Big Data

(Spark SQL)

Instructor: Trong-Hop Do

December 8th 2020

S3Lab

Smart Software System Laboratory

“Big data is at the foundation of all the megatrends that are happening today, from social to mobile to cloud to gaming.”

– Chris Lynch, Vertica Systems

Spark SQL

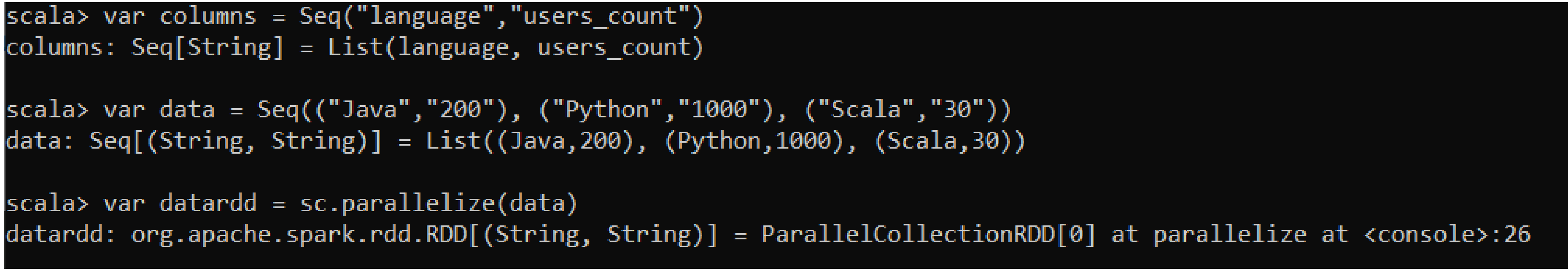
Spark SQL

- Spark SQL is Spark module for structured data processing. It runs on top of Spark Core. It offers much tighter integration between relational and procedural processing, through declarative DataFrame and Datasets API. These are the ways which enable users to run SQL queries over Spark.

Create DataFrame from RDD

Using toDF() function

- Let’s create some RDD

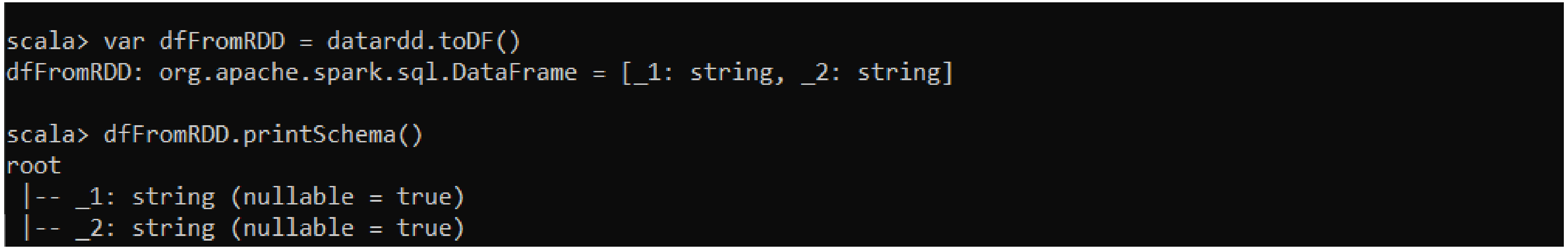


```
scala> var columns = Seq("language","users_count")
columns: Seq[String] = List(language, users_count)

scala> var data = Seq(("Java","200"), ("Python","1000"), ("Scala","30"))
data: Seq[(String, String)] = List((Java,200), (Python,1000), (Scala,30))

scala> var datardd = sc.parallelize(data)
datardd: org.apache.spark.rdd.RDD[(String, String)] = ParallelCollectionRDD[0] at parallelize at <console>:26
```

Create DataFrame from RDD

Using toDF() function

- let’s use toDF() to create DataFrame in Spark.
- Since RDD is schema-less without column names and data type, converting from RDD to DataFrame gives you default column names as _1, _2 (for two rows) and so on and data type as String.

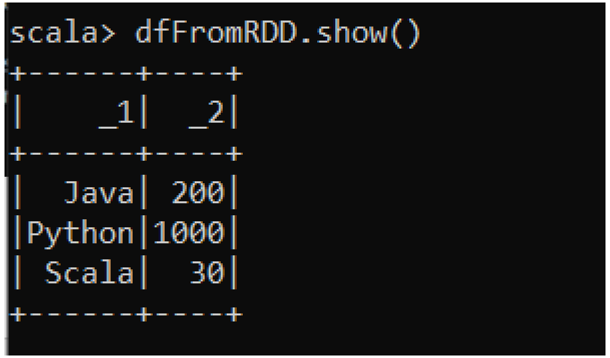


```
scala> var dfFromRDD = datardd.toDF()
dfFromRDD: org.apache.spark.sql.DataFrame = [_1: string, _2: string]

scala> dfFromRDD.printSchema()
root
 |-- _1: string (nullable = true)
 |-- _2: string (nullable = true)
```

