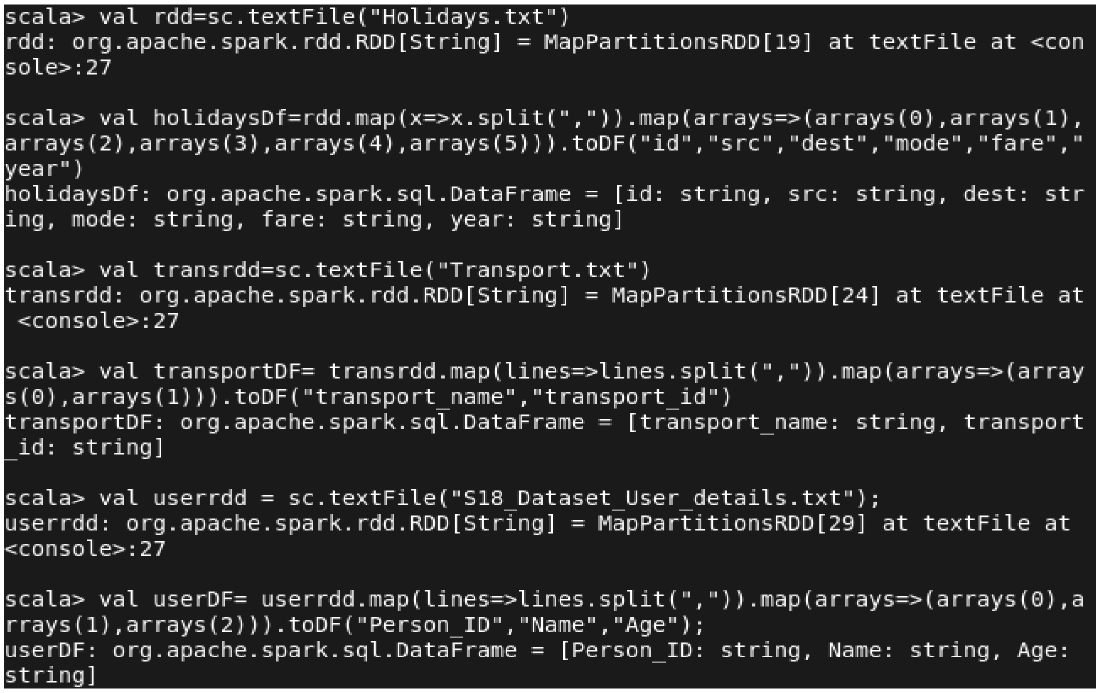
ASSIGNMENT 18.1

load data

```
scala> val rdd=sc.textFile("Holidays.txt")
rdd: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[19] at textFile at <console>:27

scala> val
holidaysDf=rdd.map(x=>x.split(",")).map(arrays=>(arrays(0),arrays(1),arrays(2),arrays(3),arrays(4),arr
ays(5))).toDF("id","src","dest","mode","fare","year")
holidaysDf: org.apache.spark.sql.DataFrame = [id: string, src: string, dest: string, mode: string, fare:
string, year: string]

scala> val transrdd=sc.textFile("Transport.txt")
transrdd: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[24] at textFile at <console>:27

scala> val transportDF=
transrdd.map(lines=>lines.split(",")).map(arrays=>(arrays(0),arrays(1))).toDF("transport_name","trans
port_id")
transportDF: org.apache.spark.sql.DataFrame = [transport_name: string, transport_id: string]

scala> val userrdd = sc.textFile("User.txt");
userrdd: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[29] at textFile at <console>:27

scala> val userDF=
userrdd.map(lines=>lines.split(",")).map(arrays=>(arrays(0),arrays(1),arrays(2))).toDF("Person_ID","
Name","Age");
userDF: org.apache.spark.sql.DataFrame = [Person_ID: string, Name: string, Age: string]
```

