大作业——简易数据库实现

（1）背景

在日常编程当中，我们有大量的数据需要进行记录、存储、查找以及修改，例如对OOP课程中每位同学的日常作业、大作业以及考试成绩进行统计和查找。在之前的先修课程中，我们已经学习了文件操作，可以来进行这些操作。但是频繁的读取文件，修改其中内容，再写回到文件中，一方面带来了操作的复杂，另一方面也影响程序的效率。因而数据库（Database）被开发出来，按照设定的数据结构来组织、存储和管理数据。

具体来说，数据库是一个管理数据的程序，并且提供了若干接口来服务于其他程序。通过这些接口，外部的程序可以创建，访问，管理，搜索和复制所保存的数据。关系型数据库管理系统(Relational Database Management System，RDBMS)是一种较为常见的数据库类型，其借助于集合代数等数学概念和方法来处理数据库中的数据，并建立关系模型来管理数据。

- RDBMS通常具有以下几个特点：
    1. 数据以表格的形式出现。
    2. 每行为各种记录名称。
    3. 每列为记录名称所对应的数据域。
    4. 许多的行和列组成一张表单。
    5. 若干的表单组成数据库。
- 一般来说，配合上述RDBMS的特点，RDBMS也相应定义了一些术语：
    1. 数据库: 数据库是一些关联表的集合。
    2. 数据表: 表是数据的矩阵。在一个数据库中的表看起来像一个简单的电子表格。
    3. 列: 一列(数据元素) 包含了相同类型的数据, 例如所有同学的班级数据。
    4. 行：一行（=元组，或记录）是一组相关的数据，例如某位同学的学号、姓名、院系等个人信息。
    5. 主键：主键是唯一的。一个数据表中只能包含一个主键。你可以使用主键来查询数据。

上述的概念可能较为抽象，理解起来可能比较困难，我们再通过一个例子来解释：

- 表一：学生信息表

学号	姓名	院系	班级
2018011343	赵甲	计算机系	计81
2018011344	钱乙	计算机系	计81
…	…	…	…

- 表二：作业成绩表

学号	第一次作业	第二次作业	第三次作业	总分
2018011343	10	10	10	30
…	…	…	…	…

- 表三：考试成绩表

学号	总分
2018011343	20
…	…

- 表四：大作业成绩表

学号	总分
2018011343	50
…	…

- 此时整个数据库便是由四个表构成。以表一来说：
    1. 表一、表二、表三以及表四就是构成整个数据库的数据表。
    2. 表一中的“学号”、“姓名”、”院系“、”班级“一行就是表头。
    3. 表一中的“赵甲”、“钱乙”、……就是一列。
    4. 表一中的“2018011343”、“赵甲”、“计算机系”、“计81”就是一行。
    5. 对于每个表而言，学号就是每个表的主键，学号的唯一性决定了使用学号来查询数据是准确的，这满足了主键的要求。

（2）第一阶段需求

你需要实现一个C++的简易数据库程序，支持数据库、表的建立，数据的增加、删除、查找、修改等操作。数据库程序能够通过标准的SQL语言进行调用。我们将使用测例代码对于需求的完成情况进行评测，同时我们也会公开**部分测例代码**帮助同学们进行需求开发。**对于变量名大小写敏感，对于SQL关键字大小写不敏感。**测例代码在MySQL上的执行结果需要与你编写的数据库输出**结果一致**。需要实现的基本指令如下：

- 数据库创建指令（`DBname` 是用户命名的数据库名称。本项目中的数据库管理系统要管理多个数据库，实现在各个数据库之间切换），占4%。
    1. `CREATE DATABASE DBname` 创建数据库指令；
    2. `DROP DATABASE DBname` 删除数据库；
    3. `USE DBname` 切换数据库；
    4. `SHOW DATABASES` 列出现有的数据库。

```
CREATE DATABASE OOP;
USE OOP;
```