Create DataFrame from RDD

Using toDF() function

- Show dataframe

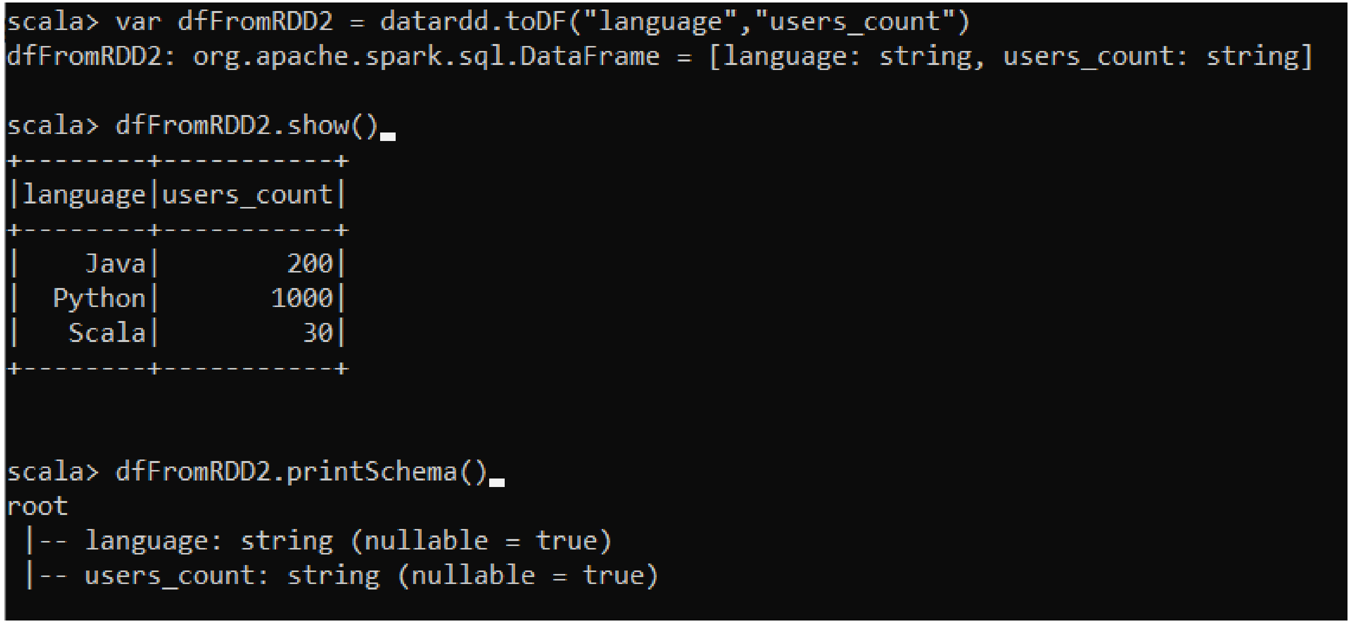


```
scala> dfFromRDD.show()
+------+----+
|    _1|  _2|
+------+----+
|  Java| 200|
|Python|1000|
| Scala|  30|
+------+----+
```

Create DataFrame from RDD

Using toDF() function

- Assign a column name

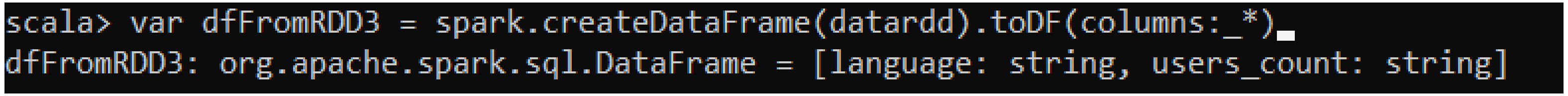


```
scala> var dfFromRDD2 = datardd.toDF("language","users_count")
dfFromRDD2: org.apache.spark.sql.DataFrame = [language: string, users_count: string]

scala> dfFromRDD2.show()
+--------+-----------+
|language|users_count|
+--------+-----------+
|    Java|        200|
|  Python|       1000|
|   Scala|         30|
+--------+-----------+


scala> dfFromRDD2.printSchema()
root
 |-- language: string (nullable = true)
 |-- users_count: string (nullable = true)
```

Create DataFrame from RDD

Using Spark createDataFrame() from SparkSession

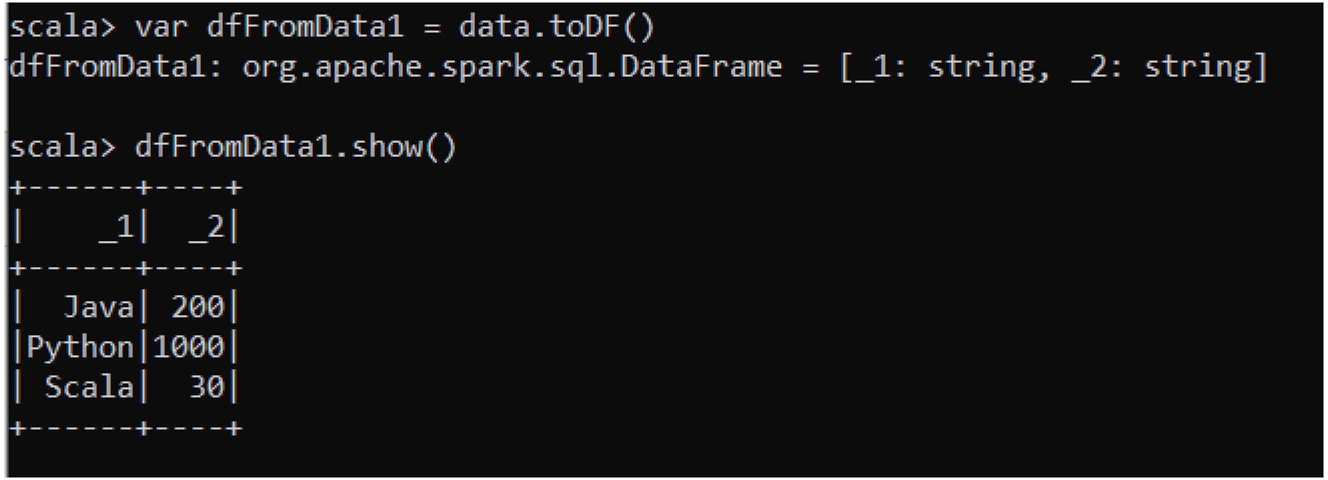
- Prior to 2.0, SparkContext used to be an entry point. Since Spark 2.0, SparkSession has become an entry point to Spark to work with RDD, DataFrame, and Dataset.
- Be default Spark shell provides "**spark**" object which is an instance of SparkSession class. We can directly use this object where required in spark-shell.
- Using createDataFrame() from SparkSession is another way to create and it takes rdd object as an argument. and chain with toDF() to specify names to the columns.



```
scala> var dfFromRDD3 = spark.createDataFrame(datardd).toDF(columns:_*)
dfFromRDD3: org.apache.spark.sql.DataFrame = [language: string, users_count: string]
```

