任务一：每日资金流入流出统计

任务概述

根据 user_balance_table 表中的数据，编写 MapReduce 程序，统计所有用户每日的资金流入与 流出情况。资金流入意味着申购行为，资金流出为赎回行为。

设计思路

1.自定义 Writable 类型 `Amounts`：

- **类功能**：`Amounts` 类用于封装每个日期的资金购买（purchase）和赎回（redeem）金额，确保 Mapper 和 Reducer 之间能够传递自定义对象。
- **序列化**：实现了 `Writable` 接口，通过 `write()` 和 `readFields()` 方法进行数据的序列化和反序列化。`toString()` 方法返回资金的字符串表示，便于输出。

2.Mapper :

- **输入**：每行数据是一个文本记录，包含多个字段。Mapper 从中提取日期、购买金额和赎回金额。
- **日期键**：提取第 1 列作为日期（`fields[1]`），作为输出的键。
- **金额处理**：通过 `parseDouble()` 方法解析购买金额（`fields[4]`）和赎回金额（`fields[8]`），并创建一个新的 `Amounts` 对象保存这两个值。
- **输出**：以日期为键，将 `Amounts` 对象作为值输出，格式为 `<date, Amounts>`。

3.Reducer:

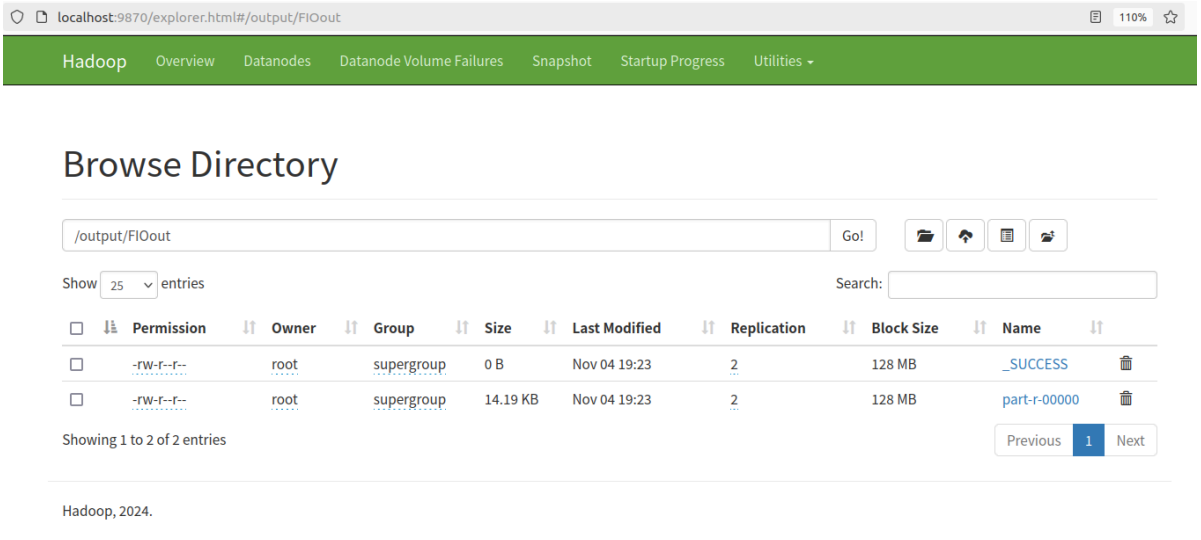
- **输入**：Reducer 收到来自 Mapper 的数据，按日期（键）聚合 `Amounts` 对象。
- **聚合**：对每个日期的 `Amounts` 对象进行求和，计算总的购买金额和赎回金额。
- **输出**：输出每个日期的总购买和赎回金额，格式为 `<date, totalPurchase,totalRedeem>`。

4.main:

- **作业配置**：配置 Hadoop 作业，指定输入路径和输出路径，设置 Mapper 和 Reducer 类，指定输出的键值类型。
- **执行作业**：调用 `job.waitForCompletion(true)` 启动作业并等待执行完成。

