CSP 554 Assignment 7

Magic Number = 191133

```
[hadoop@ip-172-31-3-0 ~]$ java TestDataGen
Magic Number = 191133
[hadoop@ip-172-31-3-0 ~]$ ls
foodplaces191133.txt  foodratings191133.txt  TestDataGen.class
[hadoop@ip-172-31-3-0 ~]$ hdfs dfs -copyFromLocal /home/hadoop/foodplaces191133.txt /user/hadoop
[hadoop@ip-172-31-3-0 ~]$ hdfs dfs -copyFromLocal /home/hadoop/foodratings191133.txt /user/hadoop
[hadoop@ip-172-31-3-0 ~]$ hadoop fs -rm /user/hadoop/foodratings191133.txt
Deleted /user/hadoop/foodratings191133.txt
[hadoop@ip-172-31-3-0 ~]$ hadoop fs -rm /user/hadoop/foodplaces191133.txt
Deleted /user/hadoop/foodplaces191133.txt
[hadoop@ip-172-31-3-0 ~]$ hdfs dfs -copyFromLocal /home/hadoop/foodratings191133.txt /user/hadoop/foodratings191133.csv
[hadoop@ip-172-31-3-0 ~]$ hdfs dfs -copyFromLocal /home/hadoop/foodplaces191133.txt /user/hadoop/foodplaces191133.csv
[hadoop@ip-172-31-3-0 ~]$ hdfs dfs -ls /user/hadoop/*191133
ls: `/user/hadoop/*191133': No such file or directory
[hadoop@ip-172-31-3-0 ~]$ hdfs dfs -ls /user/hadoop/*191133.csv
-rw-r--r--   1 hadoop hdfsadmingroup         59 2022-10-24 15:16 /user/hadoop/foodplaces191133.csv
-rw-r--r--   1 hadoop hdfsadmingroup      17489 2022-10-24 15:16 /user/hadoop/foodratings191133.csv
```

Code:
java TestDataGen

hdfs dfs -copyFromLocal /home/hadoop/foodratings191133.txt /user/hadoop/foodratings191133.csv

hdfs dfs -copyFromLocal /home/hadoop/foodplaces191133.txt /user/hadoop/foodplaces191133.csv

Question 1:

Code:
from pyspark.sql.type import StructField, StringType, StructType, LongType, IntegerType

assignment7schema = 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(assignment7schema).csv('/user/hadoop/foodratings191133.csv')

foodratings.printSchema()
foodratings.show(5)

```
SparkSession available as 'spark'.
>>> from pyspark.sql.types import StructField, StringType, StructType, LongType, IntegerType
>>> assignment7schema = StructType([
... StructField("name",StringType(),True).
... StructField("name",StringType(),Truedwjclsjwkdl])
  File "<stdin>", line 3
    StructField("name",StringType(),Truedwjclsjwkdl])
                                                    ^
SyntaxError: invalid syntax
>>> assignment7schema = 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(assignment7schema).csv('/user/hadoop/foodratings191133.csv')
>>> foodratings.printSchema()
root
 |-- name: string (nullable = true)
 |-- food1: integer (nullable = true)
 |-- food2: integer (nullable = true)
 |-- food3: integer (nullable = true)
 |-- food4: integer (nullable = true)
 |-- placeid: integer (nullable = true)

>>> foodratings.show(5)
+----+-----+-----+-----+-----+-------+
|name|food1|food2|food3|food4|placeid|
+----+-----+-----+-----+-----+-------+
| Joe|    6|   25|   21|   17|      4|
|Jill|   14|   20|   38|   18|      5|
| Joe|   44|    9|   29|   23|      5|
| Joe|   10|    1|   47|   15|      2|
| Mel|   32|   29|   41|   16|      2|
+----+-----+-----+-----+-----+-------+
only showing top 5 rows

>>>
```

Question 2:

Code:
assignment7schema2 = StructType([
StructField(“placeid”, IntegerType(), True),
StructField(“placename”, StringType(), True)])

foodplaces =
spark.read.schema(assignment7schema2).csv(‘/user/hadoop/foodplaces191133.csv’)

foodplaces.printSchema()

foodplaces.show(5)

```
>>> assignment7schema2 = StructType([
... StructField("placeid",IntegerType(),True),
... StructField("placename",StringType(),True)])
>>> foodplaces = spark.read.schema(assignment7schema2).csv('/user/hadoop/foodplaces191133.csv')
>>> foodplaces.printSchema()
root
 |-- placeid: integer (nullable = true)
 |-- placename: string (nullable = true)

>>> foodplaces.show(5)
+-------+------------+
|placeid|   placename|
+-------+------------+
|      1|China Bistro|
|      2|    Atlantic|
|      3|   Food Town|
|      4|      Jake's|
|      5|   Soup Bowl|
+-------+------------+
```

Question 3:

Step A & B:

Code:

foodratings.createOrReplaceTempView(“foodratingsT”)
foodplaces.createOrReplaceTempView(“foodplacesT”)

