Assignment 18.1

Initial Steps:

Step1: Create a temporary table User

Read dataset from /home/acadgild/assignment\_18.1/S18\_Dataset\_User\_details.txt and create RDD user\_rdd. Create case class User with field user\_id, name. Create dataframe user\_df by mapping records splitting fields by and populating the User class object. Next create temporary table User

Code is as below:

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

val user\_rdd = sc.textFile("/home/acadgild/assignment\_18.1/S18\_Dataset\_User\_details.txt")

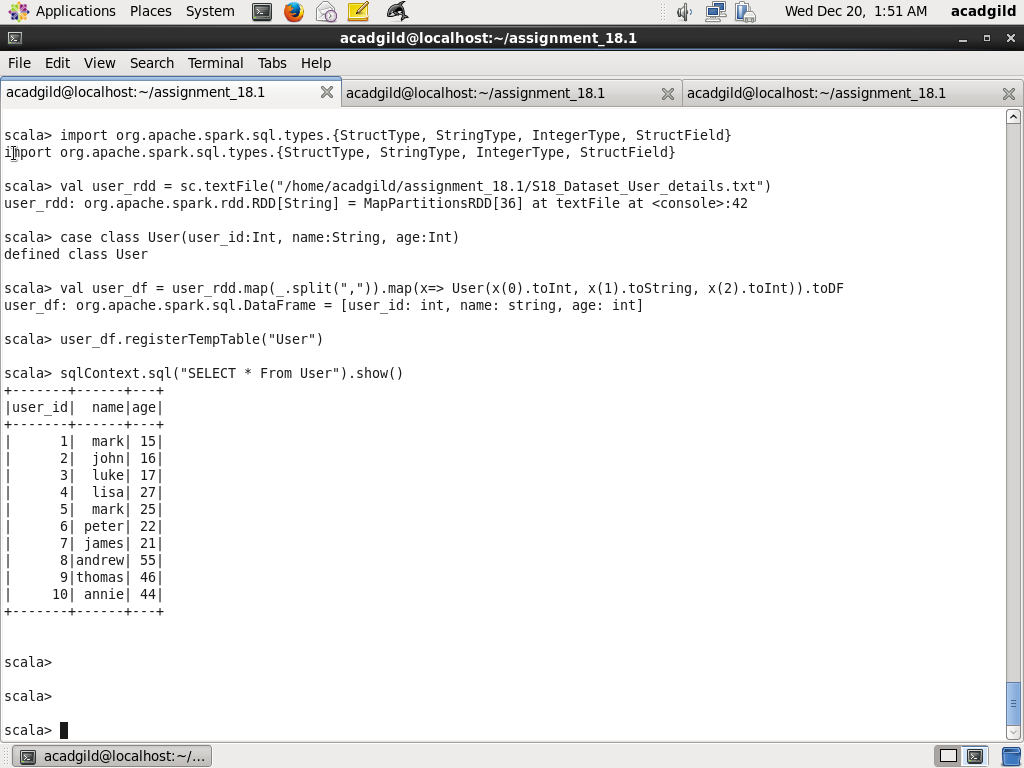
case class User(user\_id:Int, name:String, age:Int)

val user\_df = user\_rdd.map(\_.split(",")).map(x=> User(x(0).toInt, x(1).toString, x(2).toInt)).toDF

user\_df.registerTempTable("User")

sqlContext.sql("SELECT \* From User").show()

Screenshot is as below:



Step2: Create a temporary table Travel

Read dataset from /home/acadgild/assignment\_18.1/S18\_Dataset\_ Holidays.txt and create RDD user\_rdd. Create case class Travel with field user\_id, src, dest, travel\_mode distance, year\_of\_travel Create dataframe travel\_df by mapping records splitting fields by , and populating the Travel class object. Next create temporary table Travel

Code is as below:

val travel\_rdd = sc.textFile("/home/acadgild/assignment\_18.1/S18\_Dataset\_Holidays.txt")