Create DataFrame from List and Seq Collection

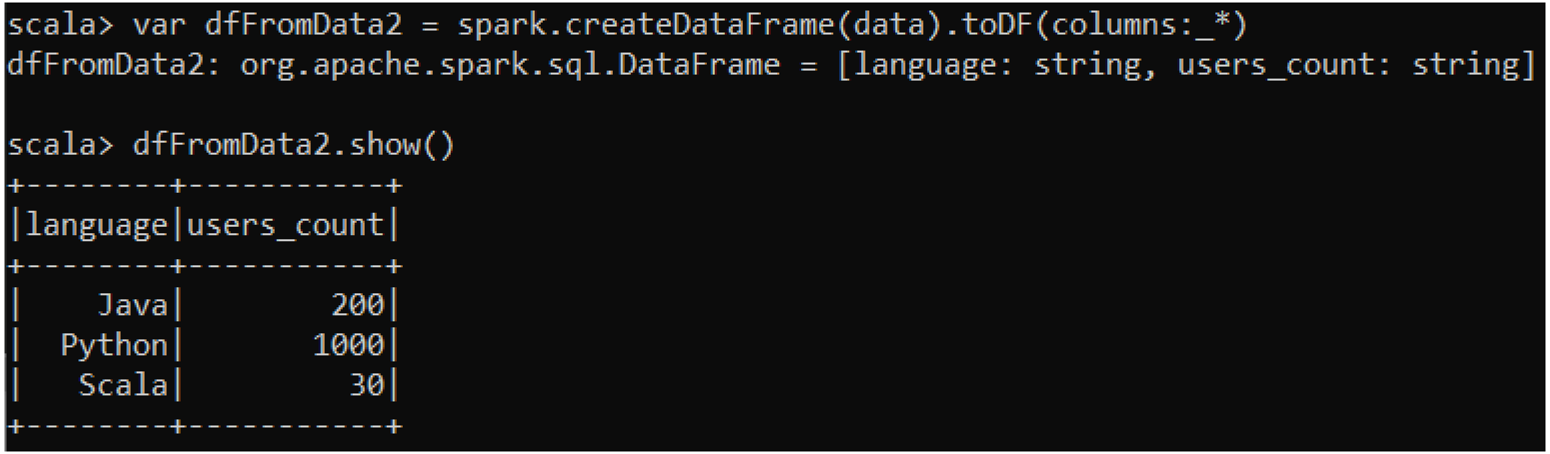
Using toDF() on List or Seq collection



```
scala> var dfFromData1 = data.toDF()
dfFromData1: org.apache.spark.sql.DataFrame = [_1: string, _2: string]

scala> dfFromData1.show()
+------+----+
|    _1|  _2|
+------+----+
|  Java| 200|
|Python|1000|
| Scala|  30|
+------+----+
```

Create DataFrame from List and Seq Collection

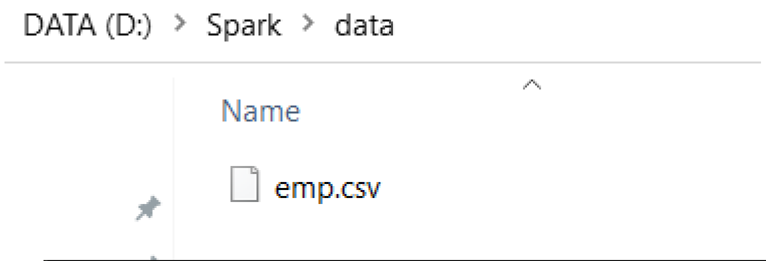
Using createDataFrame() from SparkSession



```
scala> var dfFromData2 = spark.createDataFrame(data).toDF(columns:_*)
dfFromData2: org.apache.spark.sql.DataFrame = [language: string, users_count: string]

scala> dfFromData2.show()
+--------+-----------+
|language|users_count|
+--------+-----------+
|    Java|        200|
|  Python|       1000|
|   Scala|         30|
+--------+-----------+
```

