ASSIGNMENT 18.1

Input Datasets:

We have an airline data with us:

user details:

user_id, name, age

holidays:

user_id, src, dest, travel_mode, distance, year_of_travel

transport:

travel_mode, cost_per_unit

Here are the datasets which we will be using for this assignment in all problems. It has been kept in local file system:

S18_Dataset_Holidays.txt

```
1,CHN,IND,airplane,200,1990
2,IND,CHN,airplane,200,1991
3,IND,CHN,airplane,200,1992
4,RUS,IND,airplane,200,1990
5,CHN,RUS,airplane,200,1992
6,AUS,PAK,airplane,200,1991
7,RUS,AUS,airplane,200,1990
8,IND,RUS,airplane,200,1991
9,CHN,RUS,airplane,200,1992
10,AUS,CHN,airplane,200,1993
1,AUS,CHN,airplane,200,1993
2,CHN,IND,airplane,200,1993
3,CHN,IND,airplane,200,1993
4,IND,AUS,airplane,200,1991
5,AUS,IND,airplane,200,1992
6,RUS,CHN,airplane,200,1993
7,CHN,RUS,airplane,200,1990
8,AUS,CHN,airplane,200,1990
9,IND,AUS,airplane,200,1991
10,RUS,CHN,airplane,200,1992
1,PAK,IND,airplane,200,1993
2,IND,RUS,airplane,200,1991
3,CHN,PAK,airplane,200,1991
4,CHN,PAK,airplane,200,1990
5,IND,PAK,airplane,200,1991
6,PAK,RUS,airplane,200,1991
7,CHN,IND,airplane,200,1990
8,RUS,IND,airplane,200,1992
9,RUS,IND,airplane,200,1992
10,CHN,AUS,airplane,200,1990
1,PAK,AUS,airplane,200,1993
5,CHN,PAK,airplane,200,1994
```

S18_Dataset_Holidays.txt | S18_Dataset_Transport.txt

```
airplane,170
car,140
train,120
ship,200
```

S18_Dataset_Holidays.txt | S18_Dataset_Transport.txt | S18_Dataset_User_details.txt

```
1,mark,15
2,john,16
3,luke,17
4,lisa,27
5,mark,25
6,peter,22
7,james,21
8,andrew,55
9,thomas,46
10,annie,44
```

Problem Statement:

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

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

3) Which user has travelled the largest distance till date?

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

Solution:

1. Here is the Spark code snippet to find distribution of total number of air-travelers per year:

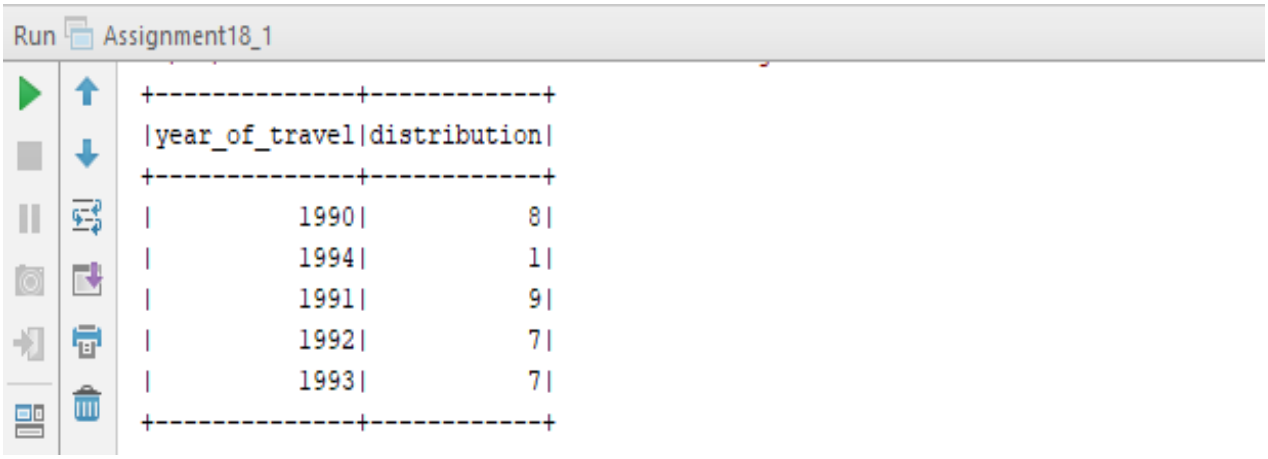
```
// import required Spark packages

import org.apache.spark.sql.SparkSession

import org.apache.spark.sql.types.{IntegerType, StringType}


object Assignment18_1 {
  def main(args: Array[String]): Unit = {
```

```
  val spark = SparkSession                    // create a SparkSession object that can be used to
    .builder()                                // create various contexts of Spark such as sqlContext
    .config("spark.sql.warehouse.dir", "file:///c:/tmp/spark-warehouse")
    .master("local[*]")
    .getOrCreate()
  val sqlContext = spark.sqlContext                          // initialize sqlContext
  val input_df = sqlContext.read.csv("E:\\Acadgild\\Session 18\\S18_Datasets\\S18_ Dataset_
Holidays.txt")                                               // load input data file – holidays.txt
  val holidaysDF = input_df.select (                         // define schema for input data loaded
    input_df("_c0").cast(IntegerType).as("user_id"),        //assign column names to the data frame
    input_df("_c1").cast(StringType).as("src"),
    input_df("_c2").cast(StringType).as("dest"),
    input_df("_c3").cast(StringType).as("travel_mode"),
    input_df("_c4").cast(IntegerType).as("distance"),
    input_df("_c5").cast(IntegerType).as("year_of_travel"))
  holidaysDF.createOrReplaceTempView("holidays")        // create a temporary view - holidays
  sqlContext.sql("SELECT year_of_travel, COUNT (*) as distribution from holidays GROUP
BY year_of_travel").show()                              // SQL query to show distribution per year
 }
}
```

