**ASSIGNMENT 6.7**

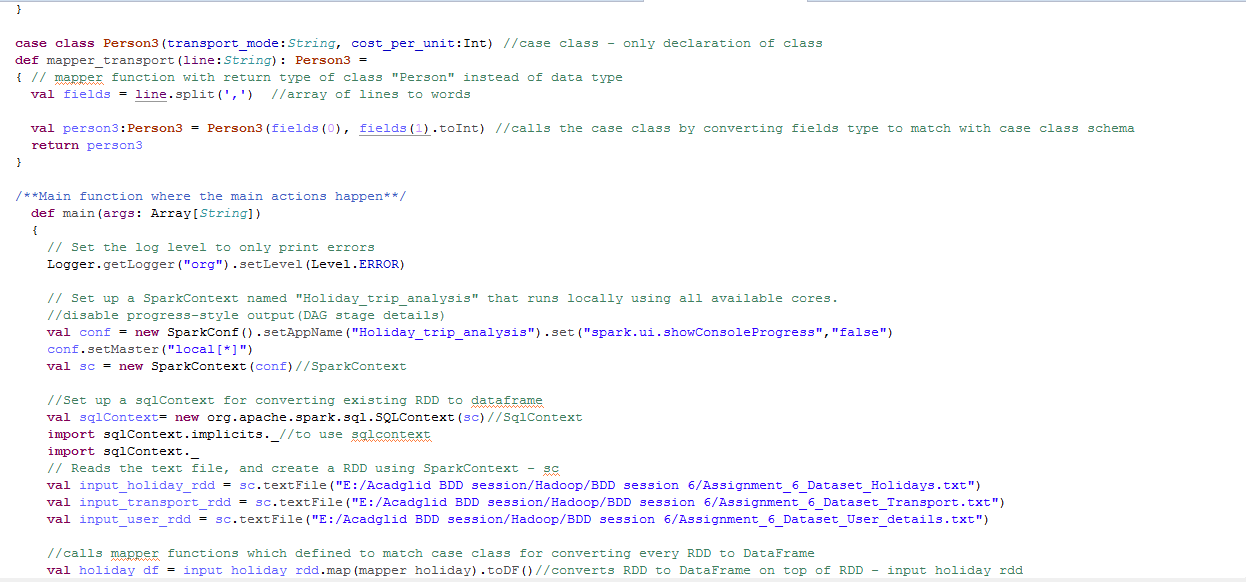
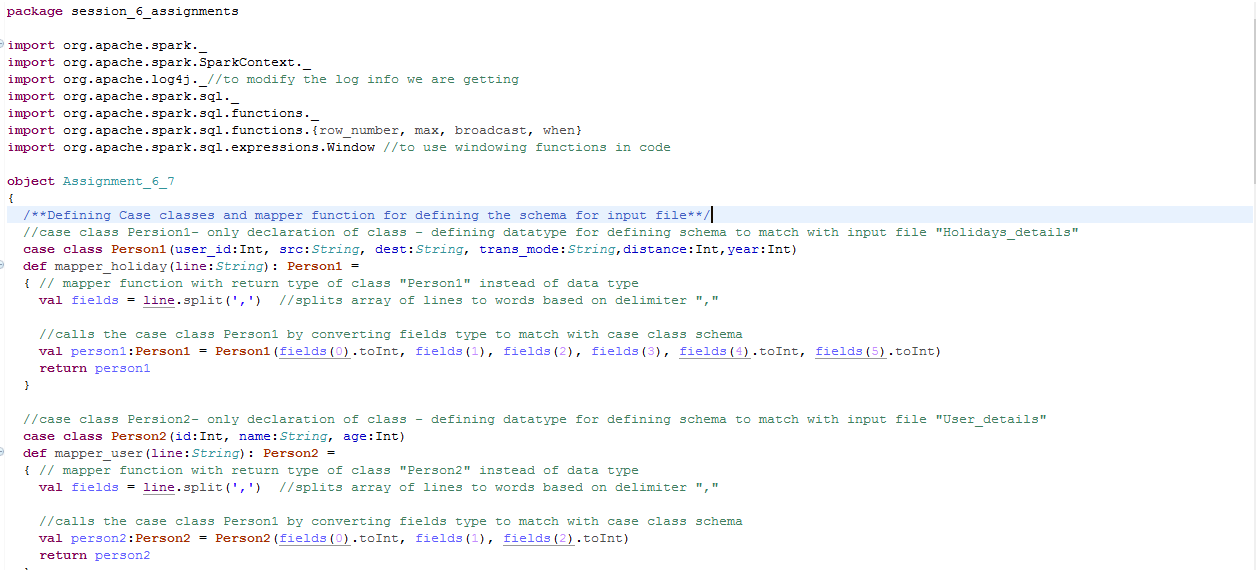
**Problem Statement:**

