第4章Spark编程进阶实训

CH04实训1：竞赛网站访问日志分析

训练要点

1. 搭建Spark工程环境。
2. Spark编程。
3. 通过spark-submit提交应用。

需求说明

某竞赛网站每年都会开展数据挖据的竞赛，在竞赛期间网站会有大量人群访问，生成了大量的用户访向记录。现在提供2016年10月到2017年6月的部分脱敏访问日志数据文件:jc_content_viewlog.txt。日志数据的基本内容如图所示，仅提供以下6个字段：

属性名称	属性解析
Id	序号
Content_id	网页ID
Page_path	网址
Userid	用户ID
Sessionid	缓存生成ID
Date_time	访问时间

要求根据提供的用户访问日志数据，利用Spark技术统计访问的用户数、被访问的不同网页个数以及每月的访问量，并将结果保存到HDFS上。

实现思路及步骤

1. 配置好Spark的IntelliJ IDEA开发环境。
2. 启动IntelliJ IDEA，并进行Spark编程。
3. 读取文件：/user/myname/jc_content_viewlog.txt，对访向记录中的网页去重，统计本周期内被访问网页的个数。
4. userid为用户注册登录的标识，对userid去重，统计登录用户的数量。
5. 按月统计访问记录数。
6. 将结果保存到三个文件中：
   1）/user/myname/out_wy_count
   2）/user/myname/out_user_count
   3）/user/myname/out_nycount。
7. 打包Spark工程，在集群使用spark-submit提交应用程序：spark-submit --master spark://master:7077

实训作业要求

1，截图IDEA上的源码

2，截图crt上运行的结果

3，截图在hdfs上的运行结果

实现参考

准备数据

```
cd /root/spark
wget https://biglab.site//b59510spark/file/jc_content_viewlog.tar.gz
tar -xzvf ./jc_content_viewlog.tar.gz
hdfs dfs -rm /user/myname/jc_content_viewlog.txt
hdfs dfs -put ./jc_content_viewlog.txt /user/myname/
```

编码参考

```scala
package chap04t

import org.apache.spark.rdd.RDD
import org.apache.spark.{SparkConf, SparkContext}
object LogCount {
  def main(args: Array[String]): Unit = {

    if (args.length < 2) {
      println("请指定input和output")
      System.exit(1) //非0表示非正常退出程序
    }

    //TODO 1.env/准备sc/SparkContext/Spark上下文执行环境
    val conf: SparkConf = new
SparkConf().setAppName("wc").setMaster("local")
    val sc: SparkContext = new SparkContext(conf)
    sc.setLogLevel("WARN")


    //TODO 2.source/读取数据
    //RDD:A Resilient Distributed Dataset (RDD):弹性分布式数据集,简单理解为分布式
集合!使用起来和普通集合一样简单!
    //RDD[就是一行行的数据]
    val logs_all: RDD[Array[String]] = sc.textFile(args(0)).map {
      _.split(",")
    }
    //TODO 3.transformation/数据操作/转换
    //对访问记录中的网页去重，统计本周期内被访问网页的个数
    //
478896,1043,/jszx/1043.jhtml,14884,F6D362B9AFAC436D153B7084EF3BA332,2017-03-
01 00:23:07
    val wy_log: RDD[String] = logs_all.map(x => (x(1).toString)).distinct()
    val wy_count: RDD[(String, Int)] = wy_log.map(("wy_zs",
_)).groupByKey().map(x => (x._1, x._2.size))
    //userid为用户注册登录的标识，对userid去重，统计登录用户的数量
    val user_log: RDD[String] = logs_all.map(x =>
(x(3).toString)).distinct()
    val user_count: RDD[(String, Int)] = user_log.map(("user_zs",
_)).groupByKey().map(x => (x._1, x._2.size))
    //按月统计访问记录数
    val logs_all_new = logs_all.map { x => (x(0), x(1), x(2), x(3), x(4),
x(5), date_time(x(5))) }
    val ny_count: RDD[(String, Int)] = logs_all_new.map(x => (x._7,
1)).reduceByKey((a, b) => a + b)

    //TODO 4.sink/输出
    //输出到指定path(可以是文件/夹)
    println("be writing to :" + args(1))
    wy_count.repartition(1).saveAsTextFile(args(1))
    println("be writing to :" + args(2))
    user_count.repartition(1).saveAsTextFile(args(2))
```

```
    println("be writing to :" + args(3))
    ny_count.repartition(1).saveAsTextFile(args(3))
    //为了便于查看Web-UI可以让程序睡一会
    Thread.sleep(1000 * 60)

    //TODO 5.关闭资源
    sc.stop()
  }

  //获取年月，时间段作为输入参数
  def date_time(date: String): String = {
    if (date.trim.length == "2017-03-01 00:23:07".length) {
      val nianye = date.trim.substring(0, 7)
      nianye
    }
    else
      "1900-01"
  }

}

