

**Mysql综合应用设计**

**学 院： 电子信息（微纳技术）学院**

**专 业： 微电子科学与工程**

**年 级： 2019级**

**姓 名： 皇甫素素**

**学 号： 2019205883**

**2021年12月20日**

**声明**

**本项目采用MIT许可协议，代码仓库如下：**

https://github.com/suyu610/qt\_mysql

**This project is licensed under MIT**

目录

[一、 需求分析 4](#_Toc11009)

[1. 基本需求 4](#_Toc1487)

[2. 拓展需求 4](#_Toc5219)

[二、 环境安装 4](#_Toc4922)

[1. 更新下载源 4](#_Toc9078)

[2. Mysql的安装与配置 4](#_Toc28860)

[(1) 使用apt-get安装mysql 4](#_Toc11053)

[(2) 配置MySQL 4](#_Toc124)

[(3) 检查MySQL启动情况 4](#_Toc28757)

[(4) 测试访问MySQL 5](#_Toc6252)

[(5) 设置访问MySQL权限 5](#_Toc9330)

[3. QT5.9安装 5](#_Toc29162)

[(1) 下载QT 5](#_Toc31128)

[(2) 安装 5](#_Toc1089)

[(3) 其他组件的安装 6](#_Toc26967)

[(4) 检查mysql 驱动是否安装好 6](#_Toc11346)

[(5) 启动QT 6](#_Toc18525)

[三、 业务逻辑 6](#_Toc21041)

[1. 数据库设计 7](#_Toc13321)

[2. 业务流程图 8](#_Toc1275)

[四、 效果展示 8](#_Toc4457)

[1. Native-C 8](#_Toc1598)

[2. QT 9](#_Toc20879)

[3. Web 9](#_Toc18846)

[五、 附录 9](#_Toc3354)

[1. 代码 9](#_Toc21149)

[2. Sql语句 13](#_Toc22878)

[3. 参考文献 14](#_Toc27058)

# 需求分析

## **基本需求**

分别使用C语言、QT和Web访问数据库中的成绩。

## **拓展需求**

能对成绩进行简单的增删改查（**未做**）

# 环境安装

## **更新下载源**

> sudo vim /etc/apt/sources.list

deb http://mirrors.aliyun.com/ubuntu/ xenial-security universe

deb http://mirrors.aliyun.com/ubuntu/ xenial-security multiverse

...

> apt-get update

> apt-get upgrade

## **Mysql的安装与配置**

1. **使用apt-get安装mysql**

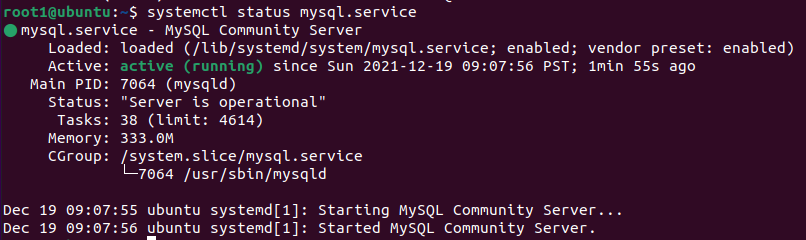
> sudo apt-get install mysql-server

1. **配置MySQL**

> sudo mysql\_secure\_installation

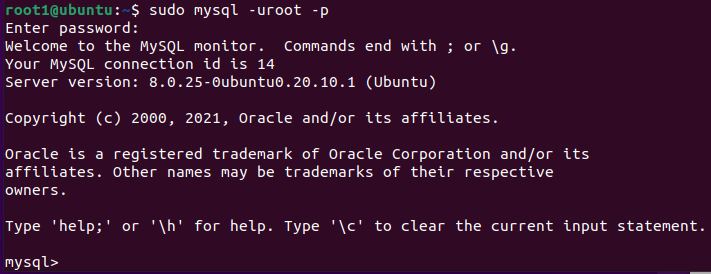
1. **检查MySQL启动情况**

> systemctl status mysql.service



1. **测试访问MySQL**

> mysql -uroot -p



1. **设置访问MySQL权限**

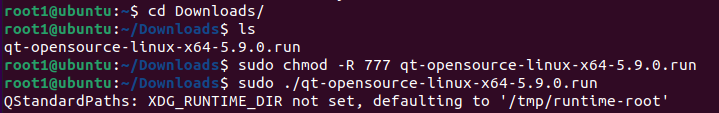
> alter user 'root'@'localhost' identified with mysql\_native\_password by 'your password'