- 在使用`USE DBname`来选定数据库后，进行数据表的创建与删除，占4%。

  1. `CREATE TABLE tableName(attrName1 Type1, attrName2 Type2, … , attrNameN TypeN NOT NULL, PRIMARY KEY(attrName1))` 创建表，注意要实现`NOT NULL`和`PRIMARY KEY`这两个关键字。`tableName`是表名，`attrName1`、`attrName2`、……是表的属性名称，`Type1`、`Type2`、……是每个属性的数据类型，`NOT NULL`表示当前的这项属性不能为空，`PRIMARY KEY(attrName1)`表示将`attrName1`设定为表的主键。数据类型支持整数（`INT`）、实数（`DOUBLE`）、字符（`CHAR`）；（注意`PRIMARY KEY`不一定出现在最后!）
  2. `DROP TABLE tableName` 删除表；
  3. `SHOW TABLES` 列出现有表；
  4. `SHOW columns from tablenName`列出制定表项的各项信息，包括属性、属性类型、主键信息。

```
CREATE TABLE oop_info(stu_id INT NOT NULL, stu_name CHAR, PRIMARY
KEY(stu_id));
```

- 表的数据加入，占3%。

  `INSERT INTO [tableName(attrName1, attrName2,…, attrNameN)] VALUES (attrValue1, attrValue2,…, attrValueN)` 向名称为`tableName`的表中加入数据，加入的数据`attrName1`的值为`attrValue1`， `attrName2`的值为`attrValue2`，……

```
INSERT INTO oop_info(stu_id, stu_name) VALUES (2018011343, "a");
```

- 表的数据删除，占3%。

  `DELETE FROM tableName WHERE whereClauses` 从名称为`tableName`的表中删除某些行，这些行的数据满足条件`whereClauses`。

```
DELETE FROM oop_info WHERE stu_id=2018011343;
```

- 表的数据修改，占3%。

  `UPDATE tableName SET attrName = attExpression WHERE whereClauses` 在名称为`tableName`的表中修改某些行，这些行的数据满足条件`whereClauses`，修改内容为将满足条件行的`attrName`改为`attExpression`。

```
UPDATE oop_info SET stu_name="b" WHERE runoob_id=2018011343;
```

- 表的数据查询，占3%。

  `SELECT AttrName FROM tableName WHERE whereClauses` 在名称为`tableName`的表中查找某些行，这些行的数据满足条件`whereClauses`，返回这些行的`attrName`属性的数据。

```
select * from oop_info;
```

- 实现多条件`whereClauses`子句，即使用 AND 或者 OR 指定多个条件，占10%：

`whereClauses`在数据的删除、修改和查询中被使用，在第一阶段对于`whereClauses`只需要进行简单的实现，**支持对于单表的多条件子句，运算符支持大于（>）、小于（<）和等于（=）**。更多关于标准SQL语句的描述可以参考 http://www.runoob.com/mysql/mysql-tutorial.html。

（3） 项目限制

只能使用C++完成，禁止使用第三方库（不包括STL或编译器自带的库）。

（4） 提交要求

1. 提交你编写的数据库代码和使其能够编译的Makefile，请遵守OOP的设计规范。
2. 在 readme.md 或 readme.txt 中写明程序的运行环境。（你的数据库最好能在跨平台下运行，并且依赖项尽量少，否则别的组将很难运行你的代码。）
3. 给出一个说明文档（markdown或word），写清数据库的结构、封装、接口。给出开发者指导，便于别的小组能够阅读你的代码。（字数不限，目标是能让别人在最短时间内明白项目结构，太短太长都不太好。）
4. 善用注释，便于别的小组能够阅读你的代码。
5. 在根目录提交一个developer.txt，写明所有组员的学号姓名。除此以外，禁止在项目的任何地方（包括代码、注释、文档）出现任何和个人相关的信息。
6. 组与组之间可以讨论，但禁止任何形式的交换代码。在第二阶段互评的过程中，我们会随机发放代码，禁止互相讨论每组手上有哪些代码或者打听分数。最终投票结果我们会在第二阶段结束后给出。这一项涉及到最终得分的公平性。