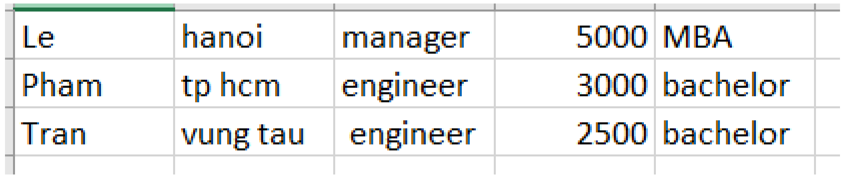
Create Spark DataFrame from CSV

Using createDataFrame() from SparkSession

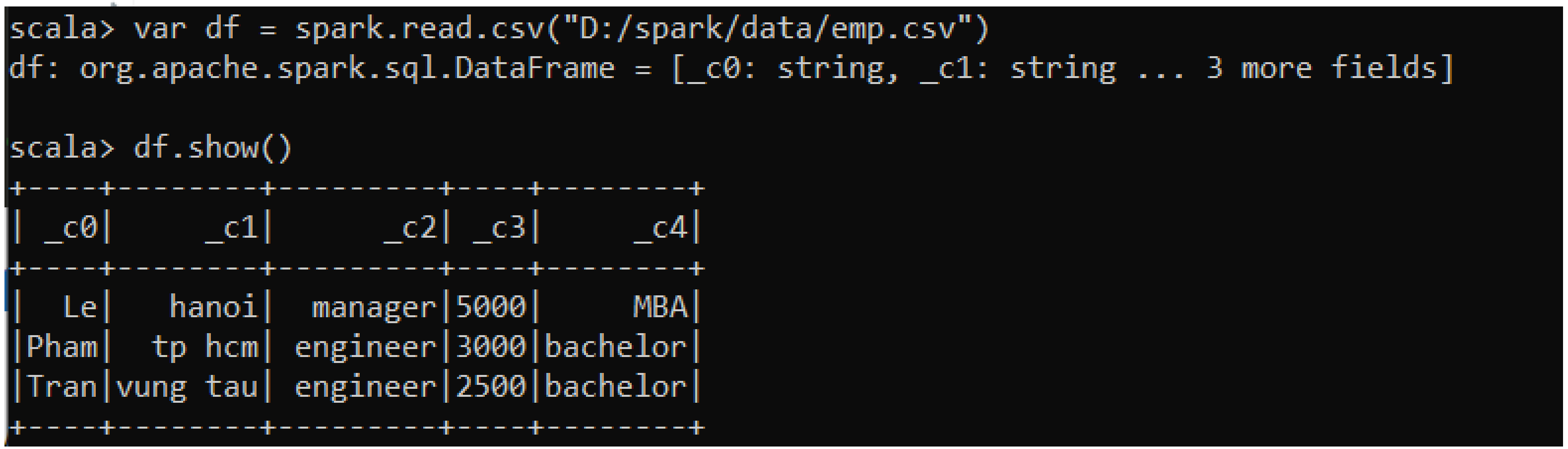

DATA (D:) › Spark › data

Name

emp.csv



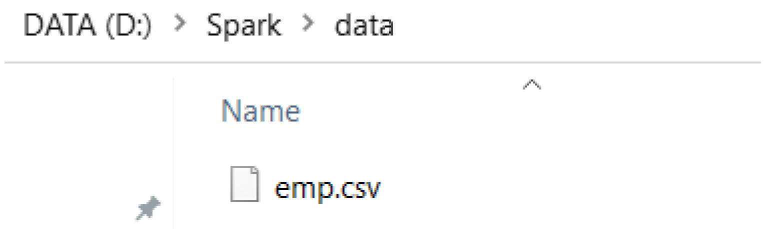
Le	hanoi	manager	5000	MBA
Pham	tp hcm	engineer	3000	bachelor
Tran	vung tau	engineer	2500	bachelor



```
scala> var df = spark.read.csv("D:/spark/data/emp.csv")
df: org.apache.spark.sql.DataFrame = [_c0: string, _c1: string ... 3 more fields]

scala> df.show()
+----+--------+--------+----+--------+
| _c0|     _c1|     _c2| _c3|     _c4|
+----+--------+--------+----+--------+
|  Le|   hanoi| manager|5000|     MBA|
|Pham|  tp hcm| engineer|3000|bachelor|
|Tran|vung tau| engineer|2500|bachelor|
+----+--------+--------+----+--------+
```

