**南昌航空大学**

**20学年—21学年第 2 学期 医疗软件技术基础 实验五**

专业名称： 生物医学工程 实验学时： 2

学号： 18084124 姓名： 杨振宇

实验题目： WEB数据库编程

实验环境： vs2008

实验目的：

1．掌握ADO.NET数据库操作组件；

2．掌握三层网络数据库构架；

3. 掌握CASE工具开发。

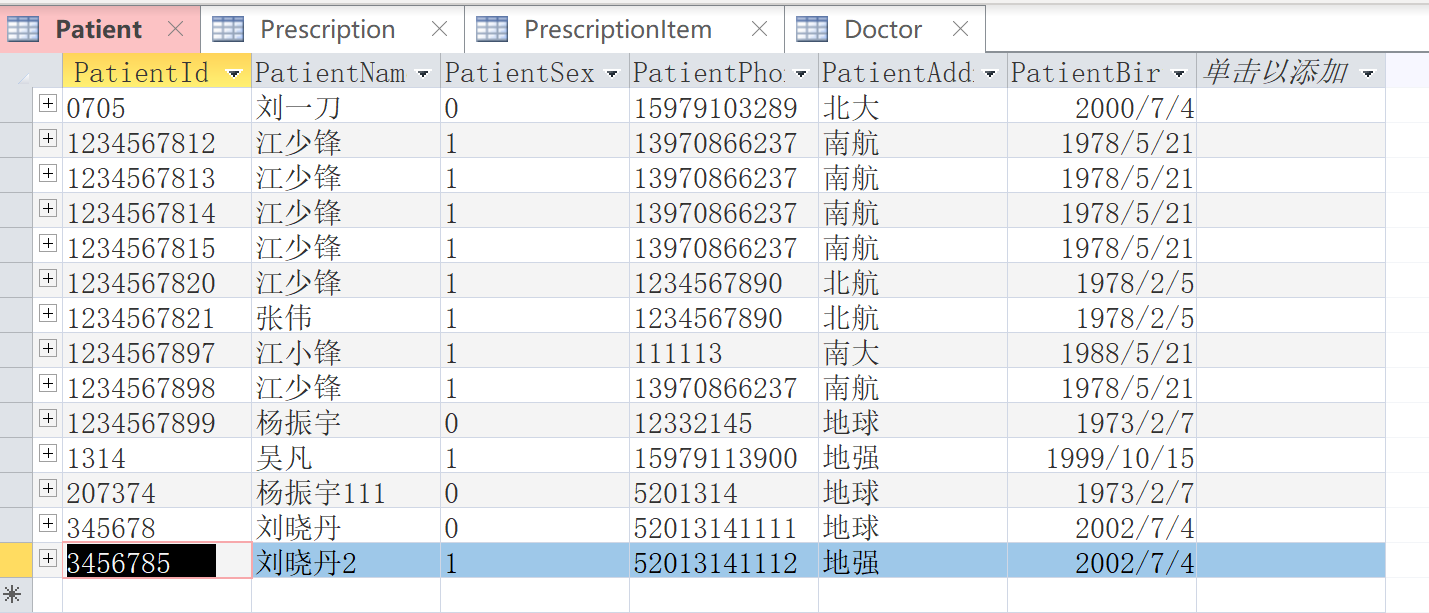
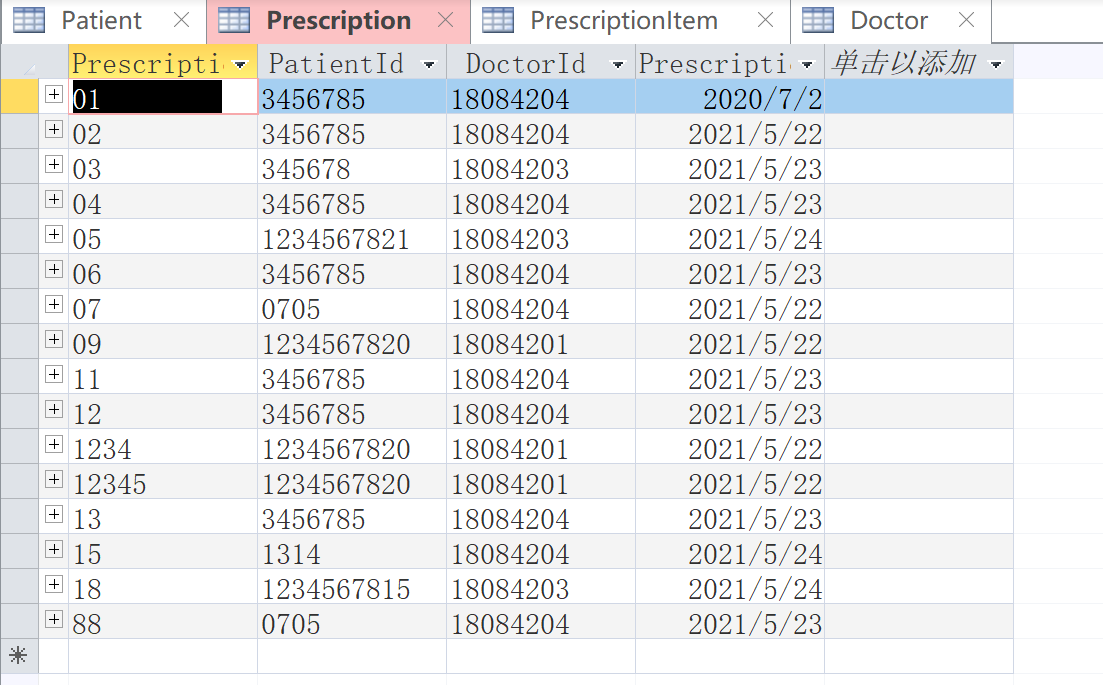
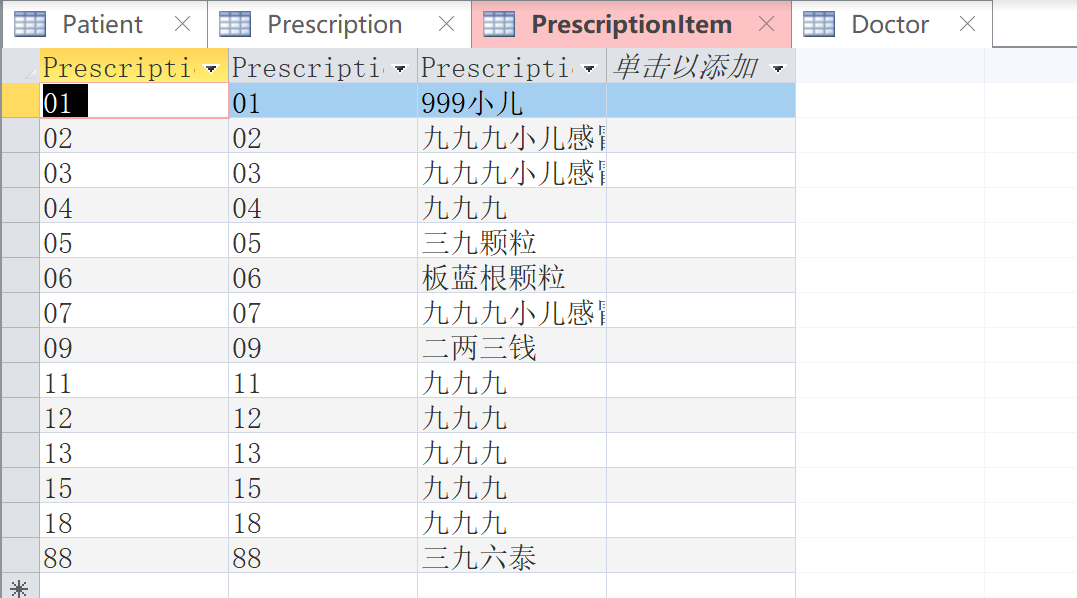
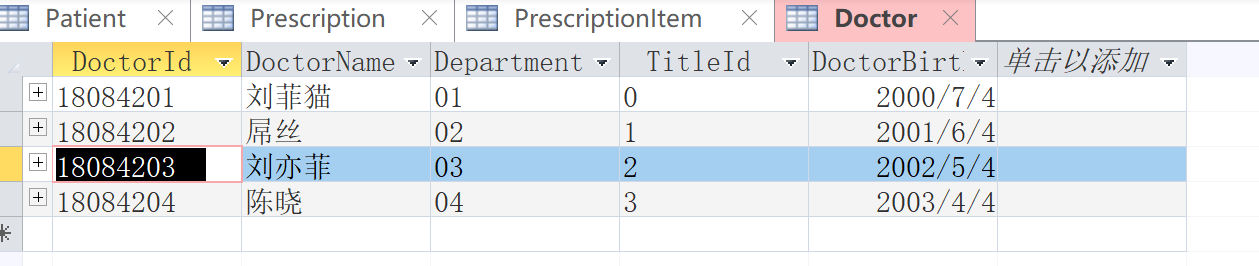
实验内容：

