|  |  |  |
| --- | --- | --- |
| 2020春《数据库系统》实验报告 | | |
| 实验3：查询器实现 | | |
| 姓名：陶飞宇 | 学号：1170300720 | 班级：1737301 |

## 1.实验分析

本次实验以现有框架为基础，相比于第二次实验，除了要熟悉本身框架内的函数，熟读每个函数的.h文件之外，还对我们对于字符串操作有了进一步的要求。

## 2.实验过程

首先要感谢群里各位同学的分享，对本身有错的框架提前做了一些更改：

①去掉了一些File前面的const，否则在编译时会报错

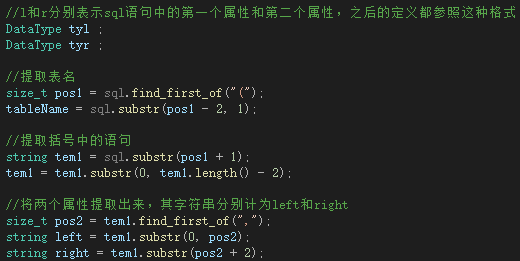
②在insert所有元素之后调用的flush函数，否则不会立即写入文件，导致之后的链接操作中没有任何元组出现。

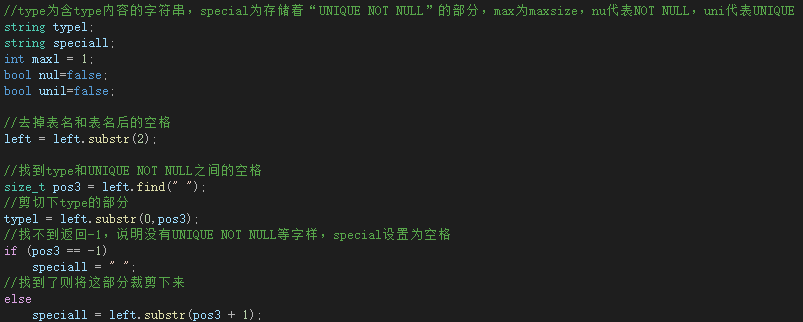
③在同学的帮助下明白了多看.h文件才能明白整个框架的结果，包括实验指导书中没有提及的部分，比如文件和页的迭代器

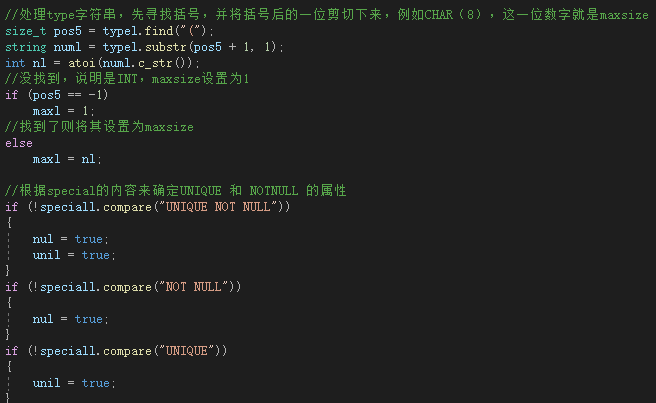
下面来分析各个函数的构建方法：

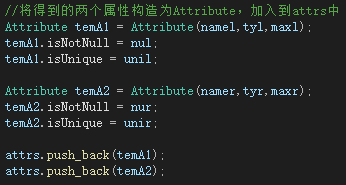
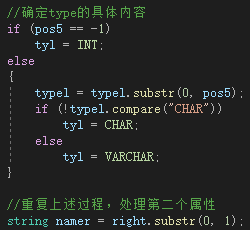
TableSchema::fromSQLStatement：该函数要求我们根据SQL语句来创建一张表，实际上是从sql语句中提取表名和属性的各项值。主要考验对于字符串裁剪的运用，这里全程使用find函数和substr函数。

在注释中已经说明了构造属性的各个步骤

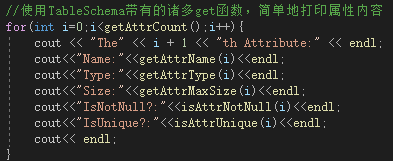






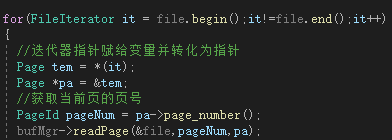


TableSchema::print()：该函数要求打印表的全部属性，使用TableSchema带有的各种get方法即可实现

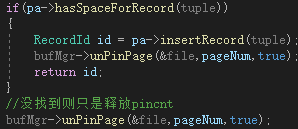


HeapFileManager::insertTuple：该函数要求根据SQL的insert语句实现元组插入，其内容和fromSQLStatement是差不多的。