输出结果

```
root@h01:/usr/local/hadoop# ./bin/hdfs dfs -ls /output/FIOout
Found 2 items
-rw-r--r--   2 root supergroup          0 2024-11-04 11:23 /output/FIOout/_SUCCE
SS
-rw-r--r--   2 root supergroup      14535 2024-11-04 11:23 /output/FIOout/part-r
-00000
root@h01:/usr/local/hadoop# ./bin/hadoop fs -cat /output/FIOout/part-r-00000
20130701        3.2488348E7,5525022.0
20130702        2.903739E7,2554548.0
20130703        2.727077E7,5953867.0
20130704        1.8321185E7,6410729.0
20130705        1.1648749E7,2763587.0
20130706        3.6751272E7,1616635.0
20130707        8962232.0,3982735.0
20130708        5.7258266E7,8347729.0
20130709        2.6798941E7,3473059.0
20130710        3.0696506E7,2597169.0
20130711        4.4075197E7,3508800.0
20130712        3.4183904E7,8492573.0
20130713        1.5164717E7,3482829.0
20130714        2.2615303E7,2784107.0
20130715        4.8128555E7,1.3107943E7
20130716        5.0622847E7,1.1864981E7
20130717        2.9015682E7,1.0911513E7
20130718        2.4234505E7,1.1765356E7
20130719        3.3680124E7,9244769.0
20130720        2.0439079E7,4601143.0
20130721        2.1142394E7,2681331.0
20130722        4.0448896E7,1.9144267E7
20130723        5.8136147E7,2.4404051E7
```



任务二：星期交易量统计

任务概述

基于任务一的结果，编写 MapReduce 程序，统计一周七天中每天的平均资金流入与流出情况，并按照资金流入量从大到小排序。

设计思路

1.Mapper：

- **解析输入数据**：每行输入包含日期和资金流入流出量。Mapper 通过 `SimpleDateFormat` 将日期字符串解析为 `Date` 对象，接着利用 `Calendar` 类提取出星期几（如 "Monday", "Tuesday"）。
- **分组数据**：将数据按照星期几进行分组，即输出的键为星期几（如 "Monday"），值为资金流入和流出量。这里假设每行的数据格式是“日期（yyyyMMdd） + 资金流入量,资金流出量”，通过逗号分隔。
- **输出格式**：输出 `<weekday, flowAmounts>`，其中 `weekday` 是星期几的名称，`flowAmounts` 是资金流入和流出量的字符串。

2.Reducer：

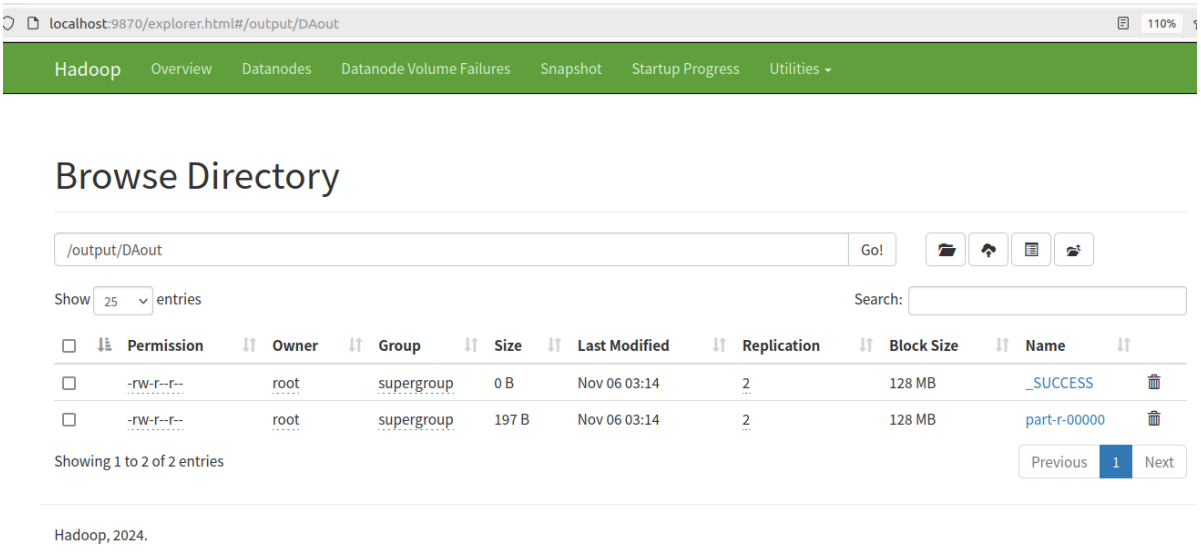
- **聚合数据**：Reducer 接收来自 Mapper 按星期几分组的所有数据。对于每个星期几，将资金流入和流出量累加，并计算日均值。
- **计算平均值**：每个星期几的总流入量和总流出量除以该星期几的记录数量，得到日均流入量和流出量。
- **排序与输出**：Reducer 还负责将星期几按资金流入量降序排序，并输出结果。