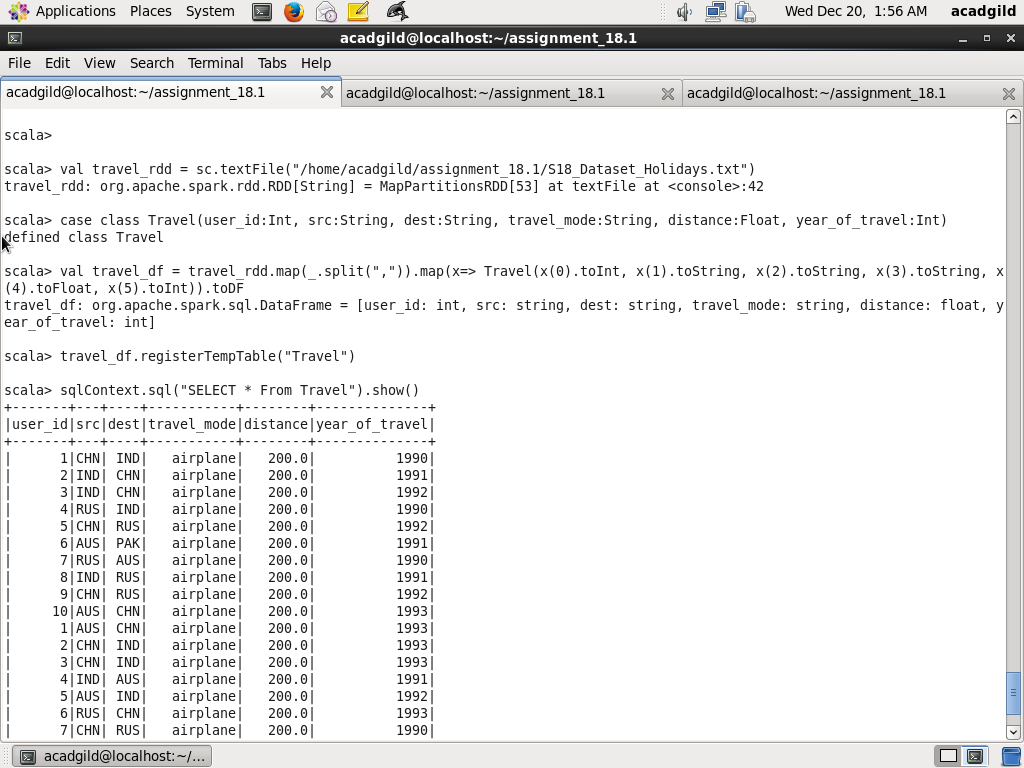
case class Travel(user\_id:Int, src:String, dest:String, travel\_mode:String, distance:Float, year\_of\_travel:Int)

val travel\_df = travel\_rdd.map(\_.split(",")).map(x=> Travel(x(0).toInt, x(1).toString, x(2).toString, x(3).toString, x(4).toFloat, x(5).toInt)).toDF

travel\_df.registerTempTable("Travel")

sqlContext.sql("SELECT \* From Travel").show()

Screenshot is as below:



Step3:

Read dataset from /home/acadgild/assignment\_18.1/S18\_Dataset\_ Transport.txt and create RDD transport\_rdd. Create case class Transport with fields travel\_mode,cost\_per\_unit Create dataframe transport\_df by mapping records splitting fields by , and populating the Transport class object. Next create temporary table Transport

val transport\_rdd = sc.textFile("/home/acadgild/assignment\_18.1/S18\_Dataset\_Transport.txt")

