五、综合题

1、 使用Spark编程对文件E:\\hello.txt中的单词进行统计，完成程序中的代码。（中）

import org.apache.spark.rdd.RDD

import org.apache.spark.{SparkConf, SparkContext}

object SparkWordCountWithScala {

def main(args: Array[String]): Unit = {

val conf = new SparkConf()

//设置运行模式为local

1.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//设置任务名

conf.setAppName("WordCount")

试卷部分：4/4

//创建SparkCore的程序入口

val sc = new SparkContext(conf)

//读取文件生成RDD

val file: RDD[String] = sc.textFile("E:\\hello.txt")

//把每一行数据按照“,”分割

2. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//让每一个单词都出现一次

3.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//单词计数

4.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//按照单词出现的次数降序排序

5.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//将最终的结果进行保存

sortRdd.saveAsTextFile("E:\\result")

sc.stop()

}

答：1.val sc = new SparkContext(conf) （2分）

2.val word: RDD[String] = file.flatMap(\_.split(",")) （2分）

3.val wordOne: RDD[(String, Int)] = word.map((\_,1)) （2分）

4.val wordCount: RDD[(String, Int)] = wordOne.reduceByKey(\_+\_) （2分）

5.val sortRdd: RDD[(String, Int)] = wordCount.sortBy(tuple => tuple.\_2,false) （2分）

2. 编写Scala程序，实现以下功能：（中）

提示：假设列表是var list = List(1,3,2,5,4,7,8,6,9,0)  
 (1) 获取列表中的前5个元素。  
 (2) 判断列表中是否包含元素0。

(3) 获取列表中的最大元素。  
 (4) 获取列表中元素的个数。

(5) 获取列表中的第一个元素。

答：object ListTest {

def main(args: Array[String]): Unit = {

var list = List(1,3,2,5,4,7,8,6,9,0) （2分）

System.out.println(list.take(5)) （2分）

System.out.println(list.contains(0)) （2分）

System.out.println(list.max) （2分）

System.out.println(list.length) （2分）

System.out.println(list.head) （2分）

}

}

3. 编写Scala程序，计算100~999的所有的水仙花数。（中）  
 提示：这里水仙花数指严格意义上的水仙花数，即若一个数满足这个数等于它的百位数、十位数、个位数的立方和，那么这个数就是水仙花数。

答：object ShuiXianhua {

def main(args: Array[String]): Unit = {

for (i<-100 to 999 ){ （2分）

val a= i % 10 （2分）

val b= i % 100 / 10 （2分）

val c = i / 100 （2分）

if (a \* a \* a + b \* b \* b + c \* c \* c == i) { （2分）

System.out.println(i);

}

}

}

}

4. 编写Spark程序，实现以下操作：（中）  
(1)通过Spark SQL读取关系数据库MySQL中的数据；  
(2)通过Spark SQL往关系数据库MySQL中插人数据。

答：（1）spark.read.format("jdbc") (1分)

.option("url", "jdbc:mysql://node01:3306/spark\_test") (1分)

.option("dbtable", "student") (1分)

.option("user", "root") (1分)

.option("password", "123456") (1分)

.load()

.show()

1. val studentDF = spark.read .option("delimiter", "\t") .schema(schema)

.csv("dataset/studenttab10k") (1分)

studentDF.write.format("jdbc")

.mode(SaveMode.Overwrite) (1分)

.option("url", "jdbc:mysql://node01:3306/spark\_test") (1分)