Assignment18_1.scala

```
6   object Assignment18_1 {
7
8     def main(args: Array[String]): Unit = {
9
10      val spark = SparkSession
11        .builder()
12        .config("spark.sql.warehouse.dir", "file:///c:/tmp/spark-warehouse")
13        .master("local[*]")
14        .getOrCreate()
15      val sqlContext = spark.sqlContext
16      val input_df = sqlContext.read.csv("E:\\Acadgild\\Session 18\\S18_Datasets\\S18_Dataset_Holidays.txt")
17      val holidaysDF = input_df.select(
18        input_df("_c0").cast(IntegerType).as("user_id"),
19        input_df("_c1").cast(StringType).as("src"),
20        input_df("_c2").cast(StringType).as("dest"),
21        input_df("_c3").cast(StringType).as("travel_mode"),
22        input_df("_c4").cast(IntegerType).as("distance"),
23        input_df("_c5").cast(IntegerType).as("year_of_travel")
24      )
25      holidaysDF.createOrReplaceTempView("holidays")
26      sqlContext.sql("SELECT year_of_travel, COUNT(*) as distribution from holidays GROUP BY year_of_travel").show()
27    }
28  }
```

Output:



2. Here is the Spark code snippet to find total air distance covered by each user per year:

```
// import required Spark packages

import org.apache.spark.sql.SparkSession

import org.apache.spark.sql.types.{IntegerType, StringType}


object Assignment18_1 {
  def main(args: Array[String]): Unit = {
    val spark = SparkSession                    // create a SparkSession object that can be used to
      .builder()                                // create various contexts of Spark such as sqlContext
      .config("spark.sql.warehouse.dir", "file:///c:/tmp/spark-warehouse")
      .master("local[*]")
      .getOrCreate()
    val sqlContext = spark.sqlContext                       // initialize sqlContext
    val input_df = sqlContext.read.csv("E:\\Acadgild\\Session 18\\S18_Datasets\\S18_ Dataset_
Holidays.txt")                                              // load input data file – holidays.txt
    val holidaysDF = input_df.select (                      // define schema for input data loaded
      input_df("_c0").cast(IntegerType).as("user_id"),      //assign column names to the data frame
      input_df("_c1").cast(StringType).as("src"),
      input_df("_c2").cast(StringType).as("dest"),
      input_df("_c3").cast(StringType).as("travel_mode"),
```

```
    input_df("_c4").cast(IntegerType).as("distance"),
    input_df("_c5").cast(IntegerType).as("year_of_travel"))
  holidaysDF.createOrReplaceTempView("holidays")        // create a temporary view - holidays
  sqlContext.sql("SELECT user_id, SUM(distance) from holidays GROUP BY user_id").show()
                                                        // SQL query to show distance covered by each user
 }
}
```

