**大数据学院**

**2018年10月30日**



**[第一单元 大数据概论和分布式介绍]**

hadoop标准化课件设计

**《大数据离线计算》**

**第九单元 mapreduce开发**

1. **本单元教学目标**
2. 知识目标：

**·**  强化MapReduce的处理流程；

* 掌握NullWritable类型

**·**  掌握MapReduce的键值类型的应用

**·**  设计MapReduce键值模型

**·**  掌握MapReduce的分区

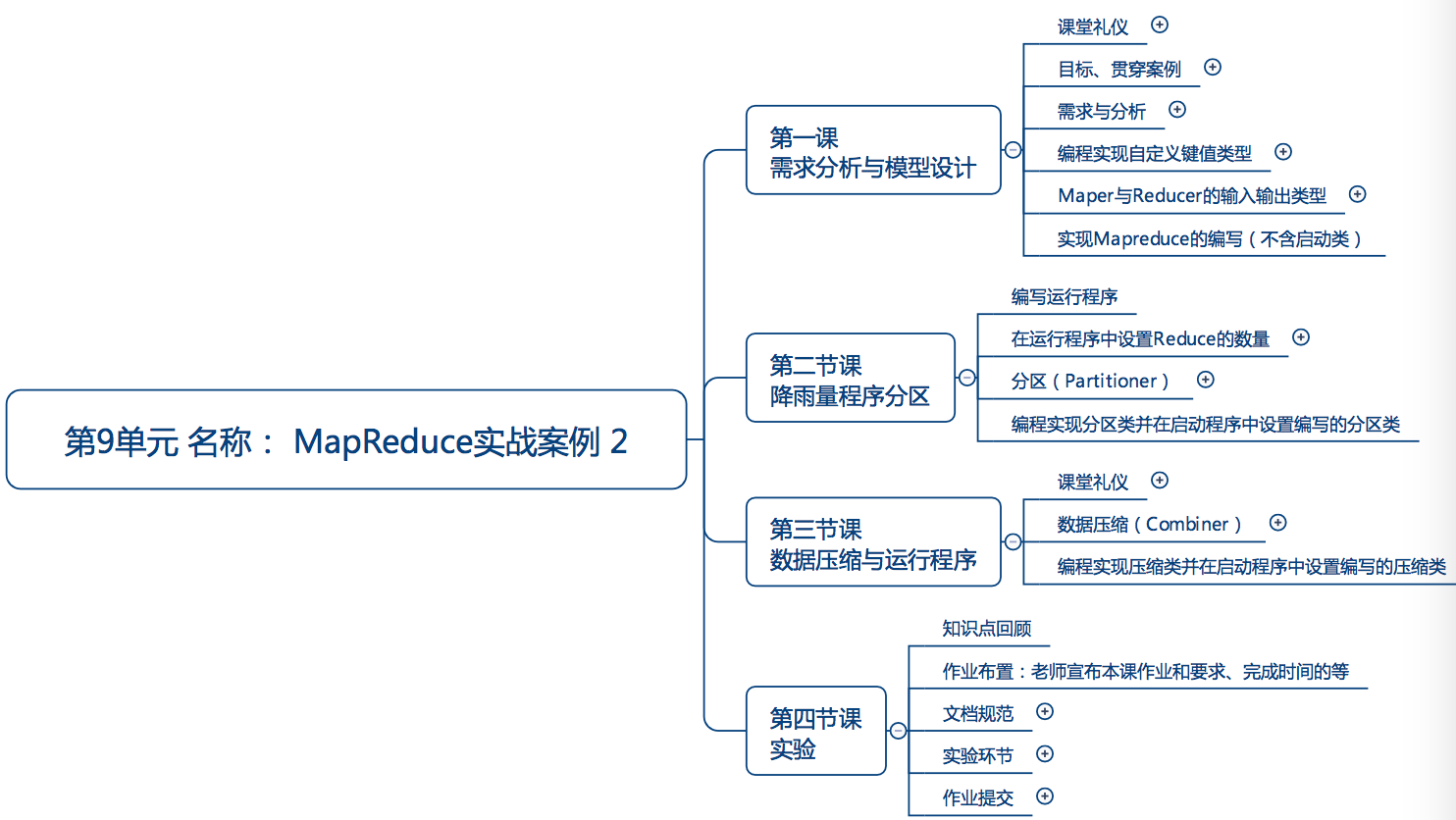
**·**  掌握MapReduce的Combiner

**·**  掌握MapReduce的自定义键值类型

1. 能力目标：

* 能够自行开发MapReduce分区处理程序
* 能够自定义键值的类型
* 能够自定义分区规则
* 能够自行开发MapReduce压缩

1. **课时分配**

****

1. **第33课时：需求分析与模型设计**

本课时通过对大数据和分布式的宏观介绍，使学生初步建立大数据知识体系，使得学生能为学习以后的课程打下基础。

1. 知识目标：

**·**  强化MapReduce模型设计；

**·** 掌握自定义键值类型；

**·**  掌握NullWritable类型

1. 能力目标：

* 能够自行开发MapReduce分区处理程序
* 能够自定义键值的类型

1. **第34课时：降雨量程序分区**
2. 知识目标：

* 什么是分区
* 了解分区的注意事项
* 掌握实现分区的方法

1. 能力目标：

**·** 能够根据需求实现分区

**·** 能够自定义分区

1. **第35课时：数据压缩与运行程序**
2. 知识目标：

* 什么是压缩
* 压缩的注意事项
* 压缩的实现方法

1. 能力目标：

* 能够自行事项压缩

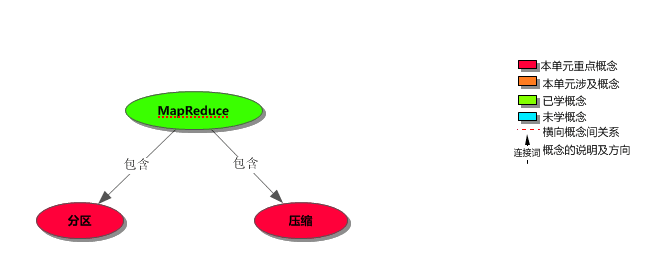
1. **第36课时：实验**
2. 知识目标：

* 强化压缩，分区，自定义键值类型

1. 能力目标：

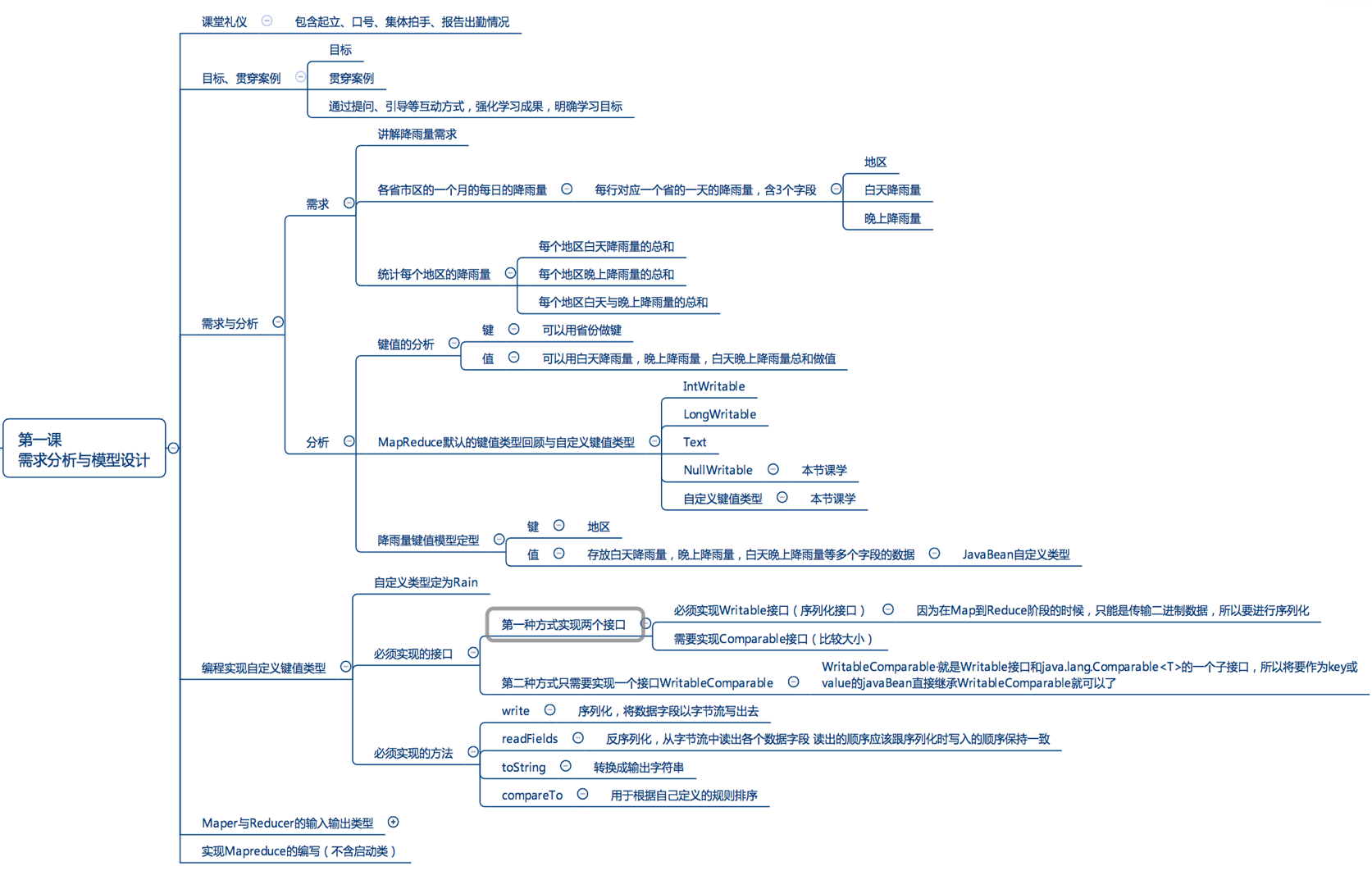
* 能够掌握并自行完成压缩，分区，自定义键值类型

**三、本单元概念图**



**四、教学过程脚本设计—第一课时**

1. **脚本设计思路**