```

打包上传

1. 在项目菜单中编译生成artifact: 选择菜单栏中的“Build->Build Artifacts->word->Build,命令
2. 将生成的word.jar上传到集群slave1的/root/spark目录下

集群执行

```
cd /root/spark
hdfs dfs -rm -r /user/myname/out_wy_count
hdfs dfs -rm -r /user/myname/out_user_count
hdfs dfs -rm -r /user/myname/out_ny_count
spark-submit --master spark://master:7077 --class chap04t.LogCount \
/root/spark/word.jar \
/user/myname/jc_content_viewlog.txt \
/user/myname/out_wy_count \
/user/myname/out_user_count \
/user/myname/out_ny_count
```

注意：spark-submit后连续6行是要合在一起执行，它是一条命令

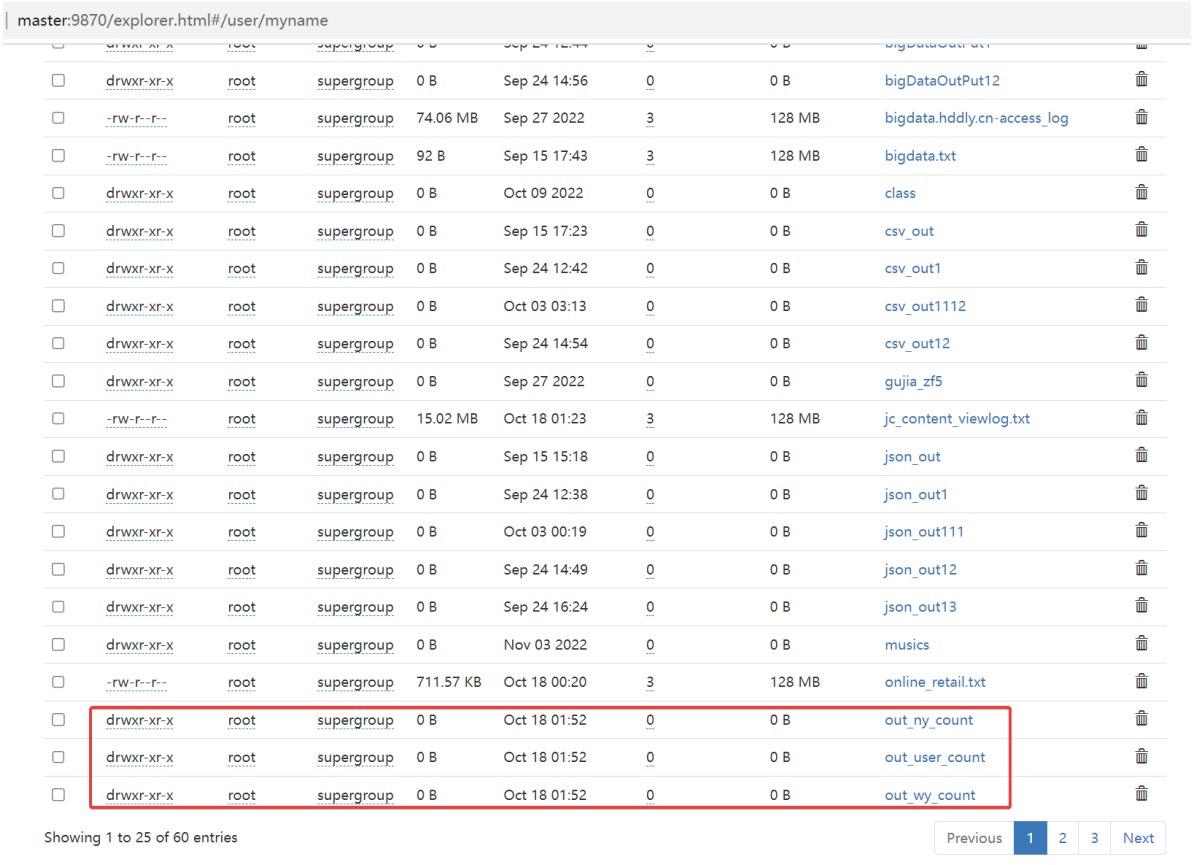
预期结果

```
[root@slave1 spark]# cd /root/spark
[root@slave1 spark]# hdfs dfs -rm -r /user/myname/out_wy_count
hdfs dfs -rm -r /user/myname/out_user_count
hdfs dfs -rm -r /user/myname/out_ny_count
spark-submit --master spark://master:7077 --class chap04t.LogCount \
/root/spark/word.jar \
/user/myname/jc_content_viewlog.txt \
/user/myname/out_wy_count \
/user/myname/out_user_count \
/user/myname/out_ny_count
```