（1）通过powderdesign设计数据库和实体类；

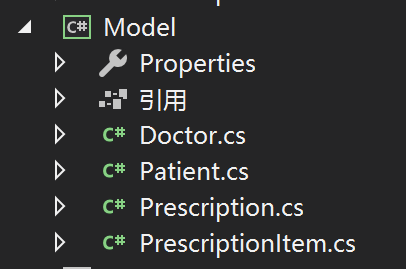
（2）分别建立表示层、业务逻辑层和数据访问层；

（3）实现数据的添加、删除、修改、查询操作；

(1) 详细说明数据库和实体类设计。

数据库与model层直接进行数据交换对应四个类



医生有 public string doctorId;

public string doctorName;

public string departmentId;

public string titleId;

public DateTime doctorBirthday;

病人有 public string patientId;

public string patientName;

public string patientSex;

public string patientPhone;

public string patientAddress;

public DateTime patientBirthday;

处方有

public string prescriptionID;

public DateTime prescriptionTime;

处方项目有

public string prescriptionItemID;

public string prescriptionItem;

public Prescription prescription;

(2) 画出网站结构图。

开始

查询所有病人信息

反映在网站上

结束

(3) 画出操作界面及说明。

查询

输入病人信息

显示查询结果

选中

选中病人信息

输出选中病人信息

删除

选中要删除的病人信息

将改病人信息从数据库中删除

(4) 给出代码和注释。

using System;

using System.Data;

using System.Configuration;

using System.Linq;

using System.Web;

using System.Web.Security;

using System.Web.UI;

using System.Web.UI.HtmlControls;

using System.Web.UI.WebControls;

using System.Web.UI.WebControls.WebParts;

using System.Xml.Linq;

/// <summary>

///doctorBLL 的摘要说明

/// </summary>

public class doctorBLL

{

    public doctorBLL()

    {

        //

        //TODO: 在此处添加构造函数逻辑

        //

    }

    public DataSet listallDoctor()//从列表中列出数据

    {

        doctorDA da = new doctorDA();

        DataSet ds = new DataSet();

        ds = da.getallDoctor();

        return ds;

    }

    public DataSet getDoctorId(string id)//获取

    {

        doctorDA da = new doctorDA();

        DataSet ds = new DataSet();

        ds = da.getdoctorbyID(id);//调用DA获取函数

        return ds;

    }

    public DataSet QueryDcotor(Doctor dr)//查询

    {

        doctorDA da = new doctorDA();