Assignment18_1.scala

```scala
object Assignment18_1 {
  def main(args: Array[String]): Unit = {
    val spark = SparkSession
      .builder()
      .config("spark.sql.warehouse.dir", "file:///c:/tmp/spark-warehouse")
      .master("local[*]")
      .getOrCreate()
    val sqlContext = spark.sqlContext
    val input_df = sqlContext.read.csv("E:\\Acadgild\\Session 18\\S18_Datasets\\S18_Dataset_Holidays.txt")
    val holidaysDF = input_df.select(
      input_df("_c0").cast(IntegerType).as("user_id"),
      input_df("_c1").cast(StringType).as("src"),
      input_df("_c2").cast(StringType).as("dest"),
      input_df("_c3").cast(StringType).as("travel_mode"),
      input_df("_c4").cast(IntegerType).as("distance"),
      input_df("_c5").cast(IntegerType).as("year_of_travel"))
    holidaysDF.createOrReplaceTempView("holidays")
    sqlContext.sql("SELECT user_id, SUM(distance) from holidays GROUP BY user_id").show()
  }
```

Output:

Run Assignment18_1

```
+-------+-------------+
|user_id|sum(distance)|
+-------+-------------+
|      1|          800|
|      6|          600|
|      3|          600|
|      5|          800|
|      9|          600|
|      4|          600|
|      8|          600|
|      7|          600|
|     10|          600|
|      2|          600|
+-------+-------------+
```

3. Here is the Spark code snippet to find which user has travelled the largest distance till date:

```
// import required Spark packages

import org.apache.spark.sql.SparkSession

import org.apache.spark.sql.types.{IntegerType, StringType}


object Assignment18_1 {
  def main(args: Array[String]): Unit = {
    val spark = SparkSession                    // create a SparkSession object that can be used to
      .builder()                                // create various contexts of Spark such as sqlContext
      .config("spark.sql.warehouse.dir", "file:///c:/tmp/spark-warehouse")
      .master("local[*]")
      .getOrCreate()
    val sqlContext = spark.sqlContext                          // initialize sqlContext
    val input_df = sqlContext.read.csv("E:\\Acadgild\\Session 18\\S18_Datasets\\S18_ Dataset_
Holidays.txt")                                                 // load input data file – holidays.txt
    val holidaysDF = input_df.select (                         // define schema for input data loaded
      input_df("_c0").cast(IntegerType).as("user_id"),        //assign column names to the data frame
      input_df("_c1").cast(StringType).as("src"),
      input_df("_c2").cast(StringType).as("dest"),
      input_df("_c3").cast(StringType).as("travel_mode"),
      input_df("_c4").cast(IntegerType).as("distance"),
      input_df("_c5").cast(IntegerType).as("year_of_travel"))
    holidaysDF.createOrReplaceTempView("holidays")        // create a temporary view - holidays
    val result = sqlContext.sql("SELECT user_id, SUM(distance) AS total_distance " +
                "FROM holidays " +
                "GROUP BY user_id " +
                "ORDER BY total_distance DESC").take(1)
                                        // SQL query to find the user who covered largest distance
     result.foreach(println)            // print the result
  }
}
```

Assignment18_1.scala

```scala
object Assignment18_1 {
  def main(args: Array[String]): Unit = {
    val spark = SparkSession
      .builder()
      .config("spark.sql.warehouse.dir", "file:///c:/tmp/spark-warehouse")
      .master("local[*]")
      .getOrCreate()
    val sqlContext = spark.sqlContext
    val input_df = sqlContext.read.csv("E:\\Acadgild\\Session 18\\S18_Datasets\\S18_Dataset_Holidays.txt")
    val holidaysDF = input_df.select(
      input_df("_c0").cast(IntegerType).as("user_id"),
      input_df("_c1").cast(StringType).as("src"),
      input_df("_c2").cast(StringType).as("dest"),
      input_df("_c3").cast(StringType).as("travel_mode"),
      input_df("_c4").cast(IntegerType).as("distance"),
      input_df("_c5").cast(IntegerType).as("year_of_travel"))
    holidaysDF.createOrReplaceTempView("holidays")
    val result = sqlContext.sql("SELECT user_id, SUM(distance) AS total_distance " +
                "FROM holidays " +
                "GROUP BY user_id " +
                "ORDER BY total_distance DESC").take(1)
    result.foreach(println)
  }
}
```

Output:

Run Assignment18_1

```
18/02/26 22:10:15 INFO DAGScheduler: Job 1 finished: take at Assignment18_1.scala:26, took 3.250373 s
18/02/26 22:10:15 INFO CodeGenerator: Code generated in 48.258411 ms
[1,800]
18/02/26 22:10:15 INFO SparkContext: Invoking stop() from shutdown hook
18/02/26 22:10:15 INFO SparkUI: Stopped Spark web UI at http://192.168.43.50:4040
18/02/26 22:10:15 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
18/02/26 22:10:15 INFO MemoryStore: MemoryStore cleared
18/02/26 22:10:15 INFO BlockManager: BlockManager stopped
```