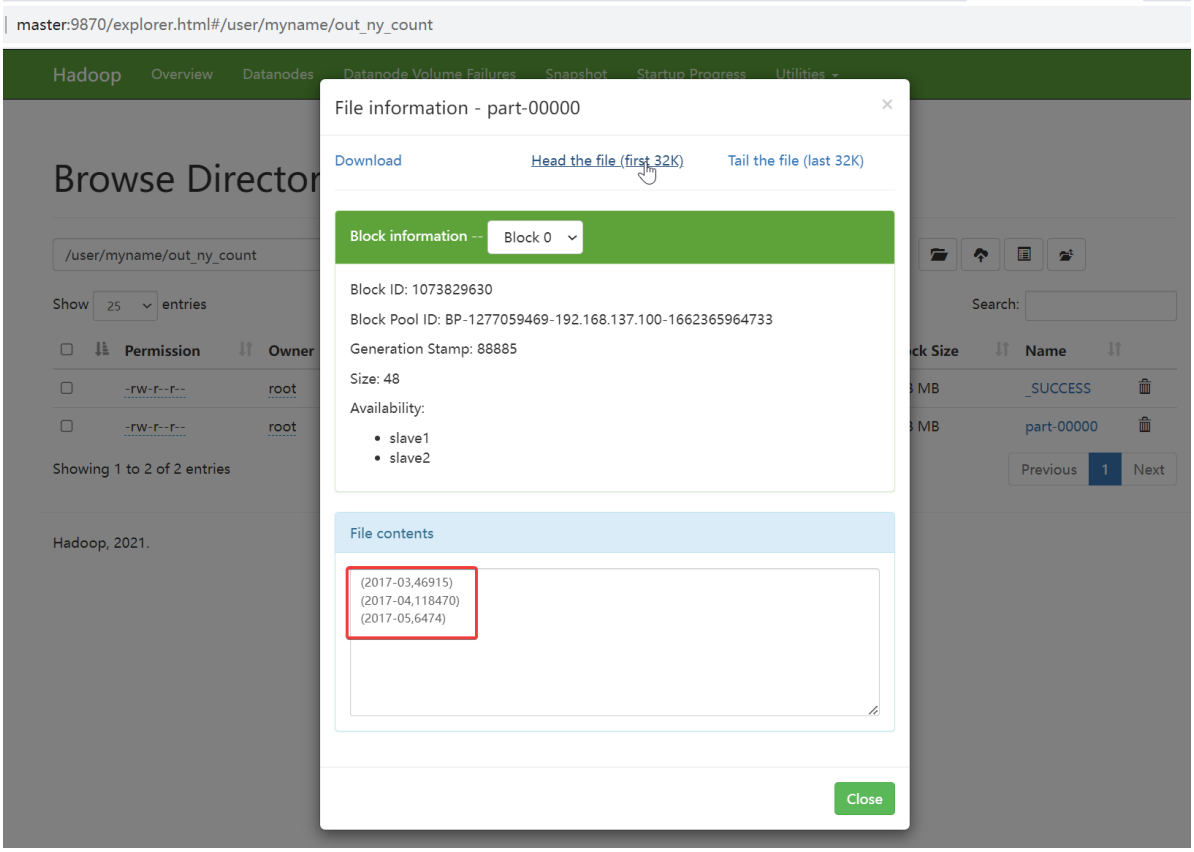
```
Deleted /user/myname/out_wy_count
[root@slave1 spark]# hdfs dfs -rm -r /user/myname/out_user_count
Deleted /user/myname/out_user_count
[root@slave1 spark]# hdfs dfs -rm -r /user/myname/out_ny_count
Deleted /user/myname/out_ny_count
[root@slave1 spark]# spark-submit --master spark://master:7077 --class chap04t.LogCount \
> /root/spark/word.jar \
> /user/myname/jc_content_viewlog.txt \
> /user/myname/out_wy_count \
> /user/myname/out_user_count \
> /user/myname/out_ny_count
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/spark/jars/slf4j-log4j12-1.7.30.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop-3.3.1/share/hadoop/common/lib/slf4j-log4j12-1.7.30.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
be writing to :/user/myname/out_wy_count
be writing to :/user/myname/out_user_count
be writing to :/user/myname/out_ny_count

[root@slave1 spark]#
```

