API 设计报告

--计算机科学与技术学院 计算机科学与技术专业 1405 班 吴陈奥 3140103809

一、 模块概述：

MiniSQL 要求实现创建表，创建索引，插入，删除，查找，删去表，删去索引等功能。分为了 interpreter/index/record/buffer/catalog 以及本模块 API。API 没有具体功能的实现主要起了一个承上启下的作用，它接住上面 interpreter 从用户那儿获得的信息，根据信息进行分类，然后调用不同的函数进行操作。

注:本模块及本工程所用语言为 C++，本模块实现时所用编译器为 VS。

二、 主要功能: 向下实现

【创建表/创建索引】

接受 interpreter 的信息来创建一个表或者类，结果返回给 interpreter。

【单/多值查找（条件查找）】

通过 interpreter 给的条件转换成合法格式传给下面的类，把查找到的结果返回给 interpreter 最后呈现给用户。

【单值插入】

上层模块给定一个值，最终反馈给用户插入成功或者失败的相关信息。多值同时插入也只需要多次调用单值插入即可。

【删除（单值/多值）】

interpreter 根据用户要求给定一个最小值和最大值，如果是单值删除则只需要将俩个值变成相等即可。调用下层函数删除相应内容最后把结果返回给 interpreter。

三、 对外提供的借口

注：本工程 API 其实就是在 main 函数里，所以其实谈不上是借口，只是一个 switch 语句里面分了各种情况，下面一一已经列出。

【创建表/创建索引】

```
case CRETAB://创建表
case CREIND://创建 index
```

调用创建表和索引的函数实行创建。

【删除表/索引】

```
case DRPTAB://删除表
case DRPIND://删除 index
```

调用 catalog 里面删除相关文件的函数删除文件也就是删除了表或者 Index。

【单/多值查找(条件查找)】

```
case SEL_NOWHERE ://没有条件的查询
case SEL_WHERE://有条件的查询
```

根据有无条件分俩种情况分别调用相关查找函数最后返回结果。

【单值插入】

```
case INSERT://插入单条记录
```

通过 interpreter 的信息把信息传给底层类来实现插入。

【删除（单值/多值）】

```
case DELETE://删除（单条/多条）记录
```

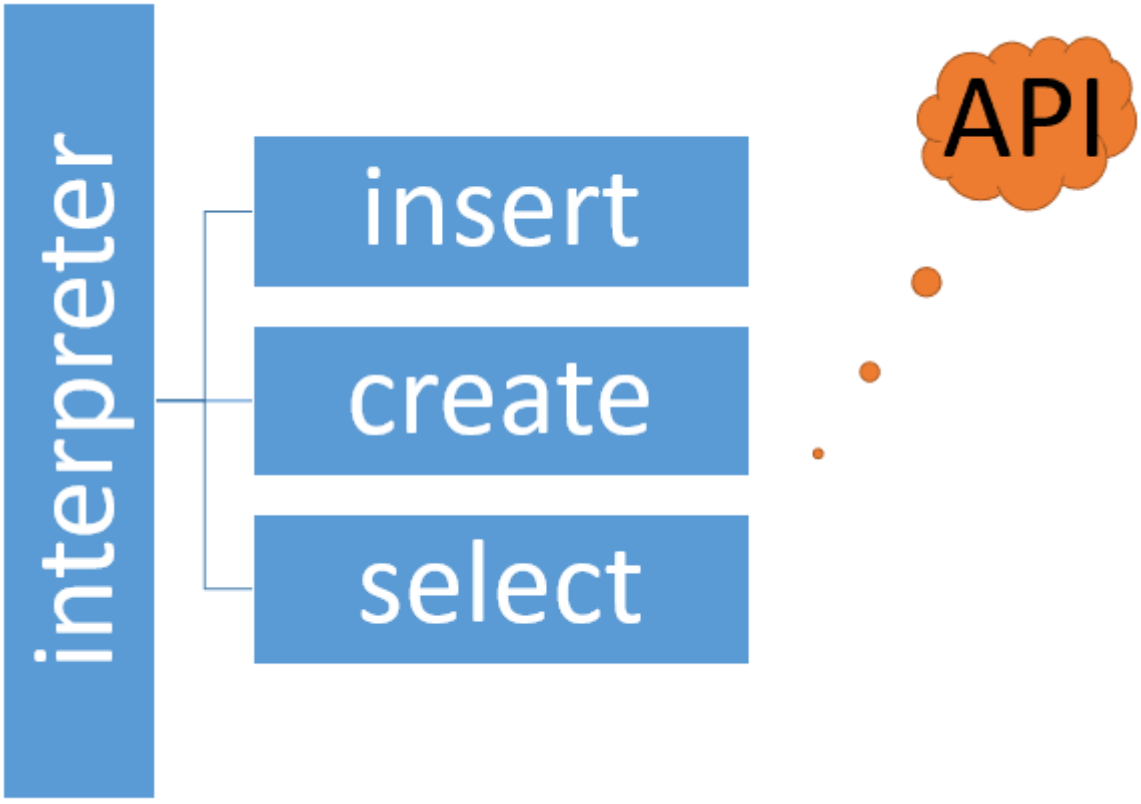
把条件告诉底层类来实现删除。

设计思路

API 没有实现一个具体的功能。所以其实也谈不上是一个具体的实现思路。大致方案就是上面所表述的那样，接受到一个 inrepreter 的一个类，类里面对操作的种类和信息已经做了存储，对种类用一个 switch 语句判断展开就可以在每个模块里面根据不同的种类，不同的信息进行不同的函数调用，实现查找、创建、删除等操作。

四、 整体架构

整体的架构主要是把整个系统整合了起来，上面的 interpreter 告诉指挥 API 通过调用不同的函数干不同的事以便实现不同的功能。而这些功能本身则不是 API 自己实现的。



五、 关键函数和代码

这条 switch 语句就是核心判断和操作的代码：

```
switch (parsetree.operation_type) {
    case SEL_NOWHERE ://没有条件的查询
        //select*
        break;
    case SEL_WHERE://有条件的查询
        checkifindex()//寻找有index的键值
