**CSP 554 – Assignment #7**

Name: Adarsh Mathad Vijayakumar

CWID: A20424847

Email ID: avijayakumar@hawk.iit.edu

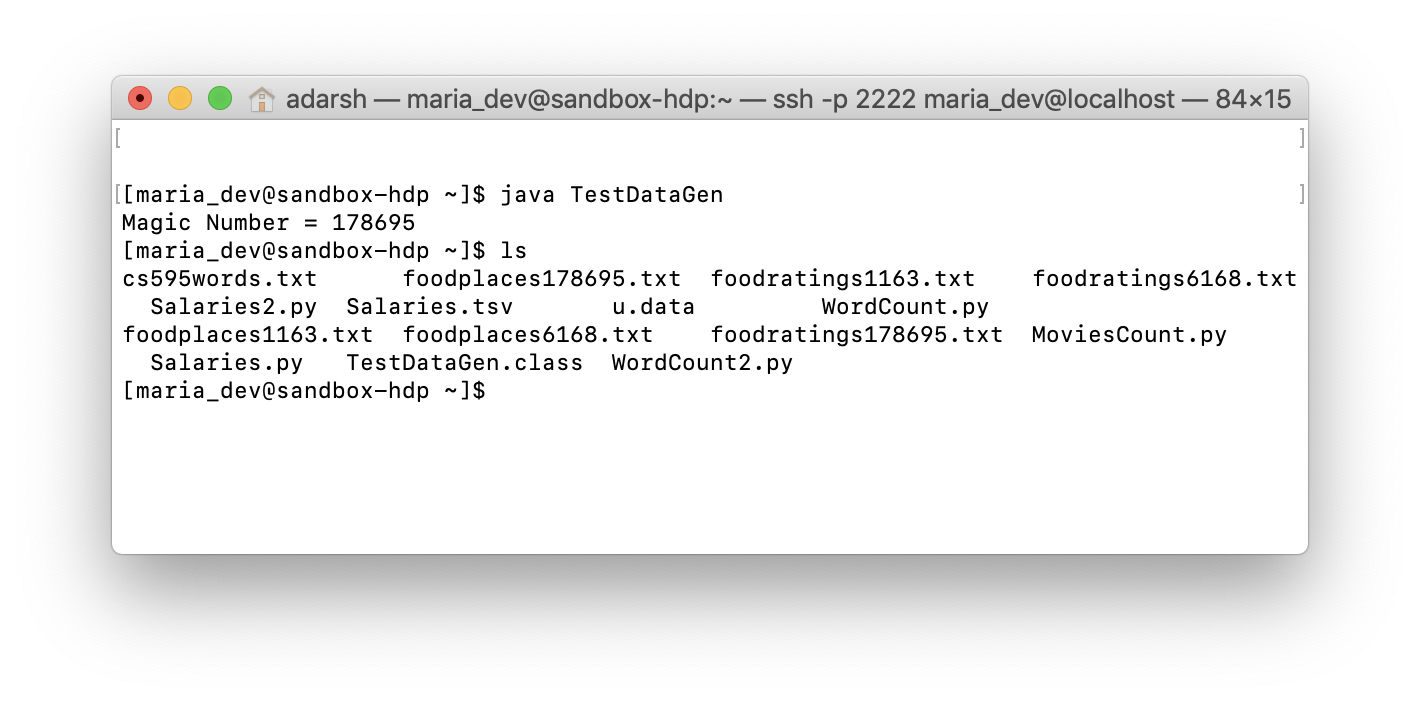
Exercise 1)

Step A

Use the TestDataGen program from previous assignments to generate new data files

Command: java TestDataGen

Output: Magic Number = 178695



Step B

Load the ‘foodratings’ file as a ‘csv’ file into a DataFrame called foodratings.

