DataBase Lab3 Report

-521120910259
-郭培宇

1 Designing Part

基于 lab1 和 lab2 的数据库中组织方式、缓冲池规范、数据骨架等构建后，本次 lab3 主要需要我们实现为 SimpleDB 编写一套操作符（operators），实现表修改（如插入和删除记录）、选择、连接和聚合。这些操作将建立在前两次 lab 的基础之上，构建一个能对多个表执行简单查询的数据库系统。

1.1 Exercise1

本次 exercise1 的主要任务是写 filter 和 join 两个操作符。这两个操作符，它们将使对于数据表的 query 操作能够执行比表扫描（table scan）更为高效的。

Filter	该操作符只返回满足谓词的元组
Join	该操作连接两个子元组

在此部分我需要修改五个 java 文件，分别是 join,filter,hashequijoin,predicate 和 joinpredicate，具体方法是在 join 和 filter 文件中添加私有成员变量并修改构造函数以加入 JoinPredicate p，目的是表示连接操作的谓词，用于指定连接条件，修改 open(),close(),rewind() 方法分别打开、关闭和重置迭代器，最后补全 fetchNext()方法，在 join 文件这种通过嵌套循环连接的算法不断从左侧和右侧的子迭代器中获取元组并根据连接谓词判断是否满足连接条件，在 filter 文件中遍历子迭代器中的元组用谓词 p 进行筛选，返回第一个符合条件的元组。最后在 predicate 和 JoinPredicate 文件中分别补全具体的初始化代码和值返回代码。

1.2 Exercise2

在此部分我完成了数据库中的聚合（aggregate）操作，具体来说是五种 SQL aggregates(`COUNT`, `SUM`, `AVG`, `MIN`,`MAX`)，关于这五种聚合，我的方法是在 aggregate 中加入以下五个私有变量：

`private final Aggregator.Op op;`

这个变量用来存储聚合操作的类型，如 SUM、AVG、MAX、MIN 等。

`private DbIterator child;`

这个变量是输入的子迭代器，即提供元组数据的源。

`private final int af, gf;`

af 存储了聚合字段的索引，即进行聚合计算的字段在元组中的位置。

gf 存储了分组字段的索引，即用于分组的字段在元组中的位置。

`private final Aggregator aggregator;`

这个变量是 Aggregator 对象，用于执行聚合操作。

`private DbIterator it;`

这个变量是输出的迭代器，用于迭代聚合后的结果元组。

同时添加三个函数，作用分别是：

`private simpledb.Type getType(int field)`

这个方法用于获取指定字段在输入子迭代器中的类型。

private simpledb.Type getAggregateType()

这个方法用于获取聚合字段的类型。

private simpledb.Type getGroupType()

这个方法用于获取分组字段的类型。

而在 IntegerAggregator 和 StringAggregator 中除以上还加入了一个哈希表以临时存储聚合结果。

1.3 Exercise3

在此部分的任务是完成支持修改表格的方法。在单个 page 和 file 的层面上操作：tuples 的添加和删除。

因为 heappage 和 heapfile 的框架已较为完善，所以我的具体工作量是在 heapfile 文件中的插入 tuples 完成以下几点：

创建 isNewPage 数组	isNewPage 数组用于标记是否创建了新的页面。
获取页面流	使用 getPagesStream 方法获取当前具有读写权限的所有页面。
过滤和查找页面	使用流式操作对页面进行过滤，只保留有空插槽的页面。（getNumEmptySlots() > 0）。然后使用 findFirst 方法获取第一个符合条件的页面。如果没有符合条件的页面，则创建新页面。如果没有可用的页面，则抛出 DbException 异常。
插入 Tuple	一旦找到了合适的页面，Tuple 被插入到页面中
写入页面并返回结果	如果 isNewPage 被设置为 true，表示新页面已经创建，则调用 writePage 方法将新页面写入到磁盘中。最后，将被修改的页面（可能是新创建的页面，也可能是已存在的页面）添加到 ArrayList<Page>对象中，并将其作为方法的返回值返回。

在 heappage 文件中的插入 tuples：

检查页面空槽位	首先，方法检查页面是否有空槽位可用。如果页面已满则抛出 DbException 异常。
检查 TupleDesc 匹配	方法接着检查传入的 Tuple 与页面的 TupleDesc 是否匹配。如果它们不匹配，则抛出 DbException 异常，表示 TupleDesc 不匹配，不允许插入。
插入 Tuple	选择第一个空的槽位进行后续插入。

而两个文件的 deleteTuples 较为简单，不再赘述。

1.4 Exercise4

在此部分完成最后的 insert 和 delete 的修改，在 insert 和 delete 文件中都加入以下三个

私有变量：

- TransactionId 用于确定删除操作所属的事务。
- DbIterator 用于从子操作符读取要删除的元组。
- TupleDesc 用于描述 Delete 操作的结果元组的结构。

并按照 exercise1 到 3 的思路补全两个文件中的 fetchNext 等函数。

2 实验心得

此次 lab3 基于 lab1 ——> lab2 的基础上继续给 SimpleDB 编写一套操作符实现数据表的修改（如插入和删除记录）、选择、连接和聚合等操作。

本次 lab 的难点可能是需要尝试一些额外导入的 jdk 不包含的 package 或者方法，以及进一步熟悉 java 对于异常处理的代码规范及处理方法，并且因为此次实验基于整个 src 文件夹中各个文件的代码，需要我们对 lab1 和 lab2 中很多步骤非常熟悉，因此在思路构建方面就需要大量的复习之前代码或者看新代码的时间。同时此次编写操作符的一系列文件 debug 难度和复杂度也相较于之前有增加。

但是整体做完感觉收获还是非常多，我认为我对这 simpleDB 的底层架构到流程实现的认知又进一步清晰了。