ASSIGNMENT 18.2

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) Which route is generating the most revenue per year?

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

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

Solution:

1. Here is the Spark code snippet to find the route generating the most revenue per year:

```
// import required Spark packages

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


object Assignment18_2 {
  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_df1 = sqlContext.read.csv ("E:\\Acadgild\\Session 18\\S18_Datasets \\S18_Dataset
_Transport.txt")                                               // load input data file – transport.txt
    val transportDF = input_df1.select(                        // define schema for input data loaded
      input_df1("_c0").cast(StringType).as("travel_mode"),     //assign column names to the data frame
      input_df1("_c1").cast(IntegerType).as("cost_per_unit"))
    transportDF.createOrReplaceTempView("transport")           // create a temporary view - transport

    val input_df2 = sqlContext.read.csv("E:\\Acadgild\\Session 18\\S18_Datasets\\S18_Dataset_
Holidays.txt")                                                 // load input data file – holidays.txt
    val holidaysDF = input_df2.select(                         // define schema for input data loaded
      input_df2("_c0").cast(IntegerType).as("user_id"),        //assign column names to the data frame
      input_df2("_c1").cast(StringType).as("src"),
      input_df2("_c2").cast(StringType).as("dest"),
      input_df2("_c3").cast(StringType).as("travel_mode"),
      input_df2("_c4").cast(IntegerType).as("distance"),
      input_df2("_c5").cast(IntegerType).as("year_of_travel"))
    holidaysDF.createOrReplaceTempView("holidays")             // create a temporary view - holidays

// SQL query to find the route generating the most revenue per year
    val result = sqlContext.sql("SELECT y.src, y.dest, y.year_of_travel " +
                          "FROM transport x, holidays y " +
                          "WHERE x.travel_mode = y.travel_mode " +
                          "GROUP BY y.src , y.dest ,y.year_of_travel " +
                          "ORDER BY SUM(x.cost_per_unit) DESC").take(6)
    result.foreach(println)                                    // print the result
  }
}
```

Assignment18_1.scala | Assignment18_2.scala

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

object Assignment18_2 {
  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_df1 = sqlContext.read.csv("E:\\Acadgild\\Session 18\\S18_Datasets\\S18_Dataset_Transport.txt")
    val transportDF = input_df1.select(
      input_df1("_c0").cast(StringType).as("travel_mode"),
      input_df1("_c1").cast(IntegerType).as("cost_per_unit"))
    transportDF.createOrReplaceTempView("transport")

    val input_df2 = sqlContext.read.csv("E:\\Acadgild\\Session 18\\S18_Datasets\\S18_Dataset_Holidays.txt")
    val holidaysDF = input_df2.select(
      input_df2("_c0").cast(IntegerType).as("user_id"),
      input_df2("_c1").cast(StringType).as("src"),
      input_df2("_c2").cast(StringType).as("dest"),
      input_df2("_c3").cast(StringType).as("travel_mode"),
      input_df2("_c4").cast(IntegerType).as("distance"),
      input_df2("_c5").cast(IntegerType).as("year_of_travel"))
    holidaysDF.createOrReplaceTempView("holidays")

    val result = sqlContext.sql("SELECT y.src, y.dest, y.year_of_travel " +
                                "FROM transport x, holidays y " +
                                "WHERE x.travel_mode = y.travel_mode " +
                                "GROUP BY y.src , y.dest ,y.year_of_travel " +
                                "ORDER BY SUM(x.cost_per_unit) DESC").take(6)
    result.foreach(println)
  }
```

Output:

```
[RUS,IND,1992]
[CHN,RUS,1992]
[AUS,CHN,1993]
[IND,AUS,1991]
[CHN,IND,1993]
[IND,RUS,1991]
```

2. Here is the Spark code to find the total amount spent by every user on air-travel per year:

// import required Spark packages

import org.apache.spark.sql.SparkSession

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

```
object Assignment18_2 {
  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_df1 = sqlContext.read.csv ("E:\\Acadgild\\Session 18\\S18_Datasets \\S18_Dataset
_Transport.txt")                                              // load input data file – transport.txt
    val transportDF = input_df1.select(                       // define schema for input data loaded
      input_df1("_c0").cast(StringType).as("travel_mode"),    //assign column names to the data frame
      input_df1("_c1").cast(IntegerType).as("cost_per_unit"))
    transportDF.createOrReplaceTempView("transport")          // create a temporary view - transport

    val input_df2 = sqlContext.read.csv("E:\\Acadgild\\Session 18\\S18_Datasets\\S18_Dataset_
Holidays.txt")                                                // load input data file – holidays.txt
    val holidaysDF = input_df2.select(                        // define schema for input data loaded
      input_df2("_c0").cast(IntegerType).as("user_id"),       //assign column names to the data frame
      input_df2("_c1").cast(StringType).as("src"),
      input_df2("_c2").cast(StringType).as("dest"),
      input_df2("_c3").cast(StringType).as("travel_mode"),
      input_df2("_c4").cast(IntegerType).as("distance"),
      input_df2("_c5").cast(IntegerType).as("year_of_travel"))
    holidaysDF.createOrReplaceTempView("holidays")            // create a temporary view - holidays

