# 关系数据库管理系统OurSQL设计文档

胡津铭 李天润

2014年12月30日

## 目录

# 1 简介

本项目实现了一个小型单用户关系数据库管理系统。项目实现大体分为六个模块：底层存储模块、索引模块、系统管理模块、命令执行模块、命令解析模块、交互模块。支持B+树索引、域完整性约束、外键约束、扩展的EMACScript语法的正则表达式模糊查询、多表连接、聚集查询、分组查询、查询结果排序。本数据库系统完整严谨、功能强大、稳定高效，模块划分科学、接口明晰、耦合度低，代码简洁明快、风格优美、平台无关。 本项目遵循MIT开源协议，托管在GitHub。但考虑到实际情况，在本课程结束之前，存储在私有仓库中。

# 2 功能特点

## 2.1 底层存储系统

独立设计并开发了页式文件系统，支持创建、删除文件，读取、写入指定页，支持动态设置页大小（创建文件时设置）。并配有使用LRU页置换算法的缓冲区管理系统，其接口与页式文件系统完全相同，支持动态设置缓冲区大小。

## 2.2 B+树索引系统

独立设计并开发了B+树索引系统。

## 2.3 数据类型

支持8位整型（tinyint）、16位整型（smallint）、32位整型（int）、64位整型（bigint）、布尔型（bool）、字符串（char, varchar）、IEEE 75432位浮点型（float）、IEEE 75464位浮点型（double），其中整型支持有符号（signed）和无符号（unsigned），共计13种数据类型。

## 2.4 聚集函数

支持AVG、COUNT、MAX、MIN、SUM五种聚集函数。

SUM函数对整数类型操作结果自动转换为64位有符号整型，对浮点数操作结果自动转换为64位浮点数。

COUNT函数返回结果为64位无符号整数。

AVG函数返回结果自动转化为64位浮点数。

MAX、MIN函数返回结果为原类型。

## 2.5 多表连接

支持多表内连接，与聚集函数、分组查询、结果排序功能兼容。

## 2.6 分组查询

支持分组查询，与聚集函数、多表连接、查询结果排序功能兼容。

## 2.7 查询排序

支持查询结果排序，与聚集查询、多表连接、分组查询功能兼容。

## 2.8 模糊匹配

对所有支持的类型进行EMACScript语法正则表达式匹配的模糊查询。注意此功能使用匹配模式而非搜索模式。

## 2.9 关系完整性约束

对于违反完整性约束的操作，数据库管理系统将拒绝此次操作，必要时进行回滚操作以保证操作的原子性。

### 2.9.1 实体完整性约束

每个表最多可创建一个主键，主键取值既不能为空也不能重复。

### 2.9.2 域完整性约束

建表时可约束每个属性的取值，约束条件是使用AND连接的条件语句。

### 2.9.3 参照完整性约束

可设置某个属性引用其他表的主键。外键只能取外表主键已有的值，或取空值。

## 2.10 命令解析系统

本项目中使用Boost Spirit进行词法分析和语法分析。Spirit用法灵活、极易扩展。

## 2.11 查询优化

在多表内连接时将有索引的放在内层，无索引的放在外层，可以大大提高性能。

## 2.12 错误处理系统

支持数据库不存在、属性名不存在、数据类型不匹配、字面量溢出、违反域完整性约束、违反参照完整性约束、聚集函数作用于不支持的类型等多达45种语法错误、运行时错误的检测和处理。

### 2.13 交互界面

支持从文件读取指令，支持指令断行、支持#风格的注释。

### 2.14 跨平台

项目源代码完全符合C++11标准（ISO/ISC 14882:2011），不使用任何平台相关库。源代码可不经任何修改在Windows、Mac OSX、Linux平台上编译通过且正确运行。运行过程中生成的数据、索引等文件也可以在任何字节序（大端模式或小端模式）相同的平台下交叉使用。

## 3 主要算法

### 3.1 缓冲区页面置换算法

使用LRU算法。利用一个哈希表和一个链表模拟一个元素不重复栈，栈具有固定大小，超出设定的大小时舍弃栈底数据。

当访问某个页面时，先在哈希表中搜索，若未搜索到，说明该页不在缓冲区中，读取该页并将页号放在链表头（栈顶），在哈希表中插入该页页号和链表的指针；若在哈希表中搜索到，则将链表中的节点移动到链表头，并更新哈希表。如此可在渐进$O(1)$时间内根据页号返回页数据。

### 3.2 B+树索引

B+树是经典算法，在此不再赘述。

### 3.3 多表连接

递归地向一个栈中加入RID，栈底是第一个表的RID，上面依次是第二个、第三个……每次加入一个数据，若不满足条件则回溯，直到栈大小等于要连接的表的个数，则得到一条连接记录。