## **QT5.9安装**

1. **下载QT**

<https://mirrors.tuna.tsinghua.edu.cn/qt/archive/qt/5.9/5.9.0/qt-opensource-linux-x64-5.9.0.run>

1. **安装**



1. **其他组件的安装**

> sudo apt-get install gcc g++

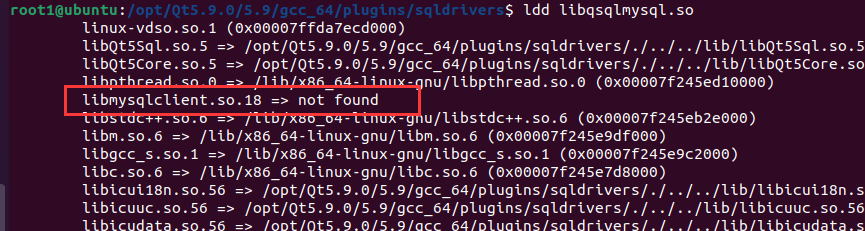
> sudo apt-get install libqt4-dev

> sudo apt-get install build-essential

1. **检查mysql 驱动是否安装好**

> cd /opt/Qt5.9.0/5.9/gcc\_64/plugins/sqldrivers

> ldd libqsqlmysql.so



如果出现libmysqlclient.so.18 => Not found

则> sudo wget -O /usr/lib/libmysqlclient.so.18 http://files.directadmin.com/services/es\_7.0\_64/libmysqlclient.so.18