        String sqlstr = "select DoctorId as 医生编号,DoctorName as 医生姓名,DepartmentId as 科诊号,TitleId as 职称号,DoctorBirthday as 生日 from Doctor where ";

        int n = 0;//n为第几个条件

        if (dr.doctorId.Length > 0)

        {

            if (n > 0) sqlstr = sqlstr + " and";

            sqlstr = sqlstr + " DoctorId=" + "'" + dr.doctorId + "'";//单引号 将值变成一个字符型

            n = n + 1;

        }

        if (dr.doctorName.Length > 0)

        {

            if (n > 0) sqlstr = sqlstr + " and";

            sqlstr = sqlstr + "DoctorName=" + "'" + dr.doctorName + "'";

            n = n + 1;

        }

        if (dr.departmentId.Length > 0)//查询地址

        {

            if (n > 0) sqlstr = sqlstr + " and";

            sqlstr = sqlstr + "DepartmentId=" + "'" + dr.departmentId + "'";

            n = n + 1;

        }

        if (n == 0) sqlstr = "select DoctorId as 编号,DoctorName as 医生姓名,DepartmentId as 科诊号,TitleId as 职称号,DoctorBirthday as 生日 from Doctor";

        return da.GetDoctorByinfo(sqlstr);//字符串作为参数传递给业务逻辑层

    }

}

using System;

using System.Data;

using System.Configuration;

using System.Linq;

using System.Web;

using System.Web.Security;

using System.Web.UI;

using System.Web.UI.HtmlControls;

using System.Web.UI.WebControls;

using System.Web.UI.WebControls.WebParts;

using System.Xml.Linq;

/// <summary>

///patientBLL 的摘要说明

/// </summary>

public class patientBLL//业务逻辑层

{

    public patientBLL()

    {

        //

        //TODO: 在此处添加构造函数逻辑

        //

    }

    public DataSet listallPatient() //从列表中列出数据

    {

        patientDA da = new patientDA();//数据访问层

        DataSet ds = new DataSet();

        ds = da.getallPatient();

        return ds;

    }

    public void removePatientId(string id)//移除 无返回值

    {

        patientDA da = new patientDA();

        da.deletepatientbyID(id);//调用DAL移除函数

    }

    public DataSet getPatientId(string id)//获取 有返回值

    {

       patientDA da = new patientDA();

       DataSet ds = new DataSet();

       ds=da.getpatientbyID(id);//调用DA选择函数

       return ds;

    }

    public void Insert(Patient pt)//添加

    {

        patientDA da = new patientDA();

        da.Insert(pt);

    }

    public void modifipatientbyID(Patient pt)//修改

    {

        patientDA da = new patientDA();

        da.updatebypatientID(pt);//调用DAL修改函数

    }

    public String IsinByid(String id)

    {

        patientDA da = new patientDA();

        return da.GetbyIdIsIn(id);//数病人个数

    }

    public DataSet QueryPatient(Patient pt)//通过一个病人的对象查询，

    {

        patientDA da = new patientDA();

        String sqlstr = "select PatientId as 编号,PatientName as 姓名,PatientSex as 性别,PatientAddress as 住址,PatientBirthday as 生日,PatientPhone as 电话 from Patient where ";

        int n = 0;//n为第几个条件

        if (pt.patientId.Length > 0)

        {

            if (n > 0) sqlstr = sqlstr + " and ";

            sqlstr = sqlstr + " PatientId="+ "'" + pt.patientId + "'";//单引号 将值变成一个字符型

            n = n + 1;

        }

        if(pt.patientName.Length > 0)

        {

            if(n > 0) sqlstr=sqlstr + " and ";

            sqlstr = sqlstr + "PatientName=" + "'" + pt.patientName + "'";

            n = n + 1;

        }

        if (pt.patientSex.Length > 0)

        {

            if (n > 0) sqlstr = sqlstr + " and ";

            sqlstr = sqlstr + " PatientSex=" + "'" + pt.patientSex + "'";

            n = n + 1;

        }

        if (pt.patientPhone.Length > 0)

        {

            if (n > 0) sqlstr = sqlstr + " and ";

            sqlstr = sqlstr + " PatientPhone=" + "'" + pt.patientPhone + "'";

            n = n + 1;

        }

        if (pt.patientAddress.Length > 0)//查询地址

        {

            if (n > 0) sqlstr = sqlstr + " and ";

            sqlstr = sqlstr + "PatientAddress=" + "'" + pt.patientAddress + "'";

            n = n + 1;

        }

        if (n == 0) sqlstr = "select PatientId as 编号,PatientName as 姓名,PatientSex as 性别,PatientAddress as 住址,PatientBirthday as 生日,PatientPhone as 电话 from Patient";

        return da.GetPatientByinfo(sqlstr);//字符串作为参数传递给业务逻辑层

    }

//以上略---------------

}

using System;

using System.Data;

using System.Configuration;

using System.Linq;

using System.Web;

using System.Web.Security;

using System.Web.UI;

using System.Web.UI.HtmlControls;

using System.Web.UI.WebControls;

using System.Web.UI.WebControls.WebParts;

using System.Xml.Linq;

/// <summary>

///prescriptionBLL 的摘要说明

/// </summary>

public class prescriptionBLL

{

    public prescriptionBLL()

    {

        //

        //TODO: 在此处添加构造函数逻辑

        //

    }

    public DataSet getPrescriptionId(string id)//获取处方id

    {

        prescriptionDA da = new prescriptionDA();

        DataSet ds = new DataSet();

        ds = da.getprescriptionbyID(id);//调用DA获取函数

        return ds;

    }

    public void Insert(Prescription pre,String drid,String ptid)//添加处方编号 病人编号 医生编号 处方时间

    {

        prescriptionDA da = new prescriptionDA();

        String str="insert into Prescription values('" + pre.prescriptionID + "','" + ptid + "','" + drid + "','" + pre.prescriptionTime + "')";

        da.Insert(str);