As the results of this exercise provide the magic number, the code you execute and screen shots of the following commands:

foodratings.printSchema()

foodratings.head(5)

Commands:

from pyspark.sql.types import \*

foodratingstruct = StructType(

[

StructField("name", StringType(), True),

StructField("food1",IntegerType(), True),

StructField("food2",IntegerType(), True),

StructField("food3",IntegerType(), True),

StructField("food4",IntegerType(), True),

StructField("placeid",IntegerType(), True)

]

)

foodratings=spark.read.schema(foodratingstruct).csv('/user/maria\_dev/foodratings178695.txt')

foodratings.printSchema()

foodratings.head(5)

Output Screenshot:



Exercise 2)

Load the ‘foodplaces’ file as a ‘csv’ file into a DataFrame called foodplaces.

As the results of this exercise provide the code you execute and screen shots of the following commands:

foodratings.printSchema()

foodratings.head(5)

Commands:

from pyspark.sql.types import \*

foodplacestruct = StructType(

[

StructField("placeid", IntegerType(), True),

StructField("placename", StringType(), True)

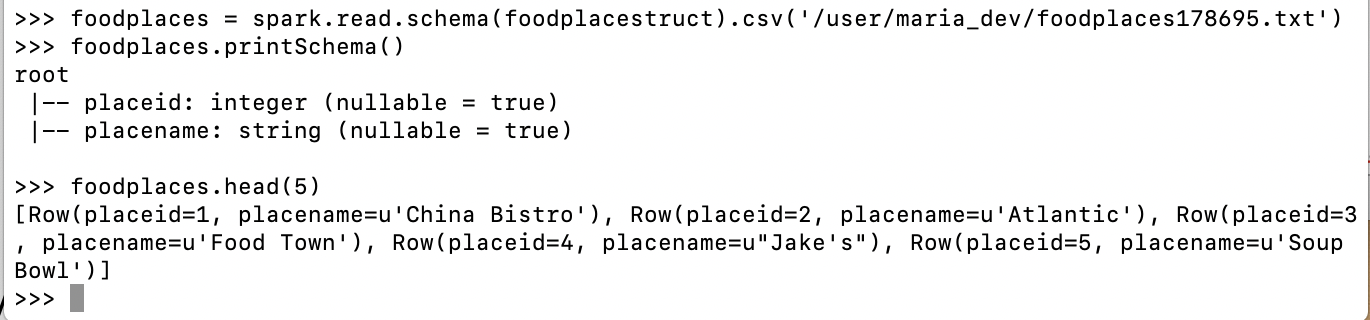
]

)

foodplaces = spark.read.schema(foodplacestruct).csv('/user/maria\_dev/foodplaces178695.txt')

foodplaces.printSchema()

foodplaces.head(5)



Exercise 3)

Step A

Register the DataFrames created in exercise 1 and 2 as tables called “foodratingsT” and “foodplacesT”

Commands:

from pyspark.sql.types import \*

foodratings.createOrReplaceTempView("foodratingsT")

foodplaces.createOrReplaceTempView("foodplacesT")