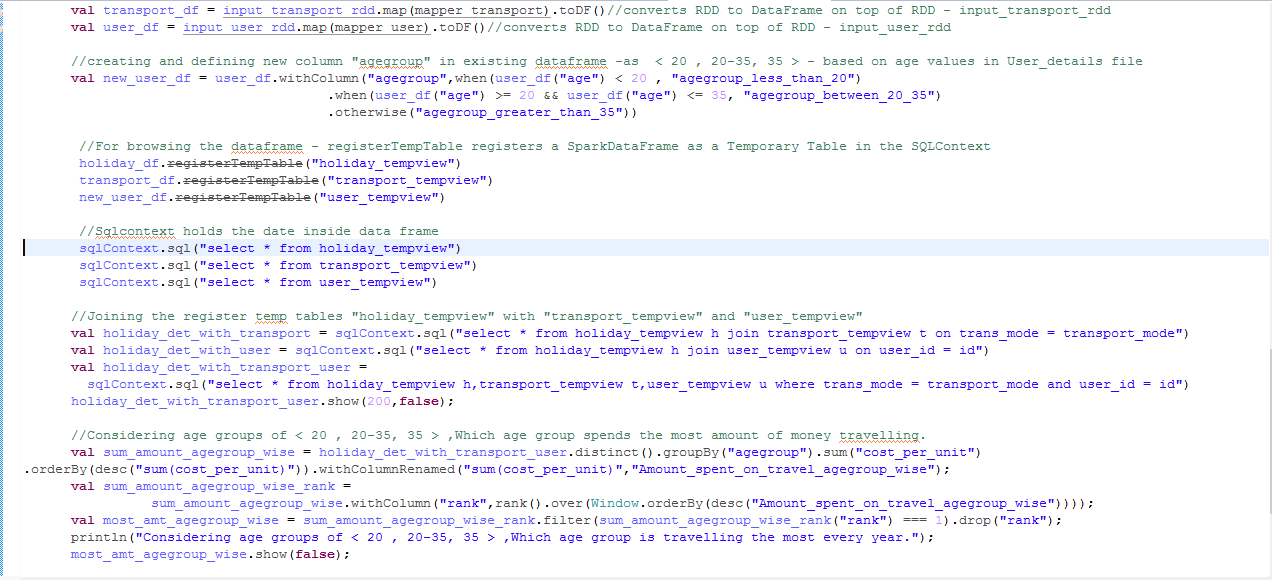
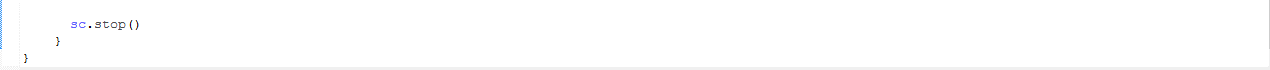
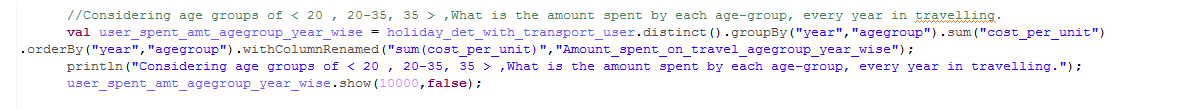
1) Considering age groups of < 20 , 20-35, 35 > ,Which age group spends the most

amount of money travelling.

2) What is the amount spent by each age-group, every year in travelling?

**Scala code:**



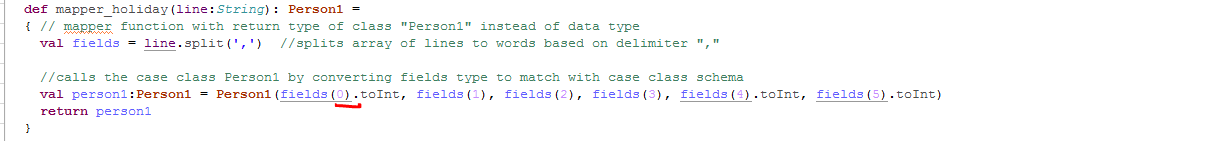
**Explanation:**

* Created 3 **case classes** called “Person1”,”Person2”,Person3” for defining the schema and name of the fields in input file

