**小型数据库**

**实验报告**

**王卓 张高翔 周钊平**

**2012年12月-2013年1月**

目录

[引言 4](#_Toc345194518)

[编写目的 4](#_Toc345194519)

[背景 4](#_Toc345194520)

[参考资料 4](#_Toc345194521)

[开发环境 5](#_Toc345194522)

[项目需求分析 5](#_Toc345194523)

[系统目标 5](#_Toc345194524)

[系统功能 5](#_Toc345194525)

[系统性能要求 6](#_Toc345194526)

[小组成员及分工 6](#_Toc345194527)

[数据结构设计 7](#_Toc345194528)

[总体结构示意图 7](#_Toc345194529)

[数据存储结构 8](#_Toc345194530)

[结点表类 8](#_Toc345194531)

[总文件表类 9](#_Toc345194532)

[算法设计与分析 10](#_Toc345194533)

[模块图 10](#_Toc345194534)

[总文件表上的操作 10](#_Toc345194535)

[表间操作 10](#_Toc345194536)

[查询操作 10](#_Toc345194537)

[插入操作 11](#_Toc345194538)

[更新操作 12](#_Toc345194539)

[删除操作 12](#_Toc345194540)

[内连接操作 12](#_Toc345194541)

[语句嵌套 12](#_Toc345194542)

[文本解析 13](#_Toc345194543)

[命令解析与运行主要方法 13](#_Toc345194544)

[命令解析与运行框架 13](#_Toc345194545)

[run函数群 14](#_Toc345194546)

[analyse函数群 16](#_Toc345194547)

[一些运算的实现 16](#_Toc345194548)

[特色 16](#_Toc345194549)

[输入数据说明 17](#_Toc345194550)

[小组成员感言 18](#_Toc345194551)

[王卓 18](#_Toc345194552)

[张高翔 18](#_Toc345194553)

[周钊平 18](#_Toc345194554)

# 引言

## 编写目的

实现一个小型数据库，将理论与实际相结合，完成数算实习课程的最后一个大作业。

## 背景

任何地方都存在数据，尤其在当今的信息社会，我们每时每刻都在产生数据。海量的数据迫使我们要找到一种有效且高效的方式去管理数据，解放我们的生产力，否则我们就很有可能在低效的数据处理方式中窒息。

同时我们还要做到用户的使用尽可能方便、数据库管理更加灵活。为此我们必须建立起一个行之有效的关系数据库管理系统（DBMS），将数据库的处理抽象成简单易懂的命令，以便用户的查询、删除、修改、连接等操作。

## 参考资料

《数据结构与算法》，2008，张铭 王腾蛟 赵海燕

本实习提供的项目说明书等文档

现实数据库的输入输出标准

## 开发环境

VS2010

# 项目需求分析

## 系统目标

系统的基本目标是：完成题目基本要求，完成对于命令的文本解析，SQL基本语句的实现，构建一个小型数据库。

系统的扩展目标是：在完成基本目标的前提下， 尝试增加对于更多的SQL语句的支持，尝试实现文件的外排序，尝试实现图形界面使得交互界面更为友好便利。

## 系统功能

1) 用户提示

提供完整的用户可进行的操作的信息，以及操作命令，提供所有可选的操作。在操作失败或者非法操作时做出适当提示。

2) 命令输入及解析

支持用户输入各种系统可以支持的SQL命令，通过文本解析，完成对于数据库的基本操作。

3) SQL命令实现

实现基本的SQL语句命令，完成对于数据的存储及维护。

4) 返回结果给用户

完成对于用户输入命令的操作，并将操作应得的结果返回给用户，或者以给定方式输出。

## 系统性能要求

1) 时间及空间复杂性

系统的响应速度应该在用户可接受的范围之内，并尽量减少除数据库的存储空间的其他运算空间，这点要求我们必须建立结构合理的数据结构，索引以及设计高效准确的查询、搜索算法。

