

哈尔滨工业大学(深圳)

# 《数据库》实验报告

## 实验五

### 查询处理算法的模拟实现

学 院: 计算机科学与技术

姓 名: 李怡凯

学 号: 190110419

专 业: 计算机科学与技术

日 期: 2021-11-7

## 一、 实验目的

理解索引的作用，掌握关系选择、连接、集合的交、并、差等操作的实现算法，理解算法的 I/O 复杂性。

## 二、 实验环境

*阐述本次实验的环境。*

操作系统：Windows10

IDE：Visual Studio 2019 Community

## 三、 实验内容

基于 ExtMem 程序库，模拟数据库系统在磁盘和内存上的调度，实现线性搜索算法，两阶段多路并归排序算法，基于索引的关系选择算法，基于排序的连接操作算法，基于排序的两趟扫描算法实现集合操作等任务。

## 四、 实验过程

*对实验中的5个题目分别进行分析,用流程图或伪代码对核心代码进行讲解,用自然语言描述解决问题的方案，并给出程序正确运行的结果截图。*

### (1) 实现基于线性搜索的关系选择算法

问题分析：

本任务需要通过线性遍历表的方式实现搜索算法，因为内存有限，一次无法完全读入整个表，所以需要分块读入并搜索。通过每次读取一个数据块，在数据块上进行搜索，当搜索到符合条件的元组时，将其写入写入块，如果写入块满，