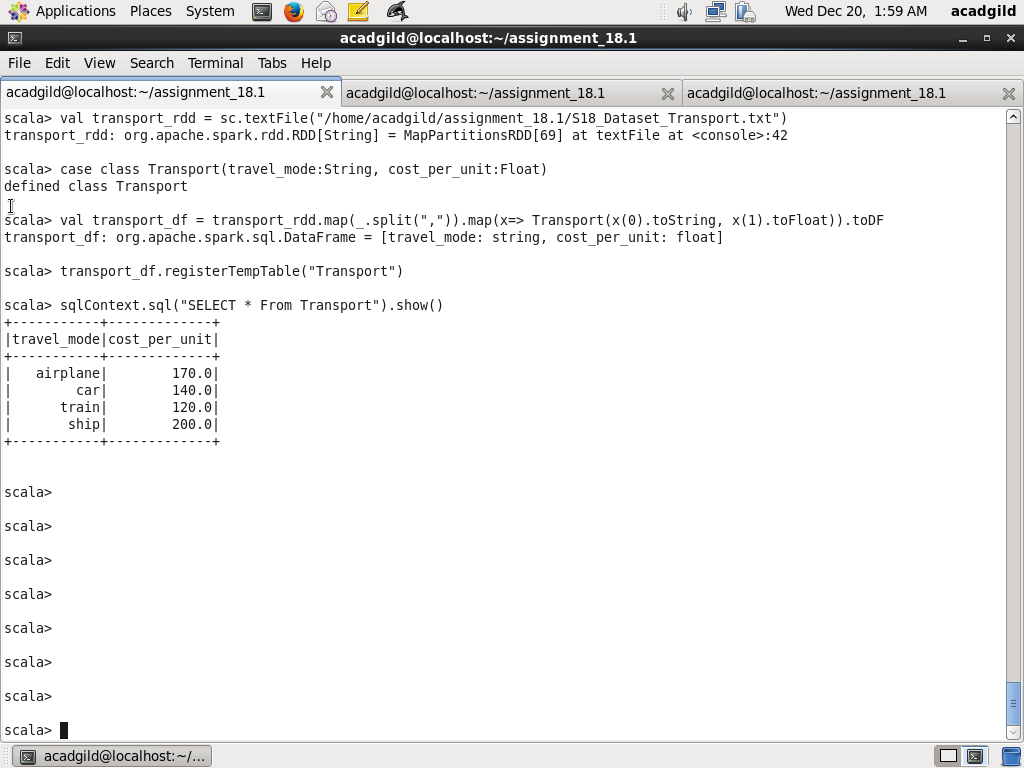
case class Transport(travel\_mode:String, cost\_per\_unit:Float)

val transport\_df = transport\_rdd.map(\_.split(",")).map(x=> Transport(x(0).toString, x(1).toFloat)).toDF

transport\_df.registerTempTable("Transport")

sqlContext.sql("SELECT \* From Transport").show()

Screenshot is as below:



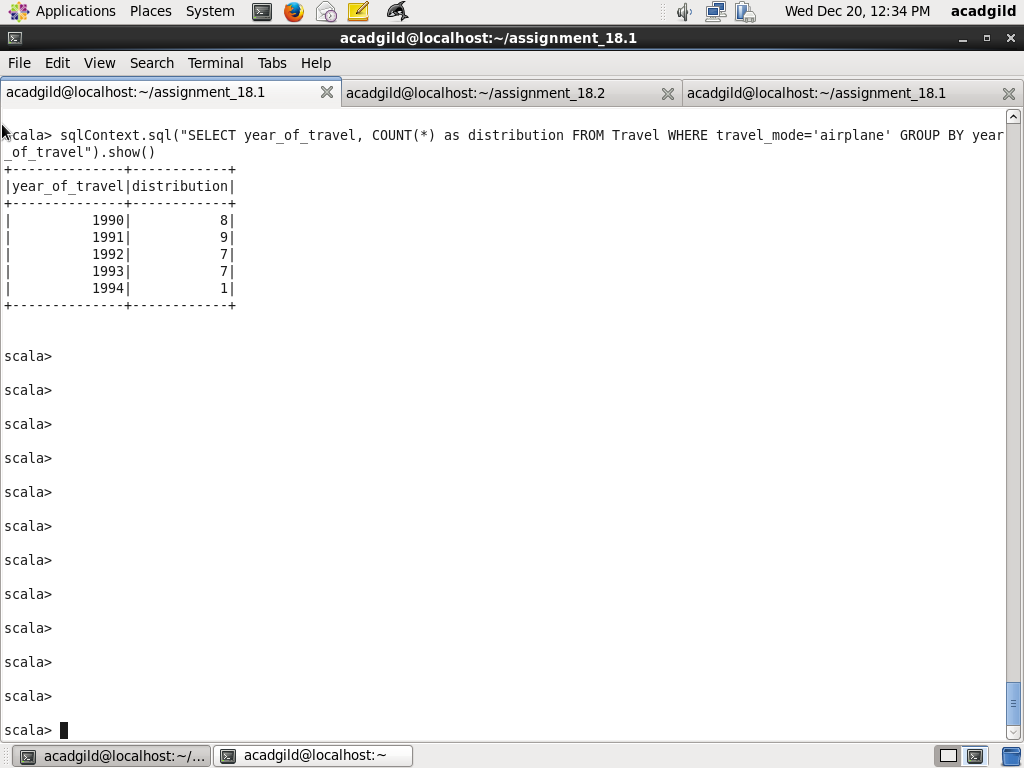
Task1: Distribution of total number of air travelers