2) 结果的准确性和合理性

对于用户输入的搜索条件，返回的结果应该尽可能的和他期望的保持一致，并且如果用户输入有误或不能满足用户要求的条件时应及时给出提示。

3) 灵活性

系统要有良好的接口，便于后面的扩充，以适应信息源的增加或者减少。

4) 其它

要求用户界面友好，运行稳定。

# 小组成员及分工

说明：我们的小组人数较少，故没有设组长一职，每次提交均由文档编辑张高翔完成。

王卓：负责结点，表，表树数据结构的编写，操作函数的实现及调试；

周钊平：负责文本解析模块的编写及调试；

张高翔：I/O，文档、数据集、输入输出标准的编写。

# 数据结构设计

## 总体结构示意图

结点通过单链表组织

结点表通过二叉树组织

结点B1

结点A2

结点A3

结点A1

结点B3

结点B2

结点表

## 数据存储结构

struct DataNode

{

char \* name; //数据主键的名称

int itemNum; //记录中项的数量

char \*\* data; //记录中的数据，每个数据以字符串存储

DataNode \* next; //下一个数据节点的指针

DataNode(); //构造函数

};

## 结点表类

class TableNode

{

public:

char \* name; //表的名称

DataNode \* head, \* tail; //表中第一项与最后一项的指针

int dataNum; //表中记录的数目

int itemNum; //表中数据项的数目

char \*\* itemName; //每个项的名称

int \* type;

TableNode \* left, \* right; //下一个表、上一个表

/\*构造函数\*/

TableNode();

TableNode(char \* tableName, char \*\* itemNameNew, int \* Type, int num);

/\*初始化\*/

void Init(TableNode \* Source);

void init(int num, char \*\* item, int \* type);

/\*析构函数\*/

~TableNode();

/\*操作函数\*/

/\*condiction = 0 1 2 (tag = 0 < = >, tag = 1 >= != <=), -1 no condiction\*/

TableNode \* Where(char \* item, char \*\* value, int isItem, int num, int condiction, int tag);

/\*插入数据节点（以数据为参数）\*/

void insertNode(char \* mainData, char \*\* data) ;

/\*插入数据节点（以数据节点为参数）\*/

void Insert(DataNode \* aim);

/\*更新数据（以数据为参数）\*/

void updateNode(TableNode \* up, int num, char \*\* itemNameNew, char \*\* data);

/\*更新数据（以数据结点为参数）\*/

void Update(DataNode \* aim, int num, int \* itemIndex, char \*\* data);

/\*删除记录（以数据为参数）\*/

void deleteNode(TableNode \* del);

/\*删除记录（以数据结点为参数）\*/

void Delete(DataNode \* pre, DataNode \* aim);

void Distinct();

/\*选取目标项\*/

TableNode \* Select(int num, char \*\* item);

TableNode \* OrderBy(int num, char \*\* item, int \* condiction);

/\*改名\*/

TableNode \* As(char \*\* item);

};

## 总文件表类

# 算法设计与分析

## 模块图

界面及I/O

数据组织

文本解析

用

户

数据

命

令

指令

函数

## 总文件表上的操作

基本类似于BST的操作。

## 表间操作

基本类似于单向链表的操作。

## 查询操作

**1.基础查询**：首先通过调用From函数模拟将待查询的源表复制、或制作出（源表数量大于1时）出，之后调用Where函数模拟where子句将符合条件的记录从源表中删选出制成新的源表，最后调用Select函数完成对于目标列的选取并制成新表。

**2.SQL运算**：当Where子句中含有and或者or等逻辑运算时，非别将逻辑运算的两个表制出，并通过对应的And和Or函数合成一个新表。

**3.AS操作**：在调用Select函数之后以及排序之后（若存在排序操作），调用As函数替换表中对应的项名称。

**4.区间操作**：对于< == >分别在比较函数中将三种运算的真值设为-1 0 1，在区间操作中将< == >的条件值定义为0,1,2并且有tag = 0定义为第一组条件值，这时在比较筛选时利用 比较函数值 + 1 == 条件值 即可筛选出符合条件的项，并将>= != <=的条件值定义为0,1,2并且将tag = 1定义为第二组条件值，这时在比较筛选时利用 比较函数值 + 1 ！= 条件值 即可筛选出符合条件的项。

**5.排序操作**：调用OrderBy函数，按照所给的项名称以及优先次序，用选择排序，依次比较表中元素选择符合次序应排在最前面的元素插入新表中，之后新表即为排序完成的表。

**6.DISTINCT操作**：直接调用Distinct函数将表中完全相等的项删除即可。

## 插入操作

1.**基础插入**：直接调用insertNode函数将想要插入的结点的数据作为参数，在insertNode函数中调用makeNode函数将数据构成一个数据结点结构，之后通过Insert函数直接将一个数据结点结构插入表中。

