|  |  |  |
| --- | --- | --- |
| 2020春《数据库系统》实验报告 | | |
| 实验3：查询器实现 | | |
| 姓名：王智义 | 学号：1160300714 | 班级：1737301 |

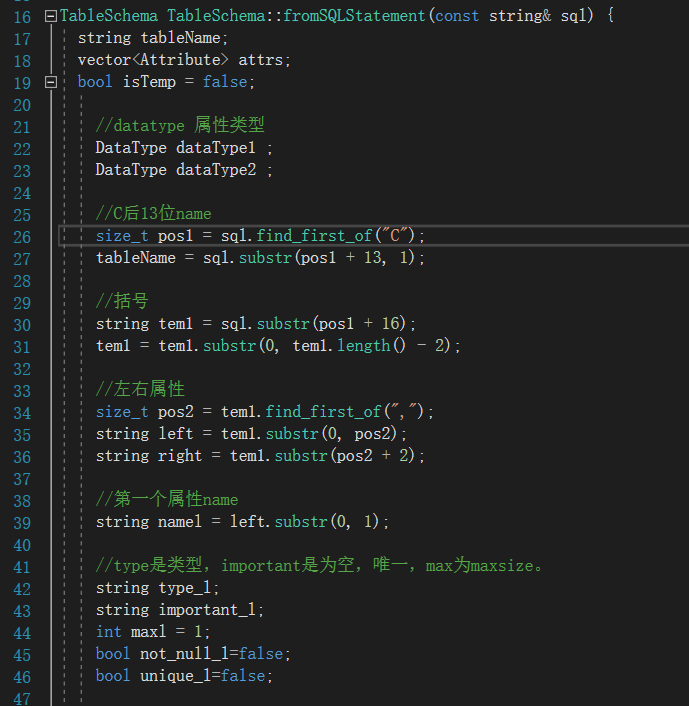
## 1.实验分析

实验三需要熟读给的.h的函数，因为实验中不仅需要用到很多其中的函数，还需要用到实验二的部分函数比如flush等，最后需要考察对字符串切割的要求，在读取sql语句以及创建元组时，需要对字符串进行切割，读取其中内容。

