Session 18

Assignment 2

|  |  |
| --- | --- |
| **Prepared For:** | AcadGild |
|  |  |
| **Document Approval:** | **AcadGild** |
|  |  |
|  |  |
|  |  |
|  |  |
| **Project Title:** | Session 18 – Assignment 2 |
|  |  |
| **Prepared By:** | Duncan Burgess |
|  |  |
|  | dburgess@duncb.com |
|  |  |
| **Primary Engineer:** | Duncan Burgess |
|  |  |
| **Document Reference:** | **Session 18 – Assignment 2** |
|  |  |
| **Start Date:** | 18/10/2017 |
|  |  |
|  |  |



# 

# Contents

[Contents 2](#_Toc496185827)

[Change History 3](#_Toc496185828)

[1. Problem Statement 4](#_Toc496185829)

[2. Datasets 4](#_Toc496185830)

[3. Solution 5](#_Toc496185831)

[4. Results 8](#_Toc496185832)

# Change History

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Document Revision** | **Date** | **Authored By** | **Authorised By** | **Sections Affected** | **Reason for Change** |
| Rev 01 | 19/10/2017 | Duncan Burgess |  | All | Initial release. |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

# Problem Statement

* Which route is generating the most revenue per year
* What is the total amount spent by every user on air-travel per year
* Considering age groups of < 20 , 20-35, 35 > ,Which age group is travelling the most every year.

Using the datasets below:

# Datasets

**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\_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

**S18\_Dataset**\_Transport.txt

airplane,170

car,140

train,120

ship,200

# Solution

I have deliberately commented and printed out all steps for enhanced learning

**Code written**

**package** com.duncb.spark

**import** org.apache.spark.\_

**import** org.apache.spark.SparkContext.\_

**import** org.apache.log4j.\_

**import** scala.math.min

//\*\* Compute the the following

**object** holiday2 {

/\*\* A function that splits a line of input into (userID,src,dest,travelMode,distance,yearOfTravel) tuples. \*/

**def** travelInfo(line: *String*) = {

// Split by commas

**val** fields = line.split(",")

**val** userID = fields(0).toInt

**val** src = fields(1).toString

**val** dest = fields (2).toString

**val** travelMode = fields (3).toString

**val** distance = fields (4).toInt

**val** yearOfTravel =fields(5).toInt

// Create a tuple that is our result.

(userID,src,dest,travelMode,distance,yearOfTravel)

}

/\*\* A function that splits a line of input into (userID,name and age) tuples. \*/

**def** userInfo(line: *String*) = {

// Split by commas

**val** fields = line.split(",")

**val** userID = fields(0).toInt

**val** userName = fields(1).toString

**val** age = fields (2).toInt

// Create a tuple that is our result.

(userID,userName,age)

}

/\*\* A function that splits a line of input into (transport,cost) tuples. \*/

**def** transportInfo(line: *String*) = {

// Split by commas

**val** fields = line.split(",")

**val** transport = fields(0).toString

**val** cost = fields (1).toInt

// Create a tuple that is our result.

(transport,cost)

}

**def** main(args: Array[*String*]) {

// Set the log level to only print errors

Logger.getLogger("org").setLevel(Level.ERROR)

// Create a SparkContext using every core of the local machine

**val** sc = **new** SparkContext("local[1]", "holidayData")

// Load each line of the source data into an holidaysRDD

**val** linesH = sc.textFile("file:///N:/Datasets/S18\_Dataset\_Holidays.txt")

**val** holidaysRDD = linesH.map(travelInfo)

// Load each line of the source data into a usersRDD

**val** linesU = sc.textFile("file:///N:/Datasets/S18\_Dataset\_User\_Details.txt")

**val** usersRDD = linesU.map(userInfo)

// Load each line of the source data into a transportRDD

**val** linesT = sc.textFile("file:///N:/Datasets/S18\_Dataset\_Transport.txt")

**val** transportRDD = linesT.map(transportInfo)

//-----------------------------------------------------------------------------------------------------

println(" Assignment 18.2 problem 1")

**val** rc1 = holidaysRDD.map(x=>(x.\_4,(x.\_2,x.\_3)))

rc1.foreach (println)

println ("-----------------------------------------------------")

**val** rc2 = rc1.join(transportRDD).map(x=>(x.\_2 .\_1 ->x.\_1) -> x.\_2.\_2)

rc2.foreach (println)

println ("-----------------------------------------------------")

**val** rc3 = rc2.groupByKey().map(x=> x.\_1 -> x.\_2.sum)

rc3.foreach (println)

println ("-----------------------------------------------------")

**val** rc4 = rc3.sortBy( x=> -x.\_2).first()

println (" The Route yielding the greatest revenue is " + rc4)

println ("-----------------------------------------------------")

println(" Assignment 18.2 problem 2")

**val** soa1= holidaysRDD.map(x=>(x.\_4,(x.\_1 -> x.\_6)))

soa1.foreach (println)

println ("-----------------------------------------------------")