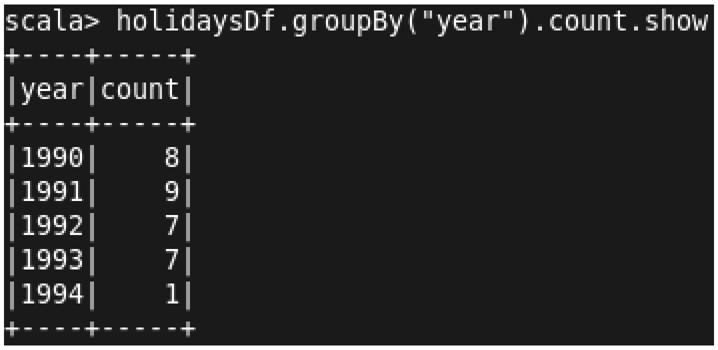


```
scala> val rdd=sc.textFile("Holidays.txt")
rdd: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[19] at textFile at <con
sole>:27

scala> val holidaysDf=rdd.map(x=>x.split(",")).map(arrays=>(arrays(0),arrays(1),
arrays(2),arrays(3),arrays(4),arrays(5))).toDF("id","src","dest","mode","fare","
year")
holidaysDf: org.apache.spark.sql.DataFrame = [id: string, src: string, dest: str
ing, mode: string, fare: string, year: string]

scala> val transrdd=sc.textFile("Transport.txt")
transrdd: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[24] at textFile at
 <console>:27

scala> val transportDF= transrdd.map(lines=>lines.split(",")).map(arrays=>(array
s(0),arrays(1))).toDF("transport_name","transport_id")
transportDF: org.apache.spark.sql.DataFrame = [transport_name: string, transport
_id: string]

scala> val userrdd = sc.textFile("S18_Dataset_User_details.txt");
userrdd: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[29] at textFile at
<console>:27

scala> val userDF= userrdd.map(lines=>lines.split(",")).map(arrays=>(arrays(0),a
rrays(1),arrays(2))).toDF("Person_ID","Name","Age");
userDF: org.apache.spark.sql.DataFrame = [Person_ID: string, Name: string, Age:
string]
```

```
scala> transportDF.registerTempTable("transport");

scala> holidaysDf.registerTempTable("holidays");

scala> userDF.registerTempTable("user");
```

1) What is the distribution of the total number of air-travelers per year

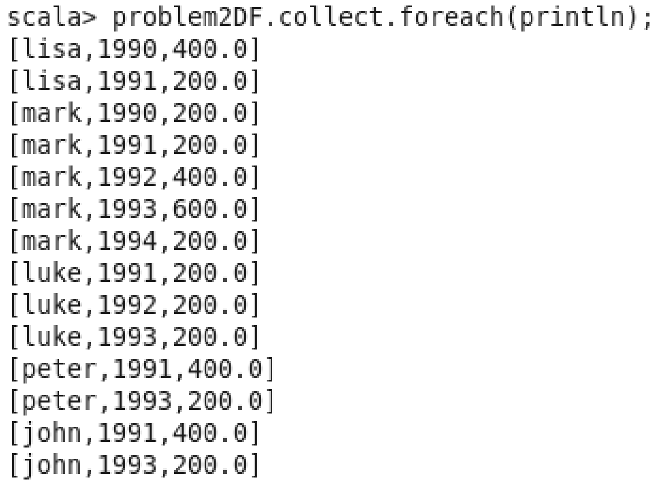
```
scala> holidaysDf.groupBy("year").count.show
+----+-----+
|year|count|
+----+-----+
|1990|    8|
|1991|    9|
|1992|    7|
|1993|    7|
|1994|    1|
+----+-----+
```



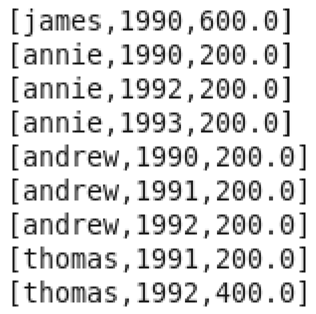
```
scala> holidaysDf.groupBy("year").count.show
+----+-----+
|year|count|
+----+-----+
|1990|    8|
|1991|    9|
|1992|    7|
|1993|    7|
|1994|    1|
+----+-----+
```

2) What is the total air distance covered by each user per year

```
scala> val joinDf=holidaysDf.as("j1").join(userDF.as("j2"),
$"j1.id"===$"j2.Person_ID").select($"j2.Name",$"j1.year",$"j1.fare");
joinDf: org.apache.spark.sql.DataFrame = [Name: string, year: string, fare: string]

scala> val Problem2Df=joinDf.groupBy("Name","year").agg(sum("fare"))
result: org.apache.spark.sql.DataFrame = [Name: string, year: string, sum(fare): double]
```