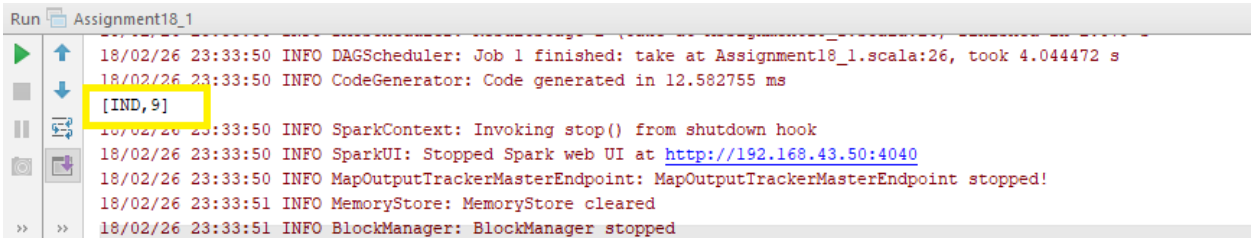
We can conclude from the above output that the user with id **1** has covered the largest distance of **800 kms** till date.

4. Here is the Spark code snippet to find the most preferred destination for all users:

```
// import required Spark packages

import org.apache.spark.sql.SparkSession

import org.apache.spark.sql.types.{IntegerType, StringType}
object Assignment18_1 {
  def main(args: Array[String]): Unit = {
```

```
  val spark = SparkSession                    // create a SparkSession object that can be used to
    .builder()                                // create various contexts of Spark such as sqlContext
    .config("spark.sql.warehouse.dir", "file:///c:/tmp/spark-warehouse")
    .master("local[*]")
    .getOrCreate()
  val sqlContext = spark.sqlContext                          // initialize sqlContext
  val input_df = sqlContext.read.csv("E:\\Acadgild\\Session 18\\S18_Datasets\\S18_ Dataset_
Holidays.txt")                                               // load input data file – holidays.txt
  val holidaysDF = input_df.select (                         // define schema for input data loaded
    input_df("_c0").cast(IntegerType).as("user_id"),        //assign column names to the data frame
    input_df("_c1").cast(StringType).as("src"),
    input_df("_c2").cast(StringType).as("dest"),
    input_df("_c3").cast(StringType).as("travel_mode"),
    input_df("_c4").cast(IntegerType).as("distance"),
    input_df("_c5").cast(IntegerType).as("year_of_travel"))
  holidaysDF.createOrReplaceTempView("holidays")        // create a temporary view - holidays
  val result = sqlContext.sql("SELECT dest, COUNT(*) AS distribution " +
            "FROM holidays " +
            "GROUP BY dest " +                               // SQL query to find the most
            "ORDER BY distribution DESC").take(1)            // preferred destination for all users
   result.foreach(println)                                   // print the result
 }
}
```

Output:



We can conclude from the above output that **India (IND)** is the most preferred destination among all **9** users.

Spark code screenshot:

Assignment18_1.scala

```scala
object Assignment18_1 {
  def main(args: Array[String]): Unit = {
    val spark = SparkSession
      .builder()
      .config("spark.sql.warehouse.dir", "file:///c:/tmp/spark-warehouse")
      .master("local[*]")
      .getOrCreate()
    val sqlContext = spark.sqlContext
    val input_df = sqlContext.read.csv("E:\\Acadgild\\Session 18\\S18_Datasets\\S18_Dataset_Holidays.txt")
    val holidaysDF = input_df.select(
      input_df("_c0").cast(IntegerType).as("user_id"),
      input_df("_c1").cast(StringType).as("src"),
      input_df("_c2").cast(StringType).as("dest"),
      input_df("_c3").cast(StringType).as("travel_mode"),
      input_df("_c4").cast(IntegerType).as("distance"),
      input_df("_c5").cast(IntegerType).as("year_of_travel"))
    holidaysDF.createOrReplaceTempView("holidays")
    val result = sqlContext.sql("SELECT dest, COUNT(*) AS distribution " +
                  "FROM holidays " +
                  "GROUP BY dest " +
                  "ORDER BY distribution DESC").take(1)
    result.foreach(println)
  }
}
```