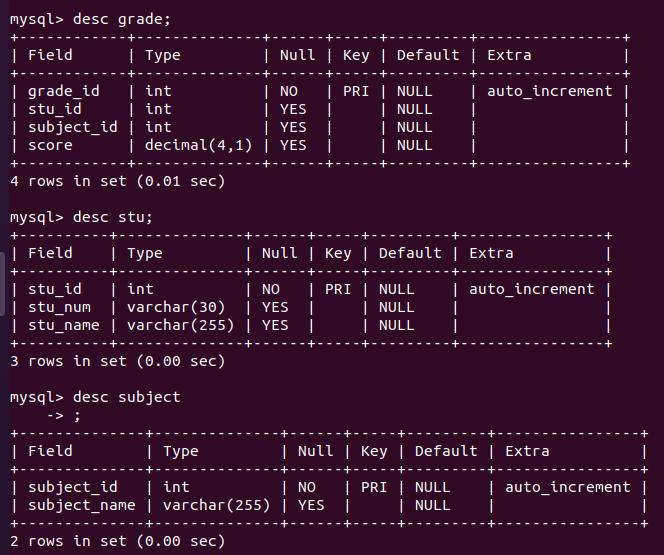
1. **启动QT**

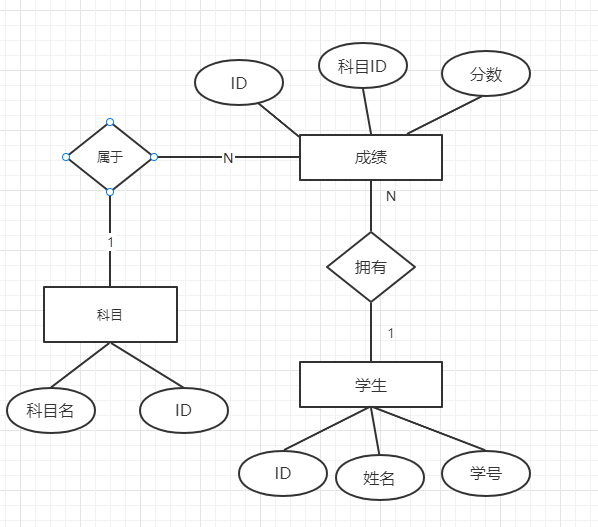
> cd opt/Qt5.9.0/Tools/QtCreator/bin

> ./qtcreator

# 业务逻辑

## **数据库设计**





## **业务流程图**

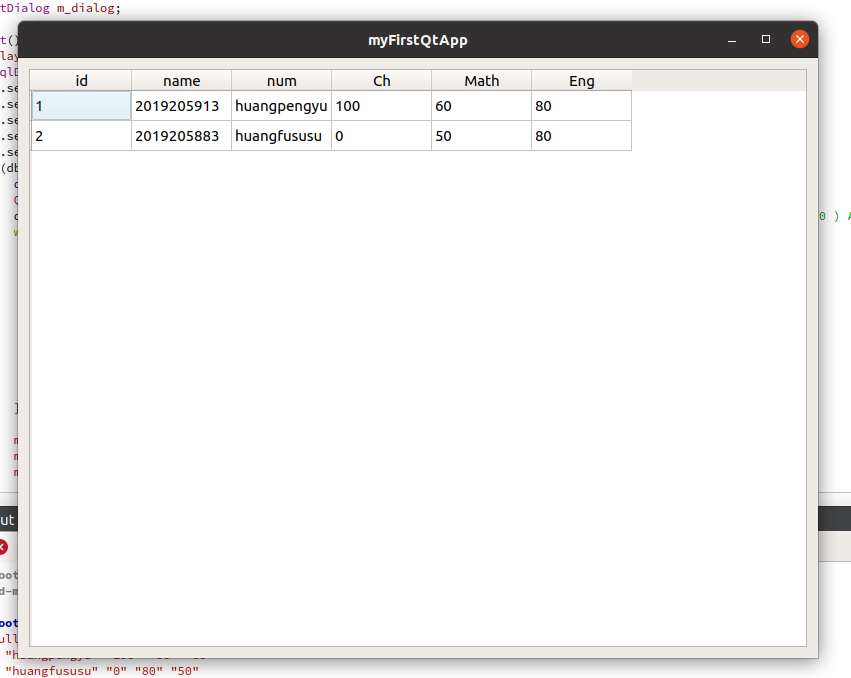
# 效果展示

## **Native-C**

> gcc -o sql\_c sql\_c.c -I/usr/include/mysql -L/usr/lib64/mysql -lmysqlclient



## **QT**



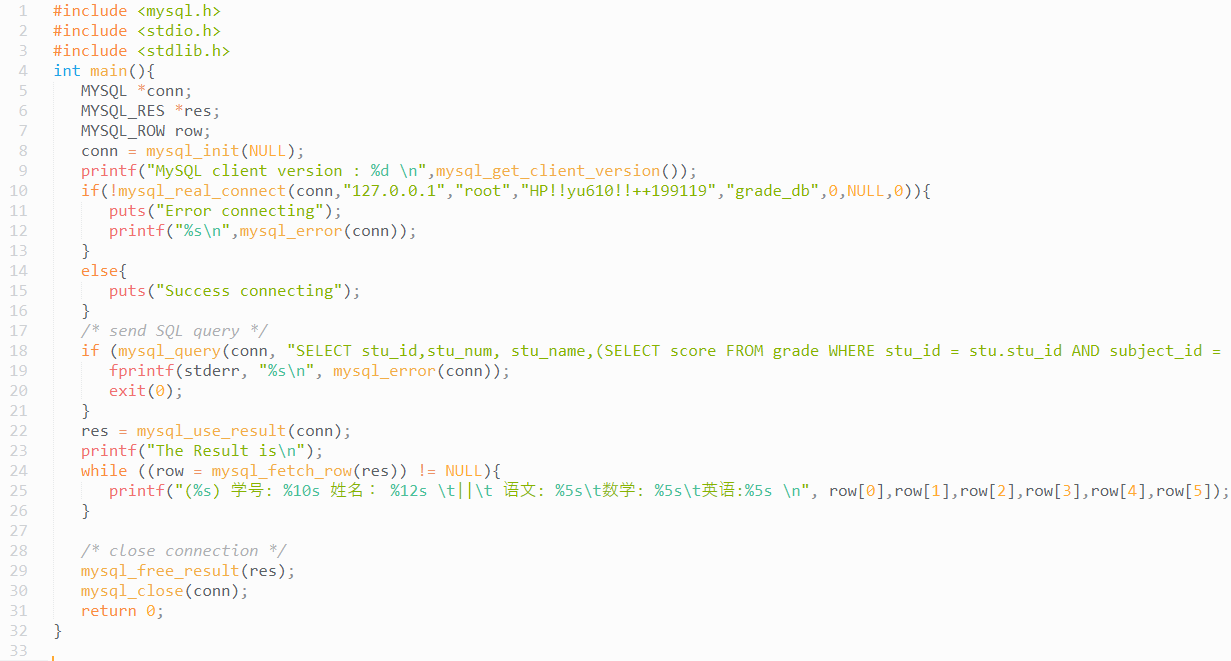
## **Web**

略

# 附录

## 代码

1. **Pure\_C连接Mysql**



#include <mysql.h>

#include <stdio.h>

#include <stdlib.h>

int main(){

MYSQL \*conn;

