03: Spark on Zeppelin – DataFrame Operations in Scala

Posted on July 25, 2018

Pre-requisite: Docker is installed on your machine for Mac OS X (E.g. $ brew cask install docker) or Windows 10. Docker interview Q&As.

This tutorial extends Apache Zeppelin on Docker Tutorial – Docker pull from Docker hub and Spark stand-alone to read a file from local file system

```
val lines = sc.textFile("file:///zeppelin/seed/emp

case class Employee (id: Integer, name: String, lo

val dfEmployees = lines.map(s => s.split(",")).map
```

300+ Java Interview FAQs

300+ Java FAQs
16+ Java Key Areas Q&As
150+ Java Architect FAQs
80+ Java Code Quality Q&As
150+ Java Coding Q&As

300+ Big Data Interview FAQs

300+ Big Data FAQs
Tutorials - Big Data
TUT - Starting Big Data
TUT - Starting Spark & Scala

1. Print the schema of the Dataframe

```
dfEmployees.printSchema()
```

root
|– id: integer (nullable = true)
|– name: string (nullable = true)
|– location: string (nullable = true)
|– salary: double (nullable = true)

2. Show contents of a Dataframe

```
dfEmployees.show()
```

```

+---+--------+---------+--------+
| id|    name| location|  salary|
+---+--------+---------+--------+
|  1|    John|      USA|100000.0|
|  2|   Peter|Australia|200000.0|
|  3|     Sam|      USA| 76000.0|
|  4|  Daniel|   France| 86000.0|
|  5|   Simon|Australia| 96000.0|
|  6|Roseanne|   France|156000.0|
+---+--------+---------+--------+

```

3. Count number of rows in a Dataframe

```
dfEmployees.count()
```

res12: Long = 6

4. Add a new column to a Dataframe

800+ Java Interview Q&As

```
dfEmployees.withColumn("bonus", dfEmployees.col("s
            .show()

```

```

+---+--------+----------+--------+------+
| id|    name|  location|  salary| bonus|
+---+--------+----------+--------+------+
|  1|    John|       USA|100000.0|2000.0|
|  2|   Peter| Australia|200000.0|4000.0|
|  3|     Sam|       USA| 76000.0|1520.0|
|  4|  Daniel|    France| 86000.0|1720.0|
|  5|   Simon| Australia| 96000.0|1920.0|
|  6|Roseanne|    France|156000.0|3120.0|
+---+--------+----------+--------+------+

```

You can drop a column with “dfEmployees.drop(“location”).show()”

5. Select a few columns from a Dataframe

```

dfEmployees.withColumn("bonus", dfEmployees.col("s
           .select("id", "bonus")
           .show()

```

```

+---+------+
| id| bonus|
+---+------+
|  1|2000.0|
|  2|4000.0|
|  3|1520.0|
|  4|1720.0|
|  5|1920.0|
|  6|3120.0|
+---+------+

```

6. Distinct values

```
dfEmployees.select("location")
           .distinct()
           .show()

```

```

+----------+
|  location|
+----------+
| Australia|
|       USA|
|    France|
+----------+

```

7. Sorting

```

dfEmployees.orderBy("location")
           .show()

```

```

+---+---------+----------+--------+
| id|     name|  location|  salary|
+---+---------+----------+--------+
|  2|    Peter| Australia|200000.0|
|  5|    Simon| Australia| 96000.0|
|  4|   Daniel|    France| 86000.0|
|  6| Roseanne|    France|156000.0|
|  1|     John|       USA|100000.0|
|  3|      Sam|       USA| 76000.0|
+---+---------+----------+--------+

```

8. Applying SQL queries

```

dfEmployees.registerTempTable("employees_tbl")

```

```

%sql

Select * from employees_tbl

```

9. Filtering by a predicate

```
dfEmployees.filter(dfEmployees.col("salary") > 100(
```

```
+---+---------+----------+---------+
| id|     name|  location|   salary|
+---+---------+----------+---------+
|  2|    Peter| Australia|200000.0|
|  6| Roseanne|    France|156000.0|
+---+---------+----------+---------+
```

10. Grouping & aggregation

```
dfEmployees.groupBy("location").agg(sum("salary"))
```

```
+----------+-----------+
|  location|sum(salary)|
+----------+-----------+
| Australia|   296000.0|
|       USA|   176000.0|
|    France|   242000.0|
+----------+-----------+
```

11. Map operations on Dataframe columns

We can apply a function on each row of DataFrame using map operation.

```
dfEmployees.select("id", "salary").map(row => (row
```

```
+---+--------+
| _1|      _2|
+---+--------+
|  1|100220.0|
|   2|200220.0|
|   3| 76220.0|
|   4| 86220.0|
|   5| 96220.0|
|   6|156220.0|
+---+---------+

```

12. Get some stats on your data

```

dfEmployees.select("salary").describe().show()

```

```

+-------+-----------------+
|summary|           salary|
+-------+-----------------+
|  count|                6|
|   mean|         119000.0|
| stddev|48493.29850608226|
|    min|          76000.0|
|    max|         200000.0|
+-------+-----------------+

```

‹ 02: Spark on Zeppelin – read a file from local file system

04: Spark on Zeppelin – DataFrame joins in Scala ›