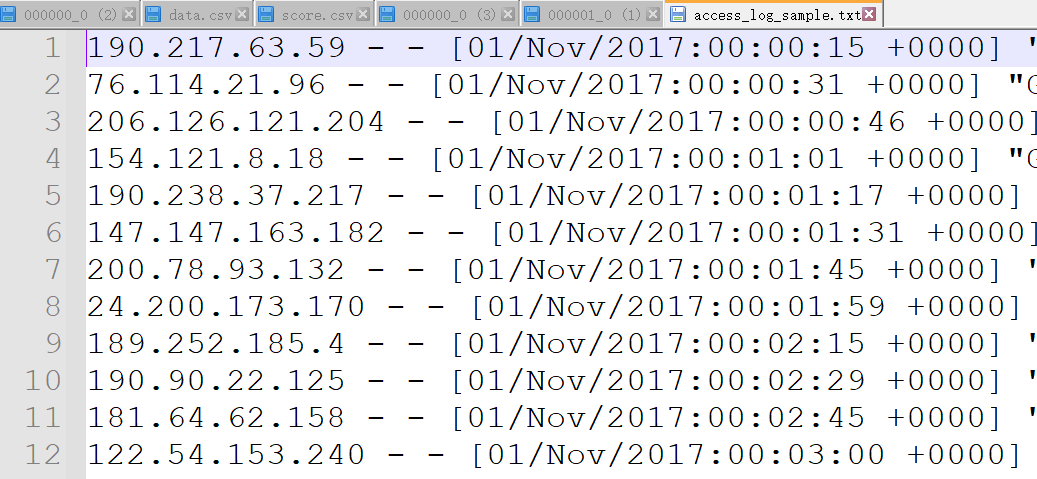
.option("dbtable", "student") (1分)

.option("user", "root")

.option("password", "123456") (1分)

.save()

5、 给定一个网站的访问记录, 俗称 Access log，请编写程序计算其中出现的独立 IP, 以及其访问的次数？日志文件名称是access\_log\_sample.txt。（中）



答:

val config = new SparkConf()

.setAppName("ip\_ana")

.setMaster("local[6]") (2分)

val sc = new SparkContext(config)

val result = sc.textFile("dataset/access\_log\_sample.txt")

.map(item => (item.split(" ")(0), 1)) (2分)

.filter(item => StringUtils.isNotBlank(item.\_1)) (2分)

.reduceByKey((curr, agg) => curr + agg) (2分)

.sortBy(item => item.\_2, false)

.take(10)

result.foreach(item => println(item)) (2分)

6、Mapreduce和Spark的都是并行计算，那么他们有什么相同和区别？ （中）

答：

1）两者都是用mr模型来进行并行计算。 （2分）

2）hadoop的⼀个作业称为job，job里面分为map task和reduce task，每个task都是在自己的进程中运行的，当task结束时，进程也会结束。 （2分）

3)spark用户提交的任务成为application，⼀个application对应⼀个sparkcontext，app中存在多个job，每触发⼀次action操作就会产生⼀个job。这些job可以并行或串行执行，每个job中有多个stage，stage是shuffle过程中DAGSchaduler通过RDD之间的依赖关系划分job⽽来的，每个stage里面有多个task，组成taskset有TaskSchaduler分发到各个executor中执行，executor 的生命周期是和app⼀样的，即使没有job运行也是存在的，所以task可以快速启动读取内存进行计算。 （2分）

4)hadoop的job只有map和reduce操作，表达能⼒⽐较欠缺⽽且在mr过程中会重复的读写hdfs，造成⼤量的io操作，多个job需 要自己管理关系。 （2分）

5）spark的迭代计算都是在内存中进行的，API中提供了⼤量的RDD操作如join，groupby等，⽽且通过DAG图可以实现良好的容错。 （2分）

7. 请描述Spark分布式集群搭建的步骤（中）

1）准备linux环境，设置集群搭建账号和用户组，设置ssh，关闭防⽕墙，关闭seLinux，配置host，hostname （2分）

2）配置jdk到环境变量 （2分）

3）搭建hadoop集群，如果要做master ha，需要搭建zookeeper集群修改hdfs-site.xml,hadoop\_env.sh,yarn-site.xml,slaves等配置⽂件 （2分）

4）启动hadoop集群，启动前要格式化namenode （1分）

5）配置spark集群，修改spark-env.xml，slaves等配置⽂件，拷贝hadoop相关配置到spark conf目录下 （2分）

6)启动spark集群。（1分）

8. hadoop和spark的shuffle相同和差异有哪？（中）

答：1）从 high-level 的角度来看，两者并没有⼤的差别。都是将 mapper（

Spark 里是 ShuffleMapTask）的输出进行 partition，不同的 partition 送到不同的 reducer（Spark 里 reducer 可能是下⼀个 stage 里的 ShuffleMapTask，也可能是 ResultTask）。Reducer 以内存作缓冲区，边 shuffle 边 aggregate 数据，等到数据 aggregate 好以后进行 reduce() （Spark 里可能是后续的⼀系列操作）。 （4分）