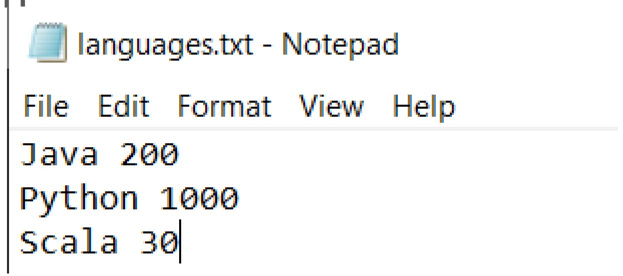
Create Spark DataFrame from text file

Using createDataFrame() from SparkSession

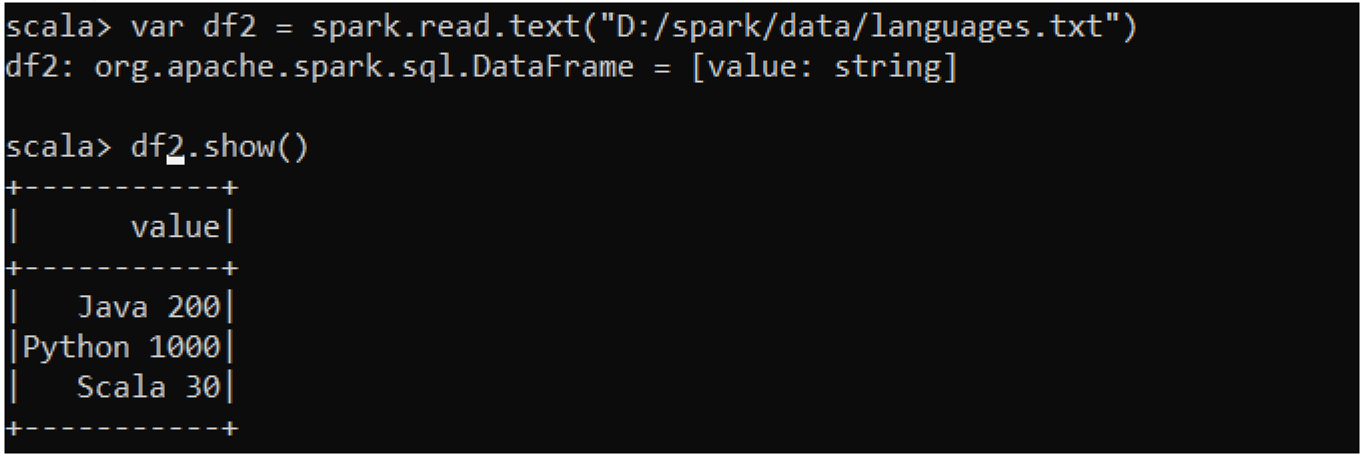

DATA (D:) › Spark › data

Name

emp.csv

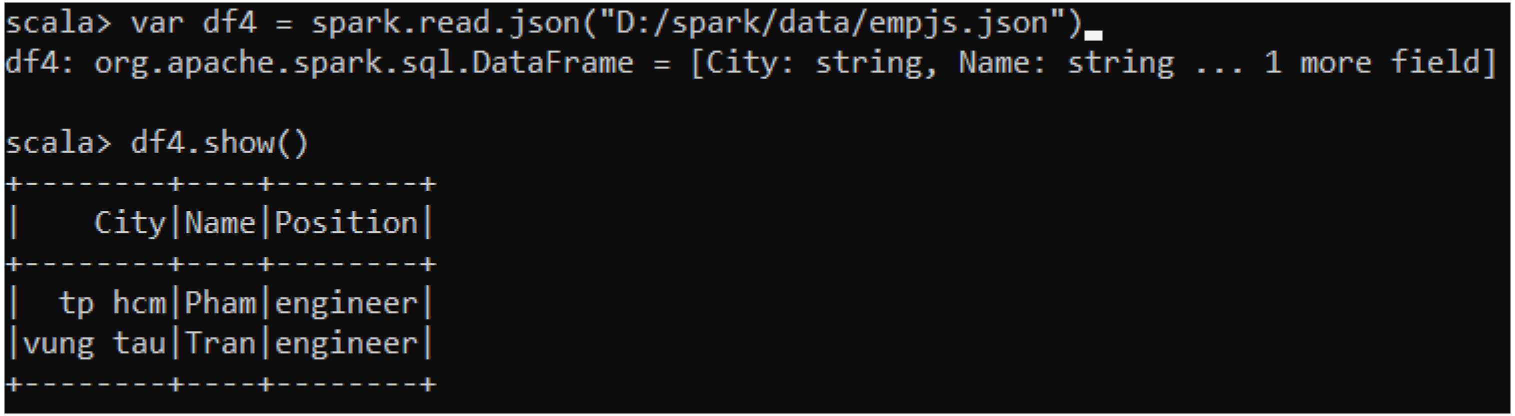


```
Java 200
Python 1000
Scala 30
```



```
scala> var df2 = spark.read.text("D:/spark/data/languages.txt")
df2: org.apache.spark.sql.DataFrame = [value: string]

scala> df2.show()
+-----------+
|      value|
+-----------+
|   Java 200|
|Python 1000|
|   Scala 30|
+-----------+
```

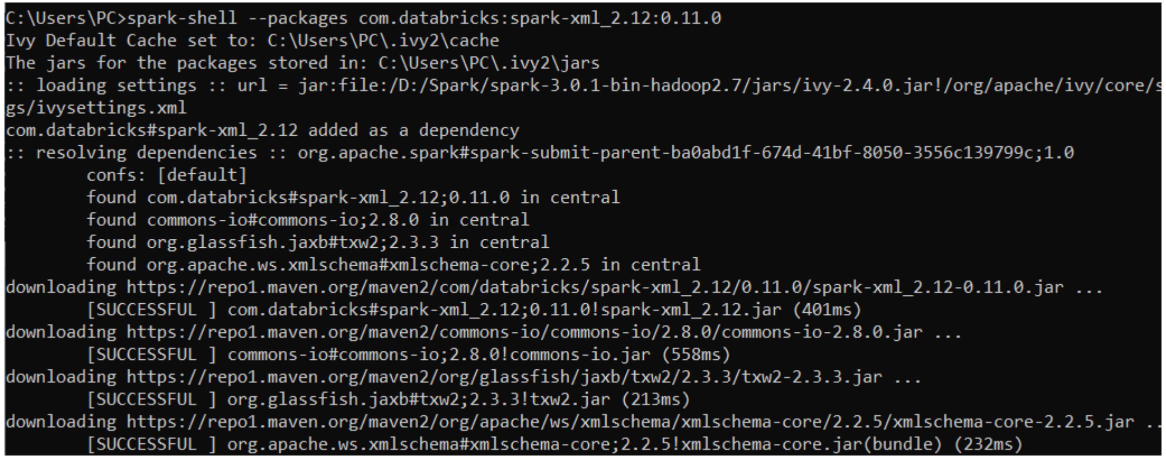
Creating from JSON file



```
scala> var df4 = spark.read.json("D:/spark/data/empjs.json")
df4: org.apache.spark.sql.DataFrame = [City: string, Name: string ... 1 more field]

scala> df4.show()
+--------+----+--------+
|    City|Name|Position|
+--------+----+--------+
|  tp hcm|Pham|engineer|
|vung tau|Tran|engineer|
+--------+----+--------+
```

