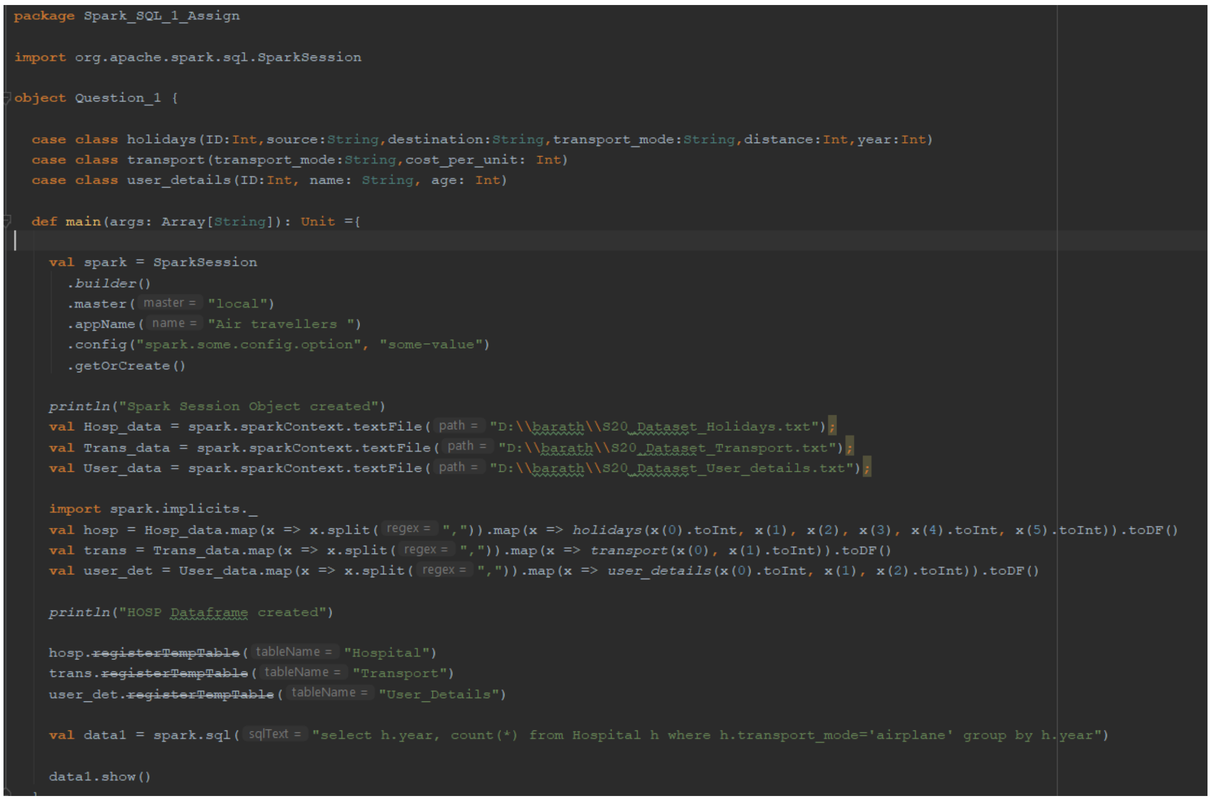
Assignment-20

Task 1

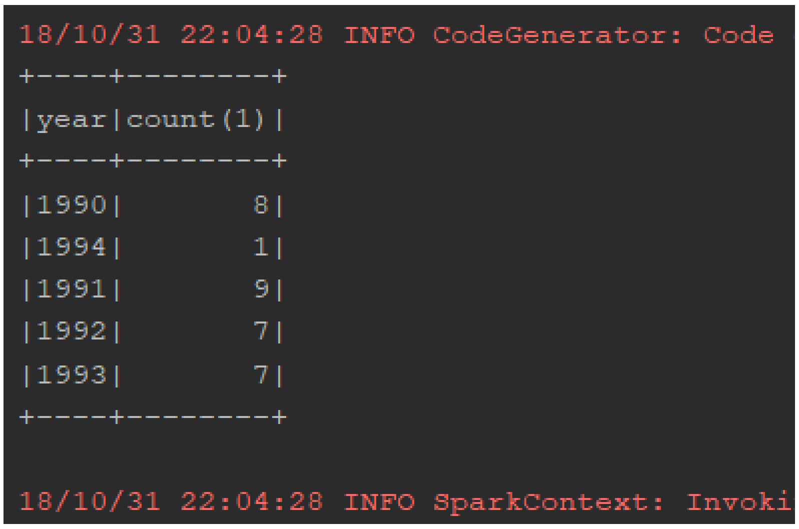
1) What is the distribution of the total number of air-travelers per year

The total number of air travellers per year has been found using the code below.



```
package Spark_SQL_1_Assign

import org.apache.spark.sql.SparkSession

object Question_1 {

  case class holidays(ID:Int,source:String,destination:String,transport_mode:String,distance:Int,year:Int)
  case class transport(transport_mode:String,cost_per_unit: Int)
  case class user_details(ID:Int, name: String, age: Int)

  def main(args: Array[String]): Unit ={

    val spark = SparkSession
      .builder()
      .master( master = "local")
      .appName( name = "Air travellers ")
      .config("spark.some.config.option", "some-value")
      .getOrCreate()

    println("Spark Session Object created")
    val Hosp_data = spark.sparkContext.textFile( path = "D:\\barath\\S20_Dataset_Holidays.txt");
    val Trans_data = spark.sparkContext.textFile( path = "D:\\barath\\S20_Dataset_Transport.txt");
    val User_data = spark.sparkContext.textFile( path = "D:\\barath\\S20_Dataset_User_details.txt");

    import spark.implicits._
    val hosp = Hosp_data.map(x => x.split( regex = ",")).map(x => holidays(x(0).toInt, x(1), x(2), x(3), x(4).toInt, x(5).toInt)).toDF()
    val trans = Trans_data.map(x => x.split( regex = ",")).map(x => transport(x(0), x(1).toInt)).toDF()
    val user_det = User_data.map(x => x.split( regex = ",")).map(x => user_details(x(0).toInt, x(1), x(2).toInt)).toDF()

    println("HOSP Dataframe created")

    hosp.registerTempTable( tableName = "Hospital")
    trans.registerTempTable( tableName = "Transport")
    user_det.registerTempTable( tableName = "User_Details")

    val data1 = spark.sql( sqlText = "select h.year, count(*) from Hospital h where h.transport_mode='airplane' group by h.year")

    data1.show()
```