则将写入块写入磁盘。

核心代码分析：

程序核心部分流程图如图 1 所示。

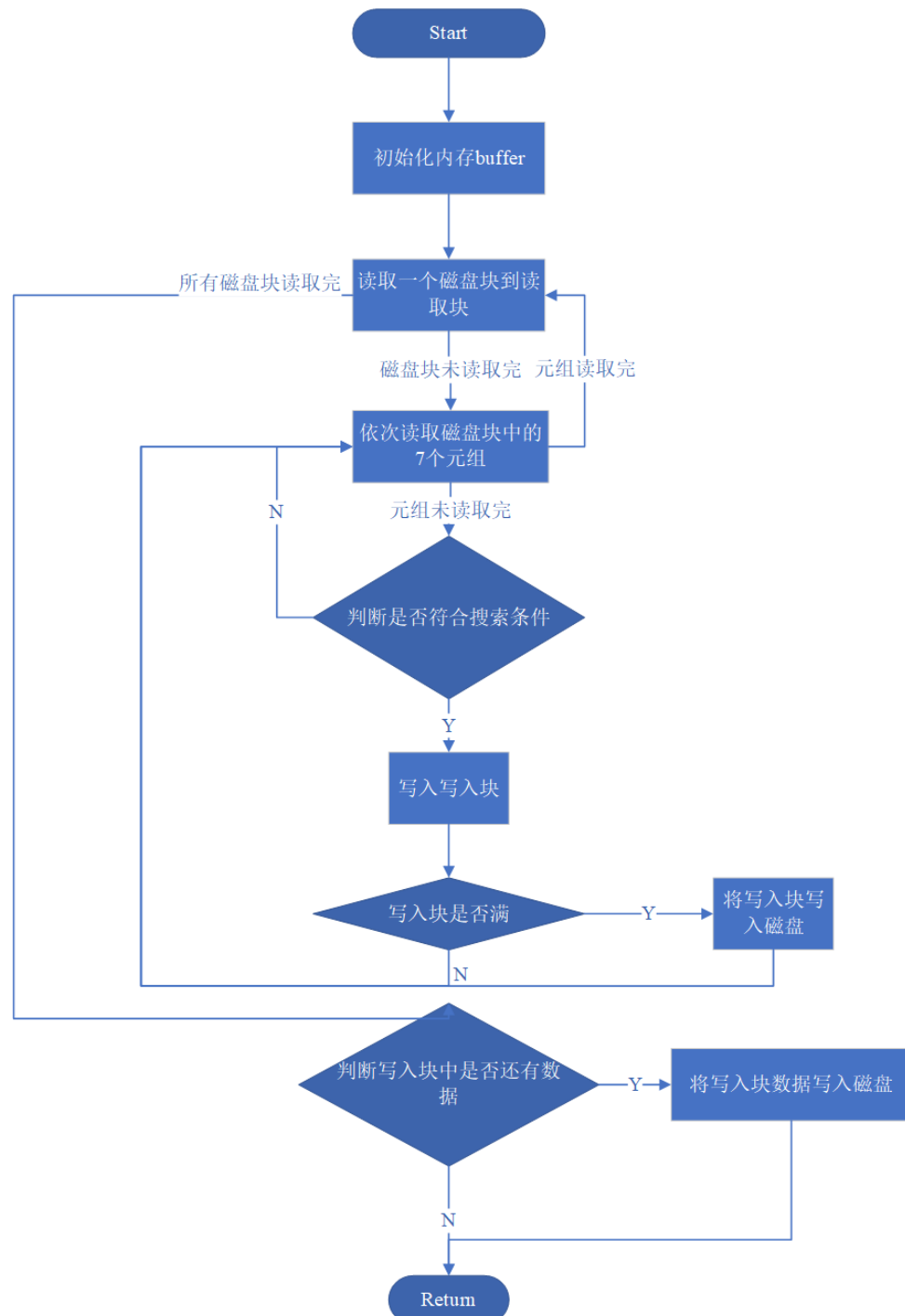


图 1 线性搜索算法流程图

实验结果：

实验结果如图 2 所示。

## =====

## 基于线性搜索的选择算法S\_C=130

## =====

```
读取磁盘块17
读取磁盘块18
(130, 4208)
(130, 5202)
读取磁盘块19
读取磁盘块20
读取磁盘块21
读取磁盘块22
读取磁盘块23
读取磁盘块24
读取磁盘块25
读取磁盘块26
读取磁盘块27
(130, 5083)
读取磁盘块28
读取磁盘块29
读取磁盘块30
读取磁盘块31
读取磁盘块32
读取磁盘块33
读取磁盘块34
读取磁盘块35
读取磁盘块36
(130, 4983)
读取磁盘块37
读取磁盘块38
读取磁盘块39
(130, 4371)
读取磁盘块40
读取磁盘块41
读取磁盘块42
读取磁盘块43
读取磁盘块44
(130, 4180)
读取磁盘块45
读取磁盘块46
读取磁盘块47
读取磁盘块48
数据存放在:100
搜索得到的元组个数:6
IO次数:33
```

图 2 任务一线性搜索算法实验结果

## (2) 实现两阶段多路归并排序算法（TPMMS）

问题分析：

本任务需要对关系进行排序，由于内存有限无法完全放入关系，所以需要采用两阶段多路并归排序，首先将关系划分为子集，使得每个子集能够被放入内存中进行排序，然后再对已排序的子集进行并归，从而得到经过排序的整个关系。