*foodratings_ex3a = spark.sql(“SELECT * FROM foodratingsT WHERE (food2<25) AND (food4>40)”)*

foodratings_ex3a.printSchema()
foodratings_ex3a.show(5)

```
>>> foodratings.createOrReplaceTempView("foodratingstemp")
>>> foodratings.createOrReplaceTempView("foodratingsT")
>>> foodplaces.createOrReplaceTempView("foodplacesT")
>>> foodratings_ex3a = spark.sql("SELECT * FROM foodratingsT WHERE (food2<25) AND (food4>40)")
22/10/24 15:47:37 WARN ObjectStore: Version information not found in metastore. hive.metastore.schema.verification is not enabled so recording the schema vers
ion 1.2.0
22/10/24 15:47:37 WARN ObjectStore: Failed to get database default, returning NoSuchObjectException
22/10/24 15:47:38 WARN ObjectStore: Failed to get database global_temp, returning NoSuchObjectException
>>> foodratings_ex3a.printSchema()
root
 |-- name: string (nullable = true)
 |-- food1: integer (nullable = true)
 |-- food2: integer (nullable = true)
 |-- food3: integer (nullable = true)
 |-- food4: integer (nullable = true)
 |-- placeid: integer (nullable = true)

>>> foodratings_ex3a.show(5)
[Stage 2:>                                                          [Stage 2:>
+----+-----+-----+-----+-----+-------+
|name|food1|food2|food3|food4|placeid|
+----+-----+-----+-----+-----+-------+
| Joy|   10|   15|   29|   46|      1|
| Sam|   37|   24|   27|   42|      5|
| Joe|   12|   23|   15|   45|      3|
|Jill|   27|   15|   38|   49|      3|
| Mel|    7|   13|   10|   45|      4|
+----+-----+-----+-----+-----+-------+
only showing top 5 rows
```

Step C:
Code:
*foodplaces_ex3b = spark.sql("SELECT * FROM foodplacesT WHERE (placeid>3)")*

foodplaces_ex3b.printSchema()
foodplaces_ex3b.show(5)

```
>>> foodplaces_ex3b = spark.sql("SELECT * FROM foodplacesT WHERE (placeid>3)")
>>> foodplaces_ex3b.printSchema()
root
 |-- placeid: integer (nullable = true)
 |-- placename: string (nullable = true)

>>> foodplaces_ex3b.show(5)
[Stage 3:>                                                          [Stage 3:>
                                    +-------+---------+
|placeid|placename|
+-------+---------+
|      4|   Jake's|
|      5|Soup Bowl|
+-------+---------+
```

Question 4:

Code:
foodratings_ex4 = foodratings.filter((foodratings.name=='Mel')&(foodratings.food3<25))

foodratings_ex4.printSchema()
foodratings_ex4.show(5)

```
>>> foodratings_ex4 = foodratings.filter((foodratings.name=='Mel')&(foodratings.food3<25))
>>> foodratings_ex4.printSchema()
root
 |-- name: string (nullable = true)
 |-- food1: integer (nullable = true)
 |-- food2: integer (nullable = true)
 |-- food3: integer (nullable = true)
 |-- food4: integer (nullable = true)
 |-- placeid: integer (nullable = true)

>>> foodratings_ex4.show(5)
[Stage 4:>                                                          [Stage 4:>
                                    +----+-----+-----+-----+-----+-------+
|name|food1|food2|food3|food4|placeid|
+----+-----+-----+-----+-----+-------+
| Mel|   46|    3|    6|   26|      3|
| Mel|   30|   31|   10|   34|      5|
| Mel|   25|    2|   14|   36|      3|
| Mel|   48|   23|   22|   10|      1|
| Mel|    7|   13|   10|   45|      4|
+----+-----+-----+-----+-----+-------+
only showing top 5 rows
```

Question 5:

Code:
foodratings_ex5 = foodratings.select('name', 'placeid')

foodratings_ex5.printSchema()
foodratings_ex5.show()

```
>>> foodratings_ex5 = foodratings.select('name', 'placeid')
>>> foodratings_ex5.printSchema()
root
 |-- name: string (nullable = true)
 |-- placeid: integer (nullable = true)

>>> foodrating_ex5.show(5)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'foodrating_ex5' is not defined
>>> foodratings_ex5.show(5)
[Stage 5:>                                                          [Stage 5:>
                                    +----+-------+
|name|placeid|
+----+-------+
| Joe|      4|
|Jill|      5|
| Joe|      5|
| Joe|      2|
| Mel|      2|
+----+-------+
only showing top 5 rows
```

Question 6:

Code:
condition = [foodplaces.placeid == foodratings.placeid]
ex6 = foodratings.join(foodplaces, condition, 'inner')
ex6.printSchema()
ex6.show(5)

```
>>> condition = [foodplaces.placeid == foodratings.placeid]
>>> ex6 = foodratings.join(foodplaces, condition, 'inner')
>>> ex6.printSchema()
root
 |-- name: string (nullable = true)
 |-- food1: integer (nullable = true)
 |-- food2: integer (nullable = true)
 |-- food3: integer (nullable = true)
 |-- food4: integer (nullable = true)
 |-- placeid: integer (nullable = true)
 |-- placeid: integer (nullable = true)
 |-- placename: string (nullable = true)

>>> ex6.show(5)
+----+-----+-----+-----+-----+-------+-------+---------+
|name|food1|food2|food3|food4|placeid|placeid|placename|
+----+-----+-----+-----+-----+-------+-------+---------+
| Joe|    6|   25|   21|   17|      4|      4|   Jake's|
|Jill|   14|   20|   38|   18|      5|      5|Soup Bowl|
| Joe|   44|    9|   29|   23|      5|      5|Soup Bowl|
| Joe|   10|    1|   47|   15|      2|      2| Atlantic|
| Mel|   32|   29|   41|   16|      2|      2| Atlantic|
+----+-----+-----+-----+-----+-------+-------+---------+
only showing top 5 rows
```