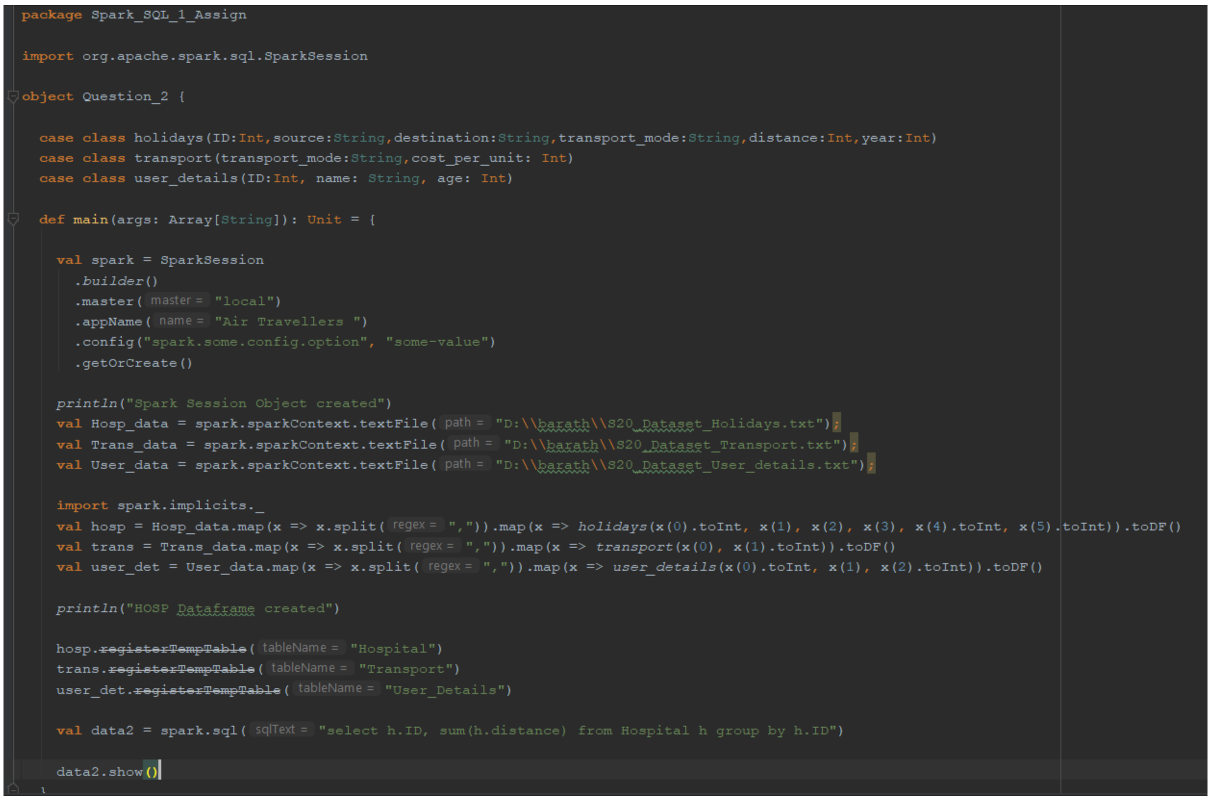
The output of the above code is shown below.



```
18/10/31 22:04:28 INFO CodeGenerator: Code
+----+--------+
|year|count(1)|
+----+--------+
|1990|       8|
|1994|       1|
|1991|       9|
|1992|       7|
|1993|       7|
+----+--------+

18/10/31 22:04:28 INFO SparkContext: Invoki
```

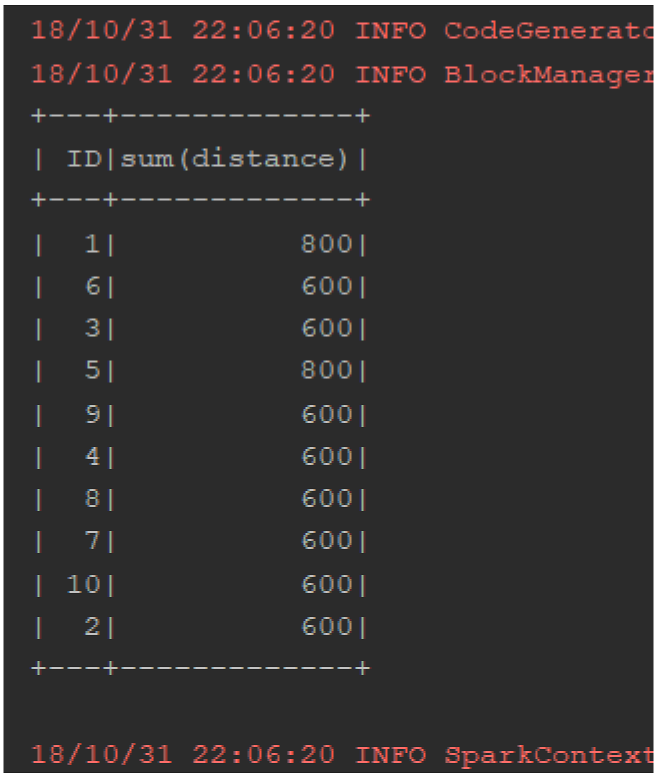
2) What is the total air distance covered by each user per year

The air distance covered by each user per year has been found using the code below.



```
package Spark_SQL_1_Assign

import org.apache.spark.sql.SparkSession

object Question_2 {

  case class holidays(ID:Int,source:String,destination:String,transport_mode:String,distance:Int,year:Int)
  case class transport(transport_mode:String,cost_per_unit: Int)
  case class user_details(ID:Int, name: String, age: Int)

  def main(args: Array[String]): Unit = {

    val spark = SparkSession
      .builder()
      .master( master = "local")
      .appName( name = "Air Travellers ")
      .config("spark.some.config.option", "some-value")
      .getOrCreate()

    println("Spark Session Object created")
    val Hosp_data = spark.sparkContext.textFile( path = "D:\\barath\\S20_Dataset_Holidays.txt");
    val Trans_data = spark.sparkContext.textFile( path = "D:\\barath\\S20_Dataset_Transport.txt");
    val User_data = spark.sparkContext.textFile( path = "D:\\barath\\S20_Dataset_User_details.txt");

    import spark.implicits._
    val hosp = Hosp_data.map(x => x.split( regex = ",")).map(x => holidays(x(0).toInt, x(1), x(2), x(3), x(4).toInt, x(5).toInt)).toDF()
    val trans = Trans_data.map(x => x.split( regex = ",")).map(x => transport(x(0), x(1).toInt)).toDF()
    val user_det = User_data.map(x => x.split( regex = ",")).map(x => user_details(x(0).toInt, x(1), x(2).toInt)).toDF()

    println("HOSP Dataframe created")

    hosp.~~registerTempTable~~( tableName = "Hospital")
    trans.~~registerTempTable~~( tableName = "Transport")
    user_det.~~registerTempTable~~( tableName = "User_Details")

    val data2 = spark.sql( sqlText = "select h.ID, sum(h.distance) from Hospital h group by h.ID")

    data2.show()
  }
```