Step B

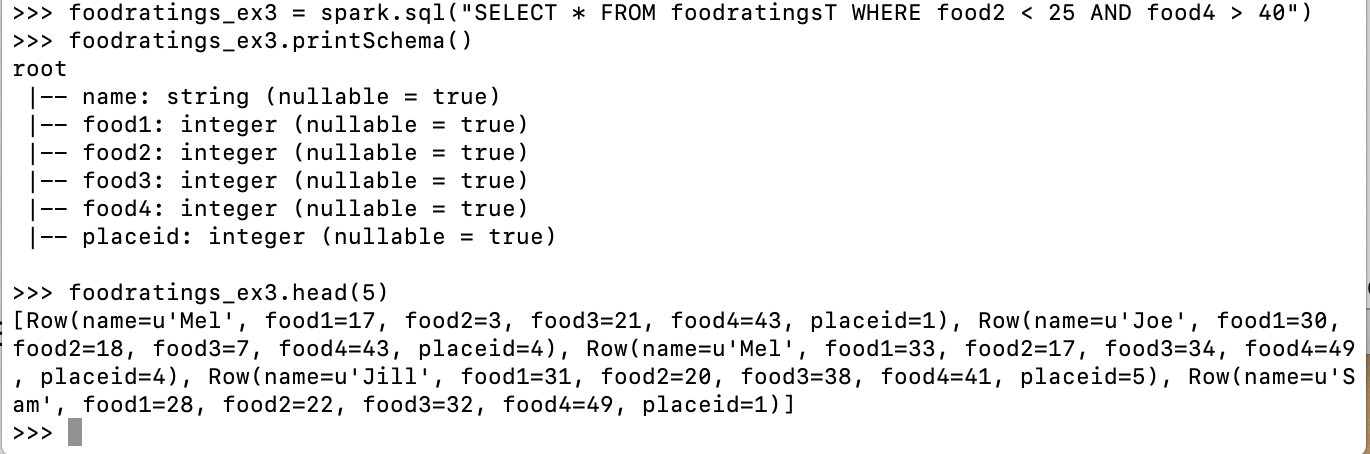
Use a SQL query on the table “foodratingsT” to create a new DataFrame called foodratings\_ex3 holding records which meet the following condition: food2 < 25 and food4 > 40

Commands:

foodratings\_ex3 = spark.sql("SELECT \* FROM foodratingsT WHERE food2 < 25 AND food4 > 40")

foodratings\_ex3.printSchema()

foodratings\_ex3.head(5)



Step C

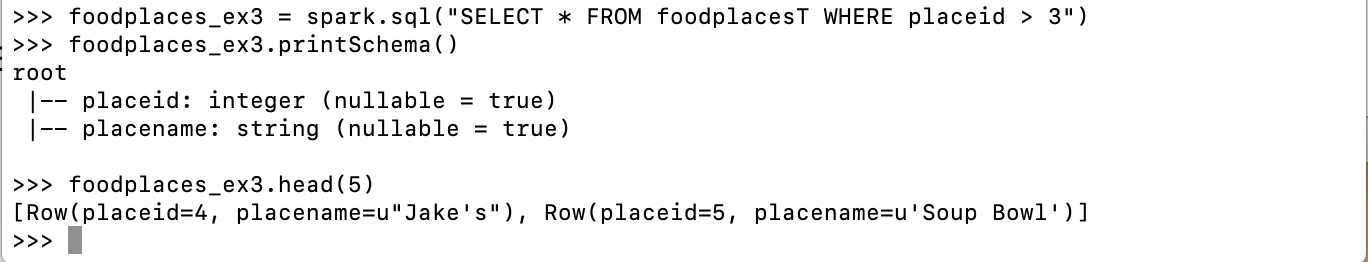
Use a SQL query on the table “foodplacesT” to create a new DataFrame called foodplaces\_ex3 holding records which meet the following condition: placeid > 3

Commands:

foodplaces\_ex3 = spark.sql("SELECT \* FROM foodplacesT WHERE placeid > 3")

foodplaces\_ex3.printSchema()

foodplaces\_ex3.head(5)



Exercise 4)

