Apache Spark DataFrames入门指南：创建DataFrame(2)

本系列文章翻译自：《scala data analysis cookbook》第二章：Getting Started with Apache Spark DataFrames。原书是基于Spark 1.4.1编写的，我这里使用的是Spark 1.6.0，丢弃了一些已经标记为遗弃的函数。并且修正了其中的错误。

三、从Scala case class中创建DataFrame

在这篇文章中，你将学到如何从Scala case class中创建DataFrame。

如何做？

1、我们首先创建一个case class，名为Employee，并且定义id和name两个参数，如下：

```
case class Employee(id: Int, name: String)
```

和先前一样，我们分别定义SparkConf、SparkContext以及SQLContext：

```
val conf = new SparkConf().setAppName("colRowDataFrame"). setMaster("local[2]")
val sc = new SparkContext(conf)
val sqlContext = new SQLContext(sc)
```

2、我们可以通过很多方式来初始化Employee类，比如从关系型数据库中获取数据以此来定义Employee类。但是在本文为了简单起见，我将直接定义一个Employee类的List，如下：

```
val listOfEmployees = List(Employee(1, "iteblog"), Employee(2, "Jason"), Employee(3, "Abhi"))
```

3、我们将listOfEmployees列表传递给SQLContext类的createDataFrame 函数，这样我们就可以创建出DataFrame了！然后我们可以调用DataFrame的printuSchema函数，打印出该DataFrame的模式，我们可以看出这个DataFrame主要有两列：name和id，这正是我们定义Employee的两个参数，并且类型都一致。

```
val empFrame = sqlContext.createDataFrame(listOfEmployees)
empFrame.printSchema
root
 |-- id: integer (nullable = false)
 |-- name: string (nullable = true)
```

之所以DataFrame打印出的模式和Employee类的两个参数一致，那是因为DataFrame内部通过反射获取到的。

4、如果你对默认反射获取到的模式名称不感兴趣，你可以通过withColumnRenamed函数来指定列名：

```
val empFrameWithRenamedColumns = sqlContext.createDataFrame(listOfEmployees).withColumnRenamed("id", "empId")
empFrameWithRenamedColumns.printSchema

root
 |-- empId: integer (nullable = false)
 |-- name: string (nullable = true)
```

5、我们可以使用Spark支持的SQL功能来查询相关的数据。在使用这个功能之前，我们必须先对DataFrame注册成一张临时表，我们可以使用registerTempTable函数实现，如下：

```
empFrameWithRenamedColumns.registerTempTable("employeeTable")
```

6、现在我们就可以使用SQL语句来查询DataFrame里面的数据了：

```
val sortedByNameEmployees = sqlContext.sql("select * from employeeTable order by name desc")
sortedByNameEmployees.show()
+-----+-------+
|empId|   name|
+-----+-------+
|    1|iteblog|
|    2|  Jason|
|    3|   Abhi|
+-----+-------+
```

它如何工作的

createDataFrame函数可以接收一切继承scala.Product类的集合对象:

```
def createDataFrame[A <: Product : TypeTag](rdd: RDD[A]): DataFrame
```

而case class类就是继承了Product。我们所熟悉的TupleN类型也是继承了scala.Product类的，所以我们也可以通过TupleN来创建DataFrame：

```
val mobiles=sqlContext.createDataFrame(Seq((1,"Android"), (2, "iPhone"))) mobiles.printSchema mobiles.show()

root
 |-- _1: integer (nullable = false)
 |-- _2: string (nullable = true)

+---+-------+
| _1|     _2|
+---+-------+
|  1|Android|
|  2| iPhone|
+---+-------+
```

我们知道，Tuple2的默认两个参数名字分别是_1和_2,同样，我们如果对这个默认的名字不是特别喜欢，我们也是可以通过withColumnRenamed函数对默认反射的列名进行重命名。