****

1. **脚本切片设计思路**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **序号** | **内容** | **小节目标** | **设计亮点** | **时间** |
| **1** | **课堂礼仪** | 集中学生的注意力，喊出士气，以饱满的精神状态投入本节课的学习 | 整齐、声音洪亮、铿锵有力，提升精气神！ |  |
| **2** | **开发项目基本步骤概述** | 概述项目大体流程 |  |  |
| **2** | **降雨量需求** | 描述降雨量需求 | 详述降雨量统计的需求 |  |
| **3** | **算法模型** | 定义键值算法模型 | 定义键值算法模型 |  |
| **4** | **设计自定义键值类型必须实现的接口** | 自行实现自定义键值类型 | 自定义类型的要求与设计实现 |  |
|  | **设计自定义键值类型必须实现的方法** |  |  |  |
|  | **自定义键值类的编写** |  |  |  |
| **5** | **Map程序开发** | 根据需求自行开发mapreduce程序 | 自定义键值类型到mapreduce程序 |  |
|  | **Reduce程序开发** |  |  |  |
|  | **知识点总结** |  |  |  |
| **6** | **分区** | 分区的实现 | 数据分区 |  |
| **7** | **开发combiner** | 自行开发combiner | 能够根据需求自行开发combine |  |
| **8** | **启动程序** | 实现启动程序能够按照要求分区 | 启动运行，能够根据设计进行分区 |  |
| **9** | **reduce的数量与partition的数量** | reduce的数量与partition的数量的注意事项 | reduce的数量与partition的数量的注意事项 |  |

1. **脚本切片一：[课堂礼仪](file:///D:\\llq\\八维\\工作\\教学标准化计划\\切片(总共14片)\\切片1.mp4)**

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| **课堂礼仪** | **使学生注意力集中并进入上课状态** |  |