核心代码分析：

程序核心代码流程图如图 3 所示。

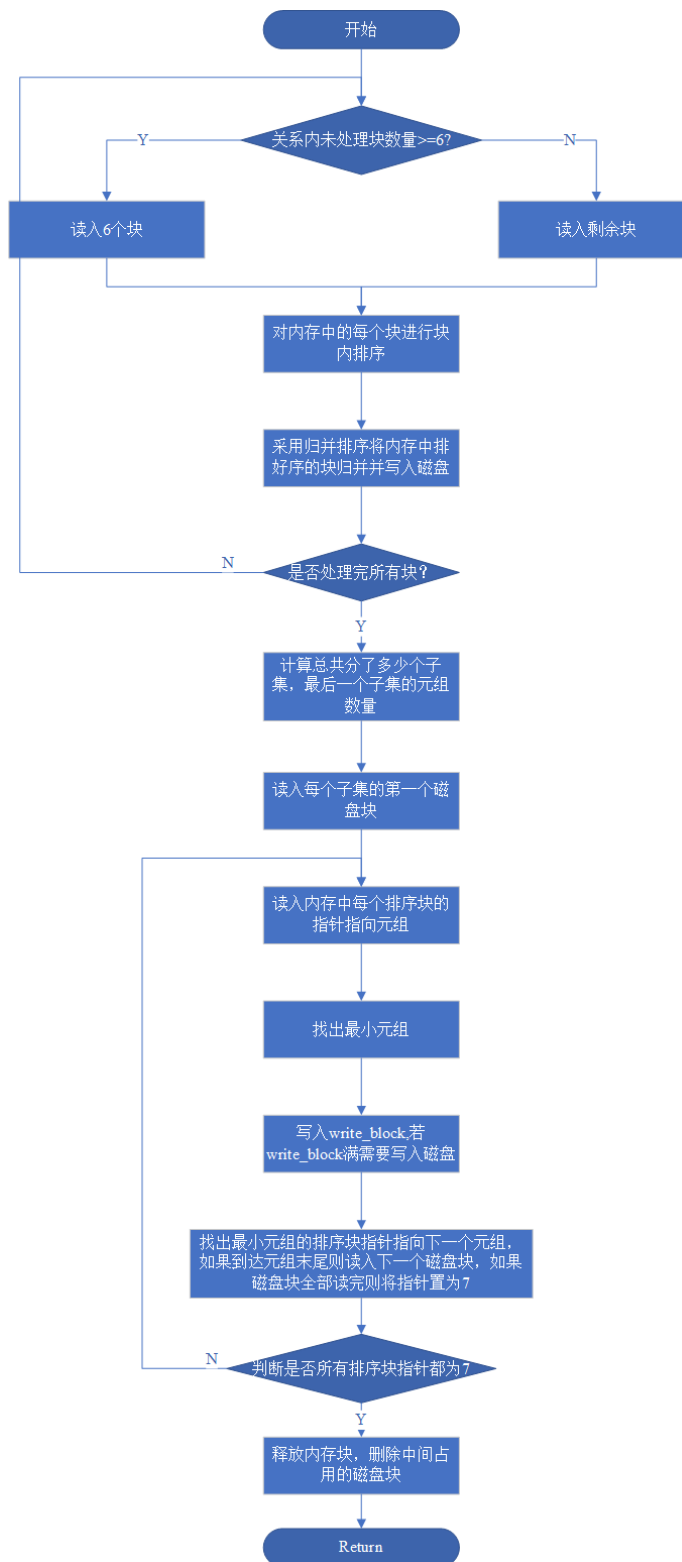


图 3 TPMMS 实现核心代码流程图

实验结果：

```
=====
两趟扫描排序算法
=====
IO次数为288
```

实验结果如图 4 所示

### (3) 实现基于索引的关系选择算法

问题分析：

本任务需要在任务二已经排好序的关系上建立索引，并利用索引实现关系选择。首先通过遍历每个磁盘块，找到磁盘块上第一个与前面所有元组在待索引字段上不同的元组，作为该文件的索引，然后在关系选择的时候首先查找索引，定位到元组所在的磁盘块范围，再在该范围的磁盘块内查找元组。