3.main：

- 配置 MapReduce 作业，设置输入输出路径，启动作业并执行。

运行结果

```
root@h01:/usr/local/hadoop# ./bin/hadoop fs -cat /output/DAout/part-r-00000
Tuesday 263582059,191769145
Monday  260305810,217463865
Wednesday       254162608,194639447
Thursday        236425594,176466675
Friday  199407923,166467960
Sunday  155914552,132427205
Saturday        148088068,112868942
```



任务三：用户活跃度分析

任务概述

根据 user_balance_table 表中的数据，编写 MapReduce 程序，统计每个用户的活跃天数，并按照活跃天数降序排列。当用户当日有直接购买（direct_purchase_amt 字段大于 0）或赎回行为（字段大于 0）时，则该用户当天活跃。

设计思路

1.Mapper：

- 读取输入数据并提取每行的用户 ID 和资金流入流出信息。
- 判断该用户是否符合活跃条件（如 `directPurchaseAmt > 0` 或 `totalRedeemAmt > 0`）。
- 如果用户符合条件，输出 `<userId, 1>`，表示该用户在某一天活跃。

2.Reducer：

- 聚合来自 Mapper 的用户活跃记录，计算每个用户的活跃天数（即用户出现的次数）。
- 将每个用户的活跃天数存储在 `userActiveDaysMap` 中。

3.排序与输出：

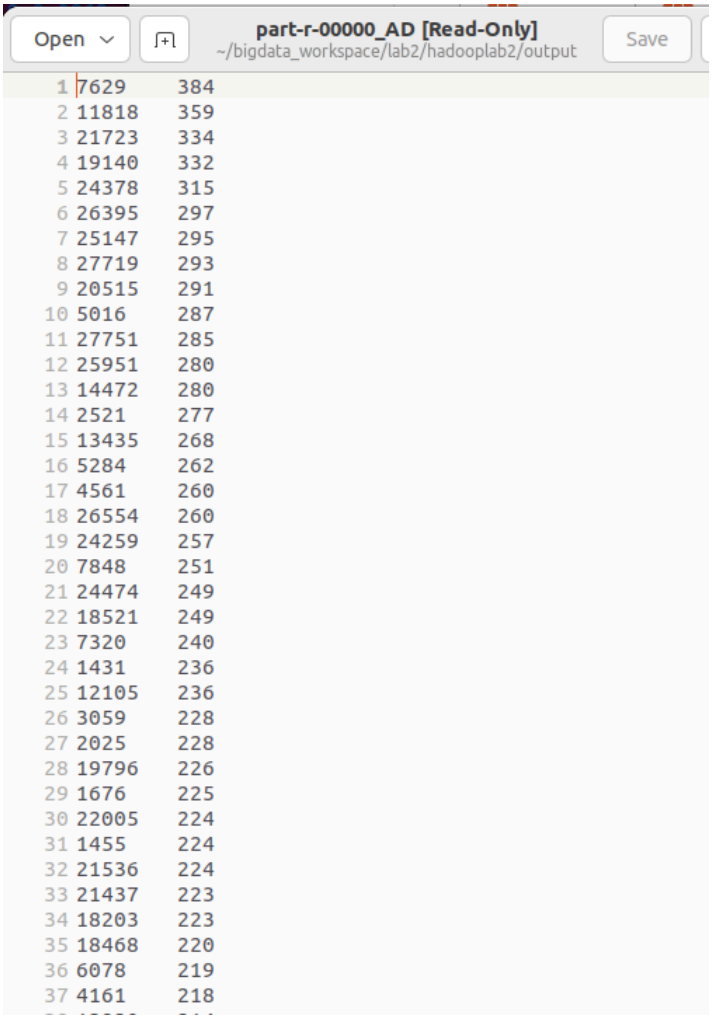
- 在 `cleanup` 方法中，将 `userActiveDaysMap` 中的用户按活跃天数降序排序。
- 输出每个用户的 ID 和活跃天数。