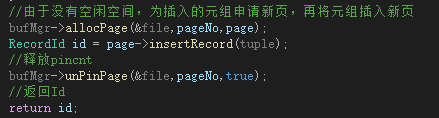
我们首先遍历整个文件，将其中的页面逐个读取到缓冲区中：



之后我们调用函数，在页面中寻找空闲空间，如果找到了则直接将其写入缓冲区，没找到则进行下一次循环，值得注意的是，readPage增加了缓冲区页框的pinCnt值，要在完成读取后调用unPinPage将其释放：



如果找完了所有的现有页面都没找到空闲空间，则调用allocPage函数分配一个空页将元组插入到空页之中：

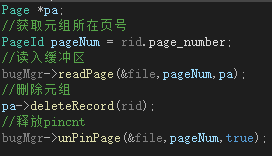


完成所有的插入操作后，还应使用flush函数将上述的更改写回文件，故而我们在main函数中增加如下字段：

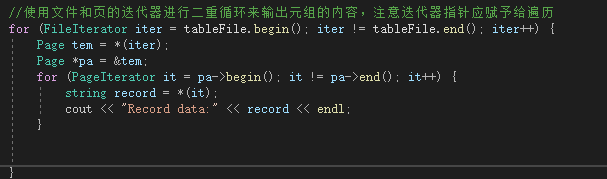


位置位于两个insert循环之后。

HeapFileManager::deleteTuple：该函数要求我们删除一个元组，由于RecordId类里面有元组所在页号的成员变量，故而可以十分容易的进行元组删除：



TableScanner::print()：该函数要求我们打印所有的元组信息，我们简单地使用二重循环来实现它，分别使用了文件和页的迭代器：

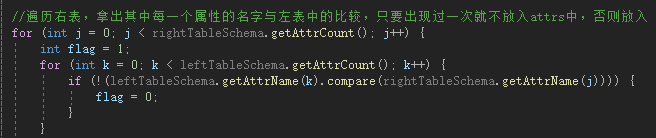


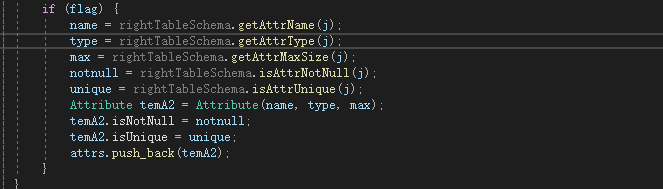
JoinOperator::createResultTableSchema：该函数要求我们将进行连接的两个关系的属性合并。我们分为两步：