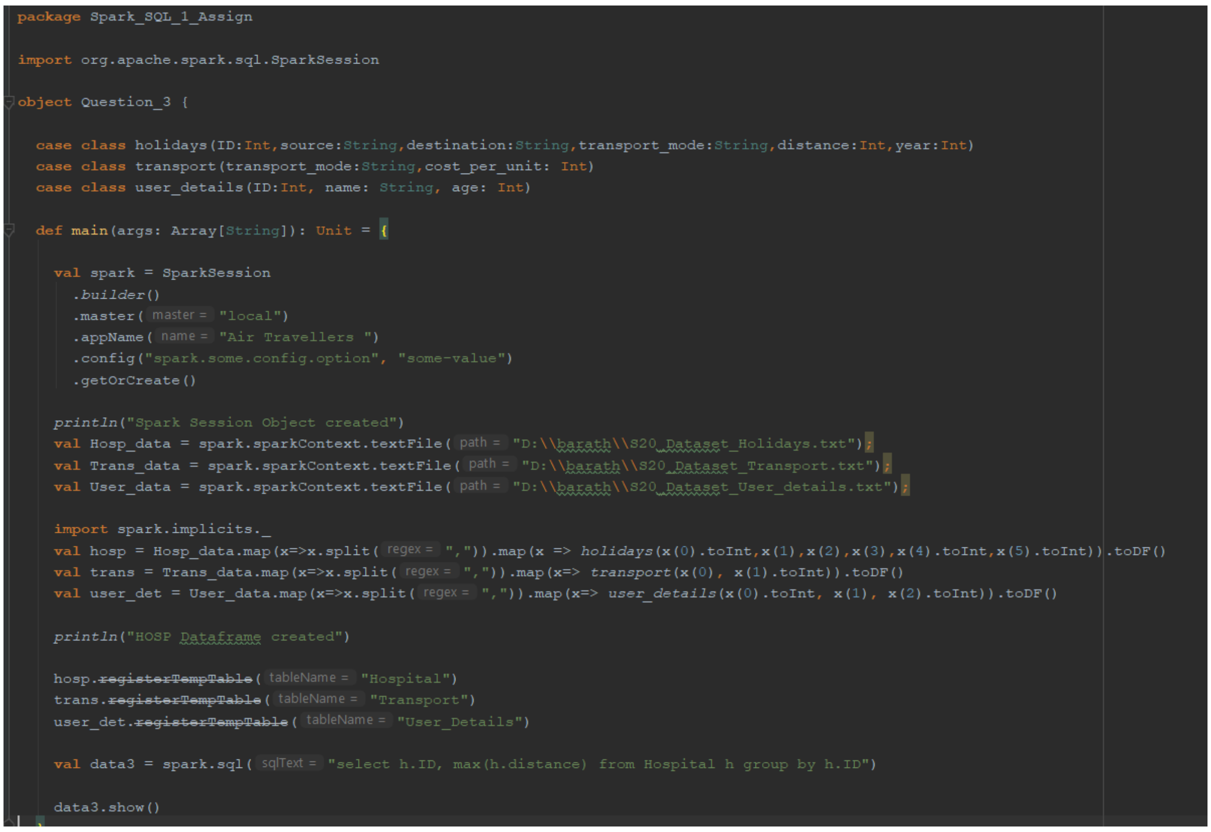
The output of the above code has been shown below.



```
18/10/31 22:06:20 INFO CodeGenerato
18/10/31 22:06:20 INFO BlockManager
+---+-------------+
| ID|sum(distance)|
+---+-------------+
|  1|          800|
|  6|          600|
|  3|          600|
|  5|          800|
|  9|          600|
|  4|          600|
|  8|          600|
|  7|          600|
| 10|          600|
|  2|          600|
+---+-------------+

18/10/31 22:06:20 INFO SparkContext
```

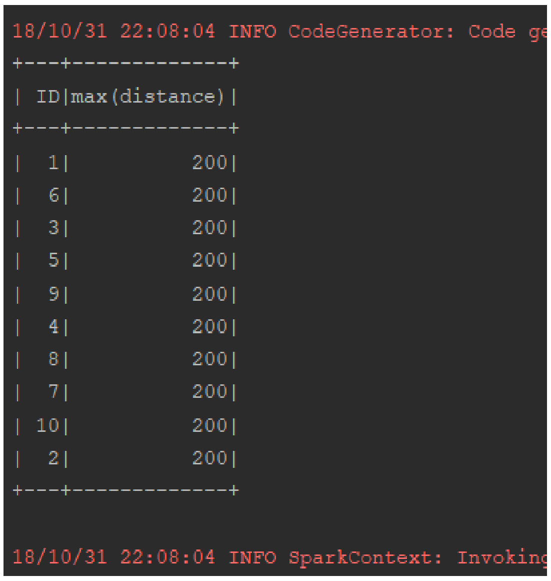
3) Which user has travelled the largest distance till date

The largest distance travelled till date has been found using the code below.



```
package Spark_SQL_1_Assign

import org.apache.spark.sql.SparkSession

object Question_3 {

  case class holidays(ID:Int,source:String,destination:String,transport_mode:String,distance:Int,year:Int)
  case class transport(transport_mode:String,cost_per_unit: Int)
  case class user_details(ID:Int, name: String, age: Int)

  def main(args: Array[String]): Unit = {

    val spark = SparkSession
      .builder()
      .master( master = "local")
      .appName( name = "Air Travellers ")
      .config("spark.some.config.option", "some-value")
      .getOrCreate()

    println("Spark Session Object created")
    val Hosp_data = spark.sparkContext.textFile( path = "D:\\barath\\S20_Dataset_Holidays.txt");
    val Trans_data = spark.sparkContext.textFile( path = "D:\\barath\\S20_Dataset_Transport.txt");
    val User_data = spark.sparkContext.textFile( path = "D:\\barath\\S20_Dataset_User_details.txt");

    import spark.implicits._
    val hosp = Hosp_data.map(x=>x.split( regex = ",")).map(x => holidays(x(0).toInt,x(1),x(2),x(3),x(4).toInt,x(5).toInt)).toDF()
    val trans = Trans_data.map(x=>x.split( regex = ",")).map(x=> transport(x(0), x(1).toInt)).toDF()
    val user_det = User_data.map(x=>x.split( regex = ",")).map(x=> user_details(x(0).toInt, x(1), x(2).toInt)).toDF()

    println("HOSP Dataframe created")

    hosp.~~registerTempTable~~( tableName = "Hospital")
    trans.~~registerTempTable~~( tableName = "Transport")
    user_det.~~registerTempTable~~( tableName = "User_Details")

    val data3 = spark.sql( sqlText = "select h.ID, max(h.distance) from Hospital h group by h.ID")

    data3.show()
```