2.**多值插入**：调用insertNode函数时没有给出的参数缺省使用”\0”即可。

## 更新操作

1.基础更新：

略

## 删除操作

**基础删除**：对于经过From和Where函数之后的新表即为待删除的记录，调用deleteNode函数将对应待删除的记录挑出调用Delete函数直接从表中删除指定数据结点。

## 内连接操作

1.**内连接算法简介**：将两个表通过其中的两个关键词及关系连接成一个表的函数。首先在From操作时将两个表直接合成一个表，其中包含所有可能的数据相合的记录，并在所有项名称添加表明前缀。

2.**JOIN操作**： 经过处理之后的函数与Select基本相同即可得出结果。

## 语句嵌套

具体内容请参考文本解析

# 文本解析

## 命令解析与运行主要方法

**方法顺序：**

① 循环读入命令，以分号隔开的一段为一个输入语句。

② 把输入语句进行预处理，主要是忽略回车符以及把逻辑比较符号单独隔开，然后做成stringstream输入流。最后以一个‘；‘字符隔开，以示输入结束。

③ 先输入输入流的第一个字符，判断该语句是属于何种操作（select/insert/delete/update），然后分别执行函数（run\_select/run\_insert/run\_delete/run\_update）函数，同时调用表的函数进行表操作。

**算法：**

Select语句的处理是按照叉乘再筛选的方式。

Delete和update的处理时按照生成并筛选出一个目标表，对原表上与目标表相同的条目进行删除或者更新处理。

## 命令解析与运行框架

**Scharp类**：封装好的健壮二维数组。在构造表运行函数的参数时十分方便。

**Main函数**： 循环接受命令，调用normalstring预处理输入流，然后变成stringstream流，最后调用Route函数分语句类处理。

**Normalstring函数**：输入流预处理。忽略回车符以及把逻辑比较符号单独隔开，命令最后用一个‘；’字符结束。

**Route函数**：读取命令流的第一个字符串，然后对其进行分语句类调用run函数群处理。

**Valueclear函数**：判断输入的是一个表项的名字还是字符串还是数字

**Char2int函数**：特殊字符串映射。

**Error函数**：约30种丰富的错误处理。

**Run函数群**：解析主语句，调用analyse函数解析子语句，调用表函数执行表的操作。

**Analyse函数群：**解析子语句，生成表操作函数的所需参数返回给run函数。

## run函数群

① Run\_select：select语句的处理

1、读入的顺序（以子语句分类，来自SQL某个标准，其中带有下划线的是必要的语句）：select、from、inner join、on、where、order by。（对这些子语句分别有各自的子语句字符处理函数analyse\_select /analyse\_from /analyse\_inner /analyse\_where /analyse\_order，经过处理之后有用的参数被返回到run\_select,方便其下一步的处理）

2、处理的算法（按照处理的顺序）：from、inner join、on、where、select、order by、distinct。

3、对子语句的具体处理算法：

(1)from table1，table2……：对多个表进行叉乘，生成列名带有点标记的新表VT1。

(2)inner join table：VT1与table叉乘后生成新表VT2。

(3)on table1.a = table2.b：对VT2进行table1.a = table2.b的筛选，生成新表VT3。

(4)where(conditions)：对VT3进行where筛选，相当于带有优先级和括号嵌套的and、or操作，具体不在run\_select中执行，而在analyse\_where中执行，返回一个筛选结果构成的新表VT4到run\_select中。