HDFS上结果

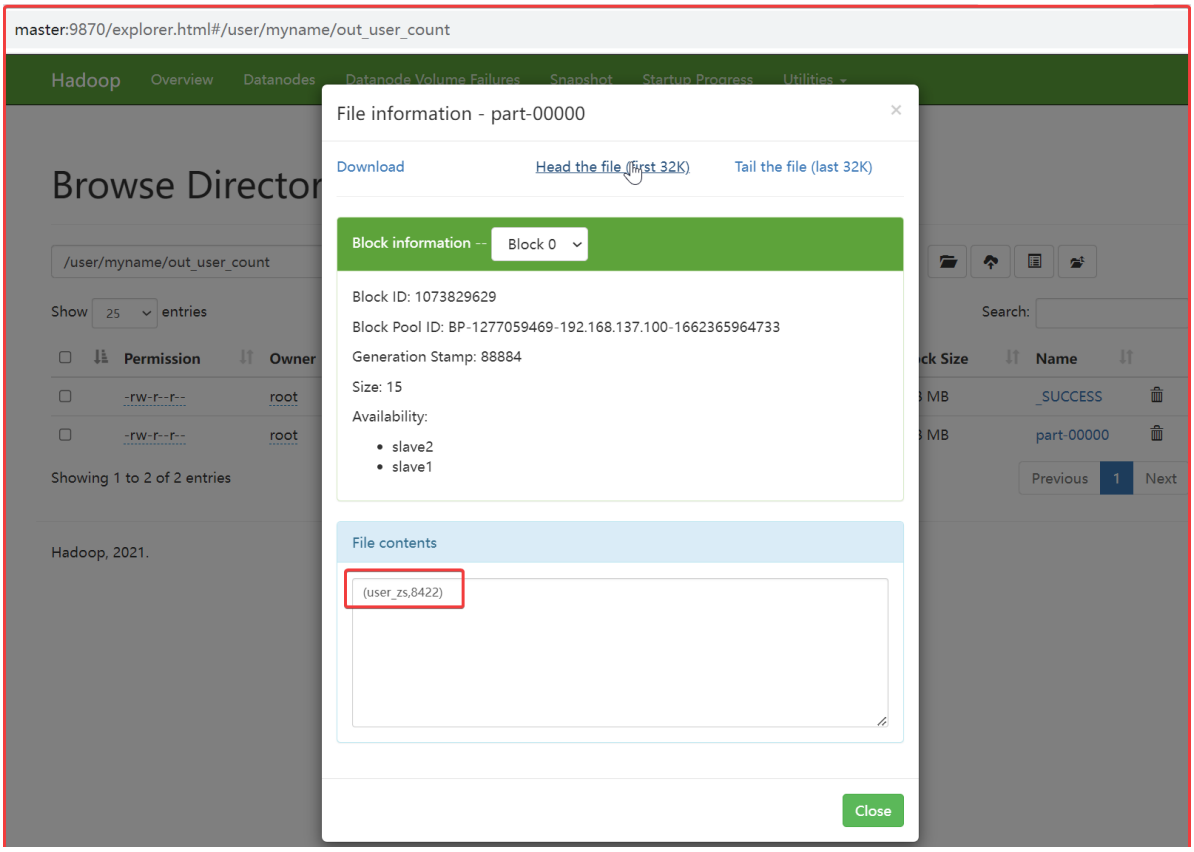
结果有三个输出文件：