第一步：



第二步：

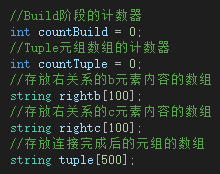




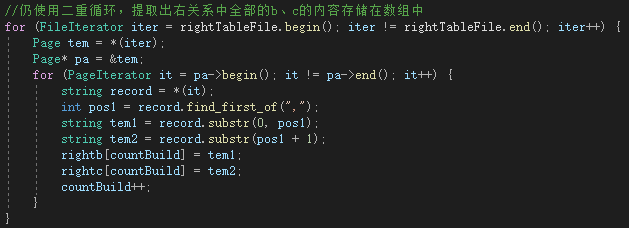
从而实现了属性的合并。

OnePassJoinOperator::execute：该函数要求我们实现一趟合并算法，分为build和probe阶段：

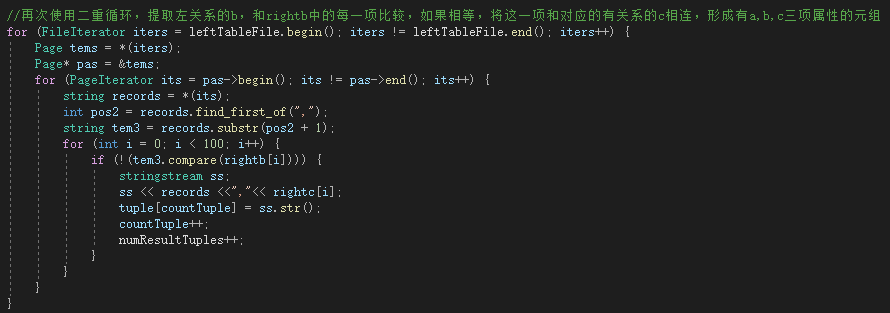
首先我们定义中间数组和计数器：



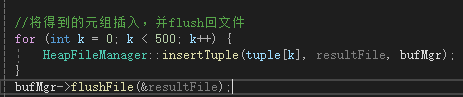
build阶段：



probe阶段：

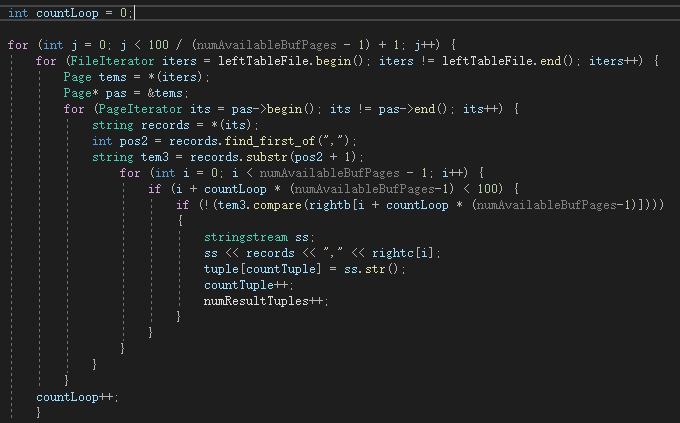


最后将构建出的元组插入到缓冲区中，并将其flush回文件：



NestedLoopJoinOperator::execute：该部分要求我们实现嵌套循环算法，和一趟算法的逻辑基本一致，只是在比较时，一次只取numAvailableBufPages-1个元组与其比较。

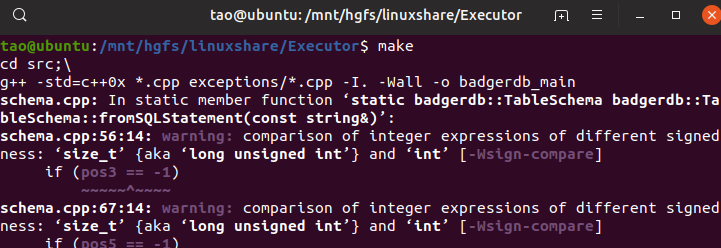
不同的部分：



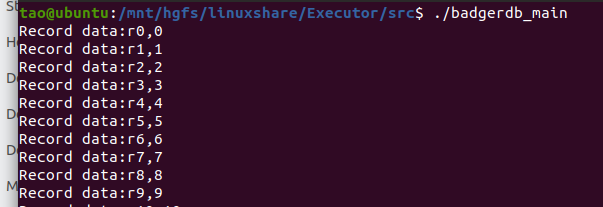
## 3.实验结果

使用make指令进行编译，虽然会抛出一些main函数中的异常，与同学讨论后发现是gcc版本的问题，不影响最终程序的运行：

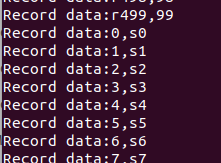
使用make，无error：



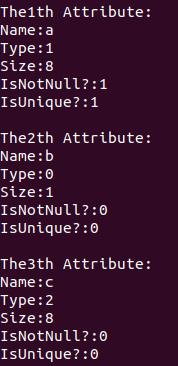
插入元组：

左关系： 

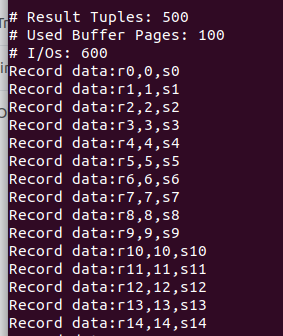
右关系：



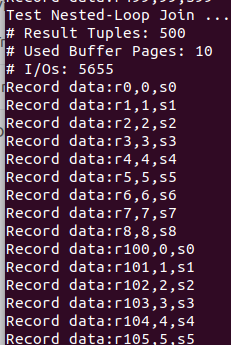
打印属性：



一趟连接：



嵌套连接：



可以看到嵌套连接每9个元组出现了断层，符合设计。

## 4.心得体会

通过重复的阅读框架代码，完成了需要构建的酒个函数，学习了C++语法下字符串裁剪的操作，收获很大