应用系统开发项目报告

题目：实验室设备查询管理系统

姓名：何兴国

学号：22920172204117

专业：人工智能

时间：2019/12/16

目录

一、 需求分析

本次实验在综合实现难度及课程应用后，我选择了实验室设备管理系统这个选题，此系统主要实现对实验室的大型设备（电脑、示波器、实验箱）及可容纳人数进行统计，并提供对数据的增加，修改，删除，查询操作。

1. 权限设置
   考虑到几乎所有系统都需要限制访问人员的权限，所以需要建立管理员表与游客表，对管理员提供增删改查操作权限，对游客仅提供查询操作。
   在登陆界面上提供游客注册的选项，而管理原的添加考虑安全性，必须由数据库管理人员从后台直接添加。
2. 增加操作
   可以向数据库中添加之前不存在的教室信息，若当前添加教室已存在，则会弹出 warning 信息，提醒管理员当前要添加的教室信息已存在。
   添加成功后弹出成功提示框
3. 修改操作
   修改已经存在的实验室信息，并同步修改相应的表，如：若选择修改的教室电脑数量为 0，则 computer 表中则不会有该教室的信息，在修改之后若不为 0，则必须在 computer 表中添加记录；
   在修改界面先输入教室号，点击选择按钮，若修改的教室不存在，则会弹出 warning 信息，提醒管理员无当前教室。
   修改成功后弹出成功提示框
4. 删除操作
   删除某间教室的信息，在此页面会显示所有实验室的大致信息，选中一间教室点击删除按钮会弹出操作确认提示，点击 yes 可以删除该教室信息，且会同步删除相应实验仪器表的信息。
   删除成功后会弹出成功提示框
5. 查询操作
   可以输入教书号点击查询按钮，在得出的结果上双击电脑或者示波器可以弹出该教室设备的详细信息。
   点击更多查询可以筛选出符合最小器材及人数要求的教室，同样提供双击查看设备详细信息的功能。

二、 系统分析

数据库方面，需要建立实验室表（Laboratory），包括六列：room（主键），电脑数量，示波器数量，实验箱数量，可容纳人数；电脑表（Computer），包括五列：address（外键），CPU，RAM，OS，数量；示波器表（Oscilloscope），包括三列：address（外键），带宽，数量；管理员表（Admin），包括两列：ID（主键），密码；游客表（Visitor），包括两列：ID（主键），密码。

三、 技术选型

本次实验数据库基于 SQL Sever2017，在用户界面及数据库交互的代码编写上，我使用了 QT for C++来实现，QT 提供了强大的用户界面设计功能，且可以通过 ODBC 方式连接到 SQL Sever，在对数据库的操作上，提供了十分方便的接口，但由于时间有限，在数据库的操作上还是使用了相对比较繁琐的 QSqlQuery 来执行 sql 语句。

四、 技术实现细节

1. 数据库连接

通过 ODBC 方式连接 SQL Sever，首先在电脑上设置 ODBC 数据源，具体方法会在说明书中说明，然后在 QT 中通过代码连接到 SQL Sever。

```
QSqlDatabase db = QSqlDatabase::addDatabase("QODBC");    //数据库驱动类型为SQL Server

bool OpenDatabase()
{

    qDebug()<<"ODBC driver?"<<db.isValid();
    QString dsn = QString::fromLocal8Bit("testdsn");     //数据源名称
    db.setHostName("localhost");                         //选择本地主机，127.0.1.1
    db.setDatabaseName(dsn);                             //设置数据源名称
    db.setUserName("sa");                                //登录用户
    db.setPassword("hxg1999323");                        //密码
    if(!db.open())                                       //打开数据库
    {
        qDebug()<<db.lastError().text();
        QMessageBox::critical(0, QObject::tr("Database error"), db.lastError().text());
        return false;                                    //打开失败
    }
    else
        qDebug()<<"database open success!";
    return true;
}
```

2. 用户登录

通过执行 select 语句在管理员表或游客表中查询，根据得到的结果集是否为空判断是否登陆成功，注册则执行 insert 语句向游客表中添加游客记录，注册必须保证用户名唯一。