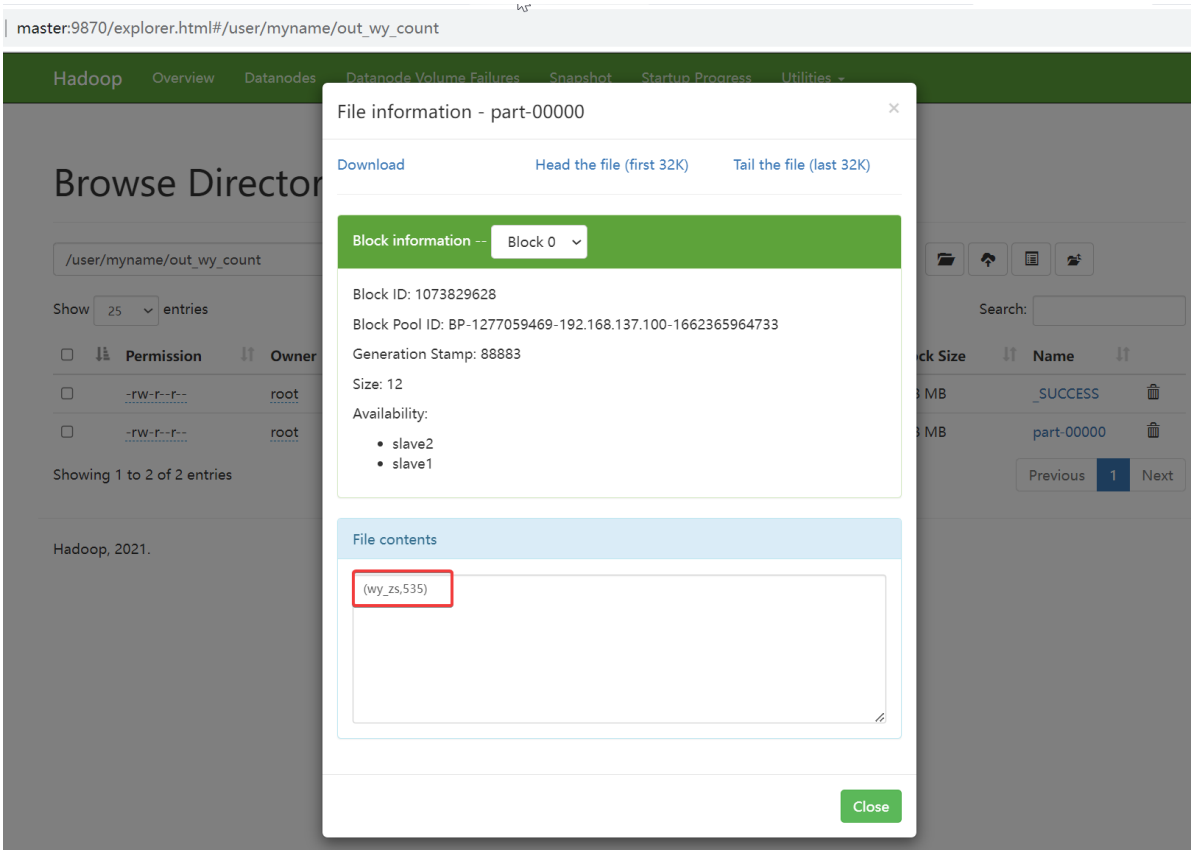
其中out_ny_count:



其中out_user_count:



其中out_wy_count:



视频学习

视频学习：

1. Hive安装和配置视频
2. 源码下载
3. 竞赛网站访问日志分析视频 补充视频2
4. 如何减小hive和spark-shell的日志输出
5. http://home.hddly.cn:50090//media/2023-11-03%2017.24.43%20biglab%209562804926/meeting_01.mp4

CH04实训2：自定义分区器实现按人物标签进行数据分析

训练要点

1. 掌握使用IntelliJ IDEA搭建spark开发环境
2. 掌握创建spark工程
3. 掌握使用spark自定义分区
4. 掌握打包spark工程
5. 掌握通过spark-submit提交应用

需求说明

中国女排曾多次夺得奥运会冠军，团结协作、顽强拼搏的女排精神始终代代相传，极大地激发了中国人的自豪之情、提高了自尊和增加了自信，为我们奋进在新征程上提供了强大的精神力量。现有一份某年度中国女排（包括国家女子排球队和国家青年女子排球队）的集训运动员数据文件Volleyball-Players.csv，包含4个数据字段，数据字段说明如表 所示排球运动员按所属位置可分为5类，分别是主攻、接应、二传、副攻和自由人。为了解中国女排各运动员的所属位置，现要求在IntelliJ IDEA中进行spark编程，通过自定义分区实现将集训运动员数据按运动员所属位置进行分区，并将程序打包，通过spark-submit提交应用。设置5个分区，第1个分区保存位置标签为“主攻"的运动员数据，第2个分区保存

位置标签为“接应"的运动员数据，第3个分区保存位置标签为“二传" 的运动员数据，第4个分区保存位置标签为“副攻"的运动员数据，第5个分区保存位置标签为“自由人"的运动员数据，将分区结果输出到HDFS上。

表:中国女排集训运动员数据字段说明

字段名称	说明
Name	中国女排集训运动员名称
Birthday	出生日期
Height	身咼，单位为cm（厘米）
Shot_Height	扣球高度(cm)
Block_Height	拦网高度(cm)
Position	位置，分为5类：主攻、接应、二传、副攻和自由人
Province	位置,省、市

Volleyball-Players.csv文件内容如：

姓名,出生日期,身高(cm),扣球高度(cm),拦网高度(cm),位置,省、市

刘晓彤,1990.02.16,187,315,300,主攻,北京

曾春蕾,1989.03.11,187,315,312,接应,北京

实现思路及步骤

1. 配置好IntelliJ IDEA和Spark开发环境，并启动IntelliJ IDEA
2. 使用textFile()方法读取Volleyball-Players.csv数据以创建RDD，并设置分区数为5
3. 使用map()方法将读取的数据按“，"（逗号）进行分割，筛选出"Position"和"Name" 字段，并转化成"(Position,Name)"的形式。
4. 自定义Mypartitioner，继承自定义partitioner类，重写partitioner类的numpartitions 和getpartition()方法，实现自定义分区。
5. 在主函数中调用自定义分区类Mypartitioner
6. 打包spark工程，并将应用程序提交至集群运行。