| **讲师：**上课！①  **班长：**起立！  **讲师：**同学们好！  **学生：**老师好！  **班长：**我们的信念是：  **全体：**知识^很简单，学习^很快乐，潜力^无极限！  **班长：**我们的口号是：  **全体：**辉煌八维大数据，引领时代高科技！人工智能显神奇，高薪就业创佳绩！  **讲师：**请坐！② | 1. 进入教室前整理着装，提前30秒面带自信的笑容的走进教室。，给学生打招呼。当上课铃声响起，喊出“上课”，声音高于授课声音。 2. 细节要求：学生发型整齐、脸部整洁、面带微笑或者礼仪笑（八颗牙齿），严禁大笑，举止合理，避免指手画脚。   IMG_256IMG_256   1. 学生礼仪内容：学习信念，学院口号，班级口号。   讲师授课过程中要面带微笑。   1. 插入合理手势，避免课堂枯燥。   ①动作表情：上课起立手势  ②动作表情：全体请坐手势 |

1. **脚本切片二：开发项目基本步骤概述**

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| 项目开发基本步骤概述 | **了解项目开发基本过程** |  |
| **讲师：**开发一个项目最基本的步骤是：  需求分析，算法模型与程序实际，本地运行调试 |  |

1. **脚本切片三**：**降雨量需求**

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| **降雨量需求** | **了解需求** | 正确理解需求 |
| **讲师：**先分析一下降雨量的需求  全国各个省市每个月的白天降雨量与晚上降雨量，得到每个省市的白天降雨量总和，晚上降雨量总和，白天和晚上的降雨量的总和。 |  |

1. **脚本切片四：**mapreduce算法模型

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| **Mapreduce计算模型** | **从需求的理解进行模型分析** | 能够正确的进行模型设计 |
| **讲师：**通过需求的分析设计简map与 reduce的键值模型：   1. map模型：map<省份,（白天雨量，晚上雨量）> 2. 第一种Reduce模型：reduce<省份,(sum（白天雨量，晚上雨量）,白天晚上和）> 3. 第二种Reduce模型：reduce<（省份，sum（白天雨量，晚上雨量）,白天晚上和））,空> |  |

