Spark RDD API扩展开发(1)

《Spark RDD API扩展开发(1)》、《Spark RDD API扩展开发(2):自定义RDD》

我们都知道，Apache Spark内置了很多操作数据的API。但是很多时候，当我们在现实中开发应用程序的时候，我们需要解决现实中遇到的问题，而这些问题可能在Spark中没有相应的API提供，这时候，我们就需要通过扩展Spark API来实现我们自己的方法。
我们可以通过两种方法来扩展Spark
API，（1）、
其中一种就是在现有的RDD中添加自定义的方法；（2）、
第二种就是创建属于我们自己的RDD。在这篇文章中，我将对这两种方法进行阐述，并赋予代码。下面我就开始介绍第一种方法。

假如我们中有一些商品的销售数据，数据的格式是CSV的。为了简单起见，假如每行数据都是由id, customerId, itemId 以及itemValue四个字段组成，我们用SalesRecord来表示：

```
class SalesRecord(val id: String,
                  val customerId: String,
                  val itemId: String,
                  val itemValue: Double) extends Comparable[SalesRecord]
with Serializable
```

所以我们可以将商品的销售数据进行解析，并存储到RDD[SalesRecord]中：

```
/**
 * User: 过往记忆
 * Date: 15-03-31
 * Time: 上午00:24
 * bolg: https://www.iteblog.com
 * 本文地址：https://www.iteblog.com/archives/1298
 * 过往记忆博客，专注于hadoop、hive、spark、shark、flume的技术博客，大量的干货
 * 过往记忆博客微信公共帐号：iteblog_hadoop
 */

val sc = new SparkContext(args(0), "iteblogRDDExtending")
val dataRDD = sc.textFile("file:///www/iteblog.csv")
val salesRecordRDD = dataRDD.map(row => {
   val colValues = row.split(",")
   new SalesRecord(colValues(0),colValues(1),
   colValues(2),colValues(3).toDouble)
```

})

如果我们想计算出这些商品的总销售额，我们会这么来写：

```
salesRecordRDD.map(_.itemValue).sum
```

虽然这看起来很简洁，但是理解起来却有点困难。但是如果我们可以这么来写，可能会很好理解：

```
salesRecordRDD.totalSales
```

在上面的代码片段中，totalSales方法让我们感觉就是Spark内置的操作一样，但是Spark是不提供这个方法的，我们需要在现有的RDD中实现我们自定义的操作。

下面我就来介绍一些如何在现有的RDD中添加我们自定义的方法。

一、定义一个工具类，来存放我们所有自定义的操作

当然，你完全没必要自定义一个类类添加我们自定义的方法，但是为了管理，还是建议你这么做。下面我们来定义IteblogCustomFunctions类，它存储所有我们自定义的方法。它是专门用来处理RDD[SalesRecord]，所以这个类中提供的操作全部是用来处理销售数据的：

```
class IteblogCustomFunctions(rdd:RDD[SalesRecord]) {
  def totalSales = rdd.map(_.itemValue).sum
}
```

二、隐形转换来实现在RDD中添加方法

我们定义了隐形的addIteblogCustomFunctions函数，这可以将所有操作销售数据的方法作用于RDD[SalesRecord]上：

```
/**
 * User: 过往记忆
 * Date: 15-03-31
```

```
 * Time: 上午00:24
 * bolg: https://www.iteblog.com
 * 本文地址：https://www.iteblog.com/archives/1298
 * 过往记忆博客，专注于hadoop、hive、spark、shark、flume的技术博客，大量的干货
 * 过往记忆博客微信公共帐号：iteblog_hadoop
 */

object IteblogCustomFunctions {
  implicit def addIteblogCustomFunctions(rdd: RDD[SalesRecord]) = new
  IteblogCustomFunctions(rdd)
}
```

三、使用自定义的方法

下面方法通过导入IteblogCustomFunctions 中的相应方法来实现使用我们自定义的方法：

```
import IteblogCustomFunctions._
println(salesRecordRDD.totalSales)
```

通过上面三步我们就可以在现有的RDD中添加我们自定义的方法。

已经已经很晚了，明天我将介绍如何自定义RDD，欢迎关注。