登录：

```
void Widget::on_pushButton_clicked()
{
    int flag=0;
    QSqlQuery qry;
    QString sql="select * from Admin where ID='"+id+"'"+"and pwd='"+pwd+"'";
    if(qry.exec(sql)){
        if(qry.next()){
            ui->label_4->clear();
            ui->label_4->setText("登陆成功");
            flag=1;
        }
        else{
            ui->label_4->clear();
            ui->label_4->setText("账号或密码错误");
        }

    }
    else{
        qDebug()<<db.lastError().text();
    }
    if(flag)
    {
        this->hide();
        Adrights a;
        a.exec();
    }
}
```

注册:

```
void ADlog::on_pushButton_clicked()
{
    ui->label_5->clear();
    QSqlQuery qry;
    QString sql="select * from Visitor where ID='"+logid+"'";
    if(qry.exec(sql)){
        if(qry.next()){

            ui->label_5->setText("用户名已存在");
        }
        else{
            if(logpwd!=logpwd2){
                ui->label_5->setText("两次密码输入不同，请重新输入");
                ui->lineEdit_2->clear();
                ui->lineEdit_3->clear();
            }
            else{
                QSqlQuery sign;
                QString signsql="insert Visitor values ('"+logid+"','"+logpwd+"')";
                if(sign.exec(signsql)){
                    ui->label_5->setText("注册成功");
                }
                else{
                    qDebug()<<db.lastError().text();
                }
            }
        }
    }
    else
    {
        qDebug()<<db.lastError().text();
    }
}
```

3. 增加记录

向实验室表和其余实验设备表中添加记录，需要注意的是，需要提前判断电脑和示波器的数量是否为 0，若为零则不需要向这两个表中添加记录

```
QString insert="insert Laboratory values ('"+room+"',"+numofcom+","+numofosc+","+numofcha+","+capaci
        ")";
if(numofcom!="0")
{
    insert+=" insert Computer values ('"+room+"','"+cpu+"','"+ram+"','"+os+"',"+numofcom+")";
}
if(numofosc!="0")
{
    insert+=" insert Oscilloscope values('"+room+"','"+bandwidth+"',"+numofosc+")";
}
```

添加时需要判断当前实验室是否存在，存在则弹出提示窗口

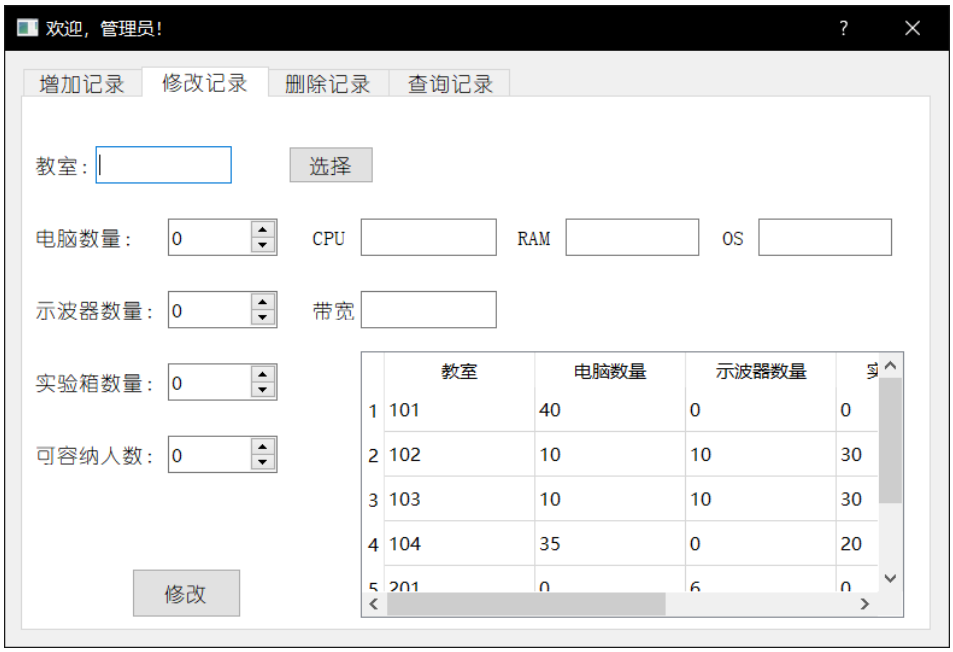
```
QString check="select *from Laboratory where room='"+room+"'";
QSqlQuery qry;
if(qry.exec(check))
{
    if(qry.next())
    {
        QString title="warning";
        QString info="当前实验室信息已存在";
        QMessageBox::warning(this,title,info,QMessageBox::Ok,QMessageBox::NoButton);
    }
```

