

数据库大作业期末报告

小组成员：

黄世宇 2013011304

陈文潇 2013011317

目 录

- 一、项目概述
- 二、编译方式
- 三、小组分工
- 四、系统结构设计
- 五、系统功能
- 六、主要模块设计原理
- 七、附加功能设计原理
- 八、实验结果
- 九、参考文献
- 十、附件

一、项目概述

该项目实现了一个能够执行基本 SQL 语句的数据库程序。底层存储采用了页式管理系统，上层手动解析 SQL 语句，调用制定好的接口进行数据库操作。本数据库具有查询速度快，鲁邦性强，模块划分明确，扩展功能多等特点。

我们支持以下扩展功能：

- 1.强大的索引支持。
- 2.基于 QT 开发的 UI 界面，显示表格，创建数据库，创建表，删除表，执行文件，执行语句等。
- 3.支持查询优化。
- 4.支持更多的数据类型：decimal 和 date。
- 5.支持三个表的连接。
- 6.支持聚集查询 AVG,SUM,MIN,MAX。
- 7.支持模糊查询。
- 8.支持查询结果按照某一属性进行排序。
- 9.支持文件输入和手动输入立即切换功能。

二、编译方式

命令行版本：

- 0.准备工作，安装 [Ubuntu14.04](#)，下载安装最新版 [CLion](#)。

- 1.新建工程，定位到 database 目录下，点击新建。
- 2.新建完工程后，点击软件右上方的运行按钮，软件会自动进行编译。
- 3.若要执行文件，先打开命令行，cd 进入~目录，执行语句：

ln -s 你的 database 的目录/sql sql

然后把需要执行的文件都放在 你的 database 的目录/sql 目录下面。

UI 版本：

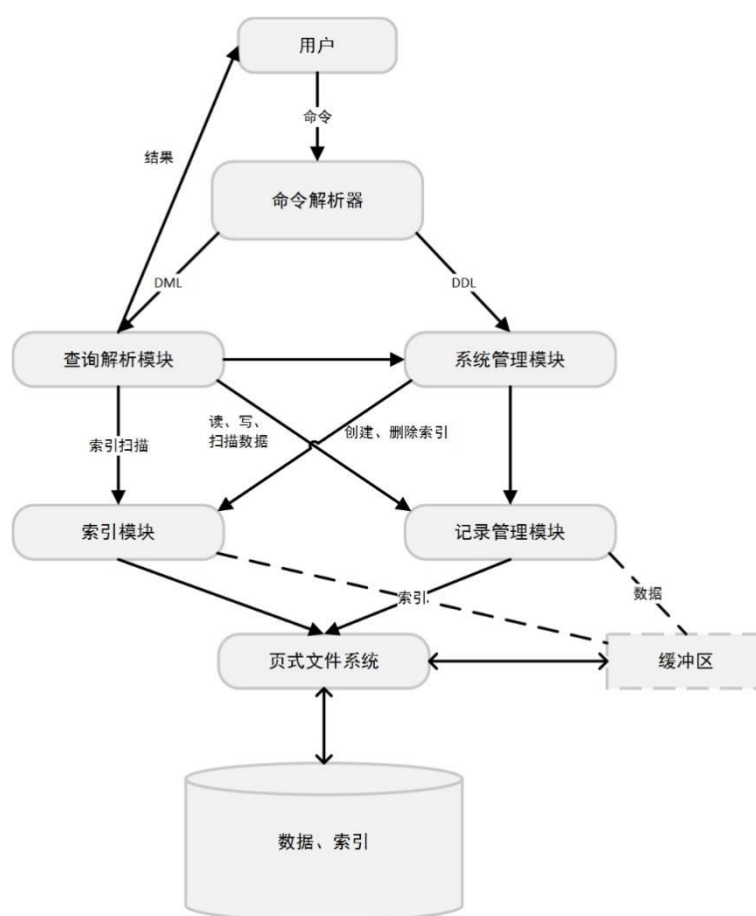
- 0.准备工作，安装 [Ubuntu14.04](#)，下载安装最新版 [Qt](#)。
- 1.用 Qt 打开 UI 目录下的 DBUI.pro 文件。
- 2.Ctrl+R 编译运行。