    }

}

using System;

using System.Data;

using System.Configuration;

using System.Linq;

using System.Web;

using System.Web.Security;

using System.Web.UI;

using System.Web.UI.HtmlControls;

using System.Web.UI.WebControls;

using System.Web.UI.WebControls.WebParts;

using System.Xml.Linq;

/// <summary>

///prescriptionItemBLL 的摘要说明

/// </summary>

public class prescriptionItemBLL

{

    public prescriptionItemBLL()

    {

        //

        //TODO: 在此处添加构造函数逻辑

        //

    }

    public DataSet getPrescriptionItemId(string id)

    {

        prescriptionItemDA da = new prescriptionItemDA();

        DataSet ds = new DataSet();

        ds = da.getprescriptionItembyID(id);//调用DA获取函数

        return ds;

    }

    public void Insert(PrescriptionItem pitem,string Preid)//添加处方信息 药品编号 处方编号 药品

    {

        prescriptionItemDA da = new prescriptionItemDA();

        String str = "insert into PrescriptionItem values('" + pitem.prescriptionItemID + "','" + Preid + "','" + pitem.prescriptionItem + "')";

        da.Insert(str);

    }

}

public doctorDA()

    {

        //

        //TODO: 在此处添加构造函数逻辑

        //

    }

    public DataSet getallDoctor()//获取所有医生，连接字符串 命令 执行 返回

    {

        OleDbConnection conn = new OleDbConnection();

        conn.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=D:\\medical\\App\_Data\\HIS.mdb;User Id=admin;Password=";

        conn.Open();

        OleDbCommand selectComm = new OleDbCommand();

        selectComm.Connection = conn;

        selectComm.CommandType = CommandType.Text;

        selectComm.CommandText = "select DoctorId as 医生编号,DoctorName as 医生姓名,DepartmentId as 科诊号,TitleId as 职称号,DoctorBirthday as 生日 from Doctor";

        OleDbDataAdapter ad = new OleDbDataAdapter();

        ad.SelectCommand = selectComm;

        DataSet ds = new DataSet();

        ad.Fill(ds);

        conn.Close();

        return (ds);//数据访问层

    }

    public DataSet getdoctorbyID(string id)//获取函数

    {

        OleDbConnection conn = new OleDbConnection();

        conn.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=D:\\medical\\App\_Data\\HIS.mdb;User Id=admin;Password=";

        conn.Open();

        OleDbCommand selectComm = new OleDbCommand();

        selectComm.Connection = conn;

        selectComm.CommandType = CommandType.Text;

        selectComm.CommandText = "select \* from Doctor where DoctorId =@Did";

        selectComm.Parameters.Add("Did", OleDbType.Char, 10);

        selectComm.Parameters[0].Value = id;//第一个参数为id

        // deleteComm.ExecuteNonQuery();

        OleDbDataAdapter ad = new OleDbDataAdapter();

        ad.SelectCommand = selectComm;

        DataSet ds = new DataSet();

        ad.Fill(ds);

        conn.Close();

        return (ds);

    }

    public DataSet GetDoctorByinfo(String sqlstr)//查询函数

    {

        OleDbConnection conn = new OleDbConnection();//建立一个连接

        conn.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=D:\\medical\\App\_Data\\HIS.mdb;User Id=admin;Password=";//设置连接字符串

        conn.Open();//连接建立

        OleDbDataAdapter adpter = new OleDbDataAdapter();//建立数据适配层

        OleDbCommand comm = new OleDbCommand();//建立数据适配对象 命令对象

        comm.Connection = conn;//命令对象的连接等于conn

        comm.CommandText = sqlstr;//命令对象执行的指令为sqlstr

        adpter.SelectCommand = comm;//数据适配层的属性selectcommand直接设置为comm

        DataSet ds = new DataSet();//建立数据层，Dataset对象

        adpter.Fill(ds);//用数据适配对象执行Fill指令，将查询的数据放到Dataset对象ds里面

        conn.Close();//连接关闭

        return ds;//返回ds

    }

    public patientDA()//默认的构造函数

    {

        //

        //TODO: 在此处添加构造函数逻辑

        //

    }

    public DataSet getallPatient()//获取所有病人（连接字符串 命令 执行 返回）

    {

        OleDbConnection conn = new OleDbConnection();

        conn.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=D:\\medical\\App\_Data\\HIS.mdb;User Id=admin;Password=";//设置连接字符串

        conn.Open();

       OleDbCommand selectComm = new OleDbCommand();

        selectComm.Connection = conn;//连接

        selectComm.CommandType = CommandType.Text;//设置类型为文本型

        selectComm.CommandText = "select PatientId as 编号,PatientName as 姓名,PatientSex as 性别,PatientAddress as 住址,PatientBirthday as 生日,PatientPhone as 电话 from patient";

        OleDbDataAdapter ad = new OleDbDataAdapter();//建立数据适配层

        ad.SelectCommand = selectComm;//指令为selectComm

        DataSet ds = new DataSet();

        ad.Fill(ds);//命令送到这来

        conn.Close();

        return (ds);//返回数据  数据访问层

    }

    public void deletepatientbyID(string id)//通过传递参数删除某个病人函数

    {

        OleDbConnection conn = new OleDbConnection();

        conn.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=D:\\medical\\App\_Data\\HIS.mdb;User Id=admin;Password=";

        conn.Open();

        OleDbCommand deleteComm = new OleDbCommand();

        deleteComm.Connection = conn;

        deleteComm.CommandType = CommandType.Text;

        deleteComm.CommandText = "delete from patient where PatientId =@Pid";

        deleteComm.Parameters.Add("Pid", OleDbType.Char, 10);

        deleteComm.Parameters[0].Value = id;//多个参数 第0个参数为id

        deleteComm.ExecuteNonQuery();//非查询指令

        conn.Close();