### 3.4 分组聚集查询

先对待分组的结果排序，之后去重，对于重复的记录若有聚集函数则使用聚集函数返回值代替原值。

### 3.5 模糊匹配

将存储类型转化为显示数据（如将二进制整型和浮点型转换为十进制），再用正则表达式匹配。

## 3.6 实体完整性约束

每次向表中插入或修改数据时，检查若发现主键重复或主键为空则拒绝此次操作。对于无主键表，为实现方便创建一个隐藏的主键，其键值为插入此记录时的纳秒级时间，对于现代CPU体系结构而言，可以保证任意两条记录主键值不同。

## 3.7 域完整性约束

每次插入或修改记录时检查是否满足域完整性约束，若不满足则拒绝此次操作。

## 3.8 参照完整性约束

修改或删除主表主键时，检查该键值是否被引用，若被引用则拒绝修改或删除。

插入或修改从表时，检查主表是否存在此键值，若不存在则拒绝插入或修改。

## 3.9 出错回滚

对于单条语句插入多条记录，UPDATE语句，为保证操作的原子性，运行过程中出错需要回滚。

在操作过程中，将每一个成功的操作压入栈中，若某次操作失败，则依次取出栈中的操作进行回滚。

# 4 模块接口说明

## 4.1 页式文件系统 DBFile

| 接口 | 描述 |
|---|---|
| (constructor) | 接受文件名 |
| create | 创建指定页大小的文件 |
| remove | 删除文件 |
| open | 打开文件 |
| close | 关闭文件 |
| writePage | 写指定页 |
| readPage | 读指定页 |
| isopen | 文件处于打开状态 |
| accessible | 文件存在且有读写权限 |
| pageSize | 返回页大小 |
| numPages | 文件总页数 |

## 4.2 缓冲区模块 DBBuffer

除在构造时接受缓冲区大小作为参数，其他与DBFile完全相同。

## 4.3 表属性管理 DBFields

| 接口 | 描述 |
|---|---|
| insert | 添加一个属性 |
| hasPrimaryKey | 有主键 |
| addPrimaryKey | 添加隐藏主键 |
| removePrimaryKey | 删除隐藏主键 |
| size | 属性个数 |
| recordLength | 记录总长度 |
| field_id | 所有属性ID |
| field_type | 所有属性类型 |
| field_length | 所有属性长度 |
| field_name | 所有属性名 |
| indexed | 各属性是否有索引 |
| notnull | 各属性取值是否不能为空 |
| primary_key_field_id | 主键ID |
| datatypepe_name_map | 属性类型名称映射表 |
| typeLength | 获取类型对应的默认长度 |
| MinGenerator | 获取类型的最小可能取值 |
| LiteralParser | 存储数据与显示数据之间转换 |
| Comparator | 比较器 |

## 4.4 系统管理 DBTableManager

| 接口 | 描述 |
|---|---|
| create | 创建表 |
| open | 打开表 |
| isopen | 表处于打开状态 |
| fildsDesc | 返回表属性描述 |
| close | 关闭表 |
| remove | 删除表 |
| insertRecord | 插入一条记录 |
| removeRecord | 删除一条记录 |
| selectRecord | 查看一条记录 |
| modifyRecord | 修改一条记录 |
| traverseRecords | 遍历所有记录 |
| findRecords | 查询满足条件的所有记录 |
| createIndex | 在某个属性创建索引 |
| removeIndex | 删除某个属性的索引 |

## 4.5 索引管理 DBIndexManager

| 接口 | 描述 |
|---|---|
| (constructor) | 接受索引文件名 |
| create | 创建索引 |
| open | 打开索引 |
| close | 关闭索引 |
| remove | 删除索引 |
| searchRecord | 查找指定键值，返回该第一个该键值的RID |
| searchRecords | 查找指定键值，返回所有该键值的RID |
| rangeQuery | 范围查询 |
| insertRecord | 插入一个键值 |
| removeRecord | 删除第一个该键值 |
| removeRecords | 删除所有该键值 |
| traverseRecords | 遍历所有键值 |
| getNumRecords | 返回键值总数 |