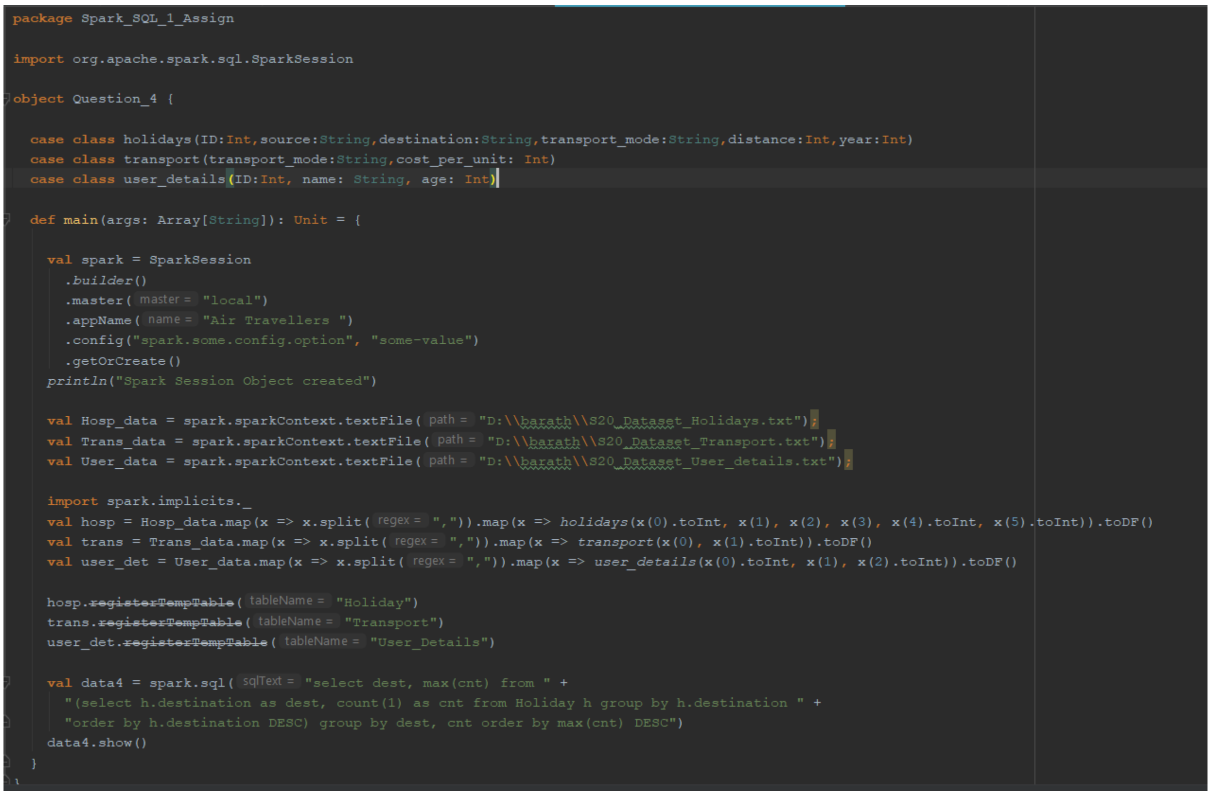
The output of the above code is shown below.



```
18/10/31 22:08:04 INFO CodeGenerator: Code ge
+---+-------------+
| ID|max(distance)|
+---+-------------+
|  1|          200|
|  6|          200|
|  3|          200|
|  5|          200|
|  9|          200|
|  4|          200|
|  8|          200|
|  7|          200|
| 10|          200|
|  2|          200|
+---+-------------+

18/10/31 22:08:04 INFO SparkContext: Invoking
```

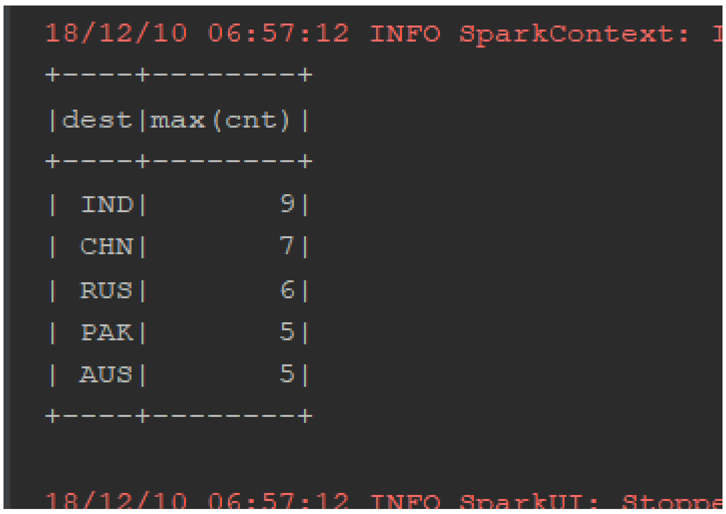
4) What is the most preferred destination for all users.

The scala code for the most preferred destinations for all users is shown below.



```
package Spark_SQL_1_Assign

import org.apache.spark.sql.SparkSession

object Question_4 {

  case class holidays(ID:Int,source:String,destination:String,transport_mode:String,distance:Int,year:Int)
  case class transport(transport_mode:String,cost_per_unit: Int)
  case class user_details(ID:Int, name: String, age: Int)

  def main(args: Array[String]): Unit = {

    val spark = SparkSession
      .builder()
      .master( master = "local")
      .appName( name = "Air Travellers ")
      .config("spark.some.config.option", "some-value")
      .getOrCreate()
    println("Spark Session Object created")

    val Hosp_data = spark.sparkContext.textFile( path = "D:\\barath\\S20_Dataset_Holidays.txt");
    val Trans_data = spark.sparkContext.textFile( path = "D:\\barath\\S20_Dataset_Transport.txt");
    val User_data = spark.sparkContext.textFile( path = "D:\\barath\\S20_Dataset_User_details.txt");

    import spark.implicits._
    val hosp = Hosp_data.map(x => x.split( regex = ",")).map(x => holidays(x(0).toInt, x(1), x(2), x(3), x(4).toInt, x(5).toInt)).toDF()
    val trans = Trans_data.map(x => x.split( regex = ",")).map(x => transport(x(0), x(1).toInt)).toDF()
    val user_det = User_data.map(x => x.split( regex = ",")).map(x => user_details(x(0).toInt, x(1), x(2).toInt)).toDF()

    hosp.registerTempTable( tableName = "Holiday")
    trans.registerTempTable( tableName = "Transport")
    user_det.registerTempTable( tableName = "User_Details")

    val data4 = spark.sql( sqlText = "select dest, max(cnt) from " +
      "(select h.destination as dest, count(1) as cnt from Holiday h group by h.destination " +
      "order by h.destination DESC) group by dest, cnt order by max(cnt) DESC")
    data4.show()
  }
}
```