核心代码分析：

核心代码流程图如图 5，图 6 所示。

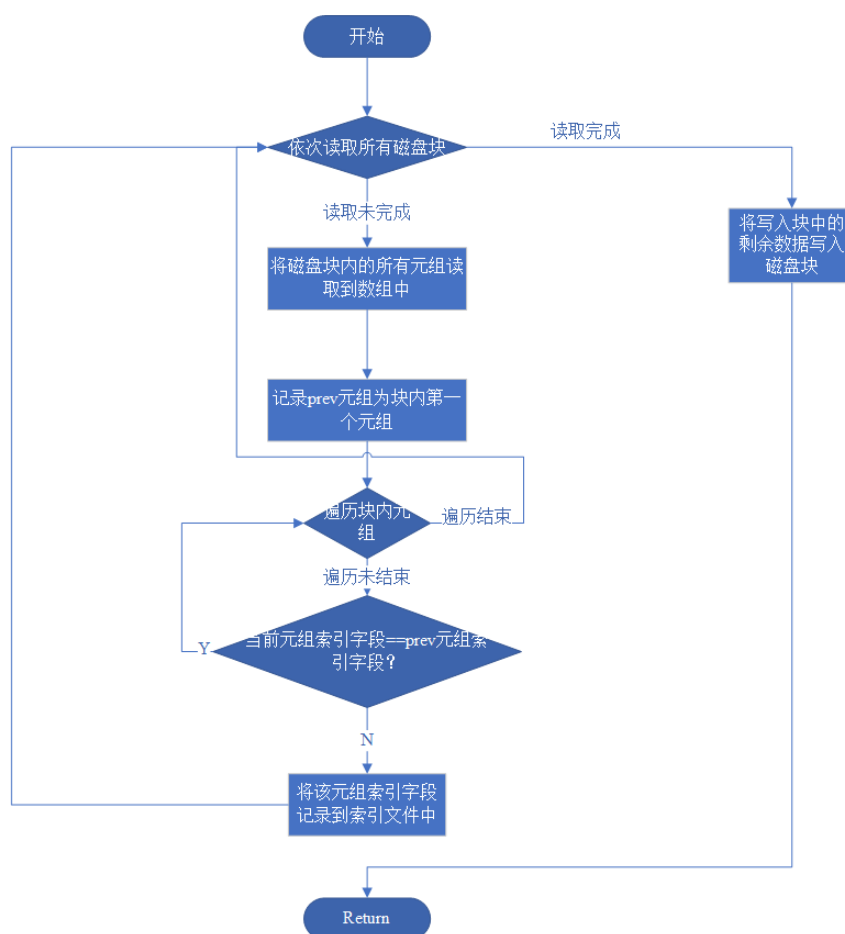


图 5 建立索引块核心流程图

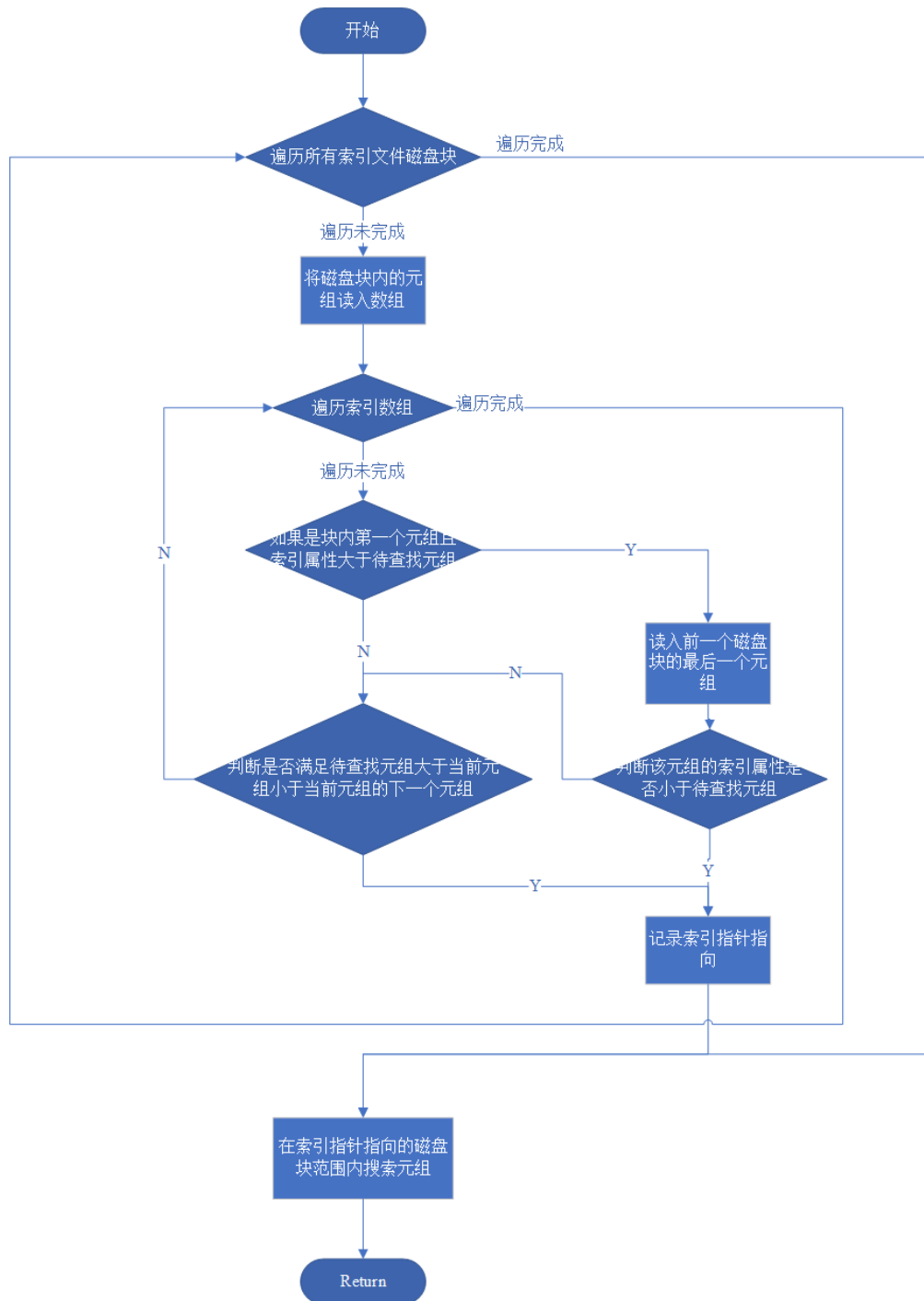


图 6 利用索引搜索元组流程图

实验结果：

实验结果如图 7 所示。

## 基于索引的搜索算法

## 建立索引

## 基于索引进行搜索

(X=130, Y=4208)

(X=130, Y=5202)

(X=130, Y=5083)

(X=130, Y=4983)

(X=130, Y=4371)

(X=130, Y=4180)

IOTimes: 10

图 7 基于索引的搜索算法运行结果

**(4) 实现基于排序的连接操作算法 (Sort-Merge-Join)**

问题分析：

本任务要实现基于排序的连接操作算法，对两个元组实现连接操作，通过选定一个关系为驱动关系，对两个元组进行遍历：如果两个元组符合连接条件则进行连接，并读入非驱动关系的下一个元组；如果驱动关系元组较大，则需要继续读入非驱动关系的下一个元组；如果非驱动关系元组较大，则需要首先判断当前元组和前一个元组的条件属性否相同，如果相同再需要再判断前一个元组是否发生了连接，如果发生了连接则需要回溯非驱动元组，进行连接操作，如果没有发生连接则可以读入驱动关系的下一个元组。

