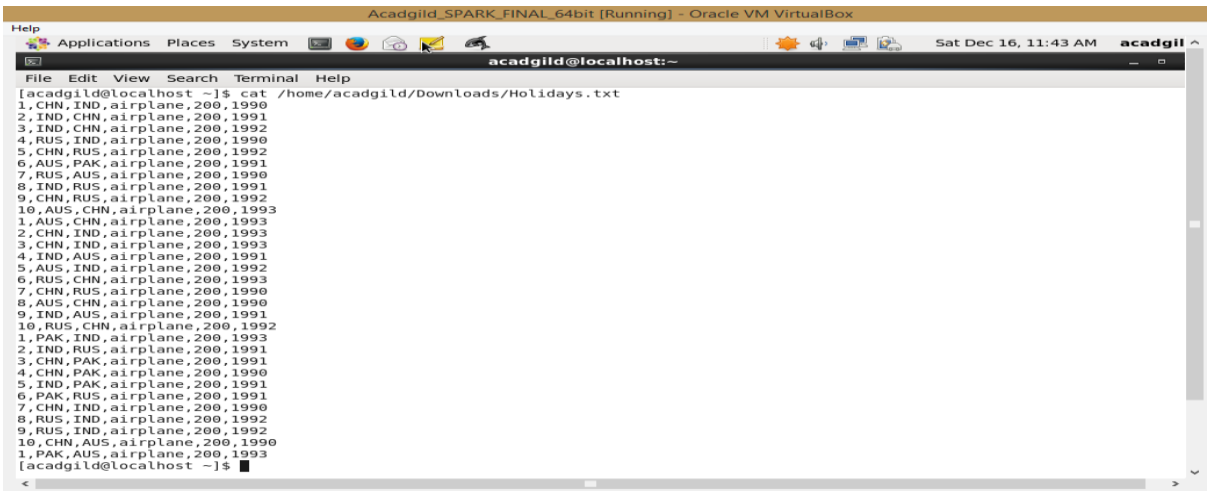
Problem Statement

- What is the distribution of the total number of air-travelers per year
- What is the total air distance covered by each user per year
- Which user has travelled the largest distance till date
- What is the most preferred destination for all users.

Dataset

https://drive.google.com/drive/folders/0B_P3pWagdIrrVThBaUdVSUtzbms

Dataset-Holidays:
The dataset is of holiday details of travelers with columns: **user_id**, **source**, **destination**, **travel_mode**, **distance**, **year_of_travel**:



Dataset-Transport:
The dataset is of transport details with columns: **travel_mode**, **cost_per_unit**:

```
[acadgild@localhost ~]$ cat /home/acadgild/Downloads/Transport.txt
airplane,170
car,140
train,120
ship,200[acadgild@localhost ~]$
```

Dataset-User_details:
The dataset is of user details of travelers with columns: **user_id**, **name,age**:

```
ship,200[acadgild@localhost ~]$ cat /home/acadgild/Downloads/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[acadgild@localhost ~]$
```

Intialization Spark-Shell:

```
Acadgild_SPARK_FINAL_64bit [Running] - Oracle VM VirtualBox
Help
Applications  Places  System                                   Sat Dec 16, 12:49 PM   acadgil
acadgild@localhost:~
File  Edit  View  Search  Terminal  Help
[acadgild@localhost ~]$ spark-shell
log4j:WARN No appenders could be found for logger (org.apache.hadoop.metrics2.lib.MutableMetricsFactory).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
Using Spark's repl log4j profile: org/apache/spark/log4j-defaults-repl.properties
To adjust logging level use sc.setLogLevel("INFO")
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.0
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_65)
Type in expressions to have them evaluated.
Type :help for more information.
17/12/16 12:35:35 WARN Utils: Your hostname, localhost.localdomain resolves to a loopback address: 127.0.0.1; using 10.0.2.15
 instead (on interface eth6)
17/12/16 12:35:35 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
Spark context available as sc.
17/12/16 12:35:53 WARN Connection: BoneCP specified but not present in CLASSPATH (or one of dependencies)
17/12/16 12:35:56 WARN Connection: BoneCP specified but not present in CLASSPATH (or one of dependencies)
17/12/16 12:36:14 WARN ObjectStore: Version information not found in metastore. hive.metastore.schema.verification is not ena
bled so recording the schema version 1.2.0
17/12/16 12:36:15 WARN ObjectStore: Failed to get database default, returning NoSuchObjectException
17/12/16 12:36:25 WARN Connection: BoneCP specified but not present in CLASSPATH (or one of dependencies)
17/12/16 12:36:26 WARN Connection: BoneCP specified but not present in CLASSPATH (or one of dependencies)
SQL context available as sqlContext.
```

