).show
```

```
+---+-----+
|   y|      avg(age) |
+---+-----+
|"yes"|41.670069956513515|
|"no" |40.83898602274435|
+---+-----+
```

```
scala> df.groupBy("y").agg(count($"marital")).show
```

```
+---+-----+
|   y|count(marital) |
+---+-----+
|"yes"|          5289 |
|"no" |          39922 |
+---+-----+
```

```
scala> df.groupBy("marital","y").count.show()
```

```
+-----+-----+-----+
| marital| y |count|
+-----+-----+-----+
| "single"|"yes"| 1912|
| "divorced"|"no"| 4585|
| "married"|"no"|24459|
| "divorced"|"yes"| 622|
| "married"|"yes"| 2755|
| "single"|"no"|10878|
+-----+-----+-----+
```

```
scala> df.groupBy("age","y").count.show()
```

```
+---+---+---+
|age| y |count|
+---+---+---+
| 31| "no"| 1790|
| 95|"yes"| 1|
| 68|"yes"| 21|
| 41|"yes"| 120|
| 69| "no"| 27|
| 42| "no"| 1131|
| 79|"yes"| 10|
| 52|"yes"| 85|
| 80| "no"| 19|
| 53| "no"| 806|
| 25|"yes"| 113|
| 26| "no"| 671|
| 90|"yes"| 2|
| 63|"yes"| 30|
| 36|"yes"| 195|
| 64| "no"| 39|
| 37| "no"| 1526|
| 74|"yes"| 13|
| 47|"yes"| 113|
| 75| "no"| 24|
+---+---+---+
```

only showing top 20 rows

```
scala> val df_new = df.withColumn("age_cat", when($"age" < 25 ,
"young").otherwise( when($"age" > 60 , "old").otherwise("mid_age") ))
df_new: org.apache.spark.sql.DataFrame = [age: int, job: string, marital:
string, education: string, default: string, balance: int, housing:
string, loan: string, contact: string, day: int, month: string, duration:
int, campaign: int, pdays: int, previous: int, poutcome: string, y:
string, age_cat: string]
```

```
scala> df_new.groupBy("age_cat","y").count.show()
```

```
+-----+-----+-----+
|age_cat| y |count|
+-----+-----+-----+
| young| "no"| 602|
```

```
| young|"yes"| 207|
| old| "no"| 686|
|mid_age| "no"|38634|
| old|"yes"| 502|
|mid_age|"yes"| 4580|
+-----+-----+-----+
```

```
scala> df.withColumn("age_cat", when($"age" < 25 , "young").otherwise(
when($"age" > 60 , "old").otherwise("mid_age")  )).show()
```

```
+---+-----+-----+-----+-----+-----+-----+-----+
+-----+---+-----+-----+-----+-----+-----+-----+
---+
|age|          job| marital| education|default|balance|housing| loan|
contact|day|month|duration|campaign|pdays|previous| poutcome|
y|age_cat|
+---+-----+-----+-----+-----+-----+-----+-----+
+-----+---+-----+-----+-----+-----+-----+-----+
---+
| 58| "management"| "married"| "tertiary"| "no"| 2143| "yes"|
"no"|"unknown"| 5|"may"| 261| 1| -1|
0|"unknown"|"no"|mid_age|
| 44| "technician"| "single"| "secondary"| "no"| 29| "yes"|
"no"|"unknown"| 5|"may"| 151| 1| -1|
0|"unknown"|"no"|mid_age|
| 33|"entrepreneur"| "married"| "secondary"| "no"| 2|
"yes"|"yes"|"unknown"| 5|"may"| 76| 1| -1|
0|"unknown"|"no"|mid_age|
| 47| "blue-collar"| "married"| "unknown"| "no"| 1506| "yes"|
"no"|"unknown"| 5|"may"| 92| 1| -1|
0|"unknown"|"no"|mid_age|
| 33| "unknown"| "single"| "unknown"| "no"| 1| "no"|
"no"|"unknown"| 5|"may"| 198| 1| -1|
0|"unknown"|"no"|mid_age|
| 35| "management"| "married"| "tertiary"| "no"| 231| "yes"|
"no"|"unknown"| 5|"may"| 139| 1| -1|
0|"unknown"|"no"|mid_age|
| 28| "management"| "single"| "tertiary"| "no"| 447|
"yes"|"yes"|"unknown"| 5|"may"| 217| 1| -1|
0|"unknown"|"no"|mid_age|
| 42|"entrepreneur"| "divorced"| "tertiary"| "yes"| 2| "yes"|
"no"|"unknown"| 5|"may"| 380| 1| -1|
0|"unknown"|"no"|mid_age|
| 58| "retired"| "married"| "primary"| "no"| 121| "yes"|
"no"|"unknown"| 5|"may"| 50| 1| -1|
0|"unknown"|"no"|mid_age|
| 43| "technician"| "single"| "secondary"| "no"| 593| "yes"|
"no"|"unknown"| 5|"may"| 55| 1| -1|
0|"unknown"|"no"|mid_age|
| 41| "admin."|"divorced"| "secondary"| "no"| 270| "yes"|
"no"|"unknown"| 5|"may"| 222| 1| -1|
0|"unknown"|"no"|mid_age|
```

29	"admin."	"single"	"secondary"	"no"	390	"yes"
"no"	"unknown"	5	"may"	137	1	-1
0	"unknown"	"no"	mid_age			
53	"technician"	"married"	"secondary"	"no"	6	"yes"
"no"	"unknown"	5	"may"	517	1	-1
0	"unknown"	"no"	mid_age			
58	"technician"	"married"	"unknown"	"no"	71	"yes"
"no"	"unknown"	5	"may"	71	1	-1
0	"unknown"	"no"	mid_age			
57	"services"	"married"	"secondary"	"no"	162	"yes"
"no"	"unknown"	5	"may"	174	1	-1
0	"unknown"	"no"	mid_age			
51	"retired"	"married"	"primary"	"no"	229	"yes"
"no"	"unknown"	5	"may"	353	1	-1
0	"unknown"	"no"	mid_age			
45	"admin."	"single"	"unknown"	"no"	13	"yes"
"no"	"unknown"	5	"may"	98	1	-1
0	"unknown"	"no"	mid_age			
57	"blue-collar"	"married"	"primary"	"no"	52	"yes"
"no"	"unknown"	5	"may"	38	1	-1
0	"unknown"	"no"	mid_age			
60	"retired"	"married"	"primary"	"no"	60	"yes"
"no"	"unknown"	5	"may"	219	1	-1
0	"unknown"	"no"	mid_age			
33	"services"	"married"	"secondary"	"no"	0	"yes"
"no"	"unknown"	5	"may"	54	1	-1
0	"unknown"	"no"	mid_age			

+---+-----+-----+-----+-----+-----+-----+-----+-----+
+-----+---+-----+-----+-----+-----+-----+-----+-----+-----+
---+

only showing top 20 rows

```
scala> df_new.groupBy("age_cat","y").count.show()
+-----+-----+-----+
|age_cat|    y|count|
+-----+-----+-----+
|  young| "no"|  602|
|  young|"yes"|  207|
|    old| "no"|  686|
|mid_age| "no"|38634|
|    old|"yes"|  502|
|mid_age|"yes"| 4580|
+-----+-----+-----+
```

```
scala>
```