**val** soa2=soa1.join(transportRDD).map(x=>(x.\_2.\_1,x.\_2 .\_2))

soa2.foreach (println)

println ("-----------------------------------------------------")

**val** soa3 = soa2.groupByKey().map(x=>(x.\_1,x.\_2.sum))

**val** soa4 =soa3.sortByKey()

println("Amount spent by user per year is ")

soa4.foreach(println)

println ("-----------------------------------------------------")

println(" Assignment 18.2 problem 3")

**val** ag1=usersRDD.map (x=> x.\_1 ->{

**if** (x.\_3 < 20)

"<20"

**else** **if** (x.\_3 > 35)

">35"

**else** "20-35"

})

ag1.foreach(println)

println ("-----------------------------------------------------")

**val** ag2 = ag1.join(holidaysRDD.map(x=> x.\_1 -> (x.\_6,1)))

ag2.foreach (println)

println ("-----------------------------------------------------")

**val** ag3 = ag2.map(x => (x.\_2.\_1 -> x.\_2.\_2.\_1 , x.\_2.\_2.\_2))

**val** ag4 = ag3.groupByKey.map(x => (x.\_1.\_2,x.\_1.\_1,x.\_2.sum))

ag4.foreach (println)

println ("-----------------------------------------------------")

//To find most populat in each year created manual filters. Tried loops and reduce (revisit)

**val** y90 =ag4.filter(x => x.\_1 ==1990).sortBy(x=> -x.\_3).first()

println( "Most popular group in " + y90)

**val** y91 =ag4.filter(x => x.\_1 ==1991).sortBy(x=> -x.\_3).first()

println( "Most popular group in " + y91)

**val** y92 =ag4.filter(x => x.\_1 ==1992).sortBy(x=> -x.\_3).first()

println( "Most popular group in " + y92)

**val** y93 =ag4.filter(x => x.\_1 ==1993).sortBy(x=> -x.\_3).first()

println( "Most popular group in " + y93)

**val** y94 =ag4.filter(x => x.\_1 ==1994).sortBy(x=> -x.\_3).first()

println( "Most popular group in " + y94)

}

}

# Results

Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties

Assignment 18.2 problem 1

(airplane,(CHN,IND))

(airplane,(IND,CHN))

(airplane,(IND,CHN))

(airplane,(RUS,IND))

(airplane,(CHN,RUS))

(airplane,(AUS,PAK))

(airplane,(RUS,AUS))

(airplane,(IND,RUS))

(airplane,(CHN,RUS))

(airplane,(AUS,CHN))

(airplane,(AUS,CHN))

(airplane,(CHN,IND))

(airplane,(CHN,IND))

(airplane,(IND,AUS))

(airplane,(AUS,IND))

(airplane,(RUS,CHN))

(airplane,(CHN,RUS))

(airplane,(AUS,CHN))

(airplane,(IND,AUS))

(airplane,(RUS,CHN))

(airplane,(PAK,IND))

(airplane,(IND,RUS))

(airplane,(CHN,PAK))

(airplane,(CHN,PAK))

(airplane,(IND,PAK))

(airplane,(PAK,RUS))

(airplane,(CHN,IND))

(airplane,(RUS,IND))

(airplane,(RUS,IND))

(airplane,(CHN,AUS))

(airplane,(PAK,AUS))

(airplane,(CHN,PAK))

-----------------------------------------------------

(((CHN,IND),airplane),170)

(((IND,CHN),airplane),170)

(((IND,CHN),airplane),170)

(((RUS,IND),airplane),170)

(((CHN,RUS),airplane),170)

(((AUS,PAK),airplane),170)

(((RUS,AUS),airplane),170)

(((IND,RUS),airplane),170)

(((CHN,RUS),airplane),170)

(((AUS,CHN),airplane),170)

(((AUS,CHN),airplane),170)

(((CHN,IND),airplane),170)

(((CHN,IND),airplane),170)

(((IND,AUS),airplane),170)

(((AUS,IND),airplane),170)

(((RUS,CHN),airplane),170)

(((CHN,RUS),airplane),170)

(((AUS,CHN),airplane),170)

(((IND,AUS),airplane),170)

(((RUS,CHN),airplane),170)

(((PAK,IND),airplane),170)

(((IND,RUS),airplane),170)

(((CHN,PAK),airplane),170)

(((CHN,PAK),airplane),170)

(((IND,PAK),airplane),170)

(((PAK,RUS),airplane),170)

(((CHN,IND),airplane),170)

(((RUS,IND),airplane),170)

(((RUS,IND),airplane),170)

(((CHN,AUS),airplane),170)

(((PAK,AUS),airplane),170)

(((CHN,PAK),airplane),170)

-----------------------------------------------------

(((RUS,AUS),airplane),170)

(((IND,CHN),airplane),340)

(((IND,AUS),airplane),340)

(((PAK,AUS),airplane),170)

(((CHN,IND),airplane),680)

(((RUS,IND),airplane),510)

(((RUS,CHN),airplane),340)