三、小组分工

主程序	: 黄世宇
接口设定	: 黄世宇
记录管理模块	: 陈文潇
系统管理模块	: 黄世宇
查询解析模块	: 陈文潇

索引模块	: 黄世宇
查询优化	: 陈文潇
图形化界面	: 黄世宇
更多数据类型	: 陈文潇
三个表连接查询	: 陈文潇
聚集查询	: 陈文潇
模糊查询	: 陈文潇
结果排序查询	: 陈文潇
文件/手动输入切换	: 黄世宇
压力测试	: 黄世宇
文档撰写	: 黄世宇

四、系统结构设计



该数据库的整体架构如上图所示，接口制定方面主要参考了斯坦福计算机系数据库课程的大作业项目接口。

首先程序进入 main 函数，创建一个数据库管理类，由这个类进行数据库的初始化，模式选择（支持自动测试模式和手动测试模式，支持文件输入模式和手动输入模式），然后运行命令解析和调用其他模块的接口进行命令执行。根据需要执行的命令是 DDL 命令还是 DML 命令进行调用系统管理模块和查询解析模块的接口。程序支持文件读入命令和手动输入命令两种模式，在任何一种模式下，只要输入 switch，就可以切换到另一种模式下。在任意模式下，输入 exit 退出程序并统一写回数据库。

五、系统功能

【记录管理模块】

数据库管理系统中的第一个模块是记录管理模块，该模块实现一些类和方法来对存储记录的文件进行管理。该模块实现的基本功能如下，直接调用页式文件 I/O 系统中的方法来实现下列功能。

新建文件、删除文件、打开文件、关闭文件。

插入记录、删除记录、更新记录、获取属性值满足特定条件的记录。

【索引模块】

在该模块为文件中的记录建立 B+树索引以提高查询速度，每一个文件会与多个索引相关联（一个属性一个索引）。这些索引也存储在文件中。该模块实现的基本功能如下：

创建索引、删除索引、打开索引、关闭索引

索引中插入节点、删除节点、获取属性值满足特定条件的节点。

【系统管理模块】

该模块以及查询解析模块要求实现解析器来解析用户的命令并执行一系列相关的操作。在该模块中解析实现的用户命令如下：

创建数据库 CREATE DATABASE DBname

删除数据库 DROP DATABASE DBname

切换数据库 USE DATABASE DBname

列出现有的所有数据库以及其包含的所有表名 SHOW DATABASE DBname

DBname 是用户命名的数据库名称。

创建表 CREATE TABLE tableName(attrName1 Type1, attrName2 Type2,..., attrNameN TypeN NOT NULL, PRIMARY KEY(attrName1))。

删除表 DROP TABLE tableName

列出现有的所有表以及其模式信息 SHOW TABLE tableName

【查询解析模块】

该模块实现解析器来解析用户的命令并执行一系列相关的操作。在该模块中解析实现下列四个常见语句的常见语法选项

INSERT INTO [tableName(attrName1, attrName2,...,

attrNameN)] VALUES (attrValue1, attrValue2,...,
attrValueN)

DELETE FROM tableName WHERE whereClauses

UPDATE tableName SET tableName.attrName =
attExpression

SELECT tableName.AttrName FROM tableName
WHERE whereClauses

其中 SELECT 支持 3 个表的连接操作。WHERE 子句的条件
包括常见条件表达式，NULL 判断，整数和字符串比较，模式
匹配等。

六、主要模块设计原理

【记录管理模块】

一个页存多条记录，每当需要访问一条记录时，调用页式
管理系统的接口得到一页的内容，并且把这一页的内容解析
成为每条记录，这样内存中就含有这条记录的内容。后来只
要调用访问内存中内容的接口，就可以得到想要的内容。对
于记录的修改，就先在内存中进行修改，最后需要写回文件
的时候再统一写回。

【索引模块】

一个索引是一个定长的记录，存储方式和普通记录的存储一样。一条索引需要存索引所指向的下个索引或者指向一条记录的位置，还需要存当前索引所表示的范围。

【系统管理模块】

首先新建一个文件保存所有数据库的信息，这样以后可以通过这个文件获得数据库的整体信息。然后对于每个数据库通过页式文件系统进行创建对应的文件，而且还要创建一个文件记录该数据库中表的信息。

【查询解析模块】

这里主要是要调用记录管理模块的一些接口，对记录进行增删改查。这里的命令解析十分繁琐。查询的功能比较多，但是基本上都是先获取到表中的所有数据，然后通过 where 语句中的条件进行筛选，得到最后的输出结果。