    }

    public DataSet getpatientbyID(string id)//获取函数,通过id得到病人信息

    {

        OleDbConnection conn = new OleDbConnection();

        conn.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=D:\\medical\\App\_Data\\HIS.mdb;User Id=admin;Password=";

        conn.Open();

        OleDbCommand selectComm = new OleDbCommand();

        selectComm.Connection = conn;

        selectComm.CommandType = CommandType.Text;

        selectComm.CommandText = "select \* from patient where PatientId =@Pid";

        selectComm.Parameters.Add("Pid", OleDbType.Char, 10);

        selectComm.Parameters[0].Value = id;

       // deleteComm.ExecuteNonQuery();

        OleDbDataAdapter ad = new OleDbDataAdapter();

        ad.SelectCommand = selectComm;

        DataSet ds = new DataSet();

        ad.Fill(ds);

        conn.Close();

        return (ds);

    }

    public void Insert(Patient pt)//添加病人信息的函数

    {

        OleDbConnection conn = new OleDbConnection();

        conn.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=D:\\medical\\App\_Data\\HIS.mdb;User Id=admin;Password=";

        conn.Open();

        OleDbCommand comm = new OleDbCommand();

        comm.Connection = conn;

        comm.CommandText = "insert into Patient (PatientId,PatientName,PatientSex,PatientAddress,PatientPhone,PatientBirthday)values('" + pt.patientId + "','" + pt.patientName + "','" + pt.patientSex + "','" + pt.patientAddress + "','" + pt.patientPhone + "','"+pt.patientBirthday+"')";

        comm.Parameters.Add("Pname", OleDbType.Char, 5);

        comm.Parameters.Add("Psex", OleDbType.Char, 1);

        comm.Parameters.Add("Paddress", OleDbType.Char, 10);

        comm.Parameters.Add("Pphone", OleDbType.Char, 18);

        comm.Parameters.Add("Pbirth", OleDbType.Date);

        comm.Parameters.Add("Pid", OleDbType.Char, 15);

        comm.Parameters[0].Value = pt.patientName;

        comm.Parameters[1].Value = pt.patientSex;

        comm.Parameters[2].Value = pt.patientPhone;

        comm.Parameters[3].Value = pt.patientAddress;

        comm.Parameters[4].Value = pt.patientBirthday;

        comm.Parameters[5].Value = pt.patientId;

        comm.ExecuteNonQuery();

        conn.Close();

    }

    public void updatebypatientID(Patient pt)//更新函数----无返回---传递进来为病人的对象

    {

        OleDbConnection conn = new OleDbConnection();

        conn.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=D:\\medical\\App\_Data\\HIS.mdb;User Id=admin;Password=";

        conn.Open();

        OleDbCommand updateComm = new OleDbCommand();

        updateComm.Connection = conn;

        updateComm.CommandType = CommandType.Text;

        updateComm.CommandText = "update Patient  set PatientName=@Pname,PatientSex=@Psex,PatientPhone=@Pphone,PatientAddress=@Paddress,PatientBirthday=@Pbirth where PatientID=@Pid";

        updateComm.Parameters.Add("Pname", OleDbType.Char, 5);

        updateComm.Parameters.Add("Psex", OleDbType.Char, 1);

        updateComm.Parameters.Add("Paddress", OleDbType.Char, 10);

        updateComm.Parameters.Add("Pphone", OleDbType.Char, 18);

        updateComm.Parameters.Add("Pbirth", OleDbType.Date);

        updateComm.Parameters.Add("Pid", OleDbType.Char, 15);

        updateComm.Parameters[0].Value = pt.patientName;

        updateComm.Parameters[1].Value = pt.patientSex;

        updateComm.Parameters[2].Value = pt.patientPhone;

        updateComm.Parameters[3].Value = pt.patientAddress;

        updateComm.Parameters[4].Value = pt.patientBirthday;

        updateComm.Parameters[5].Value = pt.patientId;

        updateComm.ExecuteNonQuery();

        conn.Close();

    }

    public String GetbyIdIsIn(String id)//数病人个数

    {

        OleDbConnection conn = new OleDbConnection();

        conn.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=D:\\medical\\App\_Data\\HIS.mdb;User Id=admin;Password=";

        conn.Open();

        OleDbCommand comm = new OleDbCommand();

        comm.Connection = conn;

        comm.CommandText = "select count(\*) from patient where PatientId=@Pid";

        comm.Parameters.Add("Pid", OleDbType.Char, 10);

        comm.Parameters[0].Value = id;

        String n = comm.ExecuteScalar().ToString();

        conn.Close();

        return n;

    }

    public DataSet GetPatientByinfo(String sqlstr) //查询函数

    {

        OleDbConnection conn = new OleDbConnection();//建立一个连接

        conn.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=D:\\medical\\App\_Data\\HIS.mdb;User Id=admin;Password=";//设置连接字符串

        conn.Open();//连接建立

        OleDbDataAdapter adpter = new OleDbDataAdapter();//建立数据适配层 建立数据适配对象

        OleDbCommand comm = new OleDbCommand();// 命令对象

        comm.Connection = conn;//命令对象的连接等于conn

        comm.CommandText = sqlstr;//命令对象执行的指令为sqlstr

        adpter.SelectCommand = comm;//数据适配层的属性selectcommand直接设置为comm

        DataSet ds = new DataSet();//建立数据层，Dataset对象

        adpter.Fill(ds);//用数据适配对象执行Fill指令，将查询的数据放到Dataset对象ds里面

        conn.Close();//连接关闭

        return ds;//返回ds

    }

using System;

using System.Data;

using System.Configuration;

using System.Linq;

using System.Web;

using System.Web.Security;

using System.Web.UI;

using System.Web.UI.HtmlControls;

using System.Web.UI.WebControls;

using System.Web.UI.WebControls.WebParts;

using System.Xml.Linq;

using System.Data.OleDb;