val input_df3 = sqlContext.read.csv("E:\\Acadgild\\Session 18\\S18_Datasets\\S18_Dataset_
User_details.txt")                                            // load input data file – user_details.txt
    val usersDF = input_df3.select(                           // define schema for input data loaded
      input_df3("_c0").cast(IntegerType).as("user_id"),       //assign column names to the data frame
      input_df3("_c1").cast(StringType).as("name"),
      input_df3("_c2").cast(IntegerType).as("age"))
```

usersDF.createOrReplaceTempView("users") // create a temporary view - users

// SQL query to find the total amount spent by every user on air-travel per year

sqlContext.sql("SELECT z.user_id, z.name, y.year_of_travel, SUM(x.cost_per_unit) as total_amount " +

"FROM transport x, holidays y, users z " +

"WHERE x.travel_mode = y.travel_mode AND x.travel_mode = 'airplane' and y.user_id = z.user_id " +

"GROUP BY z.user_id, z.name, y.year_of_travel " +

"ORDER by z.user_id, z.name, y.year_of_travel").show() // print the result

}

}

Assignment18_1.scala | Assignment18_2.scala

```scala
    val sqlContext = spark.sqlContext
    val input_df1 = sqlContext.read.csv("E:\\Acadgild\\Session 18\\S18_Datasets\\S18_Dataset_Transport.txt")
    val transportDF = input_df1.select(
      input_df1("_c0").cast(StringType).as("travel_mode"),
      input_df1("_c1").cast(IntegerType).as("cost_per_unit"))
    transportDF.createOrReplaceTempView("transport")

    val input_df2 = sqlContext.read.csv("E:\\Acadgild\\Session 18\\S18_Datasets\\S18_Dataset_Holidays.txt")
    val holidaysDF = input_df2.select(
      input_df2("_c0").cast(IntegerType).as("user_id"),
      input_df2("_c1").cast(StringType).as("src"),
      input_df2("_c2").cast(StringType).as("dest"),
      input_df2("_c3").cast(StringType).as("travel_mode"),
      input_df2("_c4").cast(IntegerType).as("distance"),
      input_df2("_c5").cast(IntegerType).as("year_of_travel"))
    holidaysDF.createOrReplaceTempView("holidays")

    val input_df3 = sqlContext.read.csv("E:\\Acadgild\\Session 18\\S18_Datasets\\S18_Dataset_User_details.txt")
    val usersDF = input_df3.select(
      input_df3("_c0").cast(IntegerType).as("user_id"),
      input_df3("_c1").cast(StringType).as("name"),
      input_df3("_c2").cast(IntegerType).as("age"))
    usersDF.createOrReplaceTempView("users")

    sqlContext.sql("SELECT z.user_id, z.name, y.year_of_travel, SUM(x.cost_per_unit) as total_amount " +
                   "FROM transport x, holidays y ,users z " +
                   "WHERE x.travel_mode = y.travel_mode AND x.travel_mode = 'airplane' and y.user_id = z.user_id " +
                   "GROUP BY z.user_id, z.name, y.year_of_travel " +
                   "ORDER by z.user_id, z.name, y.year_of_travel").show()
  }
}
```

Output:

```
+-------+------+--------------+------------+
|user_id|  name|year_of_travel|total_amount|
+-------+------+--------------+------------+
|      1|  mark|          1990|         170|
|      1|  mark|          1993|         510|
|      2|  john|          1991|         340|
|      2|  john|          1993|         170|
|      3|  luke|          1991|         170|
|      3|  luke|          1992|         170|
|      3|  luke|          1993|         170|
|      4|  lisa|          1990|         340|
|      4|  lisa|          1991|         170|
|      5|  mark|          1991|         170|
|      5|  mark|          1992|         340|
|      5|  mark|          1994|         170|
|      6| peter|          1991|         340|
|      6| peter|          1993|         170|
|      7| james|          1990|         510|
|      8|andrew|          1990|         170|
|      8|andrew|          1991|         170|
|      8|andrew|          1992|         170|
|      9|thomas|          1991|         170|
|      9|thomas|          1992|         340|
+-------+------+--------------+------------+
only showing top 20 rows
```

3. Here is the Spark code to find the age group that is travelling the most every year:

```
// import required Spark packages

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


object Assignment18_2 {
  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_df2 = sqlContext.read.csv("E:\\Acadgild\\Session 18\\S18_Datasets\\S18_Dataset_
Holidays.txt")                                                 // load input data file – holidays.txt
    val holidaysDF = input_df2.select(                         // define schema for input data loaded
      input_df2("_c0").cast(IntegerType).as("user_id"),        //assign column names to the data frame
      input_df2("_c1").cast(StringType).as("src"),
      input_df2("_c2").cast(StringType).as("dest"),
      input_df2("_c3").cast(StringType).as("travel_mode"),
      input_df2("_c4").cast(IntegerType).as("distance"),
      input_df2("_c5").cast(IntegerType).as("year_of_travel"))
    holidaysDF.createOrReplaceTempView("holidays")             // create a temporary view - holidays

val input_df3 = sqlContext.read.csv("E:\\Acadgild\\Session 18\\S18_Datasets\\S18_Dataset_
User_details.txt")                                             // load input data file – user_details.txt
    val usersDF = input_df3.select(                            // define schema for input data loaded
      input_df3("_c0").cast(IntegerType).as("user_id"),        //assign column names to the data frame
      input_df3("_c1").cast(StringType).as("name"),
      input_df3("_c2").cast(IntegerType).as("age"))
    usersDF.createOrReplaceTempView("users")                   // create a temporary view – users

// SQL query to find the age group that is travelling the most every year
     val result = sqlContext.sql("SELECT age " +
                        "FROM users x, holidays y " +
                        "WHERE x.user_id = y.user_id " +
                        "GROUP BY age " +
                        "ORDER BY COUNT(age) DESC").take(1)
    result.foreach(println)                                    // print the result
  }
}
```

Assignment18_1.scala | Assignment18_2.scala

```scala
object Assignment18_2 {
  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_df2 = sqlContext.read.csv("E:\\Acadgild\\Session 18\\S18_Datasets\\S18_Dataset_Holidays.txt")
    val holidaysDF = input_df2.select(
      input_df2("_c0").cast(IntegerType).as("user_id"),
      input_df2("_c1").cast(StringType).as("src"),
      input_df2("_c2").cast(StringType).as("dest"),
      input_df2("_c3").cast(StringType).as("travel_mode"),
      input_df2("_c4").cast(IntegerType).as("distance"),
      input_df2("_c5").cast(IntegerType).as("year_of_travel"))
    holidaysDF.createOrReplaceTempView("holidays")

    val input_df3 = sqlContext.read.csv("E:\\Acadgild\\Session 18\\S18_Datasets\\S18_Dataset_User_details.txt")
    val usersDF = input_df3.select(
      input_df3("_c0").cast(IntegerType).as("user_id"),
      input_df3("_c1").cast(StringType).as("name"),
      input_df3("_c2").cast(IntegerType).as("age"))
    usersDF.createOrReplaceTempView("users")

    val result = sqlContext.sql("SELECT age " +
                                "FROM users x, holidays y " +
                                "WHERE x.user_id = y.user_id " +
                                "GROUP BY age " +
                                "ORDER BY COUNT(age) DESC").take(1)
    result.foreach(println)
  }
}
```

Output:

Run Assignment18_2

```
18/02/27 01:14:20 INFO Executor: Running task 184.0 in stage 4.0 (TID 203)
18/02/27 01:14:20 INFO ShuffleBlockFetcherIterator: Getting 1 non-empty blocks out of 1 blocks
18/02/27 01:14:20 INFO ShuffleBlockFetcherIterator: Started 0 remote fetches in 0 ms
18/02/27 01:14:20 INFO Executor: Finished task 184.0 in stage 4.0 (TID 203). 4554 bytes result sent to driver
18/02/27 01:14:20 INFO TaskSetManager: Finished task 184.0 in stage 4.0 (TID 203) in 16 ms on localhost (199/200)
18/02/27 01:14:20 INFO Executor: Finished task 143.0 in stage 4.0 (TID 202). 4554 bytes result sent to driver
18/02/27 01:14:20 INFO TaskSetManager: Finished task 143.0 in stage 4.0 (TID 202) in 31 ms on localhost (200/200)
18/02/27 01:14:20 INFO DAGScheduler: ResultStage 4 (take at Assignment18_2.scala:36) finished in 2.588 s
18/02/27 01:14:20 INFO TaskSchedulerImpl: Removed TaskSet 4.0, whose tasks have all completed, from pool
18/02/27 01:14:20 INFO DAGScheduler: Job 3 finished: take at Assignment18_2.scala:36, took 3.233046 s
18/02/27 01:14:20 INFO CodeGenerator: Code generated in 19.830896 ms
[15]
18/02/27 01:14:20 INFO SparkContext: Invoking stop() from shutdown hook
18/02/27 01:14:20 INFO SparkUI: Stopped Spark web UI at http://192.168.43.50:4040
18/02/27 01:14:20 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
18/02/27 01:14:20 INFO MemoryStore: MemoryStore cleared
18/02/27 01:14:20 INFO BlockManager: BlockManager stopped
```

Compilation completed successfully in 4s 187ms (2 minutes ago)

Conclusion: People who are **under the age of 20 (to be precise, people with age 15)** travels the most every year.