(5)select row1，row2……：对表VT4进行割列操作。

(6)order by table1.a DESC，table2.b ASC……：对表进行依次排序

② Run\_insert：insert语句的处理

1、读入顺序（带有下划线是必要的语句）：into、rownames、value（没有rownames时即为全元素插入）。

2、处理的算法（按照处理顺序）：into、value

3、对子语句的具体处理办法：

(1)into table：读入表名

(2)(row1, row2,……)：读入列的名称，如果没有这一项那么为插入全元素条目。

(3)value (value1，value2……)：读入值。

(4)对前面列名称和值数组的读入，形成参数数组，调用表操作函数进行插入。

③ Run\_delete：delete语句的处理

1、读入顺序（带有下划线的是必要的语句）：from、where（没有where时为删除表中所有的条目）

2、处理的算法（按照处理顺序）：

(1)from table：生成一个相同项目的虚表VT1

(2)where(conditions)：对VT1调用analyse\_where函数进行筛选，得到要删除的条目构成的表VT2。

(3) 调用表删除函数，在表table中找到在VT2中存在的条目，进行删除。

④ Run\_update：update语句的处理

1、读入顺序（带有下划线的必要的语句）：update、set、where（如果没有where语句那么就是对全表进行更新）

2、处理的算法（按照处理顺序）：

(1)update table:生成一个相同虚表VT1

(2)set row = value，……：读入更新的列名和值，生成更新的参数。

(3)where(conditions)：对表VT1进行筛选。

(4)调用表更新函数，在table中找到在VT1中存在的条目，根据参数进行更新。

## analyse函数群

机械的处理函数，不需要进行介绍了。

## 一些运算的实现

嵌套：递归调用

AndOr以及括号运算：像四则运算中栈的算法。

## 特色

① 封装性（王卓的特色会更多一些）

class Scharp对于表处理函数中所需的char\*\*参数进行封装，只留下insert、changetop等接口。

Error错误处理函数对所有错误进行了综合，中有丰富的处理错误提示。

② 鲁棒性

由于命令的解析是逐词读入的，所以边读入变判错导致了强大的鲁棒性，同时又有种类繁多的错误处理提示。

# 输入数据说明

1、每个SQL命令必须以分号‘；’结束。

2、SQL中的数据最好不要以SQL中的关键字来命名，否则会有解析错误。

3、逻辑比较符号/逗号/括号可以任意贴着数据或者关键词。

4、支持冗余逗号/括号。

5、文件中输入的数据中不能有带有空格。

6、列的名称中不能带点。

7、出现多表操作时（from多个table或者内联），一个合法的子句无论是select子句还是where子句，里面的列的名称都应该带点。

本程序具有鲁棒性，您可以随性输入，但是遇到格式错误的命令会正确退出并有错误提示。

# 小组成员感言

## 王卓

## 张高翔

首先向我们的小组成员表示感谢，本实验的最关键部分是由周钊平和王卓同学完成的，在考试不断的情况下仍能坚持写出高质量的程序，最后顺利完成本作业，在学术上我确实应该向他们学习。我之前两个大作业主要是负责可视化界面部分，对实际算法接触的不是很多，希望在以后的生活中多多向他们学习。

其次表示抱歉，外排序本来是我来负责的，但是由于本人水平有限并不能处理辅修考试与大作业的关系，最后未能调试完成，耽误了本组的进度，并使本实验不能完全完成，所以对两位同学表示抱歉。

这次大作业确实是迄今为止最为复杂的一个，耗费了我们小组成员大量的时间和精力，王卓说他满脑子都是指针，周钊平满脑子都是字符，我则被外排，B树和AVL一直困住，确实在考试周的时候要兼顾作业和考试是一件比较困难的事。希望以后我能够更好地安排时间，更好地处理个人事务。

## 周钊平

这次大作业没有去学正则处理的新知识，只是用到了简单的字符串处理以及嵌套、栈运算之类的，这不得不说是一种遗憾啦，不过也提高了我大量码代码的能力，人生中的第一的一千行就这样莫名其妙地码了出来。虽然换另一种角度来说，这并不是有效率、学习能力强的表现。不过对于这次大作业我还是鞠躬尽瘁了吧。

总的来说这次大作业的题目并不是非常的有趣。不过也让我了解了一些有关与SQL的知识，真实的SQL具有极大的灵活性和强大的功能，是一种非常高级的语言。能够这么早地了解SQL，多亏了这次大作业。

在工作组织上，我们吸取了一定的教训。虽然我们只有三个人，总的来说相比于一些大组会有一点累啦，但是我们还都是扎扎实实干了不少活了。有一点值得改进的地方，那就是我们应该把所有的细节和接口都商量完后再进行编程可能会好一点，我们这次就是每讨论一次都会发现一点问题再进行相应的更改，这样做到哪讨论到哪的方式不仅低效而且会产生很多问题。

我暂时想到的只有这么多了。