1. 从 low-level 的角度来看，两者差别不⼩。 Hadoop MapReduce 是 sort-based，进⼊ combine() 和 reduce() 的 records 必须先 sort。这样的好处在于 combine/reduce() 可以处理⼤规模的数据，因为其输⼊数据可以通过外排得到（mapper 对每段数据先做排序，reducer 的 shuffle 对排好序的每段数据做归并）。（3分）
2. 目前的 Spark 默认选择的是 hash-based，通常使用HashMap 来对 shuffle 来的数据进行 aggregate，不会对数据进行提前排序。 （3分）

9.Spark技术栈有哪些组件，每个组件都有什么功能，适合什么应用场景？（较难）

答：可以画⼀个这样的技术栈图先，然后分别解释下每个组件的功能和场景

1）Spark core：是其它组件的基础，spark的内核，主要包含：有向循环图、RDD、Lingage、Cache、broadcast等，并封装了底层通讯框架，是Spark的基础。 （2分）

2）SparkStreaming是⼀个对实时数据流进行高通量、容错处理的流式处理系统，可以对多种数据源（如Kdfka、Flume、Twitter、Zero和TCP 套接字）进行类似Map、Reduce和Join等复杂操作，将流式计算分解成⼀系列短⼩的批处理作业。 （2分）

3）Spark sql：Shark是SparkSQL的前身，Spark SQL的⼀个重要特点是其能够统⼀处理关系表和RDD，使得开发⼈员可以轻松地使用SQL命令进行外部查询，同时进行更复杂的数据分析 （2分）

4）BlinkDB ：是⼀个用于在海量数据上运行交互式 SQL 查询的⼤规模并行查询引擎，它允许用户通过权衡数据精度来提升查询响应时间，其数据的精度被控制在允许的误差范围内。 （2分）

5）MLBase是Spark生态圈的⼀部分专注于机器学习，让机器学习的门槛更低，让⼀些可能并不了解机器学习的用户也能方便 地使用MLbase。MLBase分为四部分：MLlib、MLI、ML Optimizer和MLRuntime。 （2分）

10.spark的有⼏种部署模式，每种模式特点？ （较难）

1）本地模式 （2分）

Spark不⼀定⾮要跑在hadoop集群，可以在本地，起多个线程的方式来指定。将Spark应用以多线程的方式直接运行在本地，

⼀般都是为了方便调试，本地模式分三类

• local：只启动⼀个executor

• local[k]:启动k个executor

• local[\*]：启动跟cpu数目相同的 executor

2)standalone模式 （2分）

分布式部署集群， 自带完整的服务，资源管理和任务监控是Spark自己监控，这个模式也是其他模式的基础，

1. Spark on yarn模式（3分）

分布式部署集群，资源和任务监控交给yarn管理，但是目前仅⽀持粗粒度资源分配方式，包含cluster和client运行模 式，cluster适合生产，driver运行在集群子节点，具有容错功能，client适合调试，dirver运行在客户端

1. Spark On Mesos模式。（3分）

官方推荐这种模式（当然，原因之⼀是血缘关系）。正是由于Spark开发之初就考虑到⽀持 Mesos，因此，目前⽽⾔，Spark运行在Mesos上会⽐运行在YARN上更加灵活，更加自然。

11、如何配置spark master的HA？ （较难）

1)配置zookeeper （2分）

2)修改spark\_env.sh⽂件,spark的master参数不在指定，添加如下代码到各个master节点

export SPARK\_DAEMON\_JAVA\_OPTS="-Dspark.deploy.recoveryMode=ZOOKEEPER -

Dspark.deploy.zookeeper.url=zk01:2181,zk02:2181,zk03:2181 -Dspark.deploy.zookeeper.dir=/spark" （2分）

3) 将spark\_env.sh分发到各个节点 （2分）

4)找到⼀个master节点，执行./start-all.sh，会在这里启动主master,其他的master备节点，启动master命令: ./sbin/start- master.sh （2分）

5)提交程序的时候指定master的时候要指定三台master，例如 （2分）

./spark-shell --master spark://master01:7077,master02:7077,master03:7077

12、 请叙述Spark SQL的工作流程。 （较难）

参考答案:

1.在解析SQL语句之前，会创建SparkSession，涉及到表名、字段名称和字段类型的元数据都将保存在SessionCatalog中； （2分）

2.当调用SparkSession的sql()方法时就会使用SparkSqlParser进行解析SQL语句，解析过程中使用的ANTLR进行词法解析和语法解析； （2分）

3.接着使用Analyzer分析器绑定逻辑计划，在该阶段，Analyzer会使用Analyzer Rules，并结合SessionCatalog，对未绑定的逻辑计划进行解析，生成已绑定的逻辑计划；（2分）

4.然后使用Optimizer优化器优化逻辑计划，该优化器同样定义了⼀套规则（Rules），利用这些规则对逻辑计划和语句进行迭代处理； （2分）

5.接着使用SparkPlanner对优化后的逻辑计划进行转换，生成可以执行的物理计划SparkPlan； 最终使用QueryExecution执行物理计划，此时则调用SparkPlan的execute()方法，返回RDDs。 （2分）

13、 Spark SQL提供了三个功能，分别是什么，请进行描述？（较难）

参考答案:

（1）Spark SQL可以从各种结构化数据源（例如JSON、Hive、Parquet等）中读取数据，进行数据分析。 （3分）

（2）Spark SQL包含行业标准的JDBC和ODBC连接方式，因此它不局限于在Spark程序内使用SQL语句进行查询。（3分）

（3）Spark SQL可以无缝地将SQL查询与Spark程序进行结合，它能够将结构化数据作为Spark中的分布式数据集（RDD）进行查询，在Python、Scala和Java 中均集成了相关API，这种紧密的集成方式能够轻松地运行SQL查询以及复杂的分析算法。 （4分）

14、利用反射机制推断Schema代码流程是什么？（较难）

1 创建sparksession （2分）

2 创建sc （2分）

3 读取数据并加工

val PersonRDD: RDD[Person] = ttRDD.map(z=>Person(z(0).toInt,z(1),z(2).toInt)) （2分）

4 RDD转DF

val personDF: DataFrame = PersonRDD.toDF() （2分）

5 注册成表并查询 （1分）

6 关闭sc、sparksession （1分）

15、 对比Hadoop 框架和Spark框架，Spark 有哪些优势？（中）

 1）Hadoop表达能力有限。计算都必须要转化成Map和Reduce两个操作，但这并不适合所有的情况，难以描述复杂的数据处理过程；（2分）

 2）Hadoop磁盘IO开销大。每次执行时都需要从磁盘读取数据，并且在计算完成后需要将中间结果写入到磁盘中，IO开销较大；（2分）

 3）Hadoop延迟高。一次计算可能需要分解成一系列按顺序执行的MapReduce任务，任务之间的衔接由于涉及到IO开销，会产生较高延迟。而且，在前一个任务执行完成之前，其他任务无法开始，难以胜任复杂、多阶段的计算任务。（2分）

 4）Spark的计算模式也属于MapReduce，但不局限于Map和Reduce操作，还提供了多种数据集操作类型，编程模型比MapReduce更灵活；（2分）

 5）Spark提供了内存计算，中间结果直接放到内存中，带来了更高的迭代运算效率；Spark基于DAG的任务调度执行机制，要优于MapReduce的迭代执行机制。（2分）

16、 定义一个映射，包含以下学生姓名和年龄数据:（中）

"zhangsan", 30

"lisi", 40

请使用Scala代码完成以下问题

1）获取zhangsan的年龄

2）获取所有的学生姓名

3）获取所有的学生年龄

4）打印所有的学生姓名和年龄

5）获取wangwu的年龄，如果wangwu不存在，则返回-1

6）新增一个学生：wangwu, 35

7）将lisi从可变映射中移除

答：（1）val map = Map("zhangsan"->30, "lisi"->40)