七、附加功能设计原理

【UI 界面】

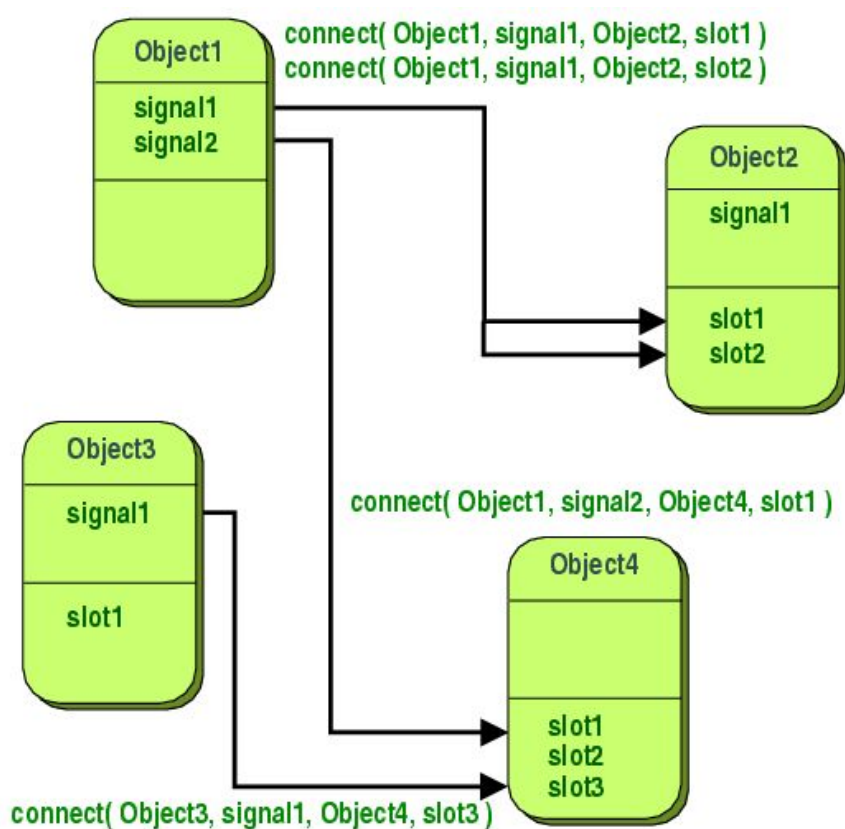
UI 界面使用了 Qt。主要原理是信号槽机制。QT 是一个跨平台的 C++ GUI 应用构架，它提供了丰富的窗口部件集，具有面向对象、易于扩展、真正的组件编程等特点，更为引人注目的是目前 Linux 上最为流行的 KDE 桌面环境就是建立在 QT 库的基础之上。QT 支持下列平台：

MS/WINDOWS-95、98、NT 和 2000 ;UNIX/X11-Linux、Sun Solaris、HP-UX、Digital Unix、IBM AIX、SGI IRIX ;EMBEDDED-支持 framebuffer 的 Linux 平台。信号和槽机制是 QT 的核心机制。信号和槽是一种高级接口，应用于对象之间的通信，它是 QT 的核心特性，也是 QT 区别于其它工具包的重要地方。信号和槽是 QT 自行定义的一种通信机制，它独立于标准的 C/C++ 语言，因此要正确的处理信号和槽，必须借助一个称为 moc (Meta Object Compiler) 的 QT 工具，该工具是一个 C++ 预处理程序，它为高层次的事件处理自动生成所需要的附加代码。在我们所熟知的很多 GUI 工具包中，窗口小部件 (widget) 都有一个回调函数用于响应它们能触发的每个动作，这个回调函数通常是一个指向某个函数的指针。但是，在 QT 中信号和槽取代了这些凌乱的函数指针，使得我们编写这些通信程序更为简洁明了。信号和槽能携带

任意数量和任意类型的参数，他们是类型完全安全的，不会像回调函数那样产生 core dumps。

所有从 QObject 或其子类（例如 QWidget）派生的类都能够包含信号和槽。当对象改变其状态时，信号就由该对象发射 (emit) 出去，这就是对象所要做的全部事情，它不知道另一端是谁在接收这个信号。这就是真正的信息封装，它确保对象被当作一个真正的软件组件来使用。槽用于接收信号，但它们是普通的对象成员函数。一个槽并不知道是否有任何信号与自己相连接。而且，对象并不了解具体的通信机制。

