databricks scala pract1

\\\*Using Dataframes and Datasets in Scala****\

```
val dataset = Seq(1,2,3).toDS()
dataset.show()
```

```
+-----+
|value|
+-----+
|    1|
|    2|
|    3|
+-----+

dataset: org.apache.spark.sql.Dataset[Int] = [value: int]
```

```
val dataset1 = Seq(("Max",33),("Adam",32),("Muller",62)).toDS()
dataset1.show()
```

```
+------+---+
|    _1| _2|
+------+---+
|   Max| 33|
|  Adam| 32|
|Muller| 62|
+------+---+

dataset1: org.apache.spark.sql.Dataset[(String, Int)] = [_1: string, _2: in
t]
```

```
val rdd = sc.parallelize(Seq((1, "Spark"), (2, "Databricks")))
val integerDS = rdd.toDS()
integerDS.show()
```

```
+---+----------+
| _1|        _2|
+---+----------+
|  1|     Spark|
|  2|Databricks|
+---+----------+

rdd: org.apache.spark.rdd.RDD[(Int, String)] = ParallelCollectionRDD[0] at p
arallelize at command-2975931743138767:1
integerDS: org.apache.spark.sql.Dataset[(Int, String)] = [_1: int, _2: strin
g]
```

```
case class Company(name: String, foundingYear: Int, numEmployees: Int)
val inputSeq = Seq(Company("ABC", 1998, 310), Company("XYZ", 1983, 904),
Company("NOP", 2005, 83))
val df = sc.parallelize(inputSeq).toDF()

val companyDS = df.as[Company]
companyDS.show()
```

```
+----+------------+------------+
|name|foundingYear|numEmployees|
+----+------------+------------+
| ABC|        1998|         310|
| XYZ|        1983|         904|
| NOP|        2005|          83|
+----+------------+------------+

defined class Company
inputSeq: Seq[Company] = List(Company(ABC,1998,310), Company(XYZ,1983,904),
Company(NOP,2005,83))
df: org.apache.spark.sql.DataFrame = [name: string, foundingYear: int ... 1
more field]
companyDS: org.apache.spark.sql.Dataset[Company] = [name: string, foundingYe
ar: int ... 1 more field]
```

```
val rdd= sc.parallelize(Seq((1,"Spark"),(2,"Databricks"),(3,"Notebook")))
val df = rdd.toDF("Id","Name")

val dataset2=df.as[(Int,String)]
dataset2.show()
```

```
+---+----------+
| Id|      Name|
+---+----------+
|  1|     Spark|
|  2|Databricks|
|  3|  Notebook|
+---+----------+

rdd: org.apache.spark.rdd.RDD[(Int, String)] = ParallelCollectionRDD[6] at p
arallelize at command-2975931743138769:1
df: org.apache.spark.sql.DataFrame = [Id: int, Name: string]
dataset2: org.apache.spark.sql.Dataset[(Int, String)] = [Id: int, Name: stri
ng]
```

```
val wordDataset = sc.parallelize(Seq("Spark I am your father","May the spark
be with your","Spark I am Your Father")).toDS()
val groupedDataset=wordDataset.flatMap(_.toLowerCase.split(" "))
                              .filter(_!="")
                              .groupBy("value")
val countDataset =groupedDataset.count()
countDataset.show()
```

```
+------+-----+
| value|count|
+------+-----+
|father|    2|
|  your|    3|
| spark|    3|
|     i|    2|
|    am|    2|
|  with|    1|
|    be|    1|
|   may|    1|
|   the|    1|
+------+-----+

wordDataset: org.apache.spark.sql.Dataset[String] = [value: string]
groupedDataset: org.apache.spark.sql.RelationalGroupedDataset = RelationalGr
oupedDataset: [grouping expressions: [value: string], value: [value: strin
g], type: GroupBy]
countDataset: org.apache.spark.sql.DataFrame = [value: string, count: bigin
t]
```

```
val rddFromFile =spark.sparkContext.textFile("/FileStore/tables/abc.txt")

rddFromFile.collect().foreach(f=>{
  println(f)
})
```

```
i am shivani
rddFromFile: org.apache.spark.rdd.RDD[String] = /FileStore/tables/abc.txt Ma
pPartitionsRDD[44] at textFile at command-2975931743138771:1
```

```
rddFromFile.count()
```

```
res30: Long = 1
```

```
rddFromFile.first()
```

```
res31: String = "i am shivani "
```

```
val lineswithshivani = rddFromFile.filter(line => line.contains("shivani"))
```

```
lineswithshivani: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[45] at
filter at command-2975931743138780:1
```

```
lineswithshivani.count()
```

```
res32: Long = 1
```

```
lineswithshivani.collect().take(1).foreach(println)
```

```
i am shivani
```

```
val dataset4=rddFromFile.toDS()
```

```
dataset4: org.apache.spark.sql.Dataset[String] = [value: string]
```

```
dataset4.show()
```

```
+-------------+
|        value|
+-------------+
|i am shivani |
+-------------+
```

```
val group1=dataset4.flatMap(_.toLowerCase.split(" "))
                              .filter(_!="")
                              .groupBy("value")
val count1 =group1.count()
count1.show()
```

```
+-------+-----+
|  value|count|
+-------+-----+
|shivani|    1|
|      i|    1|
|     am|    1|
+-------+-----+
```

```
group1: org.apache.spark.sql.RelationalGroupedDataset = RelationalGroupedDat
aset: [grouping expressions: [value: string], value: [value: string], type:
GroupBy]
count1: org.apache.spark.sql.DataFrame = [value: string, count: bigint]
```

```
+-------+-------------+
|  value|numOccurances|
+-------+-------------+
|shivani|            1|
|      i|            1|
|     am|            1|
+-------+-------------+

warning: one feature warning; for details, enable `:setting -feature' or `:r
eplay -feature'
import org.apache.spark.sql.functions._
result: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [value: str
ing, numOccurances: bigint]
```