map("zhangsan") （2分）

（2）map.keys （2分）

（3）map.values （2分）

（4）for((x,y) <- map) println(s"$x $y") （1分）

（5）map.getOrElse("wangwu", -1) （1分）

（6）map + "wangwu"->35 （1分）

（7）map - "lisi" （1分）

# 17、 求TOP值，a.txt文件包含4个字段orderid、userid、payment、productid，求TopN个payment值，请根据要求补全代码。

a.txt文件如下：

# 1,1768,50,155 2,1218,600,211 3,2239,788,242 4,3101,28,599 5,4899,290,129 6,3110,54,1201 7,4436,259,877 8,2369,7890,27

object Test {

def main(args: Array[String]): Unit = {

val spark = SparkSession.builder().appName("wc").master("local").getOrCreate()

spark.sparkContext.setLogLevel("ERROR")

val input\_path = "/Users/zz/Desktop/input/" （2分）

// 分区数目为1，1个文件

val lines = spark.sparkContext.textFile(input\_path, 1) （2分）

var num = 0

val result = lines.filter(line => (line.trim().length > 0)

&& (line.split(",").length == 4))

.map(\_.split(",")(2)) // 取第三列 （2分）

.map(x => (x.toInt, "")) // 转换为kv （2分）

.sortByKey(false)

.map(x => x.\_1).take(5)

.foreach(x => {

num = num + 1 （2分）

println(num + "\t" + x)

})

spark.stop()

}

}

18、对多个输入文件进行排序，输出文件包括rank和value两列，请补全代码。

file1.txt：

33  
37  
12  
40

file2.txt

4  
16  
39  
5

import org.apache.spark.HashPartitioner

val input\_path = "/Users/zenmen/Desktop/input/"

val output\_path = "/Users/zenmen/Desktop/output/"

// 分区数目为2

val lines = spark.sparkContext.textFile(input\_path, 2) （2分）

var index = 0

val result = lines.filter(\_.trim().length>0)

.map(line => (line.trim.toInt, "")) （2分）

.partitionBy(new HashPartitioner(1)) // 确保排序结果的准确性

.sortByKey() （2分）

.map(t => {

index += 1

(index,t.\_1) （2分）

})

result.saveAsTextFile(output\_path) （2分）

# 19、file1.txt和file2.txt两个文件中的最大值和最小值，请补全代码。

# file1.txt

129  
54  
167  
324  
111  
54  
26  
697  
4856  
3418

file2.txt

5  
329  
14  
4567  
2186  
457  
35  
267

val input\_path = "/Users/zenmen/Desktop/input/"

val lines = spark.sparkContext.textFile(input\_path, 2) （2分）

val result = lines.filter(\_.trim().length>0)

.map(line => ("key", line.trim.toInt)) （2分）

.groupByKey()

.map(x => {

var min = Integer.MAX\_VALUE

var max = Integer.MIN\_VALUE （2分）

for(num <- x.\_2){ // 对值列表进行遍历

if(num>max){max=num}

if(num<min){min=num} （2分）

}

(max,min)

}).collect

.foreach(x => {

println("max= "+x.\_1)

println("min= "+x.\_2) （2分）

})

20、给定两个文件，file1.txt为用户电影评分表，包含uid,movieId,score三个字段；file2.txt为电影信息表包括movieId,movieName两个字段，取出平均分大于4.0的电影，请补全代码。

file1.txt：

1::100::3.1  
2::101::6.2  
3::100::9.3  
4::100::3.4  
5::101::7.5  
6::102::6.6  
7::103::2.7

file2.txt：

100,huluwa  
101,aoteman  
102,huoying  
103,sunwukong

spark.sparkContext.setLogLevel("ERROR")

val input\_path\_1 = "/Users/zenmen/Desktop/input/file1.txt"

val input\_path\_2 = "/Users/zenmen/Desktop/input/file2.txt" （2分）

val output\_path = "/Users/zenmen/Desktop/output/"

val rdd1 = spark.sparkContext.textFile(input\_path\_1) （2分）

.map(line => {

val fileds = line.split("::")

(fileds(1).toInt, fileds(2).toDouble) （2分）

}).groupByKey() // 电影维度进行聚合，计算得分平均值

.map(data => {

val avg = data.\_2.sum / data.\_2.size （2分）

(data.\_1, avg)

}

)

rdd1.foreach(println) （2分）