MYSQL\_RES \*res;

MYSQL\_ROW row;

conn = mysql\_init(NULL);

printf("MySQL client version : %d \n",mysql\_get\_client\_version());

if(!mysql\_real\_connect(conn,"127.0.0.1","root","密码","grade\_db",0,NULL,0)){

puts("Error connecting");

printf("%s\n",mysql\_error(conn));

}

else{

puts("Success connecting");

}

/\* send SQL query \*/

if (mysql\_query(conn, "SELECT stu\_id,stu\_num, stu\_name,(SELECT score FROM grade WHERE stu\_id = stu.stu\_id AND subject\_id = 0 ) AS ch,(SELECT score FROM grade WHERE stu\_id = stu.stu\_id AND subject\_id = 1 ) AS math,(SELECT score FROM grade WHERE stu\_id = stu.stu\_id AND subject\_id = 2 ) AS eng FROM stu;")) {

fprintf(stderr, "%s\n", mysql\_error(conn));

exit(0);

}

res = mysql\_use\_result(conn);

printf("The Result is\n");

while ((row = mysql\_fetch\_row(res)) != NULL){

printf("(%s) 学号: %10s 姓名： %12s \t||\t 语文: %5s\t数学: %5s\t英语:%5s \n", row[0],row[1],row[2],row[3],row[4],row[5]);

}

/\* close connection \*/

mysql\_free\_result(res);

mysql\_close(conn);

return 0;

}

1. **QT**

考虑到本次项目主要目的为学习QT与MySQL在unix下的应用，所以对于业务层并没有做过多的设计与解耦。

**Main.cpp**

#include <QApplication>

#include <QDebug>

#include <QtSql>

#include <QtGui>

#if QT\_VERSION\_MAJOR > 4

#include <QtWidgets>

#endif

class StuGrade {

QString m\_stuid, m\_stuname, m\_stunum,m\_ch,m\_math,m\_eng;

public:

StuGrade(const QString & stuid, const QString & stuname, const QString & stunum, const QString & ch, const QString & math, const QString & eng) :

m\_stuid{stuid}, m\_stuname{stuname}, m\_stunum{stunum},m\_ch{ch},m\_math(math),m\_eng{eng} {}

QString stuid() const { return m\_stuid; }

QString stuname() const { return m\_stuname; }

QString stunum() const { return m\_stunum; }

QString ch() const { return m\_ch; }

QString math() const { return m\_math; }

QString eng() const { return m\_eng; }

};

class StuGradeModel : public QAbstractTableModel {

QList<StuGrade> m\_data;

public:

StuGradeModel(QObject \* parent = {}) : QAbstractTableModel{parent} {}

int rowCount(const QModelIndex &) const override { return m\_data.count(); }

int columnCount(const QModelIndex &) const override { return 6; }

QVariant data(const QModelIndex &index, int role) const override {

if (role != Qt::DisplayRole && role != Qt::EditRole) return {};

const auto & stu = m\_data[index.row()];

switch (index.column()) {

case 0: return stu.stuid();

case 1: return stu.stuname();

case 2: return stu.stunum();

case 3: return stu.ch();

case 4: return stu.eng();

case 5: return stu.math();

default: return {};

};

}

QVariant headerData(int section, Qt::Orientation orientation, int role) const override {

if (orientation != Qt::Horizontal || role != Qt::DisplayRole) return {};

switch (section) {

case 0: return "id";

case 1: return "name";

case 2: return "num";

case 3: return "Ch";

case 4: return "Math";

case 5: return "Eng";

default: return {};

}

}

void append(const StuGrade & stu) {

beginInsertRows({}, m\_data.count(), m\_data.count());

m\_data.append(stu);

endInsertRows();

}

};

class Widget : public QWidget {

QGridLayout m\_layout{this};

QTableView m\_view;

StuGradeModel m\_model;

QSortFilterProxyModel m\_proxy;

QInputDialog m\_dialog;

public:

Widget() {

m\_layout.addWidget(&m\_view, 0, 0, 1, 1);