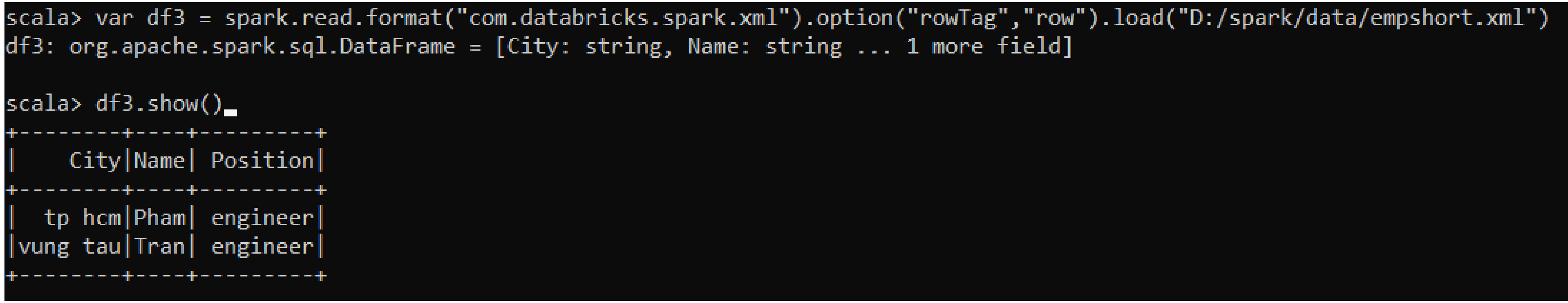
Creating from an XML file

- We use DataSource "com.databricks.spark.xml" spark-xml api from Databricks.
- Command: spark-shell --packages groupId:artifactId:version
- **spark-shell --packages com.databricks:spark-xml_2.12:0.11.0**



```
C:\Users\PC>spark-shell --packages com.databricks:spark-xml_2.12:0.11.0
Ivy Default Cache set to: C:\Users\PC\.ivy2\cache
The jars for the packages stored in: C:\Users\PC\.ivy2\jars
:: loading settings :: url = jar:file:/D:/Spark/spark-3.0.1-bin-hadoop2.7/jars/ivy-2.4.0.jar!/org/apache/ivy/core/s
gs/ivysettings.xml
com.databricks#spark-xml_2.12 added as a dependency
:: resolving dependencies :: org.apache.spark#spark-submit-parent-ba0abd1f-674d-41bf-8050-3556c139799c;1.0
        confs: [default]
        found com.databricks#spark-xml_2.12;0.11.0 in central
        found commons-io#commons-io;2.8.0 in central
        found org.glassfish.jaxb#txw2;2.3.3 in central
        found org.apache.ws.xmlschema#xmlschema-core;2.2.5 in central
downloading https://repo1.maven.org/maven2/com/databricks/spark-xml_2.12/0.11.0/spark-xml_2.12-0.11.0.jar ...
        [SUCCESSFUL ] com.databricks#spark-xml_2.12;0.11.0!spark-xml_2.12.jar (401ms)
downloading https://repo1.maven.org/maven2/commons-io/commons-io/2.8.0/commons-io-2.8.0.jar ...
        [SUCCESSFUL ] commons-io#commons-io;2.8.0!commons-io.jar (558ms)
downloading https://repo1.maven.org/maven2/org/glassfish/jaxb/txw2/2.3.3/txw2-2.3.3.jar ...
        [SUCCESSFUL ] org.glassfish.jaxb#txw2;2.3.3!txw2.jar (213ms)
downloading https://repo1.maven.org/maven2/org/apache/ws/xmlschema/xmlschema-core/2.2.5/xmlschema-core-2.2.5.jar ..
        [SUCCESSFUL ] org.apache.ws.xmlschema#xmlschema-core;2.2.5!xmlschema-core.jar(bundle) (232ms)
```