可以将很多信号与单个的槽进行连接，也可以将单个的信号与很多的槽进行连接，甚至于将一个信号与另外一个信号相连接也是可能的，这时无论第一个信号什么时候发射系统都将立刻发射第二个信号。总之，信号与槽构造了一个强大的部件编程机制。



一个关于一些信号和槽连接的摘要图

【查询优化】

每次查询时，如果有索引，通过索引可以快速知道需要查询范围的大小，选择范围小的先进行查询，这样便可以节省时间。

【更多数据类型：decimal 和 date】

decimal 只需要按照一般浮点数进行内存转换即可。而 date 我们把它看做是一个定长的字符串。

【三个表的连接】

我们三表连接和两表连接原理一样，都是先查一个表，再扫表接下来的表。

【聚集查询 AVG,SUM,MIN,MAX】

就是把查询的结果取平均，求和，取最小，取最大。

【模糊查询】

就是在所有结果中，进行字符串匹配。返回匹配正确的结果。

【排序】

主要使用了 sort 函数：

std::sort<algorithm>

default (1)

custom (2)


```
template <class RandomAccessIterator>
void sort (RandomAccessIterator first, RandomAccessIterator last);

template <class RandomAccessIterator, class Compare>
void sort (RandomAccessIterator first, RandomAccessIterator last, Compare comp);
```

Sort elements in range
Sorts the elements in the range `[first,last)` into ascending order.

The elements are compared using `operator<` for the first version, and `comp` for the second.

Equivalent elements are not guaranteed to keep their original relative order (see `stable_sort`).

 **Parameters**

first, last
Random-access iterators to the initial and final positions of the sequence to be sorted. The range used is `[first,last)`, which contains all the elements between `first` and `last`, including the element pointed by `first` but not the element pointed by `last`.
`RandomAccessIterator` shall point to a type for which `swap` is properly defined and which is both *move-constructible* and *move-assignable*.

comp
Binary function that accepts two elements in the range as arguments, and returns a value convertible to `bool`. The value returned indicates whether the element passed as first argument is considered to go before the second in the specific *strict weak ordering* it defines.
The function shall not modify any of its arguments.
This can either be a function pointer or a function object.