QSqlDatabase db = QSqlDatabase::addDatabase("QMYSQL");

db.setHostName("127.0.0.1");

db.setDatabaseName("grade\_db");

db.setPort(3306);

db.setUserName("root");

db.setPassword("HPyuko12!!");

if(db.open()){

qDebug()<<"connect successfully!";

QSqlQuery query(db);

query.exec(**"SELECT stu\_id,stu\_num, stu\_name,(SELECT score FROM grade WHERE stu\_id = stu.stu\_id AND subject\_id = 0 ) AS ch,(SELECT score FROM grade WHERE stu\_id = stu.stu\_id AND subject\_id = 1 ) AS math,(SELECT score FROM grade WHERE stu\_id = stu.stu\_id AND subject\_id = 2 ) AS eng FROM stu;"**);

while(query.next()){

QString id = query.value(0).toString();

QString stu\_num = query.value(1).toString();

QString stu\_name = query.value(2).toString();

QString ch = query.value(3).toString();

QString math = query.value(4).toString();

QString eng = query.value(5).toString();

m\_model.append({id,stu\_num,stu\_name, ch, math, eng});

qDebug()<< id << stu\_num << stu\_name << ch << math << eng;

}

m\_proxy.setSourceModel(&m\_model);

m\_proxy.setFilterKeyColumn(2);

m\_view.setModel(&m\_proxy);

}

else

qDebug()<<"failed!!!!";

}

};

int main(int argc, char \*argv[])

{

QApplication a{argc, argv};

Widget w;

w.resize(800,600);

w.show();

return a.exec();

}

## **Sql语句**

* **初始化**

CREATE DATABASE grade\_db;

DROP TABLE IF EXISTS `grade\_db`.`grade`;

DROP TABLE IF EXISTS `grade\_db`.`stu`;

DROP TABLE IF EXISTS `grade\_db`.`subject`;

CREATE TABLE `grade` (

`grade\_id` int NOT NULL AUTO\_INCREMENT,

`stu\_id` int DEFAULT NULL,

`subject\_id` int DEFAULT NULL,

`score` decimal(4,1) DEFAULT NULL,

PRIMARY KEY (`grade\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

CREATE TABLE `stu` (

`stu\_id` int NOT NULL AUTO\_INCREMENT,

`stu\_num` varchar(30) COLLATE utf8mb4\_general\_ci DEFAULT NULL,

`stu\_name` varchar(255) COLLATE utf8mb4\_general\_ci DEFAULT NULL,

PRIMARY KEY (`stu\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

CREATE TABLE `subject` (

`subject\_id` int NOT NULL AUTO\_INCREMENT,

`subject\_name` varchar(255) COLLATE utf8mb4\_general\_ci DEFAULT NULL,

PRIMARY KEY (`subject\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_general\_ci;

* **Stu.sql**

INSERT INTO `stu` (`stu\_id`, `stu\_num`, `stu\_name`) VALUES (1, '2019205913', 'hpy');

INSERT INTO `stu` (`stu\_id`, `stu\_num`, `stu\_name`) VALUES (2, '2019205883', 'huangfususu');

* **Grade.sql**

INSERT INTO `grade` (`grade\_id`, `stu\_id`, `subject\_id`, `score`) VALUES (0, 1, 0, 100.0);

INSERT INTO `grade` (`grade\_id`, `stu\_id`, `subject\_id`, `score`) VALUES (1, 1, 1, 80.0);

INSERT INTO `grade` (`grade\_id`, `stu\_id`, `subject\_id`, `score`) VALUES (2, 1, 2, 60.0);

INSERT INTO `grade` (`grade\_id`, `stu\_id`, `subject\_id`, `score`) VALUES (3, 2, 0, 63.5);

INSERT INTO `grade` (`grade\_id`, `stu\_id`, `subject\_id`, `score`) VALUES (4, 2, 1, 80.0);

INSERT INTO `grade` (`grade\_id`, `stu\_id`, `subject\_id`, `score`) VALUES (5, 2, 2, 50.0);

* **Subject.sql**

INSERT INTO `subject` (`subject\_id`, `subject\_name`) VALUES (0, 'ch');

INSERT INTO `subject` (`subject\_id`, `subject\_name`) VALUES (1, 'math');

INSERT INTO `subject` (`subject\_id`, `subject\_name`) VALUES (2, 'eng');

## **参考文献**

1. Ubuntu18.04上安装Qt5.10 步骤https://blog.csdn.net/weixin\_41477306/article/details/95743555