1. **脚本切片五：**设计自定义键值类型

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| 设计自定义键值类型必须实现的接口 | **自定义键值类型实现的接口** | 掌握自定义键值类型必须实现的接口 |
| **老师：**通过上面的要处理的文本文件可以看出有3列，如果以省份作为键，值就有 2列。多了一列，那么怎么在一个值中表示两列数据那？如果在值中有更多列的数据怎么表示那？  **老师：**可以用一个java 类封装多个内容作为键或值，也就是自定义键值  **老师：**设计自定义键值类型必须实现两个接口：   1. 必须实现Writable接口（序列化）   因为在Map到Reduce阶段的时候，只能是传输二进制数据，所以要进行序列化   1. 需要实现Comparable接口（比较大小）   因为在shuffle到Reduce阶段的合并阶段，需要根据key对数据进行排序，合并，如果不实现这个接口，运行时会出错   1. WritableComparable接口   WritableComparable就是Writable接口和java.lang.Comparable<T>的一个子接口，所以将要作为key的javaBean直接继承WritableComparable就可以了 |  |

1. **脚本切片六：**设计自定义键值类型必须实现的方法

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| 设计自定义键值类型 | **计算模型分析** | 根据需求产生的键值模型开发mapreduce |
| **讲师：**必须实现的WritableComparable接口里的所有方法：  **write**: 序列化，将数据字段以字节流写出去；  **readFields**: 反序列化，从字节流中读出各个数据字段 读出的顺序应该跟序列化时写入的顺序保持一致；  **compareTo**: 用于根据自己定义的规则排;  **toString**:转换成输出字符串。 |  |

1. **脚本切片七：**编写自定义键值类型程序

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| 编写自定义键值类型程序 | **自定义键值类型程序开发** | 掌握自定义键值类型程序实操 |
| **讲师：**现在通过实现WritableComparable接口变现自定义键值类 [ppt] | **public class Rain implements WritableComparable<Rain>{**  private String province;//省份     private long day;//白天     private long night;//晚上     private long total;//白天和晚上总和     public String getProvince() {        return province;     }     public void setProvince(String province) {        this.province = province;     }     public long getDay() {        return day;     }     public void setDay(long day) {        this.day = day;     }     public long getNight() {        return night;     }     public void setNight(long night) {        this.night = night;     }     public long getTotal() {        return total;     }     public void setTotal(long total) {        this.total = total;     }      public void set(String province,long day,long night){        this.province = province;        this.day = day;        this.night = night;        this.total = day + night;     }     @Override*序列化，将数据字段以字节流写出去*     public void **write**(DataOutput out) throws IOException {        out.writeUTF(province);        out.writeLong(day);        out.writeLong(night);        out.writeLong(total);     }       @Override*反序列化，从字节流中读出各个数据字段 读出的顺序应该跟序列化时写入的顺序保持一致*     public void **readFields**(DataInput in) throws IOException {        province = in.readUTF();        day = in.readLong();        night = in.readLong();        total = in.readLong();     }     @Override     public String **toString**() {        return province+"\t"+day+"\t"+night+"\t"+total;     }     @Override*用于根据自己定义的规则排序*     public int **compareTo**(Rain o) {        return (int) (o.getTotal()-this.total);     }  } |

1. **脚本切片八：**Map程序开发

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| **Map程序开发** | **Map程序开发** |  |
| **讲师：**根据上面设计的Map模型开发Map程序 | public class RainMapper extends Mapper<LongWritable, Text, Text, Rain>{        private Text k = new Text();//键，省份作为键        private Rain v = new Rain();//值，自定类型作为值，有4个字段：省份、白天雨量、晚上雨量、白天晚上总和量          @Override        protected void map(LongWritable key, Text value, Mapper<LongWritable, Text, Text, Rain>.Context context)              throws IOException, InterruptedException {           String[] info = StringUtils.split(value.toString(),"\t");//将当前行数据用跳格键分割           if(info.length == 3){//判断行的数据是否合法，只要数组长度是3，就合法              v.set(info[0], Integer.parseInt(info[1]), Integer.parseInt(info[2]));//将省份，白天雨量，晚上雨量存入自定义值对象              k.set(info[0]);//省份作为键（最终就会按照省份分组）              context.write(k, v);//设置键值映射，map<省份，Rain（省份，白天雨量，晚上雨量）>           }        }     } |

1. **脚本切片九：**NullWritable用法

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| **NullWritable** | **NullWritable用法** | 了解NullWritable代表什么 |
| **讲师：**如果值不需要输出任何内容，那么怎么表示值那，Hadoop中已经定义了如下表示空（Null）的类型：**NullWritable** |  |