实训作业要求

1，截图IDEA上的源码

2，截图crt上运行的结果

3，截图在hdfs上的运行结果

实现参考

准备数据

在slave1下执行

```
rm -f spark_t_data.tar.gz*
wget https://biglab.site//b59510spark/file/spark_t_data.tar.gz
tar -xzvf ./spark_t_data.tar.gz
hdfs dfs -rm /user/myname/Volleyball_Players.csv
hdfs dfs -rm -r /user/myname/output_volleyball
hdfs dfs -put ./spark_t_data/Volleyball_Players.csv /user/myname/
```

编码参考

```
package chap04t

import org.apache.spark.{Partitioner, SparkConf, SparkContext}
import org.apache.spark.rdd.RDD
object Partition {
  def main(args: Array[String]): Unit = {
    if (args.length < 2) {
      println("请指定input和output")
      System.exit(1) //非0表示非正常退出程序
    }
    val sparkConf = new
SparkConf().setMaster("local").setAppName("Players_Partition")
    val sc = new SparkContext(sparkConf)
    val input: RDD[String] = sc.textFile(args(0),5)
//    args(0) 或  "hdfs://master:9864/user/myname/Volleyball_Players.csv"
    val rdd: RDD[(String, String)] = input.map(x => {
      val data = x.split(",")
      (data(5), data(0))
    })
    rdd.foreach(println)
    val partRDD: RDD[(String, String)] = rdd.partitionBy(new MyPartitioner)
    partRDD.saveAsTextFile(args(1))
//    args(1)  或 "hdfs://master:9864/user/myname/output_volleyball"
    sc.stop()
  }

  /**
    * 自定义分区器
    * 1.继承Partition
    * 2.重写方法
    *
    */
  class MyPartitioner extends Partitioner{
    //分区数量
    override def numPartitions: Int = 5
    ///  根据数据的key值返回数据所在的分区索引（从0开始）
    override def getPartition(key: Any): Int = {

      key match{
        case "主攻" =>0
        case "接应" =>1
```

```
        case "二传" =>2
        case "副攻" =>3
        case _ =>4
      }
    }
  }

}

```

打包上传

1. 在项目菜单中编译生成artifact: 选择菜单栏中的“Build->Build Artifacts->word->Build,命令
2. 将生成的word.jar上传到集群slave1的/root/spark目录下

集群执行

```
cd /root/spark
hdfs dfs -rm -r /user/myname/output_volleyball
spark-submit --master spark://master:7077 --class chap04t.Partition \
/root/spark/word.jar \
/user/myname/Volleyball_Players.csv \
/user/myname/output_volleyball
```