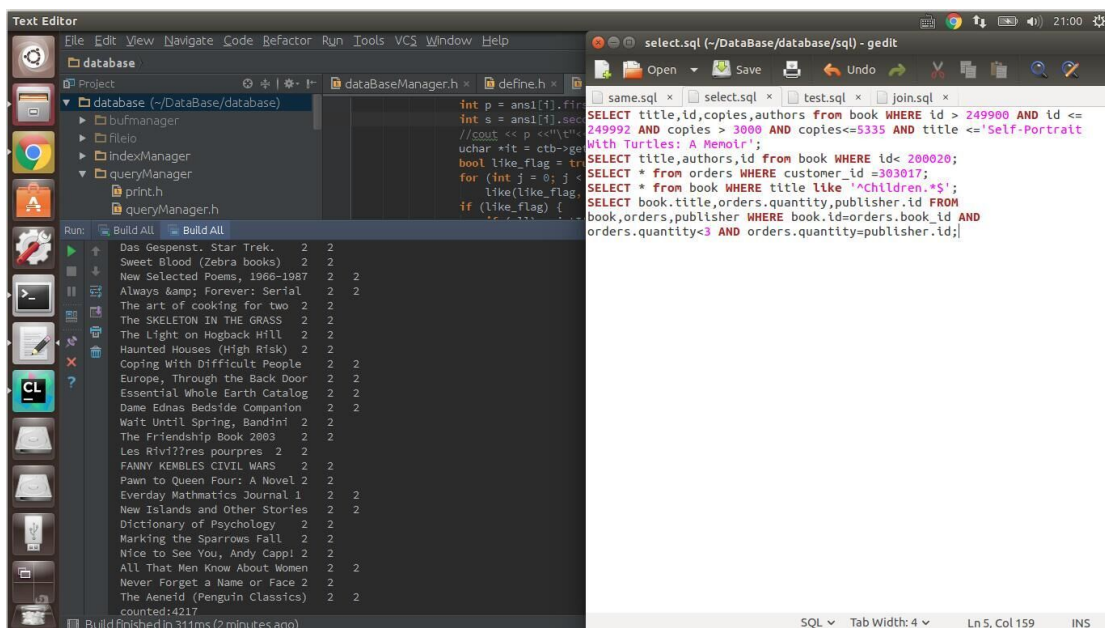
然后自定义了比较函数，用于比较整数，字符串等。

【文件输入和手动输入切换】

对于文件输入，一行一行地从文件中读入命令并解析。对于手动输入，通过用户回车结束输入。

八、实验结果

【三表连接】



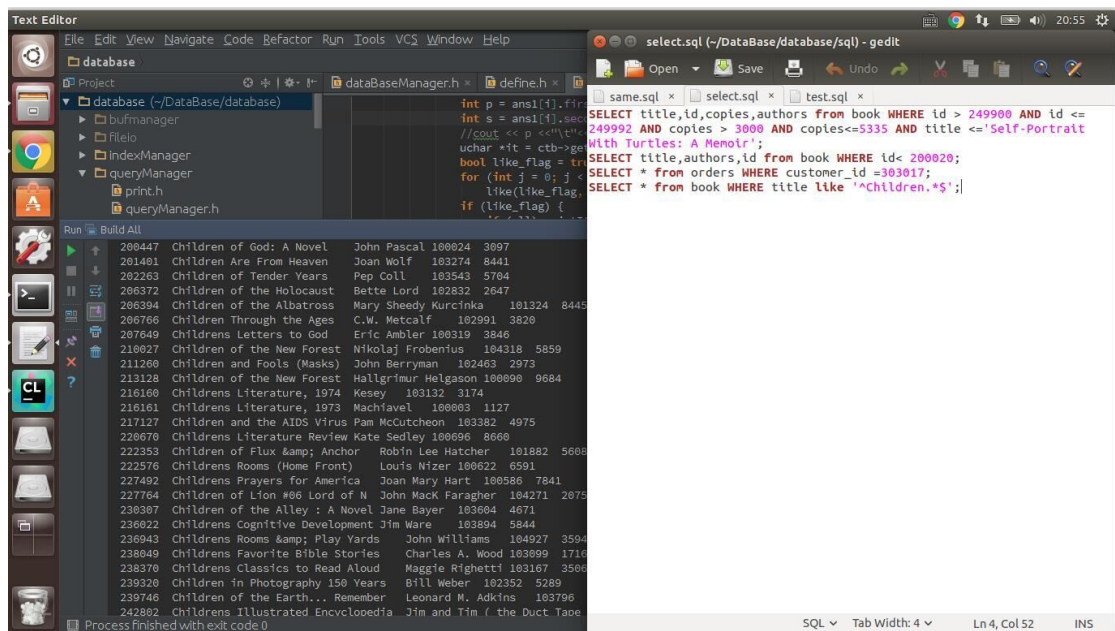
The screenshot displays a development environment with a text editor and a SQL query window. The text editor shows C++ code for a database manager, including file handling and query execution. The SQL query window shows a complex join query involving book, orders, and publisher tables.

```
int p = ans[i].first;
int s = ans[i].second;
//cout << p << "\t" << s << endl;
uchar *it = ctb->get(it);
bool like_flag = true;
for (int j = 0; j < strlen(s); j++) {
    if (s[j] != *it) {
        like_flag = false;
        break;
    }
    it++;
}
```

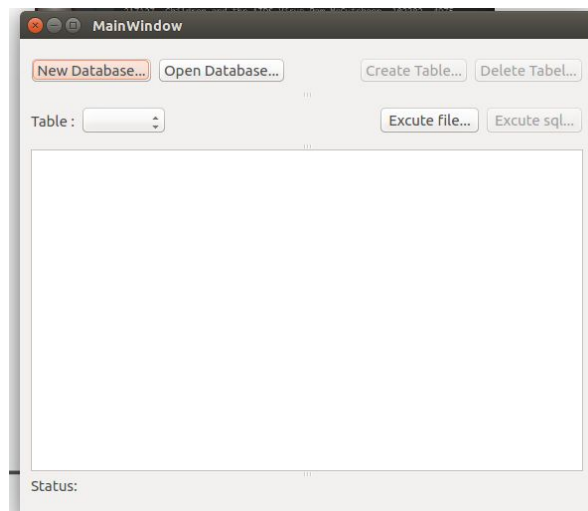
```
SELECT title,id,copies,authors from book WHERE id > 249900 AND id <= 249992 AND copies > 3000 AND copies<=5335 AND title <='Self-Portrait With Turtles: A Memoir';
SELECT title,authors,id from book WHERE id< 200020;
SELECT * from orders WHERE customer_id =303017;
SELECT * from book WHERE title like '%children.*%';
SELECT book.title,orders.quantity,publisher.id FROM book,orders,publisher WHERE book.id=orders.book_id AND orders.quantity>3 AND orders.quantity=publisher.id;
```