Here using count(\*) number of travels grouped by year calculated

Code is as below:

sqlContext.sql("SELECT year\_of\_travel, COUNT(\*) as distribution FROM Travel WHERE travel\_mode='airplane' GROUP BY year\_of\_travel").show()

Screenshot is as below:



Task2: Total air distance covered by each user per year

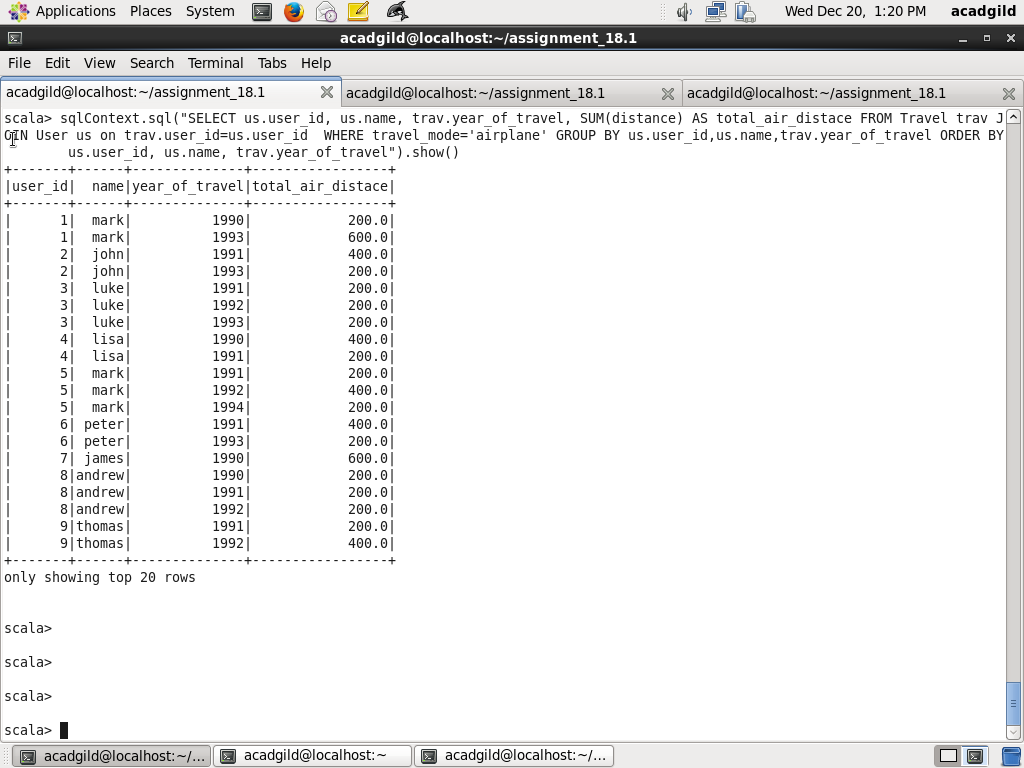
Here all the distance are summed using SUM function group by user\_id by joining the temporary tables Travel and User on user\_id column