1. **脚本切片十：**Reduce程序开发

|  |  |  |  |
| --- | --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** | **时长** |
| **Mapreduce程序开发** | **Reduce程序开发** |  | |
| **讲师：**根据上面设计的Reduce模型开发Reduce程序[ppt] | public class RainReducer extends Reducer<Text, Rain, Rain, NullWritable>{         /\*          \* 1：定义键对象，准备将Rain作为键，因为Rain里已经封装了省份，白天的雨量，晚上的雨量，白天晚上的雨量，          \* 2:reduce聚合输出时会键与值全部输出。          \* 3:综上两点，输出时Rain已经包含了所有的内容，所以可以将值设置为空，空用NullWritable表示          \* \*/    private Rain k = new Rain();        @Override        protected void reduce(Text key, Iterable<Rain> values, Reducer<Text, Rain, Rain, NullWritable>.Context context)              throws IOException, InterruptedException {           long daysum = 0l;           long nightsum = 0l;           Iterator<Rain> iterator = values.iterator();           while(iterator.hasNext()){              Rain next = iterator.next();              long day = next.getDay();              long night = next.getNight();              daysum += day;              nightsum += night;           }           k.set(key.toString(), daysum, nightsum);           context.write(k, NullWritable.get());        }     } | |

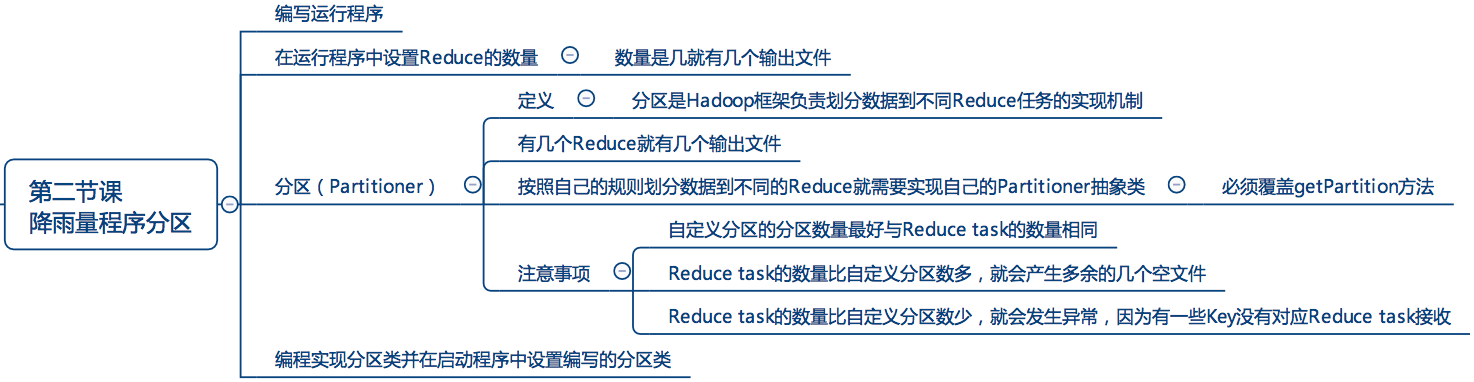
1. **脚本切片十：**启动程序要求

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| **Mapreduce程序开发** | **Reduce程序开发** |  |
| **讲师：**每个同学自行开发启动程序调试 |  |

1. **脚本切片十：课堂总结**

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| **Mapreduce程序开发** | **Reduce程序开发** |  |
| **讲师：**   1. WritableComparable接口的的作用？ 2. 自定义键值自定义键值类的定义 3. 自定义键值类必须实现的方法 |  |

**五、教学过程脚本设计—第二课时**

1. **脚本设计思路**
2. **脚本切片设计思路**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **序号** | **内容** | **小节目标** | **设计亮点** | **时间** |
| **1** | **课堂礼仪** | 集中学生的注意力，喊出士气，以饱满的精神状态投入本节课的学习 | 整齐、声音洪亮、铿锵有力，提升精气神！ |  |
| **2** | **为什么要自定义分区** | 自定义分区的原因 |  |  |
| **3** | **Reduce数量设置** | 会设置Reduce数量，了解其意义 | 图文并茂简洁说明 |  |
| **4** | **什么是分区** | 分区的定义 | 引入了分区的意义并定义 |  |
| **5** | **分区注意事项** | 自定义分区要注意什么 | 简洁介绍分区规则 |  |
| **6** | **分区实现方法** | 掌握分区集成的类 |  |  |
| **7** | **编写分区类** | 掌握如何开发分区类 | 注重实操 |  |
| **8** | **知识点总结** | 本节课知识点回顾 |  |  |

1. **脚本切片一：**编写并运行降雨量程序

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| 分区 | **了解分区原因** | 自行开发实现数据分区 |
| **讲师：**为什么要分区那？  **讲师：**mapreduce默认将所有的输出输出到一个文件，没有规则，如何将输出内容分类输出到不同的文件。这个得涉及到 内容的分区了  **讲师：**每个reduce对应一个输出文件。默认有一个reduce. |  |

1. **脚本切片二：**Reduce数量设置

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| 分区 | **设置Reduce的数量** | 掌握设置Reduce的数量 |
| **讲师：**可以设置reduce的数量，Reduce数量设置的方法是：  在启动程序中设置  job.setNumReduceTasks(5)  MapReduce是利用默认的分区将数据均匀的划分到多个reduce； |  |

1. **脚本切片三：**什么是分区

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| 什么是分区 | **分区概述** | 理解分区与分区的定义 |
| **讲师：**分区的概述  每个分区对应一个Reduce,每个Reduce对应一个输出文件，MapReduce默认使用哈希方法（HashPartitioner）分区，数据没规则，比较乱。  要按照自己定的规则将数据输出到指定的文件，就需要自定义分区。  **讲师：**分区是Hadoop框架负责划分数据到不同的Reduce任务实现机制。 |  |

1. **脚本切片四：**分区注意事项

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| 分区注意事项 | **分区注意事项** | 自行开发实现数据分区 |
| **讲师：大家看一下**分区注意事项  1）自定义分区的分区数量最好与Reduce task的数量相同；  2）Reduce task的数量比自定义分区数多，就会产生多余的几个空文件；  3）Reduce task的数量比自定义分区数少，就会发生异常，因为有一些Key没有对应Reduce task接收。 |  |

1. **脚本切片四：**分区实现方法

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| 分区注意事项 | **分区实现方法** | 自行开发实现数据分区 |
| **讲师：**分区注意事项  1）自定义分区的分区数量最好与Reduce task的数量相同；  2）Reduce task的数量比自定义分区数多，就会产生多余的几个空文件；  3）Reduce task的数量比自定义分区数少，就会发生异常，因为有一些Key没有对应Reduce task接收。 |  |

1. **脚本切片四：**编写分区类

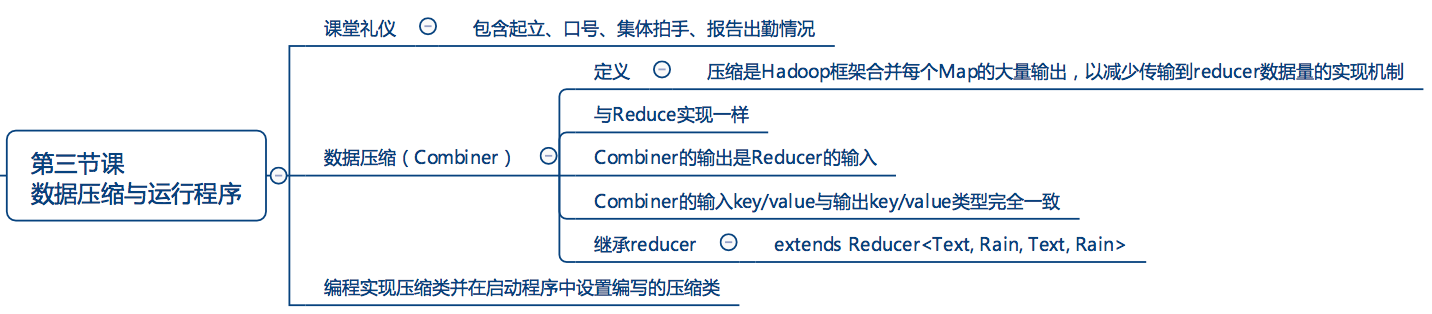
|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| 分区注意事项 | **分区实现方法** | 自行开发实现数据分区 |
| **讲师：**以下是分区实现的代码：[ppt] | public class RainPartitioner extends Partitioner<Text, Rain>{        private String[] provices = {              "河北省",              "山西省",              "辽宁省",              "吉林省",              "黑龙江省",              "江苏省",              "浙江省",              "安徽省",              "福建省",              "江西省",s              "山东省",              "河南省",              "湖北省",                   };  /\*\*计算位置 \*/        public int indexOfArr(String[] arrys,String str){           for(int i = 0;i<arrys.length;i++){              if(arrys[i].equals(str))                 return i;           }           return -1;        }        /\*\*         \* 获取分区编号，下下表从0开始.必须覆盖getpartition  \* 通过传入的值自己定义规则分区。这里通过省份在数组里的位置除以5求余得到具体的分区，也就是对应到那个文件。\*/        @Override        public int getPartition(Text key, Rain value, int numPartitions) {           String province = key.toString();           int proIndex = indexOfArr(provices, province);           return (proIndex % 5);        }     } |