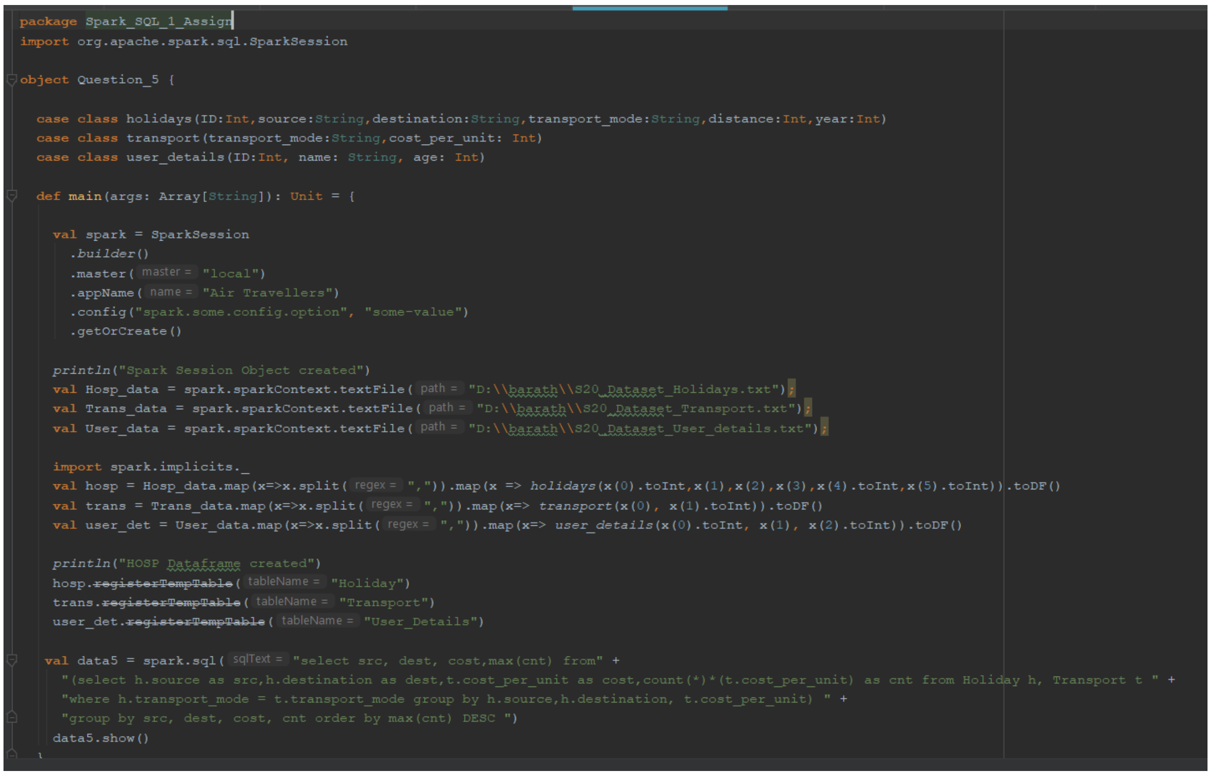
The output to the above code is shown below.



```
18/12/10 06:57:12 INFO SparkContext: I
+----+--------+
|dest|max(cnt)|
+----+--------+
| IND|       9|
| CHN|       7|
| RUS|       6|
| PAK|       5|
| AUS|       5|
+----+--------+

18/12/10 06:57:12 INFO SparkUI: Stoppe
```

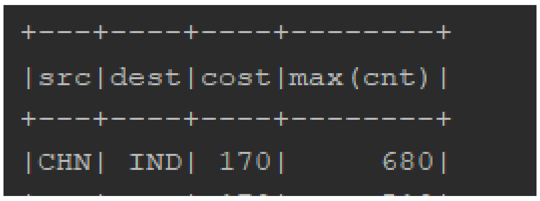
5) Which route is generating the most revenue per year

The scala code for the most revenue generating route per year is shown below.



```
package Spark_SQL_1_Assign
import org.apache.spark.sql.SparkSession

object Question_5 {

  case class holidays(ID:Int,source:String,destination:String,transport_mode:String,distance:Int,year:Int)
  case class transport(transport_mode:String,cost_per_unit: Int)
  case class user_details(ID:Int, name: String, age: Int)

  def main(args: Array[String]): Unit = {

    val spark = SparkSession
      .builder()
      .master("local")
      .appName("Air Travellers")
      .config("spark.some.config.option", "some-value")
      .getOrCreate()

    println("Spark Session Object created")
    val Hosp_data = spark.sparkContext.textFile("D:\\barath\\S20_Dataset_Holidays.txt");
    val Trans_data = spark.sparkContext.textFile("D:\\barath\\S20_Dataset_Transport.txt");
    val User_data = spark.sparkContext.textFile("D:\\barath\\S20_Dataset_User_details.txt");

    import spark.implicits._
    val hosp = Hosp_data.map(x=>x.split(",")).map(x => holidays(x(0).toInt,x(1),x(2),x(3),x(4).toInt,x(5).toInt)).toDF()
    val trans = Trans_data.map(x=>x.split(",")).map(x=> transport(x(0), x(1).toInt)).toDF()
    val user_det = User_data.map(x=>x.split(",")).map(x=> user_details(x(0).toInt, x(1), x(2).toInt)).toDF()

    println("HOSP Dataframe created")
    hosp.registerTempTable("Holiday")
    trans.registerTempTable("Transport")
    user_det.registerTempTable("User_Details")

   val data5 = spark.sql("select src, dest, cost,max(cnt) from" +
     "(select h.source as src,h.destination as dest,t.cost_per_unit as cost,count(*)*(t.cost_per_unit) as cnt from Holiday h, Transport t " +
     "where h.transport_mode = t.transport_mode group by h.source,h.destination, t.cost_per_unit) " +
     "group by src, dest, cost, cnt order by max(cnt) DESC ")
    data5.show()
  }
```