/// <summary>

///prescriptionDA 的摘要说明

/// </summary>

public class prescriptionDA

{

    public prescriptionDA()

    {

        //

        //TODO: 在此处添加构造函数逻辑

        //

    }

   /\* public DataSet getallPrescription()//连接字符串 命令 执行 返回

    {

        OleDbConnection conn = new OleDbConnection();

        conn.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=D:\\medical\\App\_Data\\HIS.mdb;User Id=admin;Password=";

        conn.Open();

        OleDbCommand selectComm = new OleDbCommand();

        selectComm.Connection = conn;

        selectComm.CommandType = CommandType.Text;

        selectComm.CommandText = "select PrescriptiontID as 处方编号,PrescriptionItemID as 药品编号,PrescriptionItem as 药品,PatientId as 编号,DoctorId as 医生编号,PrescriptionTime as 处方时间 from Prescription,PrescriptionItem,Doctor,Patient where Prescription.PatientId=Patient.PatientId and Prescription.DoctorId = Doctor.DoctorId and Prescription.PrescriptionID = PrescriptionItem.PrescriptionID ";

        OleDbDataAdapter ad = new OleDbDataAdapter();

        ad.SelectCommand = selectComm;

        DataSet ds = new DataSet();

        ad.Fill(ds);

        conn.Close();

        return (ds);//数据访问层

    }\*/

    public void Insert(String str)//添加函数

    {

        OleDbConnection conn = new OleDbConnection();

        conn.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=D:\\medical\\App\_Data\\HIS.mdb;User Id=admin;Password=";

        conn.Open();

        OleDbDataAdapter ad = new OleDbDataAdapter();

        OleDbCommand comm = new OleDbCommand();

        comm.Connection = conn;

        comm.CommandText = str;

        ad.SelectCommand = comm;

        comm.ExecuteNonQuery();

        conn.Close();

    }

    public DataSet getprescriptionbyID(string id)//获取函数

    {

        OleDbConnection conn = new OleDbConnection();

        conn.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=D:\\medical\\App\_Data\\HIS.mdb;User Id=admin;Password=";

        conn.Open();

        OleDbCommand selectComm = new OleDbCommand();

        selectComm.Connection = conn;

        selectComm.CommandType = CommandType.Text;

        selectComm.CommandText = "select \* from prescription where PrescriptionId =@Preid";

        selectComm.Parameters.Add("Preid", OleDbType.Char, 10);

        selectComm.Parameters[0].Value = id;

        OleDbDataAdapter ad = new OleDbDataAdapter();

        ad.SelectCommand = selectComm;

        DataSet ds = new DataSet();

        ad.Fill(ds);

        conn.Close();

        return (ds);

    }

}

using System;

using System.Data;

using System.Configuration;

using System.Linq;

using System.Web;

using System.Web.Security;

using System.Web.UI;

using System.Web.UI.HtmlControls;

using System.Web.UI.WebControls;

using System.Web.UI.WebControls.WebParts;

using System.Xml.Linq;

using System.Data.OleDb;

/// <summary>

///prescriptionItemDA 的摘要说明

/// </summary>

public class prescriptionItemDA

{

    public prescriptionItemDA()

    {

        //

        //TODO: 在此处添加构造函数逻辑

        //

    }

    public void Insert(string str)

    {

        OleDbConnection conn = new OleDbConnection();//建立一个连接

        conn.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=D:\\medical\\App\_Data\\HIS.mdb;User Id=admin;Password=";

        conn.Open();

        OleDbDataAdapter ad = new OleDbDataAdapter();//建立数据适配层 建立数据适配对象

        OleDbCommand comm = new OleDbCommand();// 命令对象

        comm.Connection = conn;//命令对象的连接等于conn

        comm.CommandText = str;//命令对象执行的指令为sqlstr

        ad.SelectCommand = comm;//数据适配层的属性selectcommand直接设置为comm

        comm.ExecuteNonQuery();

        conn.Close();

    }

    public DataSet getprescriptionItembyID(string id)//获取函数

    {

        OleDbConnection conn = new OleDbConnection();//建立一个连接

        conn.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=D:\\medical\\App\_Data\\HIS.mdb;User Id=admin;Password=";

        conn.Open();//连接建立

        OleDbCommand selectComm = new OleDbCommand();

        selectComm.Connection = conn;//连接

        selectComm.CommandType = CommandType.Text;

        selectComm.CommandText = "select \* from prescriptionItem where PrescriptionItemId =@Preiid";

        selectComm.Parameters.Add("Preiid", OleDbType.Char, 10);

        selectComm.Parameters[0].Value = id;//多个参数 第0个参数为id

        OleDbDataAdapter ad = new OleDbDataAdapter();//建立数据适配层

        ad.SelectCommand = selectComm;//指令为selectComm

        DataSet ds = new DataSet();//建立数据层，Dataset对象

        ad.Fill(ds);//用数据适配对象执行Fill指令，将数据放到Dataset对象ds里面

        conn.Close();//连接关闭

        return (ds);//返回数据  数据访问层

    }

}

// File:    Doctor.cs

// Author:  jspho

// Created: 2019年5月26日 20:16:25

// Purpose: Definition of Class Doctor

using System;

public class Doctor

{

   public string doctorId;

   public string doctorName;

   public string departmentId;

   public string titleId;

   public DateTime doctorBirthday;

   public System.Collections.Generic.List<Prescription> prescription;

   /// <summary>

   /// Property for collection of Prescription

   /// </summary>

   /// <pdGenerated>Default opposite class collection property</pdGenerated>

   public System.Collections.Generic.List<Prescription> Prescription

   {

      get

      {

         if (prescription == null)

            prescription = new System.Collections.Generic.List<Prescription>();

         return prescription;