注意：spark-submit后连续4行是要合在一起执行，它是一条命令

预期结果

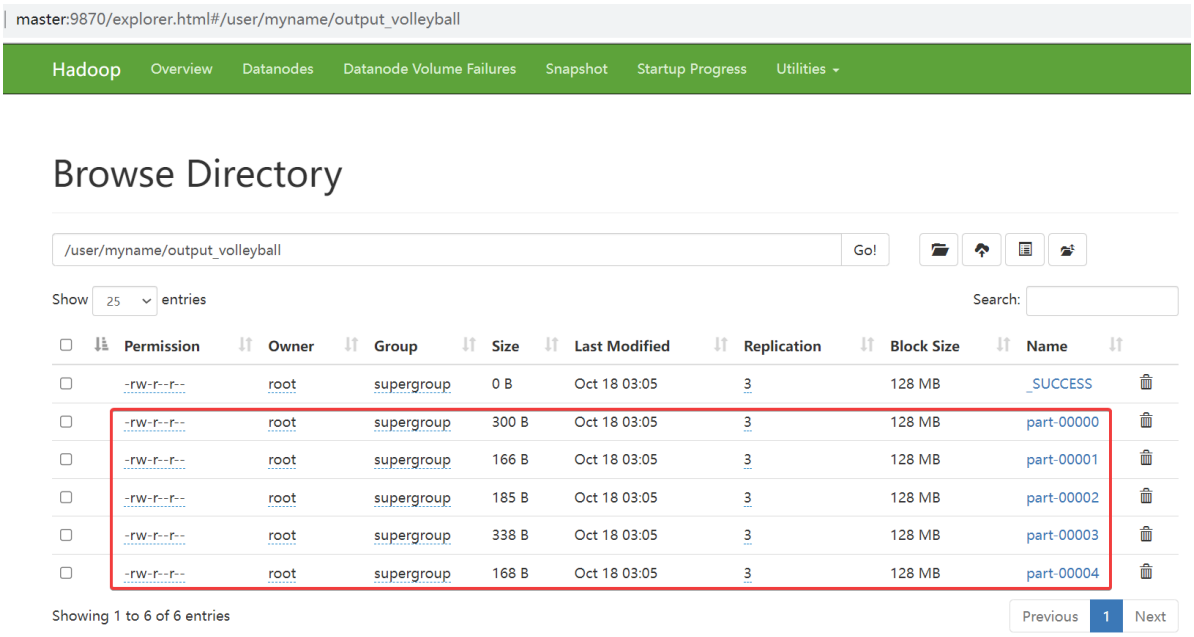
```
[root@slave1 spark]# cd /root/spark
[root@slave1 spark]# hdfs dfs -rm -r /user/myname/output_volleyball
spark-submit --master spark://master:7077 --class chap04t.Partition \
/root/spark/word.jar \
/user/myname/Volleyball_Players.csv \
/user/myname/output_volleyball
Deleted /user/myname/output_volleyball
[root@slave1 spark]# spark-submit --master spark://master:7077 --class
chap04t.Partition \
> /root/spark/word.jar \
> /user/myname/Volleyball_Players.csv \
> /user/myname/output_volleyball
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/spark/jars/slf4j-log4j12-
1.7.30.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop-
3.3.1/share/hadoop/common/lib/slf4j-log4j12-
1.7.30.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an
explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
(位置,姓名)
(主攻,刘晓彤)
(接应,曾春蕾)
(二传,姚迪)
(主攻,李盈莹)
```

(副攻,王媛媛)
(自由人,孟子旋)
(二传,丁 霞)
(副攻,颜 妮)
(副攻,胡铭媛)
(二传,孙海平)
(主攻,段 放)
(主攻,张轶婵)
(主攻,张常宁)
(接应,龚翔宇)
(二传,刁琳宇)
(自由人,倪非凡)
(接应,吴 晗)
(自由人,林 莉)
(副攻,郑益昕)
(自由人,王梦洁)
(副攻,杨涵玉)
(二传,蔡雅倩)
(主攻,杜清清)
(主攻,朱 婷)
(副攻,袁心玥)
(主攻,刘晏含)
(副攻,高 意)
(副攻,李学林)
(主攻,张景胤)
(二传,陈磊炀)
(副攻,缪阮彤)
(自由人,陈嘉杰)
(副攻,彭世坤)
(接应,陈禧龙)
(主攻,郭顺祥)
(二传,毛天一)
(主攻,袁党毅)
(接应,修成城)
(接应,唐川航)
(副攻,杜昊昱)
(自由人,马晓腾)
(接应,江川)
(主攻,张秉龙)
(副攻,王东宸)
(主攻,刘力宾)
(副攻,谷佳丰)
(副攻,陈龙海)
(二传,詹国俊)
(副攻,张哲嘉)
(主攻,戴卿尧)
(自由人,童嘉骅)
(二传,于垚辰)
(接应,刘向东)
(副攻,戴海波)
(副攻,饶书涵)
(主攻,季道帅)
(接应,王径一)
(副攻,张祖源)
(二传,李润铭)

```
77 (主攻,付侯文)
78 [root@slave1 spark]#
```

HDFS上结果

结果有一个输出目录和五个文件：/user/myname/output_volleyball



其中文件之一：part_00000