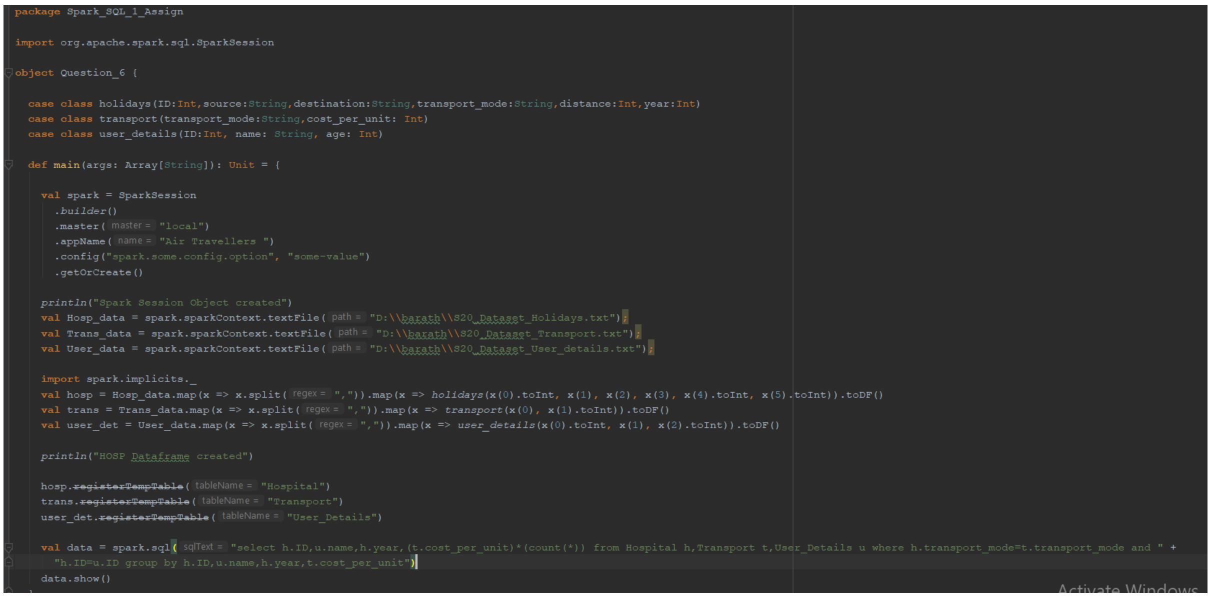
The output to the above code is shown below.



```
+---+----+----+--------+
|src|dest|cost|max(cnt)|
+---+----+----+--------+
|CHN| IND| 170|     680|
```

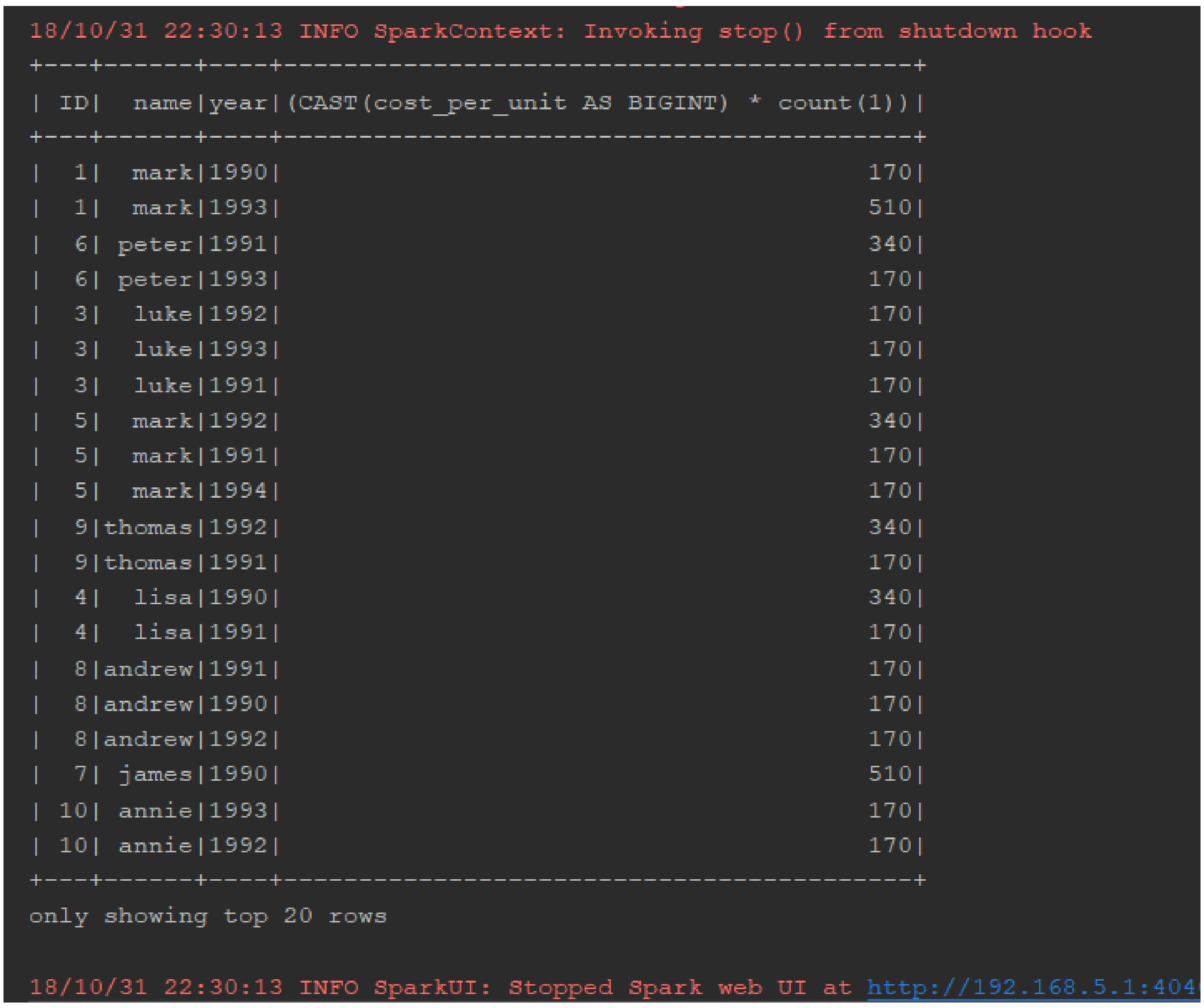
6) What is the total amount spent by every user on air-travel per year