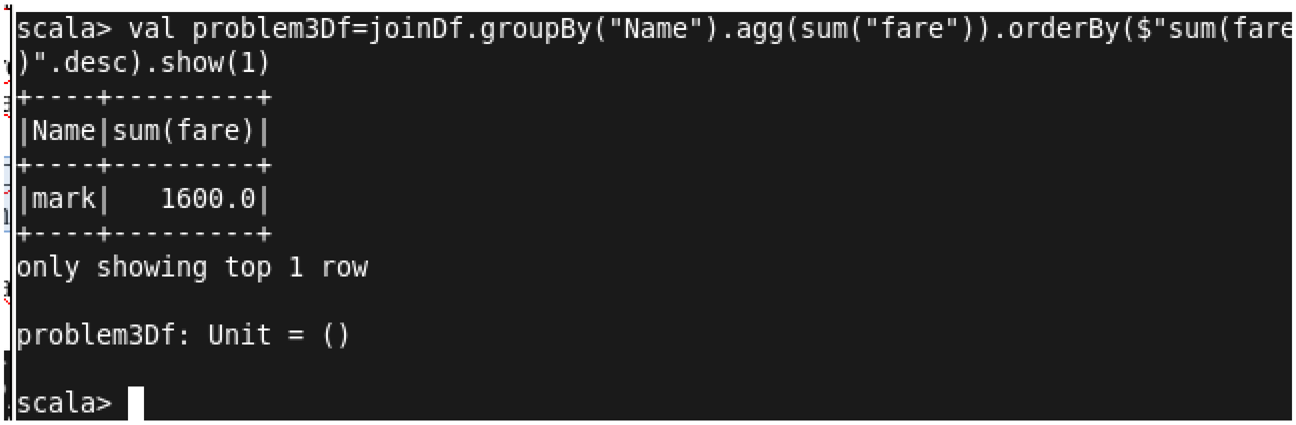


```
scala> problem2DF.collect.foreach(println);
[lisa,1990,400.0]
[lisa,1991,200.0]
[mark,1990,200.0]
[mark,1991,200.0]
[mark,1992,400.0]
[mark,1993,600.0]
[mark,1994,200.0]
[luke,1991,200.0]
[luke,1992,200.0]
[luke,1993,200.0]
[peter,1991,400.0]
[peter,1993,200.0]
[john,1991,400.0]
[john,1993,200.0]
```



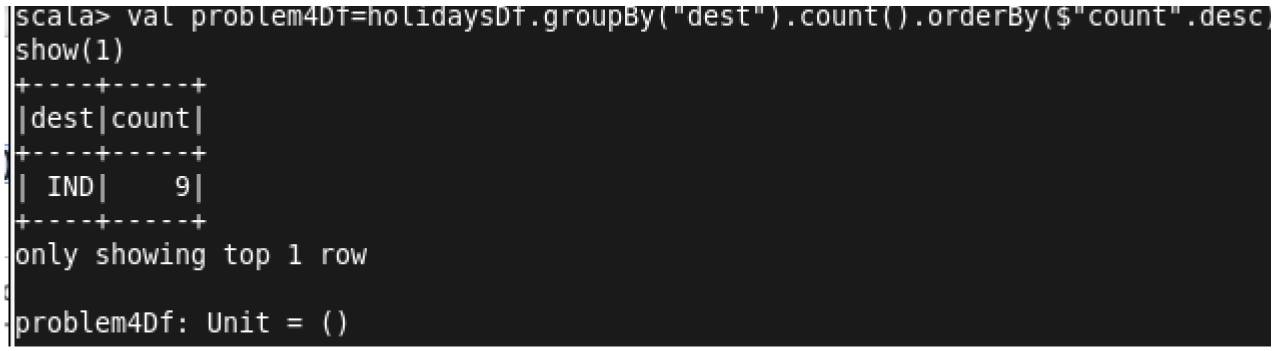
```
[james,1990,600.0]
[annie,1990,200.0]
[annie,1992,200.0]
[annie,1993,200.0]
[andrew,1990,200.0]
[andrew,1991,200.0]
[andrew,1992,200.0]
[thomas,1991,200.0]
[thomas,1992,400.0]
```

3) Which user has travelled the largest distance till date



```
scala> val problem3Df=joinDf.groupBy("Name").agg(sum("fare")).orderBy($"sum(fare
)".desc).show(1)
+----+---------+
|Name|sum(fare)|
+----+---------+
|mark|   1600.0|
+----+---------+
only showing top 1 row

problem3Df: Unit = ()

scala> 
```

4) What is the most preferred destination for all users.



```
scala> val problem4Df=holidaysDf.groupBy("dest").count().orderBy($"count".desc)
show(1)
+----+-----+
|dest|count|
+----+-----+
| IND|    9|
+----+-----+
only showing top 1 row

problem4Df: Unit = ()
```