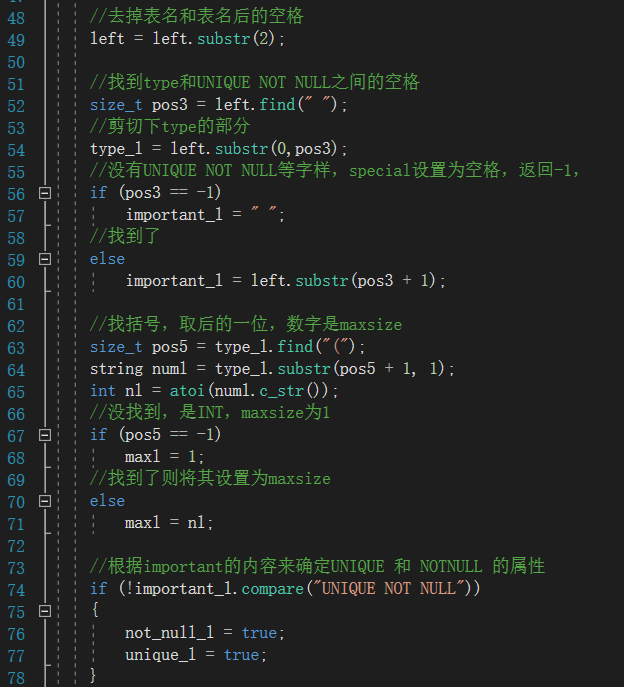
## 2.实验过程

函数解析：

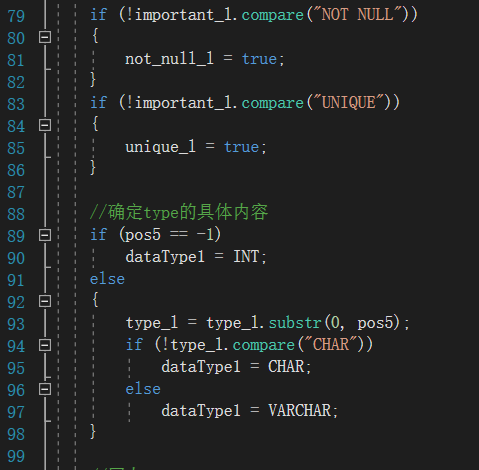
TableSchema::fromSQLStatement：该函数是从main中读取sql语句，进行字符串切割，从而提取出name 和 type等值。



从CREATE的第一个C开始，13位后是表名name，16位后市左括号（，等等，提取出所有的属性，类型，表名。

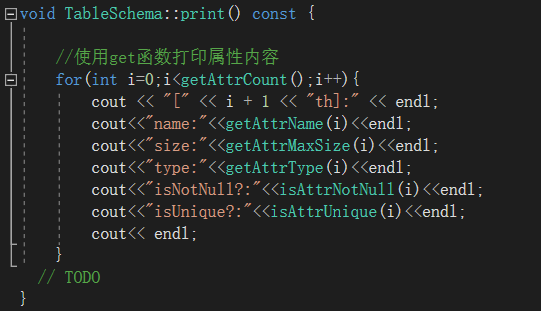


在提取类型时，如果后面没有数字，说明是INT型，若有数字说明是CHAR或者VARCHAR。



以上是对第一个sql语句的提取，第二个语句同上，就不贴截图了。

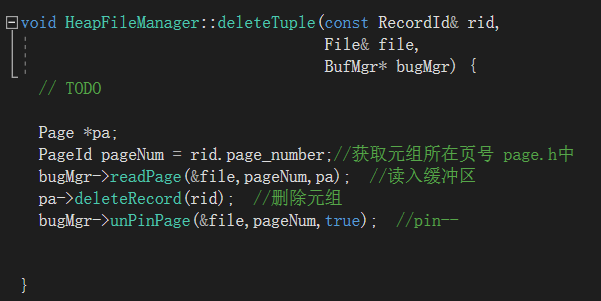
TableSchema::print：用get函数打印表的属性类型。



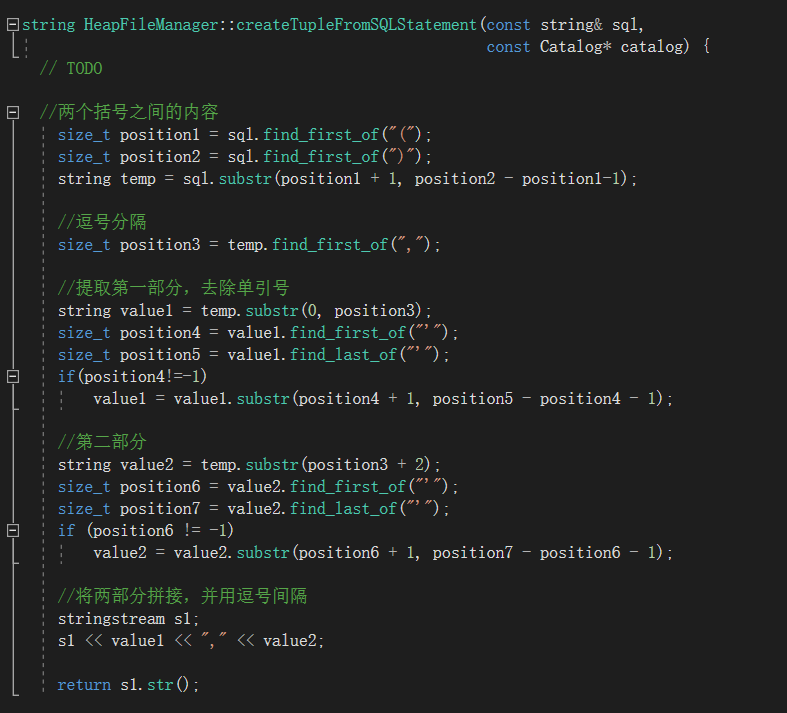
HeapFileManager::insertTuple：该函数需要插入元组，首先使用文件迭代器，不然会出错。对于搜索出空页时候，获取页号，然后插入元组，pin--，因为在搜索时pin以及++，若没有空页，也进行pin--。在for之后没有空余空间，则申请新页，再将元组插入新页，然后pin--。



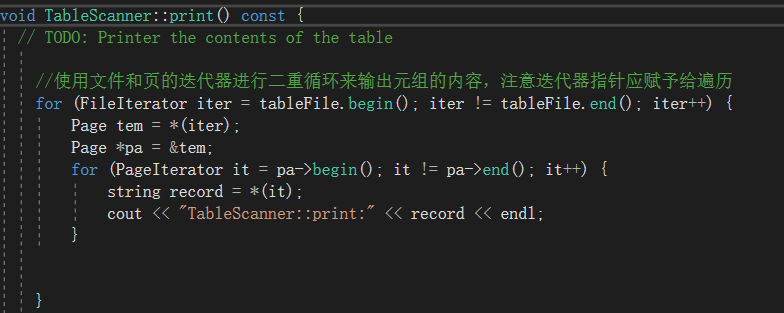
HeapFileManager::deleteTuple：该函数需要删除元组，在这里因为RecordId类里面有元组所在的页号，所以直接使用pageNum获取，然后进行删除。



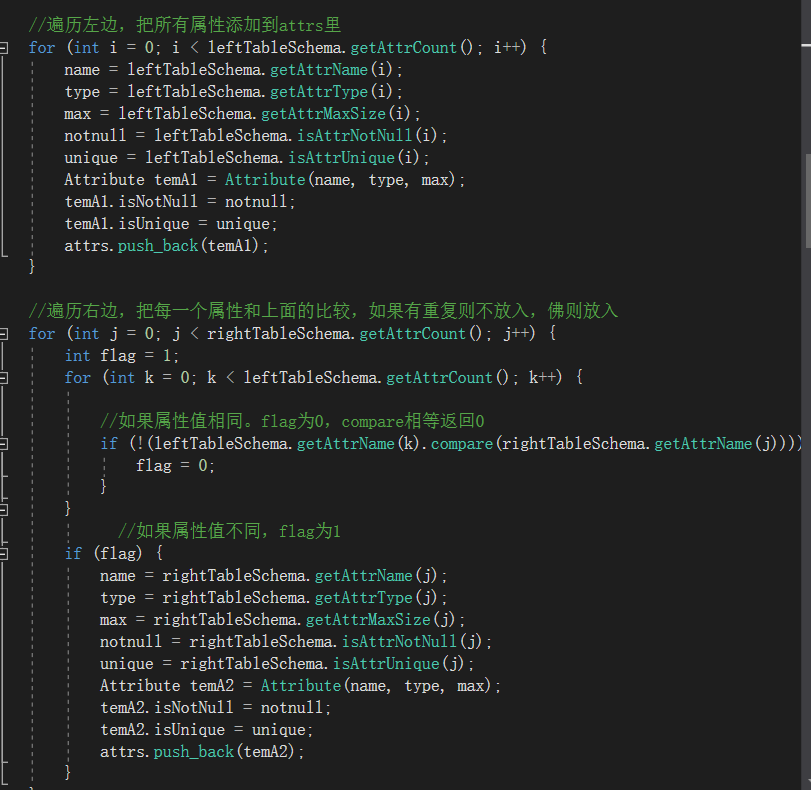
HeapFileManager::createTupleFromSQLStatement ()：该函数需要创建元组，也是使用字符串切割法，先提取括号内的内容，再用“，”分割开左右属性，然后再连接。



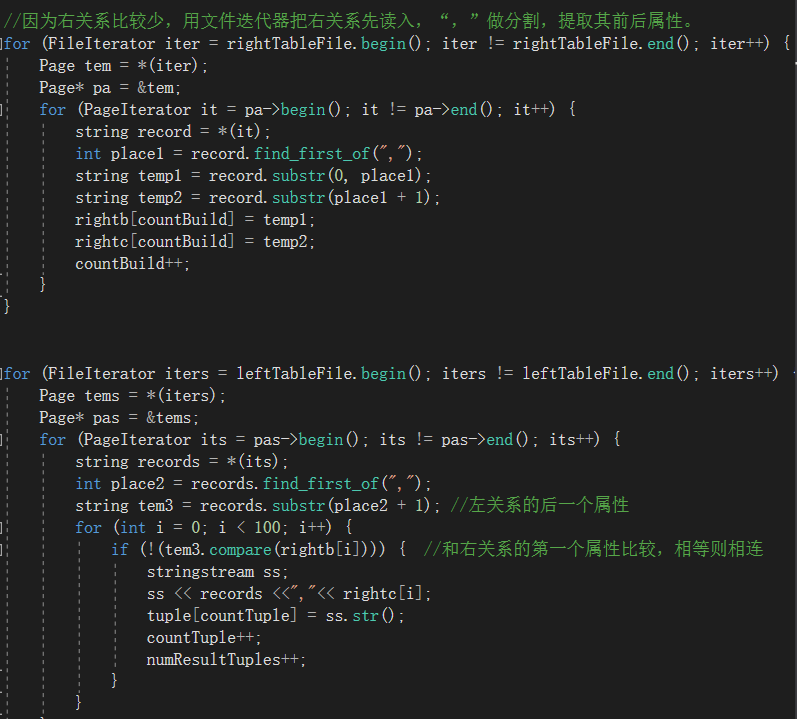
TableScanner::print：这个函数需要打印元组内容，也是使用文件迭代器



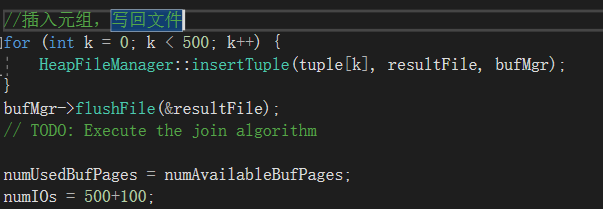
JoinOperator::createResultTableSchema：该函数需要合并两个表的属性值，首先提取左表的属性，用get函数，添加到atts中，然后提取右表的属性，和前面左表的属性进行compare对比，注意compare函数，相同时返回0，不同时返回1，若左右属性相同时，则进行连接，不同时，不连接。



OnePassJoinOperator::execute：该函数需要实现一趟连接算法。我的方法比较简单，首先右关系位100个，比较少，我们把右关系的左右属性提取出来，然后存入。之后遍历左关系，提取右属性和右关系的左属性进行比较，如果相同则连到一起。

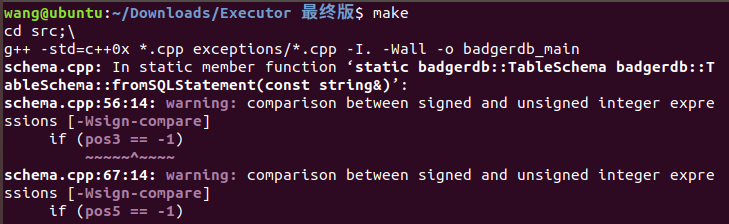


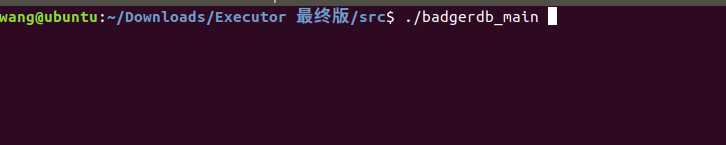
最后将元组插入，并将其flush回文件：

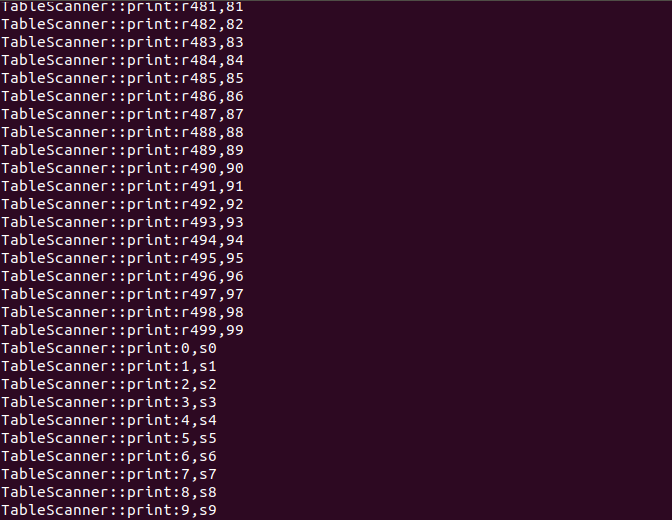


NestedLoopJoinOperator::execute：该部分需要实现嵌套循环算法，和一趟算法的逻辑差不多，但每次进入缓冲区的是umAvailableBufPages-1个元组。所以一共比较100 / (numAvailableBufPages - 1) + 1 = 11+1 = 12次

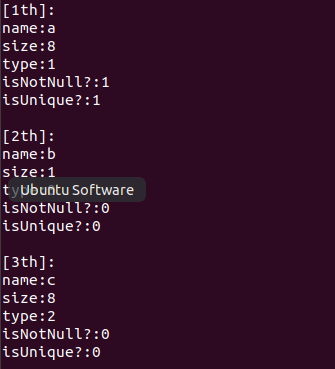
## 3.实验结果

首先make，

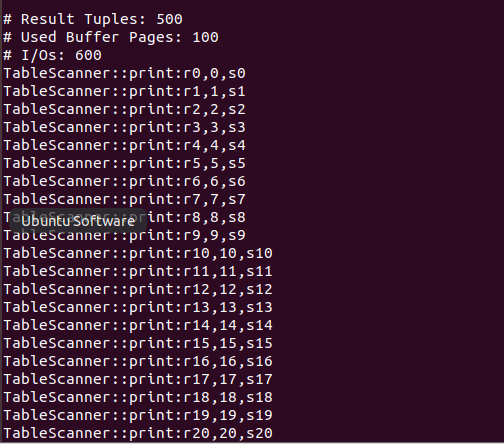
然后./运行

这里是打印左关系，500个元组，然后是右关系100个元组

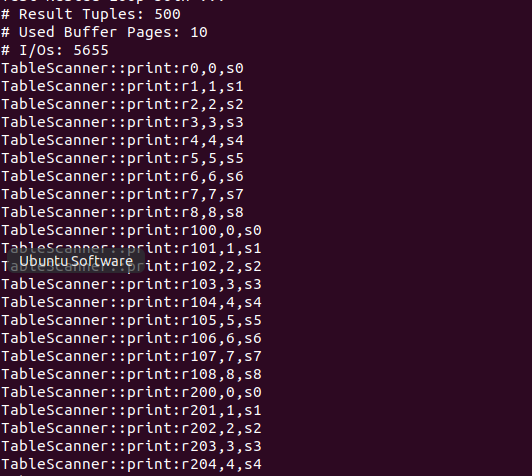
这里是表的属性类型：



一趟链接：



嵌套连接：



4.心得体会

更加熟悉了缓冲区关系和一些连接算法，实验比较难，但收获很大。