核心代码分析：

核心代码流程图如图 8 所示。

实验结果：

实验结果如图 9 所示。



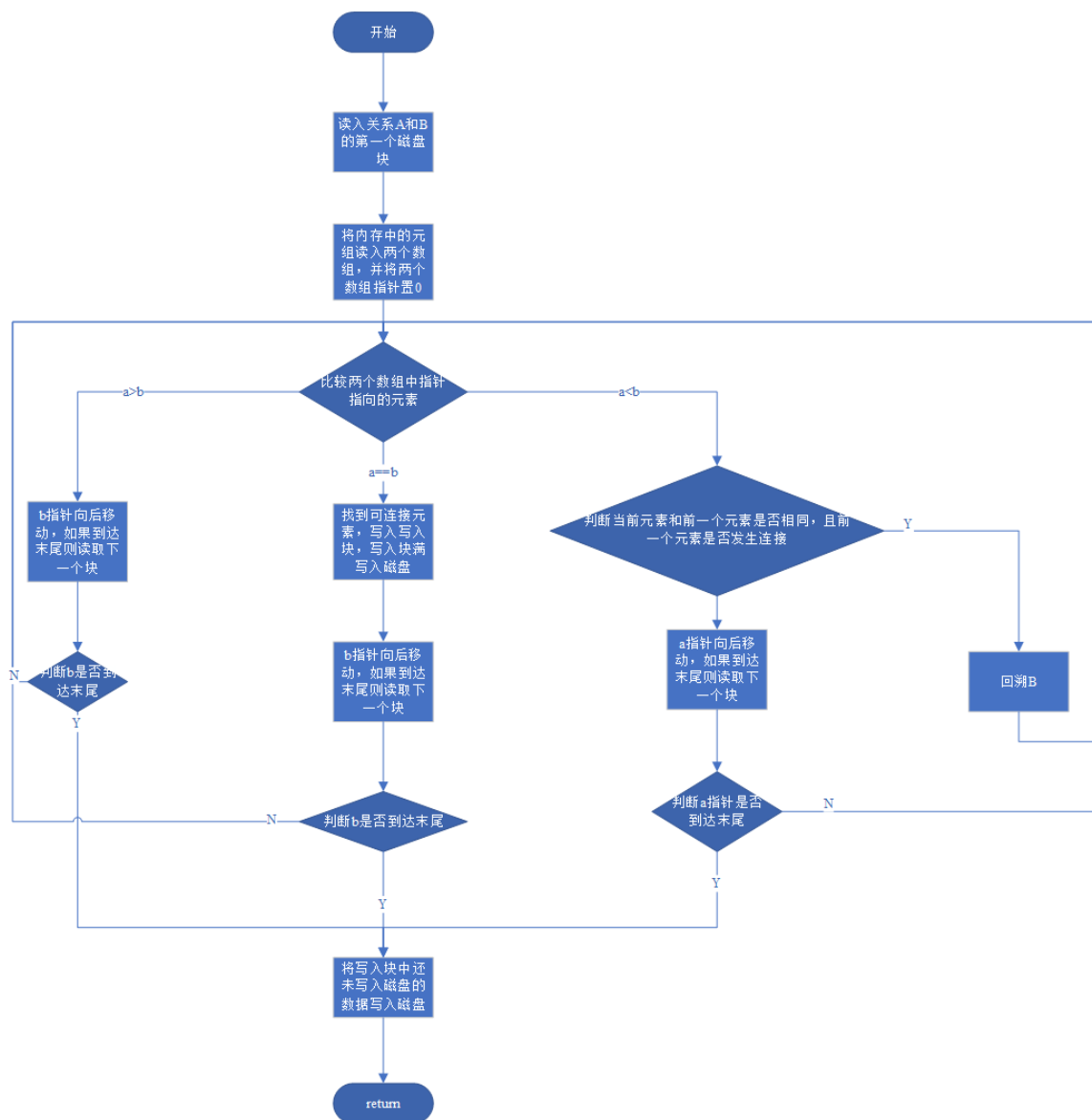


图 8 连接操作核心代码流程图

连接操作次数:347

图 9 基于排序的连接操作实验结果

(5) 实现基于散列的两趟扫描算法，实现交、并、差其中一种集合操作算法  
问题分析：

基于排序实现关系的交操作，与连接操作相似，同样选定一个驱动关系，对两个关系进行遍历，不同的地方在于，当遍历到两个关系中的元组的条件属性相同时，需要回溯非驱动关系直到第一个条件属性为当前值的位置，然后进行循

