**Maven依赖项**

<dependency>

<groupId>org.apache.spark</groupId>

<artifactId>spark-core\_2.11</artifactId>

<version>2.0.0</version>

</dependency>

<dependency>

<groupId>org.apache.spark</groupId>

<artifactId>spark-mllib\_2.11</artifactId>

<version>2.0.0</version>

</dependency>

数据类型：



**Spark切入点，创建SparkSession**

SparkSession sparkSession = SparkSession.builder().master("local").appName("CollaborativeFiltering").getOrCreate();

读取文件：

**JavaRDD类型：**

读取文本文件，RDD存储Person类，但RDD本身不了解Person类的内部结构

JavaRDD<String> logData = sparkSession.read().textFile(/user/fanyuguang/test/test.txt).javaRDD();

对读取的文件进行处理

JavaRDD<Rating> ratings = rowDataset.map(

new Function<String, Rating>() {

public Rating call(String s) throws Exception {

String[] sarray = s.trim().split("\\s+");

return new Rating(Integer.parseInt(sarray[0]), Integer.parseInt(sarray[1]), Double.parseDouble(sarray[2]));

}

}

);

遍历并打印JavaRDD类型

for (Rating r : ratings.collect()) {

System.out.println(r);

}

**Dataset<Row>类型：**

读取Json文件，可以对Dataset<Row>类型应用Spark SQL进行数据表操作

Dataset<Row> 存储Person时，存储详细的结构信息，类似于数据表。

Dataset<Row> df = sparkSession.read().json("/user/fanyuguang/test/people.json");

df.show()

df.select("name").show();

df.filter(col("age").gt(21)).show();

df.groupBy("age").count().show();

df.createOrReplaceTempView(“people”);

Dataset<Row> sqlDF = spark.sql(“SELECT \* FROM people”);

sqlDF.show();