      }

      set

      {

         RemoveAllPrescription();

         if (value != null)

         {

            foreach (Prescription oPrescription in value)

               AddPrescription(oPrescription);

         }

      }

   }

   /// <summary>

   /// Add a new Prescription in the collection

   /// </summary>

   /// <pdGenerated>Default Add</pdGenerated>

   public void AddPrescription(Prescription newPrescription)

   {

      if (newPrescription == null)

         return;

      if (this.prescription == null)

         this.prescription = new System.Collections.Generic.List<Prescription>();

      if (!this.prescription.Contains(newPrescription))

      {

         this.prescription.Add(newPrescription);

         newPrescription.Doctor = this;

      }

   }

   /// <summary>

   /// Remove an existing Prescription from the collection

   /// </summary>

   /// <pdGenerated>Default Remove</pdGenerated>

   public void RemovePrescription(Prescription oldPrescription)

   {

      if (oldPrescription == null)

         return;

      if (this.prescription != null)

         if (this.prescription.Contains(oldPrescription))

         {

            this.prescription.Remove(oldPrescription);

            oldPrescription.Doctor = null;

         }

   }

   /// <summary>

   /// Remove all instances of Prescription from the collection

   /// </summary>

   /// <pdGenerated>Default removeAll</pdGenerated>

   public void RemoveAllPrescription()

   {

      if (prescription != null)

      {

         System.Collections.ArrayList tmpPrescription = new System.Collections.ArrayList();

         foreach (Prescription oldPrescription in prescription)

            tmpPrescription.Add(oldPrescription);

         prescription.Clear();

         foreach (Prescription oldPrescription in tmpPrescription)

            oldPrescription.Doctor = null;

         tmpPrescription.Clear();

      }

   }

}

// File:    Patient.cs

// Author:  jspho

// Created: 2019年5月26日 20:16:25

// Purpose: Definition of Class Patient

using System;

public class Patient

{

   public string patientId;

   public string patientName;

   public string patientSex;

   public string patientPhone;

   public string patientAddress;

   public DateTime patientBirthday;

   public System.Collections.Generic.List<Prescription> prescription;

   /// <summary>

   /// Property for collection of Prescription

   /// </summary>

   /// <pdGenerated>Default opposite class collection property</pdGenerated>

   public System.Collections.Generic.List<Prescription> Prescription

   {

      get

      {

         if (prescription == null)

            prescription = new System.Collections.Generic.List<Prescription>();

         return prescription;

      }

      set

      {

         RemoveAllPrescription();

         if (value != null)

         {

            foreach (Prescription oPrescription in value)

               AddPrescription(oPrescription);

         }

      }

   }

   /// <summary>

   /// Add a new Prescription in the collection

   /// </summary>

   /// <pdGenerated>Default Add</pdGenerated>

   public void AddPrescription(Prescription newPrescription)

   {

      if (newPrescription == null)

         return;

      if (this.prescription == null)

         this.prescription = new System.Collections.Generic.List<Prescription>();

      if (!this.prescription.Contains(newPrescription))

      {

         this.prescription.Add(newPrescription);

         newPrescription.Patient = this;

      }

   }

   /// <summary>

   /// Remove an existing Prescription from the collection

   /// </summary>

   /// <pdGenerated>Default Remove</pdGenerated>

   public void RemovePrescription(Prescription oldPrescription)

   {

      if (oldPrescription == null)

         return;

      if (this.prescription != null)

         if (this.prescription.Contains(oldPrescription))

         {

            this.prescription.Remove(oldPrescription);

            oldPrescription.Patient = null;

         }

   }

   /// <summary>

   /// Remove all instances of Prescription from the collection

   /// </summary>

   /// <pdGenerated>Default removeAll</pdGenerated>

   public void RemoveAllPrescription()

   {

      if (prescription != null)

      {

         System.Collections.ArrayList tmpPrescription = new System.Collections.ArrayList();

         foreach (Prescription oldPrescription in prescription)

            tmpPrescription.Add(oldPrescription);

         prescription.Clear();

         foreach (Prescription oldPrescription in tmpPrescription)

            oldPrescription.Patient = null;

         tmpPrescription.Clear();

      }

   }

}

// File:    Prescription.cs

// Author:  jspho

// Created: 2019年5月26日 20:16:25

// Purpose: Definition of Class Prescription

using System;

public class Prescription

{

   public string prescriptionID;

   public DateTime prescriptionTime;

   public System.Collections.Generic.List<PrescriptionItem> prescriptionItem;

   /// <summary>

   /// Property for collection of PrescriptionItem

   /// </summary>

   /// <pdGenerated>Default opposite class collection property</pdGenerated>

   public System.Collections.Generic.List<PrescriptionItem> PrescriptionItem

   {

      get

      {

         if (prescriptionItem == null)

            prescriptionItem = new System.Collections.Generic.List<PrescriptionItem>();

         return prescriptionItem;

      }

      set

      {

         RemoveAllPrescriptionItem();

         if (value != null)

         {

            foreach (PrescriptionItem oPrescriptionItem in value)

               AddPrescriptionItem(oPrescriptionItem);

         }

      }

   }

   /// <summary>

   /// Add a new PrescriptionItem in the collection

   /// </summary>

   /// <pdGenerated>Default Add</pdGenerated>

   public void AddPrescriptionItem(PrescriptionItem newPrescriptionItem)

   {

      if (newPrescriptionItem == null)

         return;

      if (this.prescriptionItem == null)

         this.prescriptionItem = new System.Collections.Generic.List<PrescriptionItem>();

      if (!this.prescriptionItem.Contains(newPrescriptionItem))

      {

         this.prescriptionItem.Add(newPrescriptionItem);

         newPrescriptionItem.Prescription = this;