环遍历，寻找是否存在一个元组和驱动关系中的元组相同。

核心代码分析：

程序核心代码如图 10 所示。

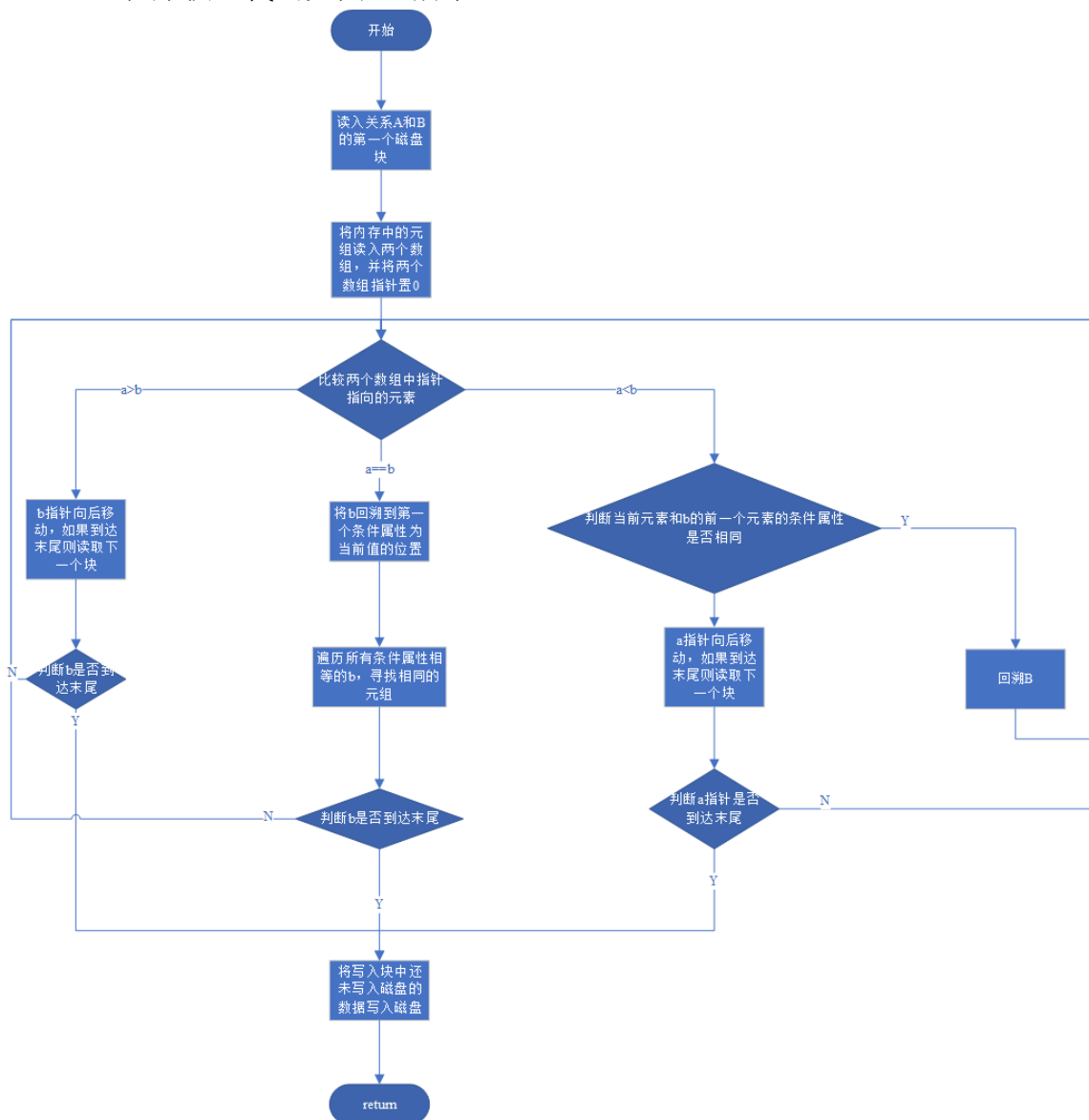


图 10 基于排序的交操作核心代码流程图

实验结果：

程序运行结果如图 11 所示。

```
=====
基于排序的关系交算法
=====
(122, 4749)
(123, 4362)
(125, 4741)
(126, 4031)
(128, 4249)
(132, 4759)
(134, 4864)
(137, 4986)
(137, 4020)
交集元组数量:9
```

图 11 基于排序的交操作运行结果

## 五、 附加题

*对剩余的 2 种集合操作进行问题分析，并给出程序正确运行的结果截图。*

## 六、 总结

*总结本次实验的遇到并解决的问题、收获及反思。*

本次实验通过在 ExtMem 第三方库的支持下，模拟实现数据库中的线性查找、两趟扫描排序算法、基于索引的搜索算法、基于排序的连接操作、基于排序的交操作，进一步加深了对数据库实现的理解，以及操作系统中的内存与磁盘管理的理解。

在本次实验中遇到的问题主要是对第三方库 ExtMem 的理解不到位导致的内存访问问题。ExtMem 库中读取磁盘块到内存的逻辑是在内存中新建一个块，然后读取磁盘内容到该块上，并不需要新建一个块再进行读取操作；ExtMem 库中写磁盘块的操作会将内存块中的内容写入磁盘，同时释放该内存块。上述两个问题通过阅读 API 无法得知，必须通过阅读源代码才能发现，容易导致错误理解，进而导致程序出现各种各样的内存访问导致的问题。