## 4.6 命令执行 DBQuery

| 接口 | 描述 |
|---|---|
| (constructor) | 接受标准输出和错误输出流 |
| execute | 执行命令 |

## 4.7 命令解析

较复杂，参见源文件。

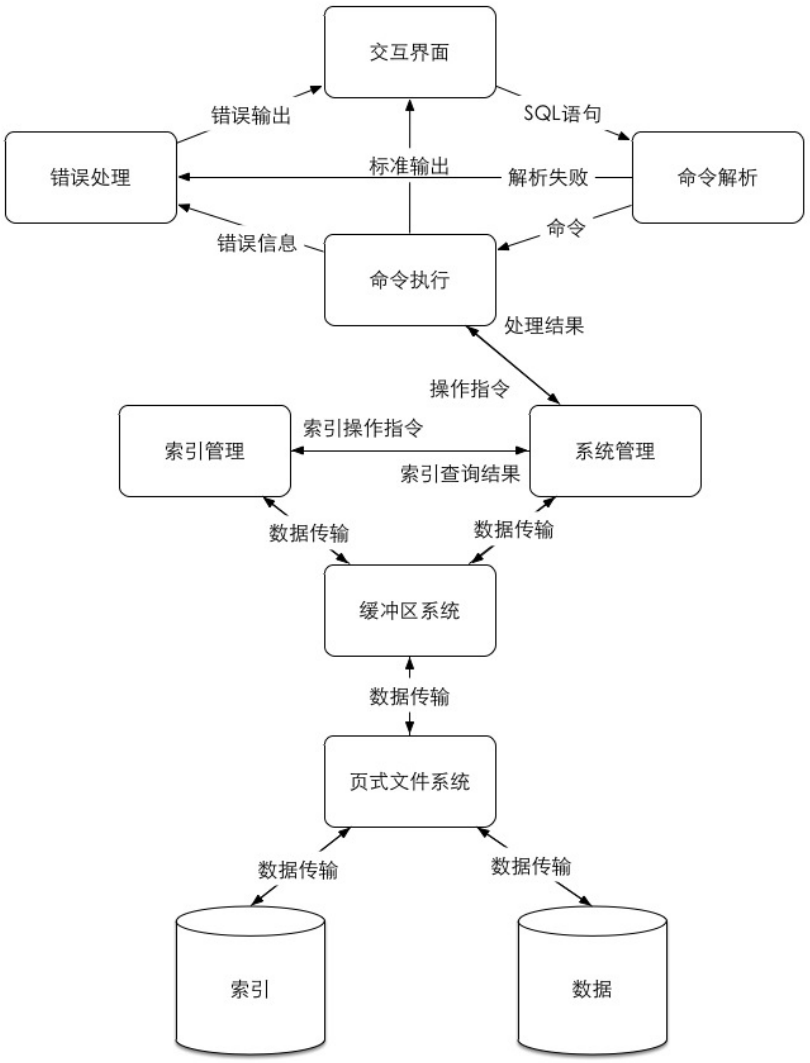
## 4.8 错误处理 DBError

| 接口 | 描述 |
|---|---|
| getInfo | 返回错误信息 |

## 4.9 交互界面 DBInterface

| 接口 | 描述 |
| --- | --- |
| feed | 输入数据到缓冲区中 |
| ready | 指令已就绪 |
| get | 取出一条指令 |
| emptyBuff | 缓冲区为空 |

图 1: 整体结构图



# 5 语法规则

1. 关键字

   不区分大小写。AND, ASC, AVG, BY, CHECK, COUNT, CREATE, DATABASE, DATABASES, DELETE, DESC, DESCRIBE, DROP, FALSE, FOREIGN, FROM, GROUP, INDEX, INSERT, INTO, IS, KEY, LIKE, MIN, MAX, NOT, NULL, ON, ORDER, PRIMARY, REFERENCES, SELECT, SET, SHOW, SIGNED, SUM, TABLE, TABLES, TRUE, UNSIGNED, UPDATE, USE, VALUES, WHERE.

2. 数据类型

表 1: 数据类型

| | |
|---|---|
| TINYINT [SIGNED] | 有符号8位整型 |
| TINYINT UNSIGNED | 无符号8位整型 |
| SMALLINT [SIGNED] | 有符号16位整型 |
| SMALLINT UNSIGNED | 无符号16位整型 |
| INT [SIGNED] | 有符号32位整型 |
| INT UNSIGNED | 无符号32位整型 |
| BIGINT [SIGNED] | 有符号64位整型 |
| BIGINT UNSIGNED | 无符号64位整型 |
| BOOL | 布尔型 |
| CHAR, VARCHAR | 字符串，必须显式指定长度 |
| FLOAT | 32位浮点型 |
| DOUBLE | 64位浮点型 |

3. 标识符

   仅包含字母或数字或下划线，以字母或下划线开头，且不能为关键字和数据类型。

4. 字面量

   整型字面量为十进制整数，浮点型字面量为十进制小数，布尔型字面量取值为TRUE（不区分大小写，下同）或FALSE，空值为NULL，字符串必须使用单引号引用，支持反斜线转义，正则表达式也使用单引号引用。