      }

   }

   /// <summary>

   /// Remove an existing PrescriptionItem from the collection

   /// </summary>

   /// <pdGenerated>Default Remove</pdGenerated>

   public void RemovePrescriptionItem(PrescriptionItem oldPrescriptionItem)

   {

      if (oldPrescriptionItem == null)

         return;

      if (this.prescriptionItem != null)

         if (this.prescriptionItem.Contains(oldPrescriptionItem))

         {

            this.prescriptionItem.Remove(oldPrescriptionItem);

            oldPrescriptionItem.Prescription = null;

         }

   }

   /// <summary>

   /// Remove all instances of PrescriptionItem from the collection

   /// </summary>

   /// <pdGenerated>Default removeAll</pdGenerated>

   public void RemoveAllPrescriptionItem()

   {

      if (prescriptionItem != null)

      {

         System.Collections.ArrayList tmpPrescriptionItem = new System.Collections.ArrayList();

         foreach (PrescriptionItem oldPrescriptionItem in prescriptionItem)

            tmpPrescriptionItem.Add(oldPrescriptionItem);

         prescriptionItem.Clear();

         foreach (PrescriptionItem oldPrescriptionItem in tmpPrescriptionItem)

            oldPrescriptionItem.Prescription = null;

         tmpPrescriptionItem.Clear();

      }

   }

   public Patient patient;

   /// <summary>

   /// Property for Patient

   /// </summary>

   /// <pdGenerated>Default opposite class property</pdGenerated>

   public Patient Patient

   {

      get

      {

         return patient;

      }

      set

      {

         if (this.patient == null || !this.patient.Equals(value))

         {

            if (this.patient != null)

            {

               Patient oldPatient = this.patient;

               this.patient = null;

               oldPatient.RemovePrescription(this);

            }

            if (value != null)

            {

               this.patient = value;

               this.patient.AddPrescription(this);

            }

         }

      }

   }

   public Doctor doctor;

   /// <summary>

   /// Property for Doctor

   /// </summary>

   /// <pdGenerated>Default opposite class property</pdGenerated>

   public Doctor Doctor

   {

      get

      {

         return doctor;

      }

      set

      {

         if (this.doctor == null || !this.doctor.Equals(value))

         {

            if (this.doctor != null)

            {

               Doctor oldDoctor = this.doctor;

               this.doctor = null;

               oldDoctor.RemovePrescription(this);

            }

            if (value != null)

            {

               this.doctor = value;

               this.doctor.AddPrescription(this);

            }

         }

      }

   }

}

// File:    PrescriptionItem.cs

// Author:  jspho

// Created: 2019年5月26日 20:16:25

// Purpose: Definition of Class PrescriptionItem

using System;

public class PrescriptionItem

{

   public string prescriptionItemID;

   public string prescriptionItem;

   public Prescription prescription;

   /// <summary>

   /// Property for Prescription

   /// </summary>

   /// <pdGenerated>Default opposite class property</pdGenerated>

   public Prescription Prescription

   {

      get

      {

         return prescription;

      }

      set

      {

         if (this.prescription == null || !this.prescription.Equals(value))

         {

            if (this.prescription != null)

            {

               Prescription oldPrescription = this.prescription;

               this.prescription = null;

               oldPrescription.RemovePrescriptionItem(this);

            }

            if (value != null)

            {

               this.prescription = value;

               this.prescription.AddPrescriptionItem(this);

            }

         }

      }

   }

}

界面：

<%@ Page Language="C#" AutoEventWireup="true"  CodeFile="Default.aspx.cs" Inherits="\_Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

    <title>无标题页</title>

</head>

<body>

    <form id="form1" runat="server">

    <asp:GridView ID="GridView1" runat="server" DataKeyNames="编号"

        onrowdeleting="GridView1\_RowDeleting"

        onselectedindexchanging="GridView1\_SelectedIndexChanging">

        <Columns>

            <asp:CommandField HeaderText="操作" ShowDeleteButton="True" ShowHeader="True"

                ShowSelectButton="True" />

        </Columns>

    </asp:GridView>

    <p style="width: 432px; height: 18px">

        病人电话<asp:TextBox ID="TextBox5" runat="server" Width="262px"></asp:TextBox>

    </p>

    <p style="width: 293px; height: 18px">

        病人编号<asp:TextBox ID="TextBox1" runat="server" Width="210px"></asp:TextBox>

    </p>

    <p style="width: 293px; height: 18px">

        病人姓名<asp:TextBox ID="TextBox2" runat="server" Width="210px"></asp:TextBox>

    </p>

    <p style="width: 293px; height: 18px">

        病人性别<asp:TextBox ID="TextBox3" runat="server" Width="208px"></asp:TextBox>

    </p>

    <p style="width: 414px; height: 18px">

        病人住址<asp:TextBox ID="TextBox4" runat="server" Width="257px"></asp:TextBox>

    </p>

    <p style="width: 675px; height: 18px">

        病人出生日期<asp:TextBox ID="TextBox6" runat="server"></asp:TextBox>

        <asp:Button ID="Button1" runat="server" onclick="Button1\_Click"

            style="height: 29px" Text="提交" />

        <asp:Button ID="Button2" runat="server" onclick="Button2\_Click" Text="查询" />

        <asp:Button ID="Button3" runat="server" onclick="Button3\_Click" Text="开药" />

    </p>

</form>

</body>

</html>

(5) 实验小结

通过本次实验，我了解了三层架构的基本模式，了解了vs软件版本越高安全性越好同时编写也更为严格，写起来也更难，了解了class文件调用的安全保护机制，public，而写成友元是一个很不好的习惯。了解网站运行的机理，了解了htm就是一个界面，了解了网络的配置，和vs .APT的基本框架，了解了解决方案与项目的关系