(((CHN,RUS),airplane),510)

(((PAK,RUS),airplane),170)

(((AUS,PAK),airplane),170)

(((AUS,CHN),airplane),510)

(((PAK,IND),airplane),170)

(((IND,PAK),airplane),170)

(((CHN,PAK),airplane),510)

(((AUS,IND),airplane),170)

(((CHN,AUS),airplane),170)

(((IND,RUS),airplane),340)

-----------------------------------------------------

The Route yielding the greatest revenue is (((CHN,IND),airplane),680)

-----------------------------------------------------

Assignment 18.2 problem 2

(airplane,(1,1990))

(airplane,(2,1991))

(airplane,(3,1992))

(airplane,(4,1990))

(airplane,(5,1992))

(airplane,(6,1991))

(airplane,(7,1990))

(airplane,(8,1991))

(airplane,(9,1992))

(airplane,(10,1993))

(airplane,(1,1993))

(airplane,(2,1993))

(airplane,(3,1993))

(airplane,(4,1991))

(airplane,(5,1992))

(airplane,(6,1993))

(airplane,(7,1990))

(airplane,(8,1990))

(airplane,(9,1991))

(airplane,(10,1992))

(airplane,(1,1993))

(airplane,(2,1991))

(airplane,(3,1991))

(airplane,(4,1990))

(airplane,(5,1991))

(airplane,(6,1991))

(airplane,(7,1990))

(airplane,(8,1992))

(airplane,(9,1992))

(airplane,(10,1990))

(airplane,(1,1993))

(airplane,(5,1994))

-----------------------------------------------------

((1,1990),170)

((2,1991),170)

((3,1992),170)

((4,1990),170)

((5,1992),170)

((6,1991),170)

((7,1990),170)

((8,1991),170)

((9,1992),170)

((10,1993),170)

((1,1993),170)

((2,1993),170)

((3,1993),170)

((4,1991),170)

((5,1992),170)

((6,1993),170)

((7,1990),170)

((8,1990),170)

((9,1991),170)

((10,1992),170)

((1,1993),170)

((2,1991),170)

((3,1991),170)

((4,1990),170)

((5,1991),170)

((6,1991),170)

((7,1990),170)

((8,1992),170)

((9,1992),170)

((10,1990),170)

((1,1993),170)

((5,1994),170)

-----------------------------------------------------

Amount spent by user per year is

((1,1990),170)

((1,1993),510)

((2,1991),340)

((2,1993),170)

((3,1991),170)

((3,1992),170)

((3,1993),170)

((4,1990),340)

((4,1991),170)

((5,1991),170)

((5,1992),340)

((5,1994),170)

((6,1991),340)

((6,1993),170)

((7,1990),510)

((8,1990),170)

((8,1991),170)

((8,1992),170)

((9,1991),170)

((9,1992),340)

((10,1990),170)

((10,1992),170)

((10,1993),170)

-----------------------------------------------------

Assignment 18.2 problem 3

(1,<20)

(2,<20)

(3,<20)

(4,20-35)

(5,20-35)

(6,20-35)

(7,20-35)

(8,>35)

(9,>35)

(10,>35)

-----------------------------------------------------

(4,(20-35,(1990,1)))

(4,(20-35,(1991,1)))

(4,(20-35,(1990,1)))

(1,(<20,(1990,1)))

(1,(<20,(1993,1)))

(1,(<20,(1993,1)))

(1,(<20,(1993,1)))

(6,(20-35,(1991,1)))

(6,(20-35,(1993,1)))

(6,(20-35,(1991,1)))

(3,(<20,(1992,1)))

(3,(<20,(1993,1)))

(3,(<20,(1991,1)))

(7,(20-35,(1990,1)))

(7,(20-35,(1990,1)))

(7,(20-35,(1990,1)))

(9,(>35,(1992,1)))

(9,(>35,(1991,1)))

(9,(>35,(1992,1)))

(8,(>35,(1991,1)))

(8,(>35,(1990,1)))

(8,(>35,(1992,1)))

(10,(>35,(1993,1)))

(10,(>35,(1992,1)))

(10,(>35,(1990,1)))

(5,(20-35,(1992,1)))

(5,(20-35,(1992,1)))

(5,(20-35,(1991,1)))

(5,(20-35,(1994,1)))

(2,(<20,(1991,1)))

(2,(<20,(1993,1)))

(2,(<20,(1991,1)))

-----------------------------------------------------

(1990,20-35,5)

(1991,>35,2)

(1991,<20,3)

(1991,20-35,4)

(1994,20-35,1)

(1990,>35,2)

(1993,20-35,1)

(1993,<20,5)

(1992,20-35,2)

(1992,<20,1)

(1992,>35,4)

(1990,<20,1)

(1993,>35,1)

-----------------------------------------------------

Most popular group in (1990,20-35,5)

Most popular group in (1991,20-35,4)

Most popular group in (1992,>35,4)

Most popular group in (1993,<20,5)

Most popular group in (1994,20-35,1