The scala code for the total amount spent by every user on air-travel per year is shown below.



```
package Spark_SQL_1_Assign

import org.apache.spark.sql.SparkSession

object Question_6 {

  case class holidays(ID:Int,source:String,destination:String,transport_mode:String,distance:Int,year:Int)
  case class transport(transport_mode:String,cost_per_unit: Int)
  case class user_details(ID:Int, name: String, age: Int)

  def main(args: Array[String]): Unit = {

    val spark = SparkSession
      .builder()
      .master( master = "local")
      .appName( name = "Air Travellers ")
      .config("spark.some.config.option", "some-value")
      .getOrCreate()

    println("Spark Session Object created")
    val Hosp_data = spark.sparkContext.textFile( path = "D:\\barath\\S20_Dataset_Holidays.txt");
    val Trans_data = spark.sparkContext.textFile( path = "D:\\barath\\S20_Dataset_Transport.txt");
    val User_data = spark.sparkContext.textFile( path = "D:\\barath\\S20_Dataset_User_details.txt");

    import spark.implicits._
    val hosp = Hosp_data.map(x => x.split( regex = ",")).map(x => holidays(x(0).toInt, x(1), x(2), x(3), x(4).toInt, x(5).toInt)).toDF()
    val trans = Trans_data.map(x => x.split( regex = ",")).map(x => transport(x(0), x(1).toInt)).toDF()
    val user_det = User_data.map(x => x.split( regex = ",")).map(x => user_details(x(0).toInt, x(1), x(2).toInt)).toDF()

    println("HOSP Dataframe created")

    hosp.registerTempTable( tableName = "Hospital")
    trans.registerTempTable( tableName = "Transport")
    user_det.registerTempTable( tableName = "User_Details")

    val data = spark.sql( sqlText = "select h.ID,u.name,h.year,(t.cost_per_unit)*(count(*)) from Hospital h,Transport t,User_Details u where h.transport_mode=t.transport_mode and " +
      "h.ID=u.ID group by h.ID,u.name,h.year,t.cost_per_unit")
    data.show()
```

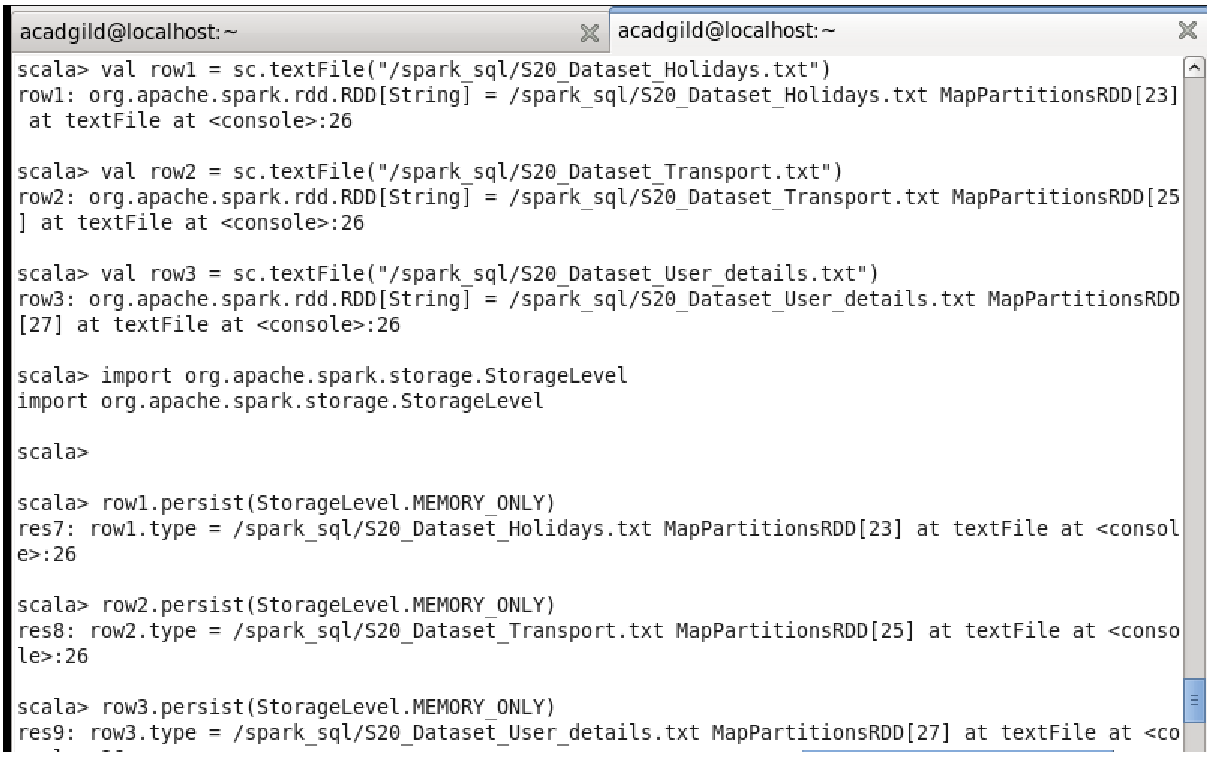
The output to the above code is shown below.



```
18/10/31 22:30:13 INFO SparkContext: Invoking stop() from shutdown hook
+---+------+----+----------------------------------------+
| ID|  name|year|(CAST(cost_per_unit AS BIGINT) * count(1))|
+---+------+----+----------------------------------------+
|  1|  mark|1990|                                     170|
|  1|  mark|1993|                                     510|
|  6| peter|1991|                                     340|
|  6| peter|1993|                                     170|
|  3|  luke|1992|                                     170|
|  3|  luke|1993|                                     170|
|  3|  luke|1991|                                     170|
|  5|  mark|1992|                                     340|
|  5|  mark|1991|                                     170|
|  5|  mark|1994|                                     170|
|  9|thomas|1992|                                     340|
|  9|thomas|1991|                                     170|
|  4|  lisa|1990|                                     340|
|  4|  lisa|1991|                                     170|
|  8|andrew|1991|                                     170|
|  8|andrew|1990|                                     170|
|  8|andrew|1992|                                     170|
|  7| james|1990|                                     510|
| 10| annie|1993|                                     170|
| 10| annie|1992|                                     170|
+---+------+----+----------------------------------------+
only showing top 20 rows

18/10/31 22:30:13 INFO SparkUI: Stopped Spark web UI at http://192.168.5.1:404
```