同步更新增加记录、修改记录、删除记录中的 tableview 以保持数据的一致性

```
model->setQuery("select *from Laboratory ",db);
model2->setQuery("select *from Laboratory ",db);
model3->setQuery("select *from Laboratory ",db);

ui->tableView->setModel(model);
ui->tableView_2->setModel(model2);
ui->tableView_5->setModel(model3);
```

4. 修改记录



修改已有的记录，先填写教室号码，点击选择，若教室不存在则会弹出提示框

```
QString room2=ui->lineEdit_6->text();
QString check="select *from Laboratory where room='"+room2+"'";
QSqlQuery qry;
if(qry.exec(check))
{
    if(qry.next()) {...}
    else{
        QString title="warning";
        QString info="无当前实验室信息";
        QMessageBox::warning(this,title,info,QMessageBox::Ok,QMessageBox::NoButton);
    }
}
```

然后将查询到的信息填在相应的位置，根据需要进行修改，点击修改按钮后完成修改并弹出成功对话框。在修改操作中，若原电脑数量为 0，修改后不为零则需要在 Computer 表在添加记录，示波器同理：

```
update="update Laboratory set numofcom="+ui->spinBox_5->text()+"where room='" +ui->lineEdit_
update+="insert Computer values ('"+ui->lineEdit_6->text()+"','"+ui->lineEdit_7->text()+"',
        "','"+ui->lineEdit_9->text()+"',"+ui->spinBox_5->text()+")";
if(!qry.exec(update))
{
    flagcom=false;
}
```

```
update="update Laboratory set numofosc="+ui->spinBox_6->text()+"where room='" +ui->lineEdit_
update+="insert Oscilloscope values ('"+ui->lineEdit_6->text()+"','"+ui->lineEdit_10->text()
if(!qry.exec(update))
{
    flagosc=false;
}
```

若原数量不为零，修改后为零，则需要在相应的表中删除记录：

```
if(ui->spinBox_5->value()==0)
{
    update="update Laboratory set numofcom=0 where room='" +ui->lineEdit_6->text()+"'";
    if(!qry.exec(update))
    {
        flagcom=false;
    }
    update="delete from Computer where address='"+ui->lineEdit_6->text()+"'";
    if(!qry.exec(update))
    {
        flagcom=false;
    }
}
if(ui->spinBox_6->value()==0)
{
    update="update Laboratory set numofosc=0 where room='" +ui->lineEdit_6->text()+"'";
    if(!qry.exec(update))
    {
        flagosc=false;
    }
    update="delete from Oscilloscope where address='"+ui->lineEdit_6->text()+"'";
    if(!qry.exec(update))
    {
        flagosc=false;
    }
}
```

其余的修改就是简单的判断与原数据是否相同并使用 update 语句进行修改，如：

```
//带宽修改
if(ui->lineEdit_10->text()!=bandwidth2){
    if(ui->spinBox_6->value()!=0)
    {
        update="update Oscilloscope set bandwidth='"+ui->lineEdit_10->text()+"' where address='" +ui->li
        if(!qry.exec(update))
        {
            flagbw=false;
        }
    }
}
```

同样修改后需要同步更新三个 tableview 以保持数据的一致性

5. 删除记录

为了防止误操作，增加了操作确认提示框：

```
QString dlgTitle="Question消息框";
QString strInfo="确认删除 "+delroom+" 实验室信息吗？";
QMessageBox::StandardButton  defaultBtn=QMessageBox::NoButton; //缺省按钮
QMessageBox::StandardButton result;//返回选择的按钮
result=QMessageBox::question(this, dlgTitle, strInfo,
                        QMessageBox::Yes|QMessageBox::No |QMessageBox::Cancel,
                        defaultBtn);
if(result== QMessageBox::Yes) {...}
```