【模糊查询】

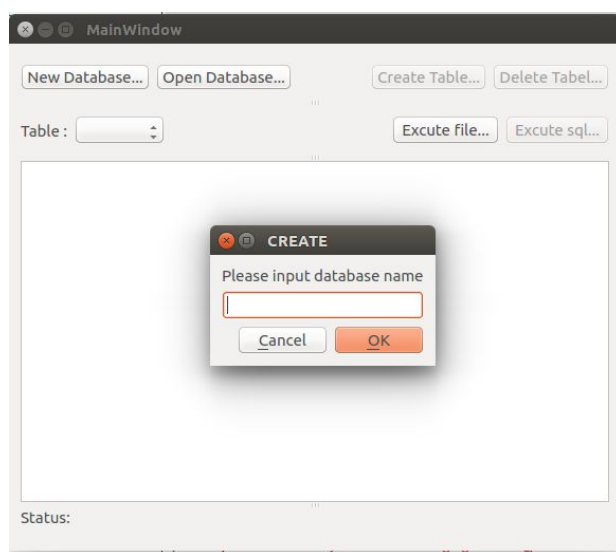
数据库大作业期末报告 / 黄世宇 陈文潇



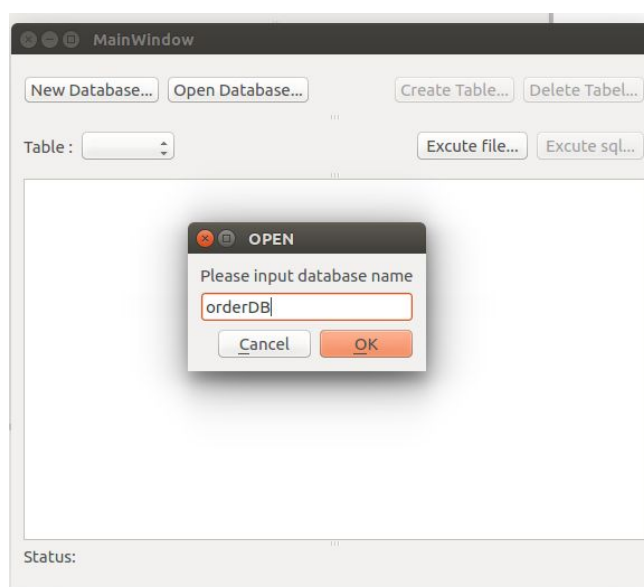
【UI 界面】



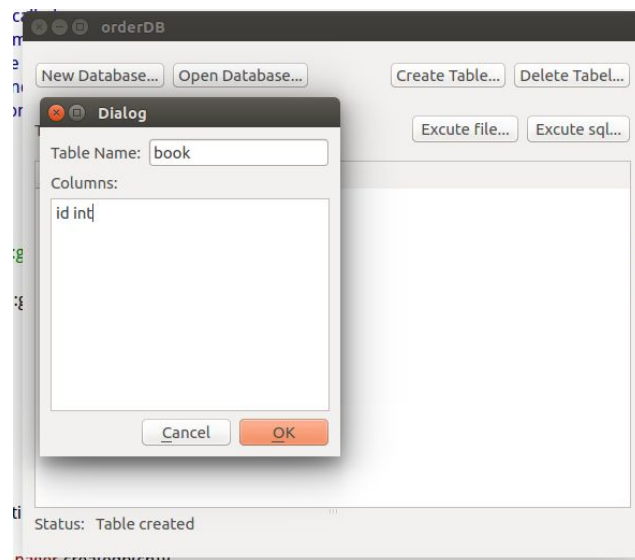
主界面



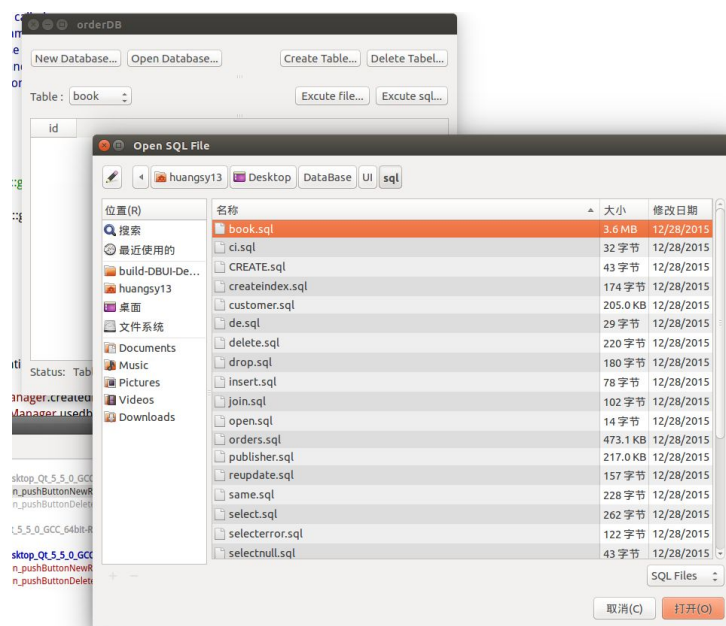
创建数据库



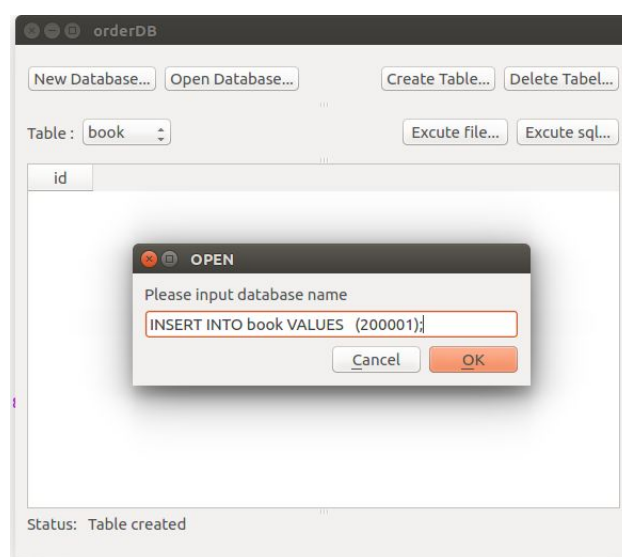
打开数据库



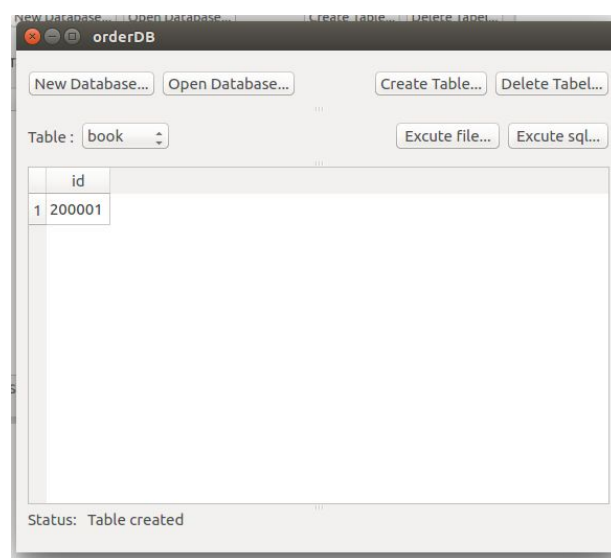
创建表



选择需要执行的文件



执行单条语句



执行结果

十、参考文献

[1]斯坦福 RedBase : <https://web.stanford.edu/class/cs346/2014/project.html>

[2]RedBase Project: <https://github.com/junkumar/redbase>

[3]DataBase Project: <https://github.com/jia-kai/uSQL>

[4]DataBase Project: <https://github.com/liqile/shujuku>

[5]SQL 教程：<http://www.w3school.com.cn/sql/index.asp>

十一、附件

项目连接：<https://github.com/ChenHuang13/DataBase>