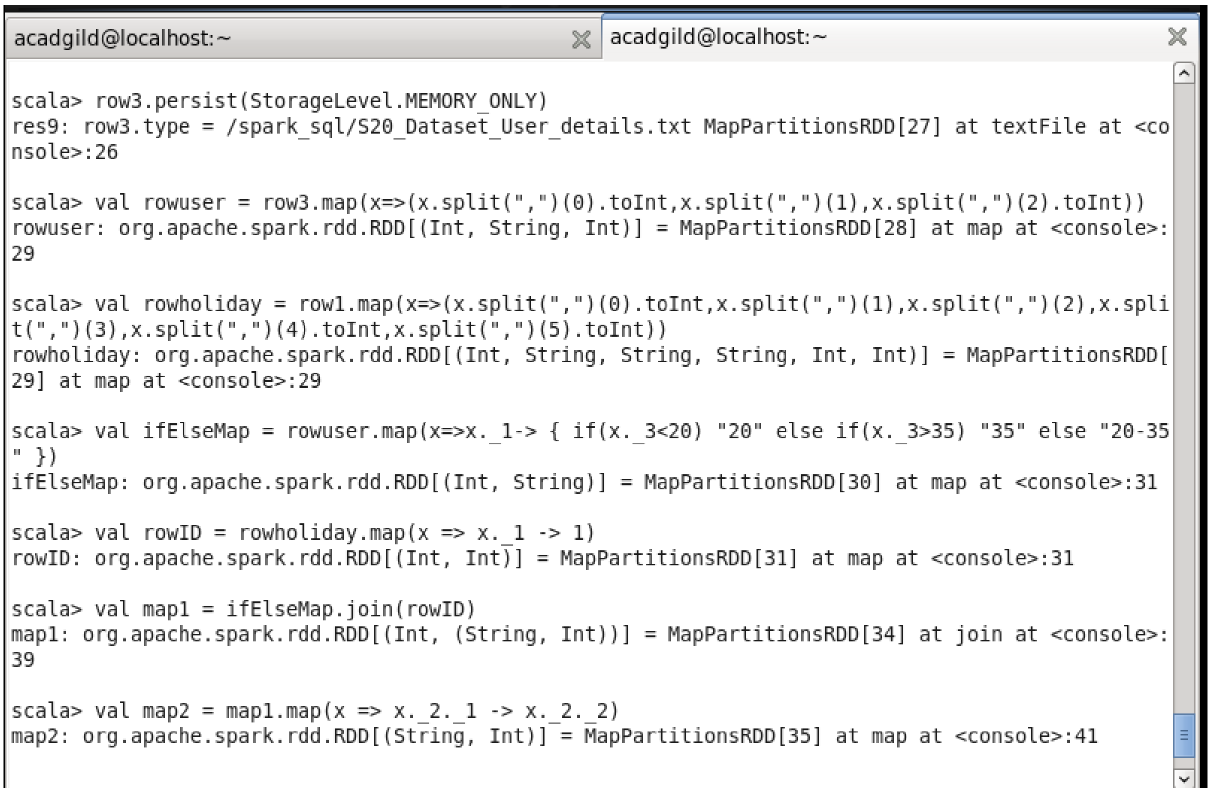
7) Considering age groups of < 20, 20-35, 35 >, which age group is travelling the most every year.

The scala code for the above question is shown in the below screenshot.



acadgild@localhost:~ acadgild@localhost:~

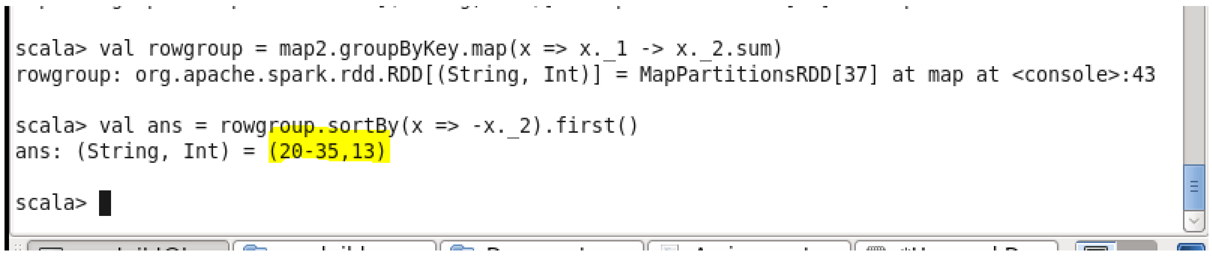
```
scala> val row1 = sc.textFile("/spark_sql/S20_Dataset_Holidays.txt")
row1: org.apache.spark.rdd.RDD[String] = /spark_sql/S20_Dataset_Holidays.txt MapPartitionsRDD[23]
 at textFile at <console>:26

scala> val row2 = sc.textFile("/spark_sql/S20_Dataset_Transport.txt")
row2: org.apache.spark.rdd.RDD[String] = /spark_sql/S20_Dataset_Transport.txt MapPartitionsRDD[25
] at textFile at <console>:26

scala> val row3 = sc.textFile("/spark_sql/S20_Dataset_User_details.txt")
row3: org.apache.spark.rdd.RDD[String] = /spark_sql/S20_Dataset_User_details.txt MapPartitionsRDD
[27] at textFile at <console>:26

scala> import org.apache.spark.storage.StorageLevel
import org.apache.spark.storage.StorageLevel

scala>

scala> row1.persist(StorageLevel.MEMORY_ONLY)
res7: row1.type = /spark_sql/S20_Dataset_Holidays.txt MapPartitionsRDD[23] at textFile at <consol
e>:26

scala> row2.persist(StorageLevel.MEMORY_ONLY)
res8: row2.type = /spark_sql/S20_Dataset_Transport.txt MapPartitionsRDD[25] at textFile at <conso
le>:26

scala> row3.persist(StorageLevel.MEMORY_ONLY)
res9: row3.type = /spark_sql/S20_Dataset_User_details.txt MapPartitionsRDD[27] at textFile at <co
```



acadgild@localhost:~ acadgild@localhost:~

```
scala> row3.persist(StorageLevel.MEMORY_ONLY)
res9: row3.type = /spark_sql/S20_Dataset_User_details.txt MapPartitionsRDD[27] at textFile at <co
nsole>:26

scala> val rowuser = row3.map(x=>(x.split(",")(0).toInt,x.split(",")(1),x.split(",")(2).toInt))
rowuser: org.apache.spark.rdd.RDD[(Int, String, Int)] = MapPartitionsRDD[28] at map at <console>:
29

scala> val rowholiday = row1.map(x=>(x.split(",")(0).toInt,x.split(",")(1),x.split(",")(2),x.spli
t(",")(3),x.split(",")(4).toInt,x.split(",")(5).toInt))
rowholiday: org.apache.spark.rdd.RDD[(Int, String, String, String, Int, Int)] = MapPartitionsRDD[
29] at map at <console>:29

scala> val ifElseMap = rowuser.map(x=>x._1-> { if(x._3<20) "20" else if(x._3>35) "35" else "20-35
" })
ifElseMap: org.apache.spark.rdd.RDD[(Int, String)] = MapPartitionsRDD[30] at map at <console>:31

scala> val rowID = rowholiday.map(x => x._1 -> 1)
rowID: org.apache.spark.rdd.RDD[(Int, Int)] = MapPartitionsRDD[31] at map at <console>:31

scala> val map1 = ifElseMap.join(rowID)
map1: org.apache.spark.rdd.RDD[(Int, (String, Int))] = MapPartitionsRDD[34] at join at <console>:
39

scala> val map2 = map1.map(x => x._2._1 -> x._2._2)
map2: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[35] at map at <console>:41
```

The output of age group travelling the most every year is highlighted in the below screenshot.



```
scala> val rowgroup = map2.groupByKey.map(x => x._1 -> x._2.sum)
rowgroup: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[37] at map at <console>:43

scala> val ans = rowgroup.sortBy(x => -x._2).first()
ans: (String, Int) = (20-35,13)

scala> 
```