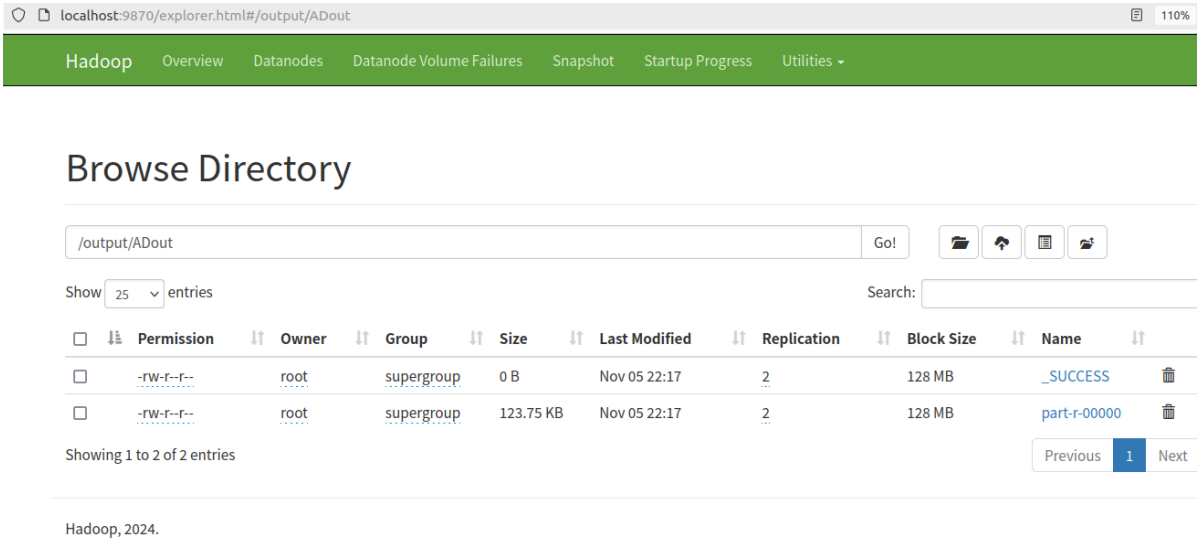
4.main：

- 配置和启动 MapReduce 作业，设置输入输出路径和相关参数，执行作业并返回结果。

运行结果

```
root@h01:/usr/local/hadoop# ./bin/hadoop jar hadooplab2-1.0-SNAPSHOT.jar com.exa
mple.ActiveDay /input/user_balance_table.csv /output/ADout
2024-11-05 14:17:36,668 INFO client.DefaultNoHARMFailoverProxyProvider: Connecti
ng to ResourceManager at h01/172.18.0.2:8032
2024-11-05 14:17:37,136 WARN mapreduce.JobResourceUploader: Hadoop command-line
option parsing not performed. Implement the Tool interface and execute your appl
ication with ToolRunner to remedy this.
2024-11-05 14:17:37,157 INFO mapreduce.JobResourceUploader: Disabling Erasure Co
ding for path: /tmp/hadoop-yarn/staging/root/.staging/job_1730809269004_0007
2024-11-05 14:17:37,466 INFO input.FileInputFormat: Total input files to process
 : 1
2024-11-05 14:17:37,579 INFO mapreduce.JobSubmitter: number of splits:2
2024-11-05 14:17:37,740 INFO mapreduce.JobSubmitter: Submitting tokens for job:
job_1730809269004_0007
2024-11-05 14:17:37,741 INFO mapreduce.JobSubmitter: Executing with tokens: []
2024-11-05 14:17:37,951 INFO conf.Configuration: resource-types.xml not found
2024-11-05 14:17:37,952 INFO resource.ResourceUtils: Unable to find 'resource-ty
pes.xml'.