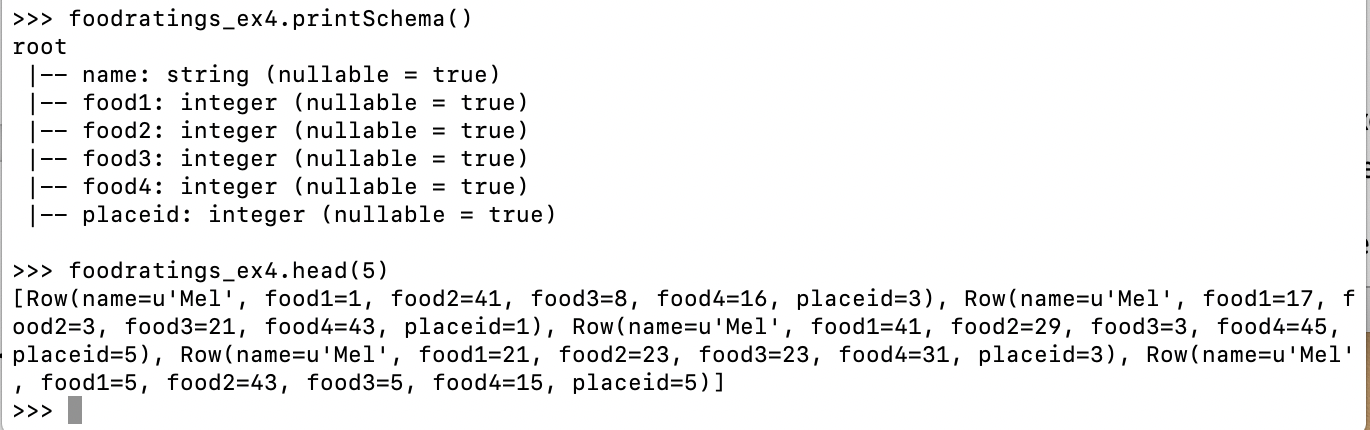
Use an operation (not a SQL query) on the DataFrame ‘foodratings’ create in exercise 1 to create a new DataFrame called foodratings\_ex4 that includes only those records (rows) where the ‘name’ field is “Mel” and food3 < 25.

Commands:

foodratings\_ex4 = foodratings.filter(foodratings.name == "Mel").filter(foodratings.food3 < 25)

foodratings\_ex4.printSchema()

foodratings\_ex4.head(5)



Exercise 5)

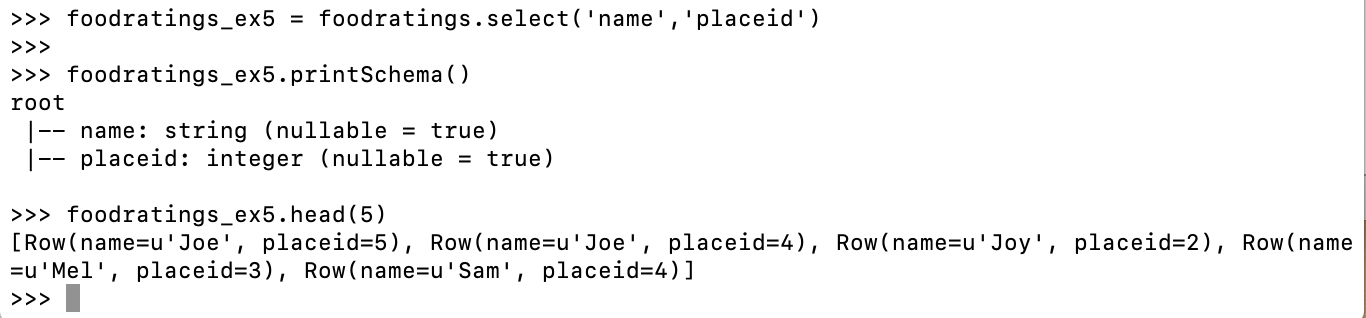
Use an operation (not a SQL query) on the DataFrame ‘foodratings’ create in exercise 1 to create a new DataFrame called foodratings\_ex5 that includes only the columns (fields) ‘name’ and ‘placeid’

Commands:

foodratings\_ex5 = foodratings.select('name','placeid')

foodratings\_ex5.printSchema()

foodratings\_ex5.head(5)



Exercise 6)

Use an operation on the DataFrame called ex6 which is the inner join, on placeid, of the DataFrames ‘foodratings; and ‘foodplaces’ created in exercises 1 and 2

Commands:

ex6 = foodratings.join(foodplaces, foodratings.placeid == foodplaces.placeid,"inner").drop(foodratings.placeid)

ex6.printSchema()

ex6.head(5)

