Linux下C++访问Mysql数据库(增删改查及事务提交和回滚操作)

一、C++对Mysql数据库的增删改查操作

> 在这个程序中，实现了通过 C++ 对数据表进行添加，修改，插入，删除的功能。

头文件-myDB.h:

```
#ifndef myDB_class
#define myDB_class

#include <iostream>
#include <string>
#include <mysql/mysql.h>

class myDB
{
public:
    myDB();
    ~myDB();
    int initDB(std::string host, std::string user, std::string password, std::string
db_name);
    void Run();

private:
    int exeSQL();
    int insertSQL();
    int deleteSQL();
    int updateSQL();
    MYSQL *connection;
    MYSQL_RES *result;
    MYSQL_ROW row;
};

#endif
```


myDB.cpp

```
#include "myDB.h"
#include <iostream>
#include <cstdlib>

using namespace std;

myDB::myDB()
{
    connection = mysql_init(NULL); // 初始化数据库的连接变量
    if(connection == NULL)
        {
            std::cout << "error:" << mysql_error(connection);
            exit(1);            // exit(1)表示发生错误后退出程序， exit(0)表示正常退出
        }
}

myDB::~myDB()
{
    if(connection != NULL)
        {
            mysql_close(connection); // 关闭数据库连接
        }
}

int myDB::initDB(std::string host, std::string user, std::string password, std::string
db_name)
{
        char value = 1;
        mysql_options(connection_,MYSQL_OPT_RECONNECT,(char*)&value);//mysql自动重连

    connection = mysql_real_connect(connection, host.c_str(), user.c_str(),
password.c_str(), db_name.c_str(), 0, NULL, 0); // 建立数据库连接
    if(connection == NULL)
    {
        std::cout << "error:" << mysql_error(connection);
        exit(1);            // exit(1)表示发生错误后退出程序， exit(0)表示正常退出
    }
    return 0;
}

int myDB::exeSQL()
{
    // mysql_query()执行成功返回0，失败返回非0值。与PHP中不一样
    std::string sql;
    sql = "select * from user;";
    if(mysql_query(connection, sql.c_str()))
    {
        std::cout << "Query Error:" << mysql_error(connection);
        exit(1);
    }
```

```
    else
    {
        result = mysql_use_result(connection); // 获取结果集
        // mysql_field_count()返回connection查询的列数
        for(int i=0; i < mysql_field_count(connection); ++i)
        {
            // 获取下一行
            row = mysql_fetch_row(result);
            if(row <= 0)
            {
                break;
            }
            // mysql_num_fields()返回结果集中的字段数
            for(int j=0; j < mysql_num_fields(result); ++j)
            {
                std::cout << row[j] << " ";
            }
            std::cout << endl;
        }
        std::cout << endl;
        // 释放结果集的内存
        mysql_free_result(result);
    }
    return 0;
}

int myDB::insertSQL()
{
    std::string username;
    std::string psd;
    std::string level;
    std::cout << "请输入id、username、password、level: " << std::endl;
    std::cin >> username >> psd >> level;
    string sql = "insert into user(id,username,password,level) values ( NULL,'" +
username +"','"+ psd  +"', "+ level +");";
    if(mysql_query(connection, sql.c_str()))
    {
        std::cout << "Query Error:" << mysql_error(connection);
        exit(1);
    }
    std::cout << endl;
    exeSQL();
    return 0;
}

int myDB::deleteSQL()
{
    std::string id;
    std::cout << "输入要删除用户的ID: " << std::endl;
    std::cin >> id;
    std::string sql;
```

```
    sql = "delete from user where id = " + id + ";";
    if(mysql_query(connection, sql.c_str()))
    {
        std::cout << "Query Error:" << mysql_error(connection);
        exit(1);
    }
    std::cout << endl;
    exeSQL();
    return 0;

}

int myDB::updateSQL()
{
    std::string id;
    std::string thing;
    std::cout << "请输入要修改的用户的id与内容（eg：修改1号用户的姓名，输入：1 【回车】
username = 'Mary'）" << std::endl;
    std::cin >> id >> thing;
    std::string sql;
    sql = "update user set " + thing + "where id =" + id + ";";
    if(mysql_query(connection, sql.c_str()))
    {
        std::cout << "Query Error:" << mysql_error(connection);
        exit(1);
    }
    std::cout << endl;
    exeSQL();
    return 0;

}

void myDB::Run()
{
        int i = 1;
    while(i != 0)
        {
            int sel;
            std::cout << "1、查询    2、添加    3、删除    4、修改    5、退出" <<
std::endl;
            std::cin >> sel;
            switch(sel)
            {
            case 1:exeSQL();
                break;
            case 2:insertSQL();
                break;
            case 3:deleteSQL();
                break;
            case 4:updateSQL();
                break;
```

```
            case 5:i = 0;
                break;
            default:std::cout << "error" << std::endl;
                break;
            }
        }
}
```

main.cpp

```
#include <iostream>
#include "myDB.h"

int main()
{
    myDB db;
    db.initDB("localhost", "root", "wei123", "wei");
    db.Run();
    return 0;
}
```

以上为全部代码,接下来我makefile文件

makefile

```
mydb:main.cpp myDB.cpp
    g++ -o mydb main.cpp myDB.cpp -lmysqlclient

clean:
    rm -f *.o mydb
```

二、C++对Mysql数据库的事务提交及回滚

> 之前在做有关数据库的操作时发现，有些内容应该作为一个事务一起提交，而不是每个都单独提交，这就需要把这些操作当做一个事务来处理。因为 mysql 默认的是自动提交，我们就需要用到 api——mysql_commit ()。

```
mysql_commit(MYSQL* mysql, my_bool mode);

PS:mode 为 1 时表示 ON，mode 为 0 时表示 OFF。
```

> 在关掉自动提交后，以后我们的操作就需要自己手动提交了，而这就是我们所需要的了，对于一系列的操作，如果都成功了，那我们就把他们提交上去，如果失败了就执行回滚，这里需要用到另外两个 api：

```
mysql_commit()
提交事务。

mysql_rollback()
回滚事务。
```

示例代码

```
void myDB::test()
{
    int i = 0;
    std::string sql_1 = "insert into user(id,username,password,level) values (
1,'username ','psd', 1 );";
    std::string sql_2 = "insert into user(id,username,password,level) values (
1,'username ','psd','1');";
//  std::string sql_2 = "update user set id = 4 where id =5;";
    int ON = 1;
    int OFF = 0;
    mysql_autocommit(connection,OFF);
    mysql_autocommit(connection,OFF);
    if(mysql_query(connection,sql_1.c_str()))
    {
        std::cout << "sql_1 was error" << std::endl;
        i = 1;
    }
    if(mysql_query(connection,sql_2.c_str()))
    {
        std::cout << "sql_2 was error" << std::endl;
        i = 1;
    }


    if(i == 1)
    {
        mysql_rollback(connection);
    }
    else
    {
        mysql_commit(connection);
    }
    mysql_autocommit(connection,ON);
}
```

这个代码可以直接用在第一部分C++对mysql数据库的增删改查操作里的，因为之前设置 `id` 为主键，不能重复，所以最后的结果就是 `sql_2` 执行错误，之前的操作全部回滚。