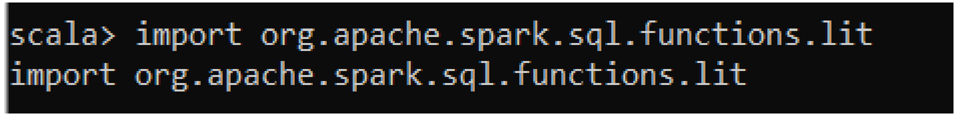
Creating from an XML file



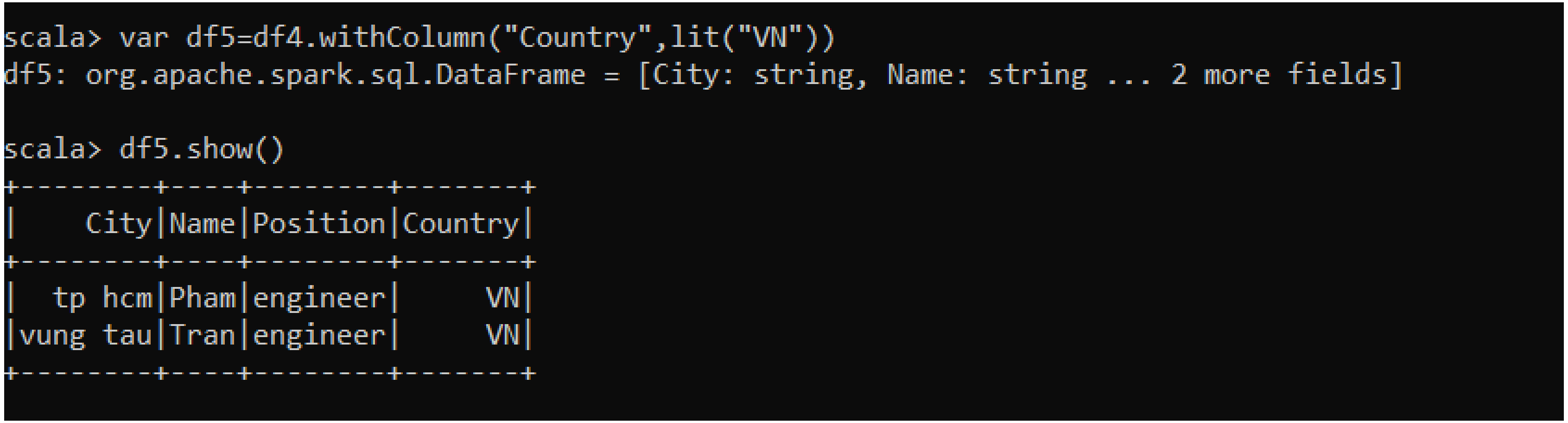
```
scala> var df3 = spark.read.format("com.databricks.spark.xml").option("rowTag","row").load("D:/spark/data/empshort.xml")
df3: org.apache.spark.sql.DataFrame = [City: string, Name: string ... 1 more field]

scala> df3.show()
+--------+----+---------+
|    City|Name| Position|
+--------+----+---------+
|  tp hcm|Pham| engineer|
|vung tau|Tran| engineer|
+--------+----+---------+
```

Working with column in DataFrame

Add new column



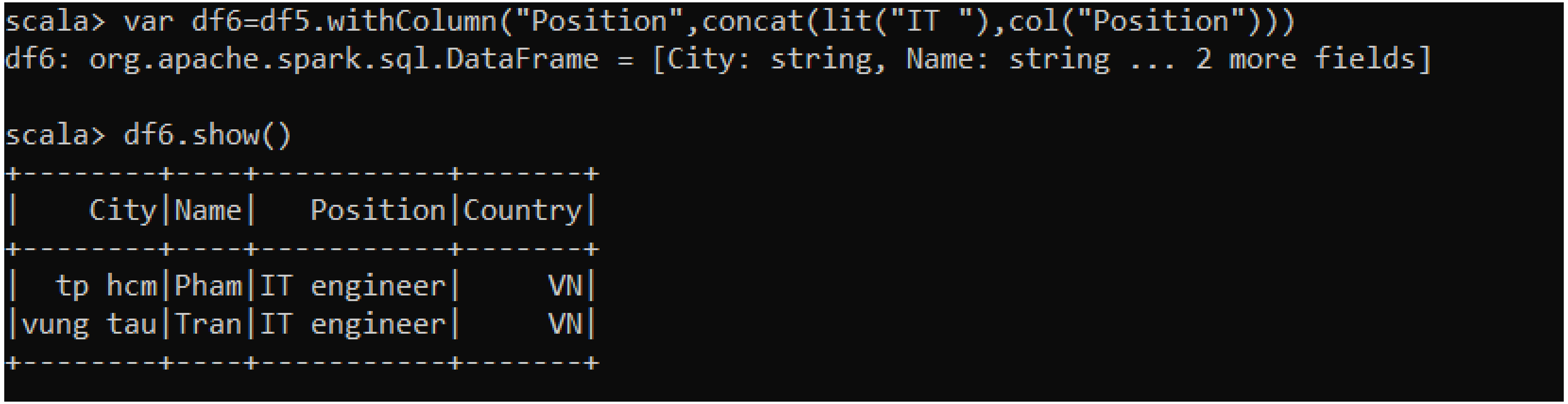
```
scala> import org.apache.spark.sql.functions.lit
import org.apache.spark.sql.functions.lit
```



```
scala> var df5=df4.withColumn("Country",lit("VN"))
df5: org.apache.spark.sql.DataFrame = [City: string, Name: string ... 2 more fields]

scala> df5.show()
+--------+----+--------+-------+
|    City|Name|Position|Country|
+--------+----+--------+-------+
|  tp hcm|Pham|engineer|     VN|
|vung tau|Tran|engineer|     VN|
+--------+----+--------+-------+
```

Working with column in DataFrame

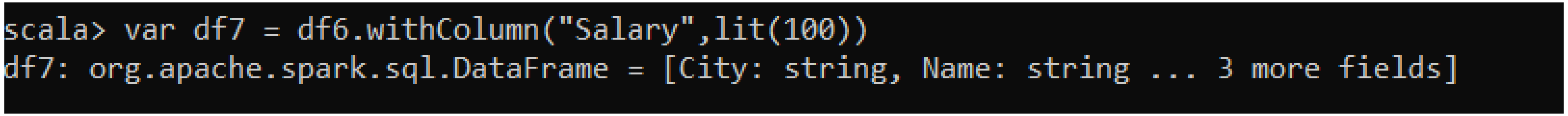
Change Value of an Existing Column



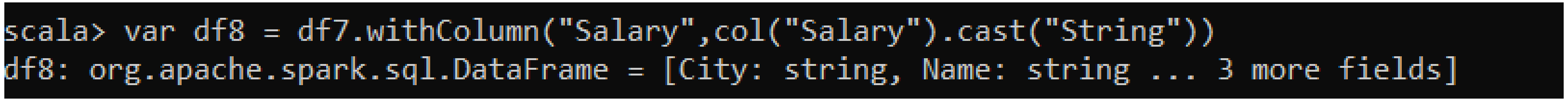
```
scala> var df6=df5.withColumn("Position",concat(lit("IT "),col("Position")))
df6: org.apache.spark.sql.DataFrame = [City: string, Name: string ... 2 more fields]

scala> df6.show()
+--------+----+-----------+-------+
|    City|Name|   Position|Country|
+--------+----+-----------+-------+
|  tp hcm|Pham|IT engineer|     VN|
|vung tau|Tran|IT engineer|     VN|
+--------+----+-----------+-------+
```