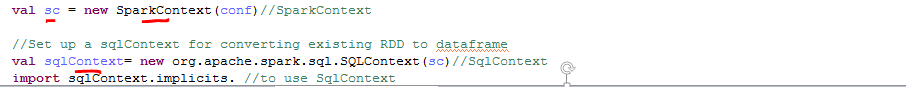
Person1(Holidays file),Person2(Transport file),Person3(User\_details file)



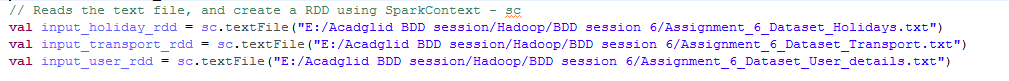
* Created 3 **mapper functions** for all 3 input files – which will convert the datatype of the fields to match with corresponding Person class schema and returns “Person” type



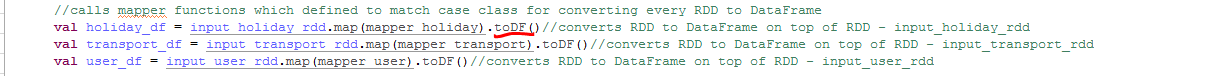
* Set the **log level to “Error”** to display only error messages - Logger.getLogger("org").setLevel(Level.ERROR) **and disabled progress-style** output(DAG stage details) - set("spark.ui.showConsoleProgress","false")
* Set up a **SparkContext** named – sc – for application "Holiday\_trip\_analysis" that runs locally using all available cores.
* Set up a **sqlContext** for converting existing RDD to **dataframe**



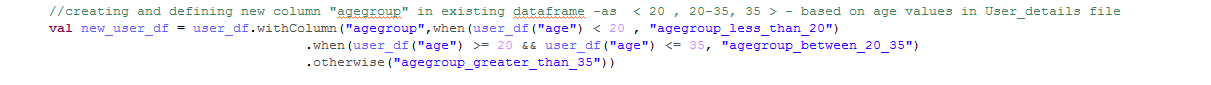
* Reads the text file, and create a RDD using SparkContext – sc



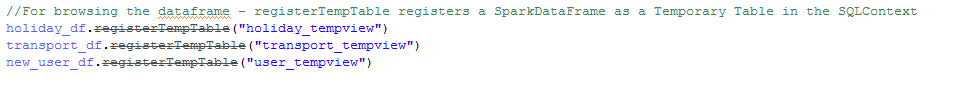
* Calls mapper functions which defined to match case class for converting every RDD to DataFrame
* Calls mapper functions by passing mapper function with RDD and retunrs Dataframe as toDF() is used



* For the problem statement, **created and defined a new column "agegroup" in existing dataframe** -as < 20 , 20-35, 35 > - based on age values in User\_details file using withColumn – which **is to create or add columns dynamically to the new data frame from existing DF**



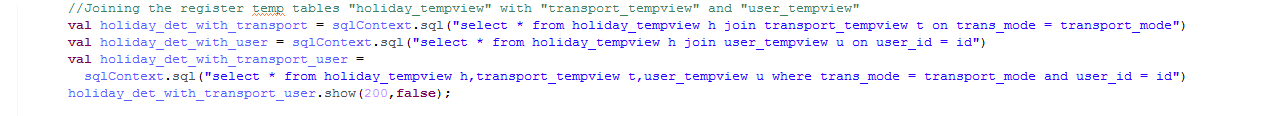
* For browsing the dataframe **- registerTempTable registers a SparkDataFrame** as a Temporary Table in the SQLContext and Sqlcontext holds the date of data frame



* Now as the data from RDD is converted to Dataframe,it can be queried with “SparkContext”.

**Joins:**

* Joined the registered temp table “"holiday\_tempview" with "transport\_tempview" based on transport\_mode to get the value of “cost\_per\_unit” from transport file with SqlContext
* Joined the registered temp table “"holiday\_tempview" with "user\_tempview" based on id to get the value of “age” from user file.
* Joined the registered temp table “"holiday\_tempview" with "transport\_tempview" and “user\_tempview” based on id and transport\_mode **(three table join**) to get the value of **“agegroup” from “user\_details”** file and **“cost\_per\_unit” from “transport”** file for each user\_id for every **“year” from “holidays” file**(holiday\_det\_with\_trasnport\_user).