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

Creating tupleRdd Holidays

```
scala> var Holidays = sc.textFile("/home/acadgild/Downloads/Holidays.txt").map(x=>
     | {
     | val row = x.split(",")
     | (row.apply(0).toInt,row.apply(1),row.apply(2),row.apply(3),row.apply(4).toInt,row.apply(5).toInt)
     | })
Holidays: org.apache.spark.rdd.RDD[(Int, String, String, String, Int, Int)] = MapPartitionsRDD[2] at map at <console>:27

scala>
```

Displaying all data tupleRdd Holidays

```
scala> Holidays.foreach(println)
(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)

scala>
acadgild@localhost:~
```

Code:

Finding Total Number Of Air Travellers Per Year

```
scala> val Total_Air_Travellers =Holidays.map(x=>(x._1,x._6)).map(x=>x._2->1).groupByKey().map(x=>x._1->x._2.sum)
Total_Air_Travellers: org.apache.spark.rdd.RDD[(Int, Int)] = MapPartitionsRDD[6] at map at <console>:29
```

Output:

Displaying Total Number of Air Travellers per year

```
scala> Total_Air_Travellers.foreach(println)
(1994,1)
(1992,7)
(1990,8)
(1991,9)
(1993,7)

scala>
```

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

Code:

Finding Total air Distance covered by each user per year

```
scala> val Total_Air_Distance = Holidays.map(x=>(x._1->x._6->x._5)).groupByKey().map(x=>x._1->x._2.sum).sortByKey()
Total_Air_Distance: org.apache.spark.rdd.RDD[((Int, Int), Int)] = ShuffledRDD[10] at sortByKey at <console>:29
```

Output:

Displaying Total air Distance covered by each user per year

```
scala> Total_Air_Distance.foreach(println)
((1,1990),200)
((1,1993),600)
((2,1991),400)
((2,1993),200)
((3,1991),200)
((3,1992),200)
((3,1993),200)
((4,1990),400)
((4,1991),200)
((5,1991),200)
((5,1992),400)
((5,1994),200)
((6,1991),400)
((6,1993),200)
((7,1990),600)
((8,1990),200)
((8,1991),200)
((8,1992),200)
((9,1991),200)
((9,1992),400)
((10,1990),200)
((10,1992),200)
((10,1993),200)

scala>
```

- Which user has travelled the largest distance till date

Code:

Finding user who travelled largest distance till date

```
scala> val User_travelled_large_distance = Total_Air_Distance.map(x=>(x._1._1,x._2)).groupByKey.map(x=>(x._1,x._2.sum)).sortB
y(x => -x._2).take(2)
User_travelled_large_distance: Array[(Int, Int)] = Array((1,800), (5,800))
```

Output:

Displaying user who travelled largest distance till date

```
scala> User_travelled_large_distance.foreach(println)
(1,800)
(5,800)

scala>
```

- What is the most preferred destination for all users.

Code:

Finding the most preferred destination for all users

```
scala> val Most_prefered_Distination = Holidays.map(x => x._3 -> 1).groupByKey().map(x => x._1 -> x._2.sum)
Most_prefered_Distination: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[25] at map at <console>:29
```

Output:

Displaying the most preferred destination for all users

```
scala> Most_prefered_Distination.sortBy(x => -x._2).first()
res4: (String, Int) = (IND,9)
```