1. **脚本切片六：**知识点总结

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| 知识点回顾 | **知识点回顾** | 自行开发实现数据分区 |
| 1. 分区的意义。 2. 分区类必须继承的类。 3. 分区类必须覆盖的方法getPartition的理解 |  |

**六、教学过程脚本设计—第三课时**

1. **脚本设计思路**



1. **脚本切片设计思路**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **序号** | **内容** | **小节目标** | **设计亮点** | **时间** |
| **1** | **课堂礼仪** | 集中学生的注意力，喊出士气，以饱满的精神状态投入本节课的学习 | 整齐、声音洪亮、铿锵有力，提升精气神！ |  |
| **2** | **为什么要压缩** | 压缩的原因与好处 | 图文并茂介绍压缩的原因呢 |  |
| **3** | **压缩的定义** | 会设置Reduce数量，了解其意义 | 简洁定义理解压缩 |  |
| **4** | **压缩的注意事项** | 分区的定义 | 简洁介绍压缩的注意细节 |  |
|  | **压缩需要继承的类** | 掌握压缩需要继承的类 | 压缩的实现方法 |  |
|  | **编写压缩类** | 掌握开发压缩类 | 注重实操 |  |
|  | **编写启动程序运行** | 学会设置压缩与分区 |  |  |
| **8** | **知识点回顾** |  |  |  |

**3．脚本切片一：**为什么要压缩

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| 为什么要压缩 | **了解应用压缩的原因以及好处** | 了解为什么要用压缩以及压缩的优点 |
| **讲师：**为什么要压缩？  1）每一个map可能会产生大量的输出，Combiner的作用就是在map端对输出先做一次合并，以减少传输到reducer的数据量。  2)Combiner最基本是实现本地key的归并，Combiner具有类似本地的reduce功能，  如果不用Combiner，那么，所有的结果都是reduce完成，效率会相对低下。  使用Combiner，先完成的map会在本地聚合，提升速度。 |  |

**4.脚本切片二：**压缩定义

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| **压缩的定义** | 压缩的定义 | 理解压缩的定义 |
| **讲师：**  Hadoop框架中将每个map的大量输出在Map端先做一次合并，以减少传输到reducer的数据量的实现机制 |  |

**5.脚本切片三：**压缩的注意事项

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| **压缩的注意事项** | **了解压缩的注意事项** |  |
| **讲师：**压缩要注意   1. Combiner的输出是Reducer的输入；   2）如果Combiner是可插拔的，添加Combiner绝不能改变最终的计算结果；  3）Combiner只应该用于那种Reduce的输入key/value与输出key/value类型完全一致，且不影响最终结果的场景。比如累加，最大值等。 |  |

**6.脚本切片四：**压缩需要继承的类

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| **压缩需要继承的类** | **了解压缩需要继承的类** |  |
| **讲师：**与编写Reducer类需要继承的类是一样的  继承Reducer类 |  |

**7.脚本切片五：编写压缩类**

|  |  |  |
| --- | --- | --- |
| **模块** | **教学内容/台词** | **教学标准/素材** |
| **开发combiner** | **开发压缩程序** | 能自行开发combiner |
| 讲师：开发combiner的意义：[ppt] | /\*\* Combiner，map的输出是combiner的输入，combiner的输出是reducer的输  因为继承Reducer ，所以实现的方法与Reducer 大体一样，只需要注意返回值\*/  public class CombinerClass extends Reducer<Text, Rain, Text, Rain>{        private Rain v = new Rain();//定义降雨量对象        protected void reduce(Text key, Iterable<Rain> values, Reducer<Text, Rain, Text, Rain>.Context context)              throws IOException, InterruptedException {           long daysum = 0l;//累加器           long nightsum = 0l;//累加器           Iterator<Rain> iterator = values.iterator();           while(iterator.hasNext()){//遍历器              Rain next = iterator.next();              long day = next.getDay();              long night = next.getNight();              daysum += day;              nightsum += night;           }           v.set(key.toString(), daysum, nightsum);           context.write(key, v);//\*\*输入与输出类型一致        }     } |