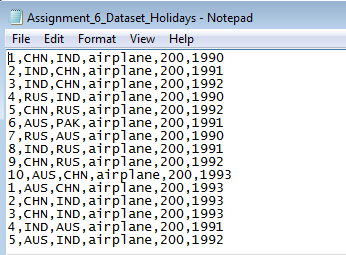


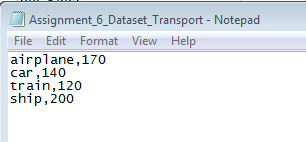
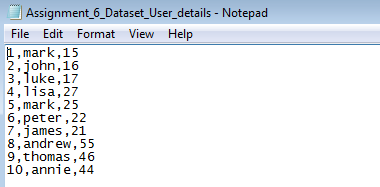
**Snapshot of input file:**

User: id, name, age – Assignment\_6\_Dataset\_User\_details

Transport: transport\_mode, cost\_per\_unit - Assignment\_6\_Dataset\_Transport

Holidays: id, src, dest, transport\_mode, distance, year - Assignment\_6\_Dataset\_Holidays

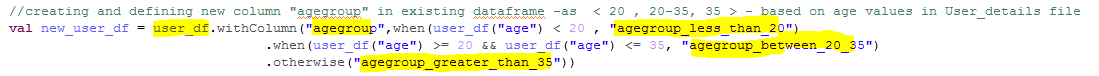


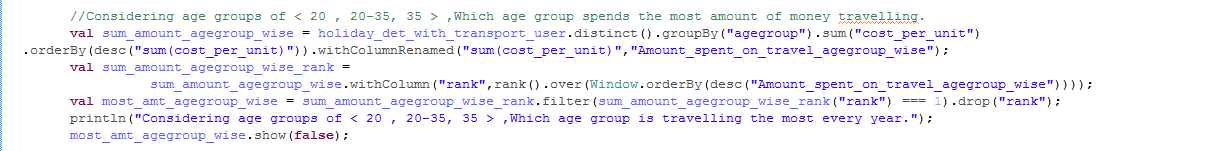
**1) Considering age groups of < 20 , 20-35, 35 > ,Which age group spends the most**

**amount of money travelling.**

Created a **new column age group “agegroup\_less\_than\_20”,” agegroup\_between\_20-35”,” agegroup\_greater\_than\_35>”**already based on age present in user\_details file

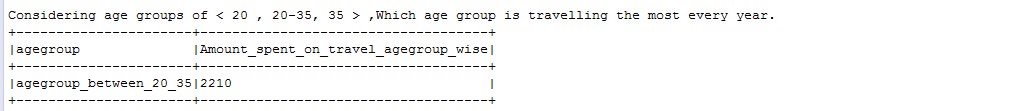


* Grouped data based **on agegroup and taken sum(cost\_per\_unit)** in every group – will hold the **sum(amount\_spent) according to each agegroup.**
* Using Windowing **function generated rank column based (ordered) based on sum(amount\_spent)** - will display agegroup and sum(amount\_spent) in all years according to agegroup ordered based on amount in descending
* And **taken first ro**w – to get **the age group which spends most of the money** in travelling– will hold the agegroup name which had spent most of the money in travelling across all year along with amount spent.



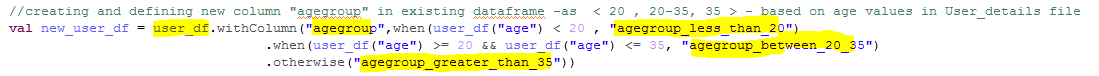
**Output:**

Displays the agegroup which had spent most of the money in travelling across all year along with amount spent by that agegroup.

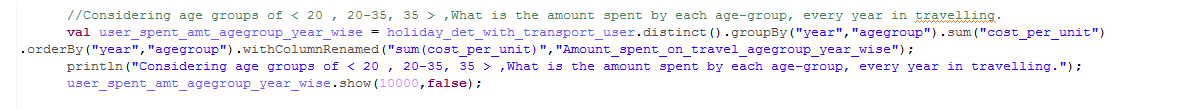


**2) What is the amount spent by each age-group, every year in travelling?**

Created a **new column age group “agegroup\_less\_than\_20”,” agegroup\_between\_20-35”,” agegroup\_greater\_than\_35>”**already based on age present in user\_details file



* Grouped data based on **agegroup,year and taken sum(cost\_per\_unit)** in every group – will hold the sum(amount\_spent) spent by each agegroup year wise.
* Displays the grouped data using show(false) – which will disable the truncation of column values in output console

**Output:**

Displays the amount spent by each age group, every year in travelling.