2024-11-05 14:17:38,028 INFO impl.YarnClientImpl: Submitted application applicat
ion_1730809269004_0007
2024-11-05 14:17:38,067 INFO mapreduce.Job: The url to track the job: http://h01
:8088/proxy/application_1730809269004_0007/
2024-11-05 14:17:38,068 INFO mapreduce.Job: Running job: job_1730809269004_0007
2024-11-05 14:17:44,192 INFO mapreduce.Job: Job job_1730809269004_0007 running i
n uber mode : false
2024-11-05 14:17:44,193 INFO mapreduce.Job:  map 0% reduce 0%
2024-11-05 14:17:52,339 INFO mapreduce.Job:  map 50% reduce 0%
2024-11-05 14:17:55,381 INFO mapreduce.Job:  map 100% reduce 0%
2024-11-05 14:17:58,429 INFO mapreduce.Job:  map 100% reduce 100%
2024-11-05 14:17:58,437 INFO mapreduce.Job: Job job_1730809269004_0007 completed
```





任务四：交易行为影响因素分析

我考虑的是过去7天日均收益率（参考支付宝收益率表 fd_day_share_interest ）对申购 / 赎回行为的影响，将日均收益率分为了5个区间，利用任务一获得的每日资金流入流出量，算出过去7天日均收益率在每个区间中所有日期的平均资金流入流出量。

任务概述

根据 fd_day_share_interest 表，编写一个MapReduce程序，将过去 7 天的每日收益率 mfd_7daily_yield 划分为不同的区间（4到4.5、4.5到5、5到5.5、5.5到6、大于6，五个区间分别标号1、2、3、4、5），统计每个区间下的日均资金流入和流出总量，其中每天的资金流入和流出量是由任务一得出的。

设计思路

1.Mapper：

- 读取输入数据并提取每行的收益率（yield）、资金流入量（inflow）和资金流出量（outflow）。
- 根据收益率将数据分为不同的区间（如 4%-4.5%、4.5%-5% 等）。定义了 `getInterval()` 方法来为每条记录分配一个收益区间。
- 将每条记录的流入量（正值）和流出量（负值）分别输出，键是区间编号，值是资金流量。