Code is as below:

sqlContext.sql("SELECT us.user\_id, us.name, trav.year\_of\_travel, SUM(distance) AS total\_air\_distace FROM Travel trav JOIN User us on trav.user\_id=us.user\_id WHERE travel\_mode='airplane' GROUP BY us.user\_id,us.name,trav.year\_of\_travel ORDER BY us.user\_id, us.name, trav.year\_of\_travel").show()

(NOTE: I have used both user\_id and user\_name as there are two different users who have same name but different id. For example, user\_id 1 and 5 both have name as Mark)

Screenshot is as below:



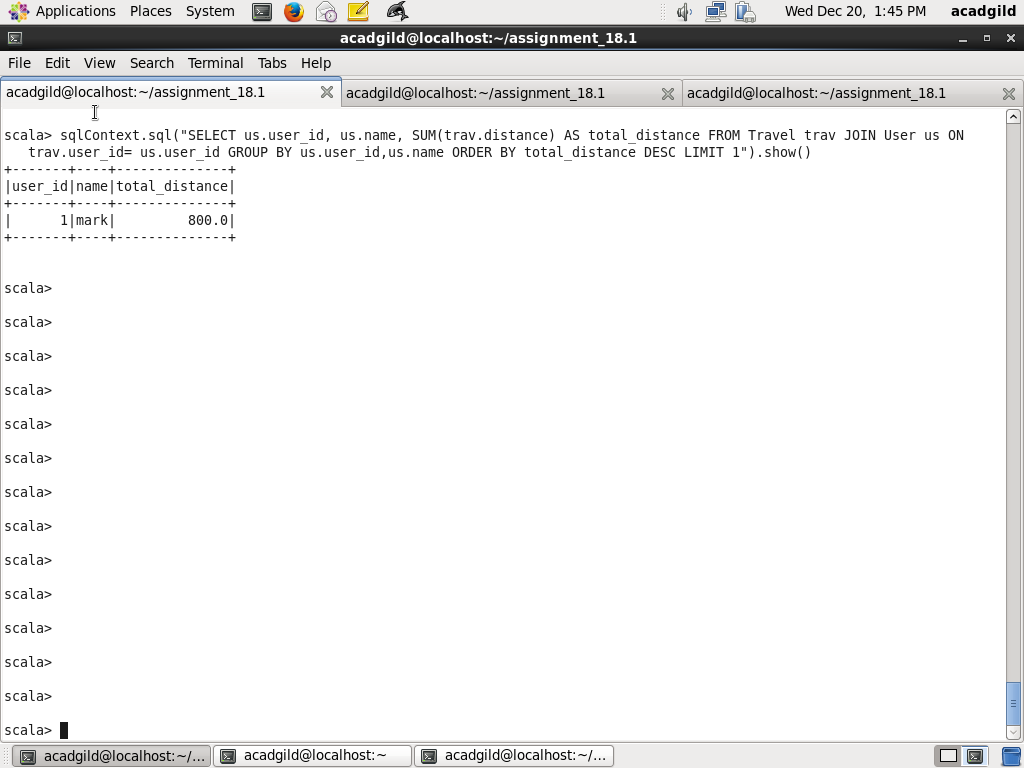
Task3: Which user has travelled largest distance till date

Here all the distance are summed using SUM function group by name, user\_id by joining tables Travel, User column user\_id then sorted in descending order by total\_air\_distance and first record is taken using LIMIT 1

Code is as below:

sqlContext.sql("SELECT us.user\_id, us.name, SUM(trav.distance) AS total\_distance FROM Travel trav JOIN User us ON trav.user\_id= us.user\_id GROUP BY us.user\_id,us.name ORDER BY total\_distance DESC LIMIT 1").show()

Screenshot is as below:



Task4: What is the most preferred destination of all users

Here using count(\*) number of travels as distributed grouped by dest calculated and sorted by distribution and first record is taken

Code is as below:

sqlContext.sql("SELECT dest, COUNT(\*) AS distribution FROM Travel GROUP BY dest ORDER BY distribution DESC LIMIT 1").show()

Screenshot is as below:

