数据库系统概论-大作业报告

2018011324

计82

尤艺霖

1.设计思路

写记录管理和索引的时候是按照Redbase的思路写，分成总管理、分支管理和扫描三个层次的模块。在对系统管理和查询管理两个模块思考后发现不能完全按照Redbase的体系来，就修改了很多之前的设计，并把最后几个模块全部集成到命令解析器中。最后的开发思路是从命令解析器中寻找需求，并根据需求实现具体函数。

2.文件组成&使用方法

根目录：

- head.h 提供所有其他文件需要的封装好的类型和一些通用的函数
- main.cpp 向用户提供对于命令解析器的接口
- compare.h 提供比较函数
- report.md 大作业文档Markdown格式
- report.pdf 大作业文档pdf格式

ix目录：存放索引管理模块

- handle.h 提供对于单个索引文件的插入删除等操作
- manager.h 管理所有的索引文件
- scan.h 允许对于单个索引文件按一定规则进行扫描

parser目录：存放命令解析，提供数据库的主要功能

- parser.h 主要提供命令解析器功能和一些琐碎功能
- all.h 用来方便main.cpp进行include的文件，没有实际作用
- alter.h 用来实现alter命令下的各种操作
- createdrop.h 用来实现单个数据库下的create和drop两类操作
- insdel.h 用来实现delete和insert两类操作
- updateselect.h 用来实现update和select两类操作

rm目录：存放记录管理模块

- handle.h 提供对于单个记录文件的插入删除等操作
- manager.h 管理所有的记录文件
- scan.h 允许对于单个记录文件进行顺序扫描

ql目录：存放一些辅助文件，没有实际功能

pf目录：存放助教提供的页式文件系统，有细微改动

使用方法：

用支持c++11标准的编译器直接编译main.cpp，需要确保页式文件系统在当前计算机上可以正常工作。

运行编译后的程序即可输入命令，关闭需要通过EXIT命令关闭，否则缓存区不会被写入文件。

3.实现功能

- CREATE命令，支持创建表格/索引/Database。
- DROP命令，支持删除表格/索引/Database。
- INSERT命令，支持向表插入，同时检查约束。
- UPDATE命令，支持单表的修改，不支持WHERE子句。
- SELECT命令，支持朴素的多表查询、一定程度上支持聚集函数。
- DELETE命令，支持WHERE子句。
- ALTER命令，允许加入/删除主键/外键，并判断合法性。
- 一些其他辅助性的命令，如USE、DESC、SHOW、COPY等。

4.架构设计

分成三个模块，记录管理模块、索引模块、命令解析模块。记录管理模块和索引模块与助教提供的页式文件系统对接，并隐藏页式文件系统。命令解析模块直接操作以上两个模块完成操作和分析。

记录管理和索引两个模块使用类似的结构，来自于Redbase。每个模块都分为Manager、Handle、Scan三个层次。Manager层次实现全局的管理，Handle层次实现单个文件的管理，如插入删除等，Scan层次实现对于单个文件的按条件遍历。

命令解析模块由于时间原因没有过多思考，导致集成了太多功能。只有一个对外的类QueryParser，所有的功能都写成它的成员函数。有一个起控制作用的成员函数是对main.cpp的接口，其他的为了方便实现，分散在了各个文件里。这一部分没有什么特别的结构，基本是想到哪写到哪。

5.实现细节

在记录管理部分，使用的是相对朴素的实现。在索引部分，由于扫描的过程中不能进行B+树结构上的修改，故选择懒惰删除法，在较小范围的测试下影响不会很大，如果需要在较大数据集上测试，可以参考替罪羊树进行定期重构。类似的懒惰处理在VARCHAR的存储中也有使用，目前对VARCHAR的处理方式是存在一个统一的文件里，在初始化时全部读入，并在内存里完成对VARCHAR的处理，然后在退出的时候写回文件。理想的情况是用类似缓存的方式，选择性存储一部分VARCHAR。同时删除文件的时候，也采取懒惰删除的方法，只是将这一侧能到达该文件的东西删掉，同时将FileManager中打开文件的参数改为w+，如此在重新找到该文件时已经自动清空。

在查询解析部分，首先是考虑SELECT命令，我将*也作为了一个聚集函数；处理链接时对所有的表进行处理，挑出符合条件的元组并暴力枚举组合。UPDATE命令则是以全部删除再全部插入来实现的。然后是Where子句的问题，为了简化处理，强制要求'值'的左右必须有括号，否则视作列。最后考虑约束的问题，我将主键和外键共同定义为Constraint，对主键有默认建立的索引，每次需要判断约束是否满足，都使用的是朴素做法。

在head.h的部分，AttrTypeList、FieldList、ValueList这三个封装的类可以将代码对细节的需求转向对逻辑的需求，从而方便检查，并可以将很大一部分查错交给编译器。其次很多文件头类型的数据可以直接用fstream文件流处理，不一定要页式文件系统。但由于head.h是除比较外唯一的公用头文件，导致内容又多又杂，比较麻烦。

6.待改进的地方

对于聚集函数，必须确保SELECT命令中包含有效的WHERE子句才可以正确工作，具体原因暂不明确。

不能支持ALTER命令中对于整列的修改。不能支持ALTER命令中RENAME的操作。

不能有效率地支持SELECT多表链接，目前采取的做法偏向于暴力，在对于每个表处理出合法元组后暴力枚举组合；实际上可以考虑用图论的做法，来使得每一个WHERE子句中的元素都可以尽可能的利用到索引。

在少数时候会出现writeBack错误的问题，会每隔200个字节左右产生1字节偏置，原因暂不明确，但大部分时候不影响测试。

由于时间问题，开发的过程很匆忙，没有注意面向对象，也导致了很多地方还不够严谨。

7.接口整理（即输入命令的格式）

```
COPY tbname FROM path
CREATE DATABASE dbname
CREATE TABLE tbname ( fieldlist )
CREATE INDEX idxname ON tbname ( collist )
USE dbname
DROP DATABASE dbname
DROP TABLE tbname
DROP INDEX tbname ON idxname
SHOW DATABASES
SHOW INDEXS
SHOW TABLES
DESC tbname
INSERT INTO tbname ( collist ) VALUES ( valuelist )
SELECT selector FROM fromclause WHERE whereclause
SELECT selector FROM fromclause
UPDATE tbname SET setclause
DELETE FROM tbname WHERE whereclause
DELETE FROM tbname
ALTER ADD PRIMARY KEY tbname(collist)
ALTER ADD FOREIGN KEY csname tbname(collist) tbname(collist)
ALTER DROP FOREIGN KEY csname
ALTER DROP PRIMARY KEY tbname
```

8.参考资料

Redbase对应的一些链接：

https://web.stanford.edu/class/cs346/2015/redbase-rm.html

https://web.stanford.edu/class/cs346/2015/redbase-ix.html

https://web.stanford.edu/class/cs346/2015/redbase-sm.html

https://web.stanford.edu/class/cs346/2015/redbase-ql.html