        if (/*indexinfor是否为空*/) {
            switch (conditions[0].op) {
            case Lt:
                break;
            case Le:
                break;
            case Gt:
                break;
            case Ge:
                break;
            case Eq:
                indexM.selectEqual(parsetree.indexInfo, conditions[0].eqValue);
                //index的返回值需要修改
                break;
            case Ne:
                //直接跳出
                break;
            }
        }
        //传给借口
        break;
    case CRETAB://创建表
        cata.CreateTable_catalog();
        cata.CreateTable_catalog(tableinf);
        break;
    case CREIND://创建index
        cata.CreateIndex_catalog(parsetree.indexInfo);
        indexdata = recorder.CreateIndexData();//从table获取已经有的记录
        indexM.createIndex(parsetree.indexInfo, indexdata);
        break;
    case DRPTAB://删除表
        cata.DropTable_catalog();//在catalog里直接删掉文件
        break;
    case DRPIND://删除index
```

```
        cata.Dropindex_catalog();//在catalog里直接删除index的文件
        break;
    case DELETE://删除（单条/多条）记录
        //select先执行一遍
        //选择delete有index和没有的情况
        while (/*还有index没有删*/) {
            //获得indexcolumn
            indexM.deleteValue();//在该index删除记录
            whatindex.pop_back();
        }

        break;
    case INSERT://插入单条记录
        whatindex = cata.check_index();//查找哪些属性有index
        while (/*还有index没有插*/) {
            indexdata = recorder.GetIndexData();//获取当前属性的index的key值
            whatindex.pop_back();
            //insertindex
        }
        break;
    case QUIT://退出
        break;
        }
```