5. 运算符

   运算符左值必须是属性名，右值可以是属性名或字面量。当用在表连接的SELECT语句中，属性名必须显示指定表名。

6. CREAETE DATABASE <name>;

   创建名为name的数据库。

7. SHOW DATABASES;

   列出所有数据库。

8. USE <name>;
   打开名为name的数据库。
9. DROP DATABASE <name>;
   删除名为name的数据库。
10. CREATE TABLE <table_name> (<field_name> <type>[(<size>)] [NOT NULL] [,<filed_name> <type>[(<size>)] [NOT NULL]]* [, PRIMARY KEY (<field_name>)] [, CHECK (<conditions>)] [, FOREIGN KEY (<field_name>) REFERENCES <table_name>(<field_name>)]* );
    创建名为table_name的表，包含若干个feild，并设置关系完整性约束。
11. SHOW TABLES;
    列出当前数据库的所有表。
12. DESC[RIBE] <name>;
    显示名为name的表信息。
13. DROP TABLE <name>;
    删除名为name的表。
14. CREATE INDEX ON <table_name> (<field_name>);
    在表table_name的属性field_name上创建索引。
15. DROP INDEX ON <table_name> (<field_name>);
    删除表table_name的属性field_name上的索引。
16. INSERT INTO <table_name> VALUES (<value> [, <value>]*) (, <value> [, <value>]*)*;
    插入一系列记录到表table_name中。

表 2: 运算符

| | |
|---|---|
| = | 相等 |
| <>, != | 不相等 |
| > | 大于 |
| < | 小于 |
| >= | 大于等于 |
| <= | 小于等于 |
| IS | 值为空，右值只能是NULL |
| IS NOT | 值不为空，右值只能是NULL |
| LIKE | 正则表达式匹配 |
| NOT LIKE | 正则表达式不匹配 |

17. DELETE FROM <table_name> [WHERE <conditions>];

    从表table_name中删除所有满足条件的记录。

18. UPDATE <table_name> SET <field_name> = <new_value> [, <field_name> = <new_value>]* WHERE <conditions>;

    更新表table_name中满足条件的记录的部分字段。

19. SELECT [AVG|SUM|MAX|MIN|COUNT(]<field_name | *>[)] [, [AVG|SUM|MAX|MIN|COUNT(]<field_name | *>[)]]* FROM <table_name> [WHERE <conditions>] [, GROUP BY <field_name>] [, ORDER BY <field_name> [ASC | DESC]];

    从表table_name中选择满足条件的记录，按照某个属性分组，使用聚集函数，按照某个属性排序， 显示部分属性。

20. SELECT [AVG|SUM|MAX|MIN|COUNT(]<table_name>.<field_name>[)] [, [AVG|SUM|MAX|MIN|COUNT(]<table_name>.<field_name>[)]]* FROM <table_name> [, <table_name>]* [WHERE <conditions>] [, GROUP BY <table_name>.<field_name>] [, ORDER BY <table_name>.<field_name> [ASC | DESC]];

    多表的内连接。

# 6 任务分工

本项目开发过程中分工明确、协作配合到位。

本项目框架由胡津铭设计。胡津铭负责开发页式文件系统、缓冲区模块、系统管理模块、命令解析模块、交互模块。李天润负责开发B+树索引模块、命令执行模块、错误处理模块。

# 7 源文件说明

## 7.1 编译说明

需要安装Boost 1.57.0 或以上版本。

需要编译器支持C++11。

Microsoft C/C++ Optimizing Compiler Version 18.00.21005.1 编译器因尚未支持全部C++11特性，不能通过编译。

## 7.2 文件结构

```
/
└── design ..................................................... 设计资料
    ├── data_file_format ....................... 数据文件存储格式
    └── index_file_format ...................... 索引文件存储格式
```

```
    ├── supported_statements ..... 支持的SQL语句说明
├── src ..... 源代码
    ├── db_buffer.h ..... 缓冲区管理
    ├── db_common.h ..... 公用常量、函数
    ├── db_error.h ..... 错误处理
    ├── db_fields.h ..... 表属性描述
    ├── db_file.h ..... 页式文件管理
    ├── db_indexmanager.h ..... B+树索引
    ├── db_interface.h ..... 交互界面
    ├── db_outputer.h ..... 交互界面对齐输出器
    ├── db_query.h ..... 命令执行
    ├── db_query_analyser.h ..... 命令解析
    ├── db_table_manager.h ..... 系统管理
    ├── oursql.cc ..... 主函数
    └── Makefile ..... Makefile
└── LICENSE ..... 开源协议
```

# 8 参考资料


[1] 冯建华，周立柱，郝小龙. 数据库系统设计与原理[M]. 北京：清华大学出版社, 2007-6-2

[2] Hector Garcia-Molina, Jeffrey D. Ullman, Jennifer D. Widom. Database System Implementation[M]. 2th ed. New Jersey: Prentice Hall, 1999-6-11

[3] Joel de Guzman, Hartmut Kaiser. Boost Spirit 2.5.2[OL]. 2014-10-30.http://www.boost.org/doc/libs/1_57_0/libs/spirit/doc/html/index.html