由于 Computer 表、Oscilloscope 表和 Laboratory 表存在外键关系，所以在删除是需要先删除前两个表的相关记录，再删除 Laboratory 表中的记录：

```
if(qry.value(1).toInt()!=0)
{
    QSqlQuery qry2;
    QString del="delete from Computer where address='"+delroom+"'";
    if(!qry2.exec(del))
    {
        flag=false;
        QString title="error";
        QString info="出现错误！";
        QMessageBox::critical(this,title,info,QMessageBox::Ok,QMessageBox::NoButton);
    }
}
if(qry.value(2).toInt()!=0)
{
    QSqlQuery qry2;
    QString del="delete from Oscilloscope where address='"+delroom+"'";
    if(!qry2.exec(del))
    {
        flag=false;
        QString title="error";
        QString info="出现错误！";
        QMessageBox::critical(this,title,info,QMessageBox::Ok,QMessageBox::NoButton);
    }
}
QSqlQuery qry2;
QString del="delete from Laboratory where room='"+delroom+"'";
if(!qry2.exec(del))
{
    flag=false;
    QString title="error";
    QString info="出现错误！";
    QMessageBox::critical(this,title,info,QMessageBox::Ok,QMessageBox::NoButton);
}
```

6. 查询记录

按教室查询则输入教室信息点击查询即可

```
void Adrights::on_pushButton_9_clicked()
{
    QString searchroom=ui->lineEdit_21->text();
    QSqlQuery qry;
    QString sel="select *from Laboratory where room='"+searchroom+"'";
    if(qry.exec(sel)){
        if(!qry.next()){
            QString title="warning";
            QString info="无当前实验室信息";
            QMessageBox::warning(this,title,info,QMessageBox::Ok,QMessageBox::NoButton);
        }
        else{
            model4->setQuery(sel,db);
            model4->setHeaderData(0,Qt::Horizontal,QObject::tr("教室"));
            model4->setHeaderData(1,Qt::Horizontal,QObject::tr("电脑数量"));
            model4->setHeaderData(2,Qt::Horizontal,QObject::tr("示波器数量"));
            model4->setHeaderData(3,Qt::Horizontal,QObject::tr("实验箱数量"));
            model4->setHeaderData(4,Qt::Horizontal,QObject::tr("可容纳人数"));
            ui->tableView_6->setModel(model4);
        }
    }
}
```

可以双击电脑或示波器的单元格显示当前教室电脑或示波器的详细信息，使用双击作为槽函数触发：

```
//查询双击显示详细信息
void Adrights::on_tableView_6_doubleClicked(const QModelIndex &index)
{
    QAbstractItemModel *Imodel=ui->tableView_6->model();
    QModelIndex Iindex = Imodel->index(index.row(),0);  //index.row()为算选择的行号。1为所选中行的第一列。
    QVariant datatemp=Imodel->data(Iindex);
    QString name=Imodel->data(index).toString();        //name即为所选择行的第一列的值
    qDebug()<<name<<datatemp.toString()<<index.column();
    if(index.column()==1){
        if(name!="0"){
            QString room=datatemp.toString();
            detail_inf="select *from Computer where address='"+room+"'";
            details d;
            d.exec();
        }
    }
    else if(index.column()==2){
        if(name!="0"){
            QString room=datatemp.toString();
            detail_inf="select *from Oscilloscope where address='"+room+"'";
            details d2;
            d2.exec();
        }
    }
}
```

7. 更多查询

更多查询提供了按电脑等器材数量和可容纳人数进行查询的方法，此界面与游客登录后的界面相同。查询时同样使用 select 语句查询，查询结果也支持查看详细信息的功能，更多查询的代码在 searchlog.cpp 中实现，在此就不予展示了。

五、 项目总结

作为数据挖掘的大作业，在课上只讲了 SQL Sever 和 SQL 语句的前提下完成这个作业，还是有比较大的难度的，使用什么开发工具，如何连接到数据库，如何通过自编写的界面操作数据库等等，通过查阅大量资料，并结合实际情况我选择了这个选题并加以实现，从结果上看，实现了预期的功能，但也存在很多遗憾，比如界面不够美观，代码结构有些混乱，功能可以更加完善等等，这些问题有机会的话后续会逐步解决。总而言之本次作业使我对数据库的操作有了更深刻的理解，编程能力得到提升。