Working with column in DataFrame

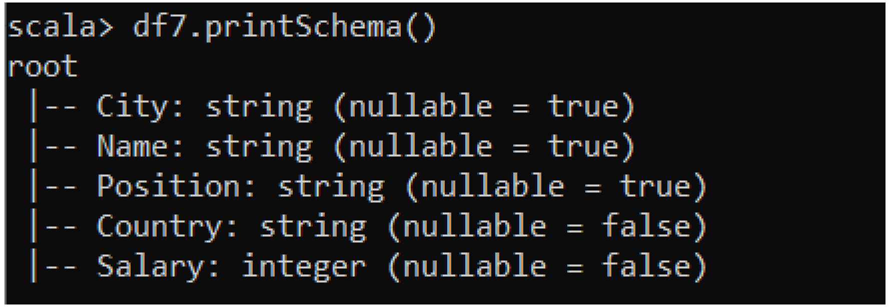
Change Column Data Type



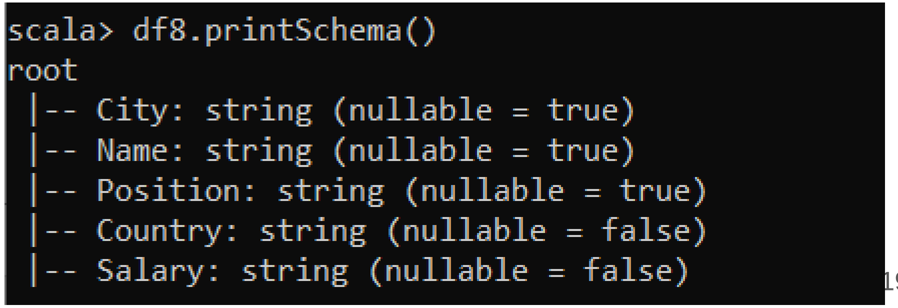
```
scala> var df7 = df6.withColumn("Salary",lit(100))
df7: org.apache.spark.sql.DataFrame = [City: string, Name: string ... 3 more fields]
```



```
scala> var df8 = df7.withColumn("Salary",col("Salary").cast("String"))
df8: org.apache.spark.sql.DataFrame = [City: string, Name: string ... 3 more fields]
```



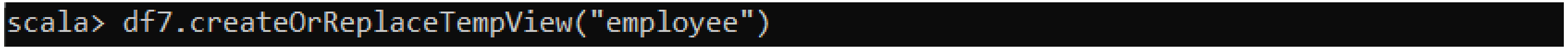
```
scala> df7.printSchema()
root
 |-- City: string (nullable = true)
 |-- Name: string (nullable = true)
 |-- Position: string (nullable = true)
 |-- Country: string (nullable = false)
 |-- Salary: integer (nullable = false)
```



```
scala> df8.printSchema()
root
 |-- City: string (nullable = true)
 |-- Name: string (nullable = true)
 |-- Position: string (nullable = true)
 |-- Country: string (nullable = false)
 |-- Salary: string (nullable = false)
```

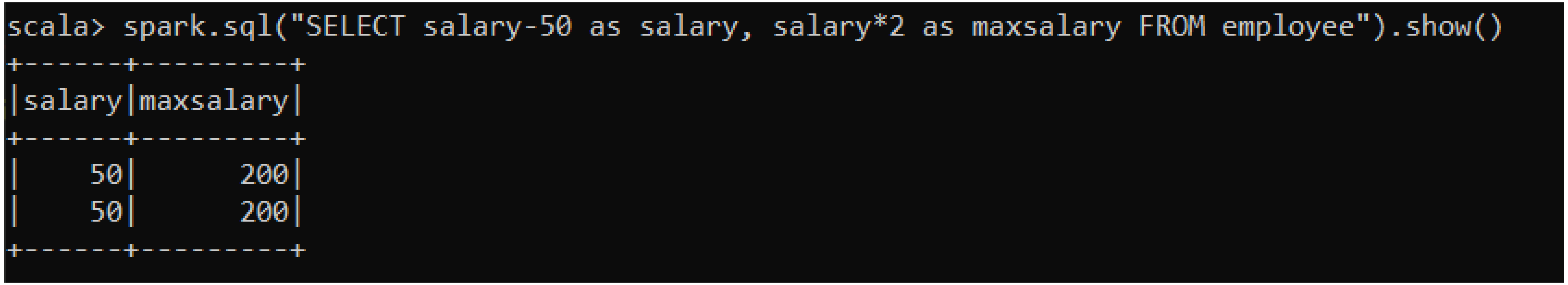
Working with column in DataFrame

Query some columns



```
scala> df7.createOrReplaceTempView("employee")
```

- CreateOrReplaceTempView will create a temporary view of the table on memory it is not presistant at this moment but you can run sql query on top of that .



```
scala> spark.sql("SELECT salary-50 as salary, salary*2 as maxsalary FROM employee").show()
+------+---------+
|salary|maxsalary|
+------+---------+
|    50|      200|
|    50|      200|
+------+---------+
```

Q & A

Cảm ơn đã theo dõi

Chúng tôi hy vọng cùng nhau đi đến thành công.