2.Reducer：

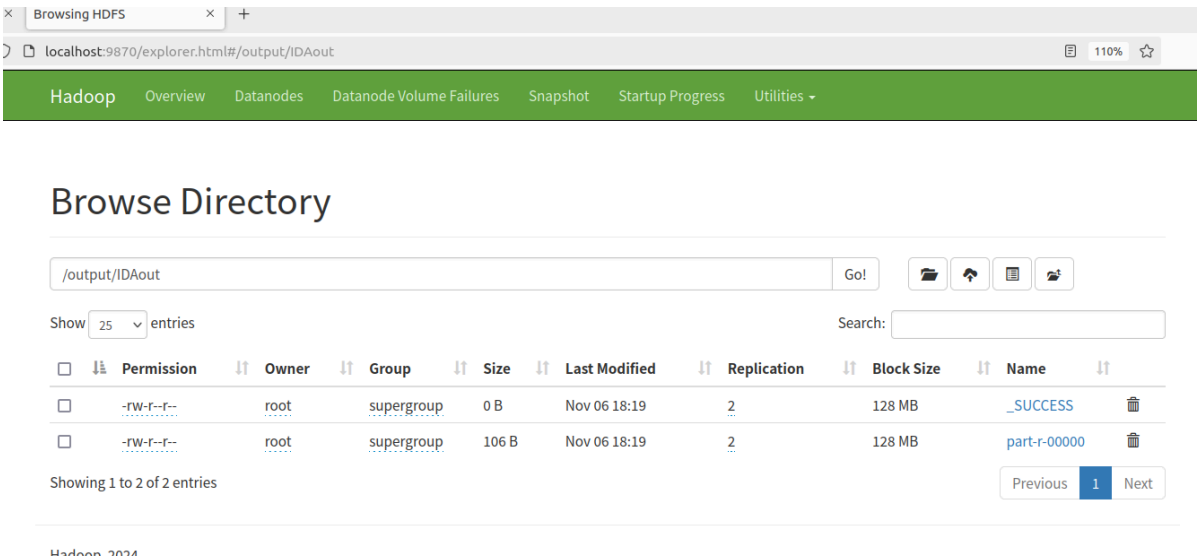
- 聚合来自 Mapper 的数据，分别计算每个收益区间的资金流入和流出量的总和。
- 计算日均流入和流出量，通过将总流入和总流出除以记录的数量来得到平均值。
- 输出每个收益区间的平均流入和流出量。

3.main：

- 配置并启动 MapReduce 作业，设置输入输出路径和相关参数，执行作业并返回结果。

运行结果

```
root@h01:/usr/local/hadoop# ./bin/hadoop fs -cat /output/IDAout/part-r-00000
1       106182558,115049208
2       64182993,53684732
3       106703560,84379756
4       148682813,124718609
5       189052899,97614010
root@h01:/usr/local/hadoop#
```



主要命令行

```
#开启各节点
~$ sudo docker start h01 h02 h03 h04 h05
~$ sudo docker exec -it h01 /bin/bash

#查看各容器
sudo docker ps

#在主节点中开启运行
root@h01:/usr/local/hadoop/sbin# ./start-all.sh

#代码传入容器
sudo docker cp ~/bigdata_workspace/lab2/hadooplab2/target/hadooplab2-1.0-SNAPSHOT.jar e5c3902c27e5:/usr/local/hadoop

#查看.jar目录
jar tf hadooplab2-1.0-SNAPSHOT.jar

#运行MapReduce程序
./bin/hadoop jar hadooplab2-1.0-SNAPSHOT.jar com.example.ActiveDay /input/user_balance_table.csv /output/ADout

#查看output目录
./bin/hadoop fs -ls /output/ADout

#查看输出结果
./bin/hadoop fs -cat /output/ADout/part-r-00000

#递归删除output目录下的输出
./bin/hadoop fs -rm -r /output/ADout

#更改输出文件的名字
./bin/hdfs dfs -get /output/FIOout/part-r-00000 /tmp/new_name.txt

#将输出文件传回主机
sudo docker cp e5c3902c27e5:/tmp/part-r-00000_AD ~/bigdata_workspace/lab2/hadooplab2/output
```

改进

任务4中使用到的任务1的输出结果是人为添加到csv表格中的，这种方法一方面有损于数据的精度，另一方面如果数据量较大或是数据较复杂实现起来会比较困难。

然而在实际操作当中不知道运行一个MapReduce如何有两个输入文件，试行过`./bin/hadoop jar hadooplab2-1.0-SNAPSHOT.jar com.example.InterestDayAverage /input/mfd_bank_shibor.csv /output/FIOout/part-r-00000 /output/IDAout`，但是会报错，会将

`/output/FIOout/part-r-00000`当作输出目录。

后续改进可以考虑先编写程序进行两个输入文件的合并，然后再进行MapReduce。