**8.脚本切片六：编写并启动程序**

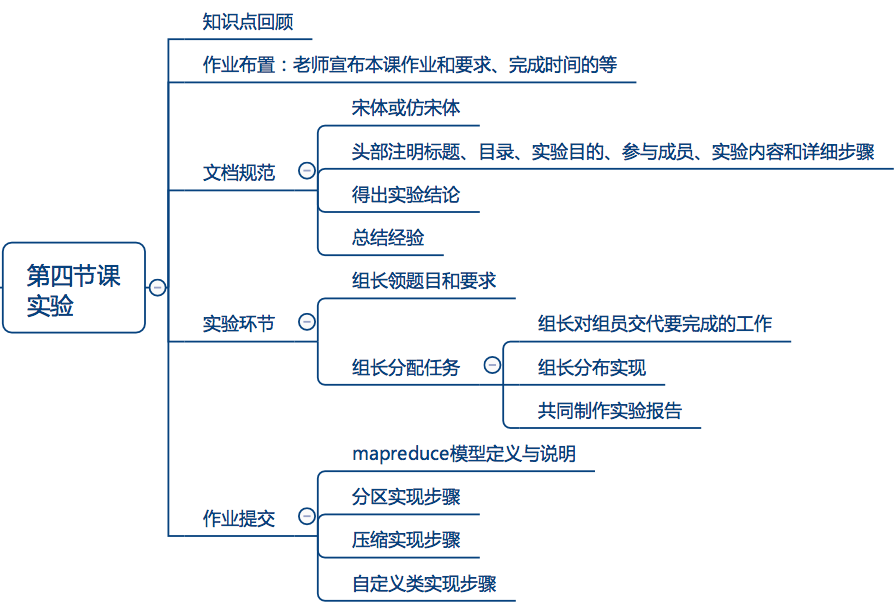
|  |  |  |
| --- | --- | --- |
| **模块** | **教学内容/台词** | **教学标准/素材** |
| **编写并启动程序** | **开发启动程序并使分区与压缩起作用** | 掌握如何在启动程序中设置分区、压缩 |
| 讲师：根据以前的map的启动程序开发降雨量启动程序 | public static void main(String[] args) throws Exception {    Path input = new Path("/Users/wangjianfeng/rain.txt");    Path output = new Path("/Users/wangjianfeng/rain");    Configuration conf = new Configuration();    Job job = Job.getInstance(conf,"root");    job.setJarByClass(SumJob.class);    //设置Map类和Reducer类    job.setMapperClass(RainMapper.class);      job.setCombinerClass(CombinerClass.class);    job.setPartitionerClass(RainPartitioner.class);      job.setNumReduceTasks(5);  job.setReducerClass(RainReducer.class);      //设置Map的输出类型    job.setMapOutputKeyClass(Text.class);    job.setMapOutputValueClass(Rain.class);        //设置Reduce的输出类型      job.setOutputKeyClass(Rain.class);      job.setOutputValueClass(NullWritable.class);        //设置输入路径和输出路径      FileInputFormat.setInputPaths(job, input);      FileSystem fs = FileSystem.get(conf);      if(fs.exists(output))         fs.delete(output, true);      FileOutputFormat.setOutputPath(job, output);      //启动任务      job.waitForCompletion(true);    } |

**9.脚本切片七：知识点总结**

|  |  |  |
| --- | --- | --- |
| **模块** | **教学内容/台词** | **教学标准/素材** |
| **开发combiner** | 知识总结 |  |
| **讲师：**   1. 压缩的意义。 2. 压缩注意事项。 3. 压缩的实现方法。 4. 启动类中设置压缩方法。 5. 启动类中设置分区方法。 6. 启动类中设置分区数量。 |  |

1. **教学过程脚本设计—第四课时**

**1．脚本设计思路**



**2.脚本切片设计思路**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **序号** | **内容** | **小节目标** | **设计亮点** | **时间** |
| **1** | **降雨量程序实践** | 理解，掌握，自行编写 |  |  |
| **2** | **实践文档提交要求** | 文档的提交要求 |  |  |

**3.脚本切片三：**降雨量程序实践

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| 降雨量程序实践 | **降雨量程序** | 自行开发降雨量程序 |
| **讲师：**  本课时根据上课讲授的内容自行理解，掌握并编写降雨量程序。  要求：   1. 自行实现降雨量程序； 2. 自行实现降雨量自定义键值类，分区，压缩； 3. 运行并查看结果； |  |

**4.脚本切片四：**实践文档提交要求

|  |  |  |
| --- | --- | --- |
| **课堂模块** | **教学内容/台词** | **教学标准/素材** |
| 实践文档提交要求 | 明确实践文档的要求 |  |
| **讲师：**  本课时根据上节课的安装讲解，由教师进行安装辅导，学生小组成员为每4个人一组（按照班级已有组别）进行MapReduce程序的调试。下课提交Word版实验报告，要求每组每个同学调试并运行通讲课的案例。  步骤5教师会在下节课细讲实验步骤，学生请比较与自己的实现有什么不同。  实验报告要求：   1. 提交MapReduce的运行流程说明（map阶段业务说明，reduce阶段业务说明）； 2. 提交自定义键值的描述，提交分区与压缩的实现机制以及运行原理说明；代码要有注释   3. 提交团队成员实验主要分工；  4. 对mapreduce中的示例代码进行开发调试(1组、2组……)；  5.提交疑难问题列表  6. 程序分析疑难问题解决 |  |