第 5 章

HIS 系统的开发

随着网络技术的迅速发展，网络应用软件的开发日益普及。在众多的网络应用软件中，其中一类应用软件采用了最常用的客户端/服务器（Client/Server，C/S）的工作模式，在这个工作模式下，客户程序向服务器发出服务请求时，服务器程序做出应答，并为客户程序提供相应的服务，客户端和服务器端都需要单独开发各自的专门程序。

在 C/S 模式中，最常见的是浏览器/服务器（Browser/Server，B/S）模型，在该模型中，客户机上通过浏览器程序访问服务器，浏览器是客户端采用的统一程序，以此模式为基础的系统称为基于 Web 的系统，在 Internet 上，我们常用的百度、淘宝等应用程序采用的都是典型的 B/S 模型。

B/S 模型中，应用程序保存在 Web 应用服务器上，程序涉及的大量数据则保存在数据库服务器中，这样浏览器、应用程序服务器和数据库服务器构成了 Web 系统的三层结构。

本章的 HIS（Hospital Information System，医院信息系统）也是一个基于 Web 的医院信息管理系统，主要功能模块包括门诊管理、医生管理、住院管理、收费管理、开药管理、药品管理，在每个模块中可以对信息进行增加、修改、删除和查询操作。

虽然基于 Web 的应用系统应用领域涉及多个方面，但相关的技术和开发过程是相似的，这类系统中涉及的主要技术有 WebForm 应用程序的创建、网页之间数据的传递、数据库的设计和创建、数据库访问技术等，可以使用的开发工具有许多，本章使用的开发工具是 Visual Studio 2008 和 SQL Server 2005，程序设计语言是 C#。

通过本章 Web 系统的开发学习，使读者初步掌握基于 Web 的系统开发的基本过程，了解开发 Web 应用程序采用的技术，常见开发工具的使用。

5.1 数据库基础

数据库技术是计算机应用的一个重要方面。使用数据库技术的目的包括对信息进行有效的管理，以及对数据进行查询和处理。数据库技术中最基本的概念有数据库、数据库管理系统和应用程序。

5.1.1 基本概念

数据库（DataBase，DB）是指按某个特定的组织方式将数据以文件形式保存在存储介质上形成的文件，即数据库文件。

数据库管理系统（DataBase Management System，DBMS）是对数据库进行操纵和管理的大型软件。这类软件用统一的方式管理和维护数据库，接收并完成用户提出的访问数据

的各种请求，同时还具有开发应用程序的功能。

应用程序是指系统开发人员使用数据库管理系统、数据库资源和某个开发工具如 Visual Studio 开发的、应用于某一个实际问题的应用软件。例如，库房物资的管理系统、财务管理系统、医院信息管理系统等，本章最终开发的 HIS 就是一个应用程序。

1. 关系模型

在数据库文件中，不仅包含数据本身，也包含数据之间的联系即组织方式，不同的组织方式构成特定的数据模型。数据模型通常有层次模型、网状模型和关系模型，常用的是关系模型。

关系模型中，可以表达实体之间的先后关系（线性关系）。如果将每个实体从上到下排列成行，由于每个实体包括若干个数据，这些数据又构成了若干个列，这样，每个实体的数据之间的联系可以用二维表格的形式来形象地表示。

图 5-1 所示的用户表就是一个关系模型。

用户表

用户名	密码	性别	地址	电话
admin	admin	男	交大医学院	1300000000
张三	123456	男	交大计教中心	1300001111
李四	111111	女	交大医学院	1300002222
王五	222222	女	交大电信学院	1300004444

字段

记录

图 5-1　关系模型的组成

关系模型中常用到下面的一些术语。

（1）字段

二维表中，垂直方向上的每一列称为一个字段，每一个字段都有一个字段名。例如，用户表中有 5 个字段:“用户名”“密码”“性别”“地址”和“电话”。

（2）记录

二维表中从第二行起的每一行称为一条具体的记录，图中由 5 个字段 4 条记录组成。

（3）表结构

二维表中的第一行，是组成该表的各个字段的名称。在具体的数据库文件中，还应该详细地指出各个字段的类型、取值范围和宽度等，这些都称为字段的属性，一个表中所有字段的名称和属性的集合称为该表的结构。

这样，一个完整的二维表由表结构和记录两部分组成。在进行创建表和修改表的操作时，要先分清是对表结构进行的操作还是对记录进行的。

目前，计算机厂商推出的数据库管理系统几乎都是以关系模型为基础的，也称为关系型数据库管理系统。常用的关系型数据库管理系统软件有 Visual FoxPro、Access、SQL Server、Oracle、MySQL、DB2、 SYBASE、INFORMIX 等，本章使用的是 SQL Server。

2. 关系中的键

键（Key）也称为码（Code），在一个关系中，可以有几种不同的键。

在一个关系中可以用来唯一地标识每条记录的字段或字段的组合，称为候选键（Candidate Key），一个关系中，可以有多个候选键。

例如，在户籍信息表中，“身份证号”字段可以作为候选键；学生信息表中，“学号”字段可以作为候选键；在考生报名表中，“准考证号”和“身份证号”都可以作为候选键，该表中就有两个候选键。

如果一个关系中有多个候选键，可以从候选键中指定其中的一个作为主键（Primary Key）。设置主键后，表中的记录其主键的值既不能为空值，也不能有重复值，实现了关系的实体完整性约束规则。

5.1.2　在 SQL Server 中创建数据库和表

SQL Server 中的数据库是保存在盘上的文件，一个数据库中可以包含多个表。

1. 创建数据库

在 SQL Server 2005 中创建名为 hospital 的数据库，并在数据库中创建名为 userInfo 的表，操作过程如下。

（1）选择菜单“开始”→“所有程序”→Microsoft SQL Server 2005→SQL Server Management Studio Express，启动后的界面如图 5-2 所示。

图 5-2　启动 SQL Server

（2）单击“连接”按钮，打开 SSMS 窗口，如图 5-3 所示。

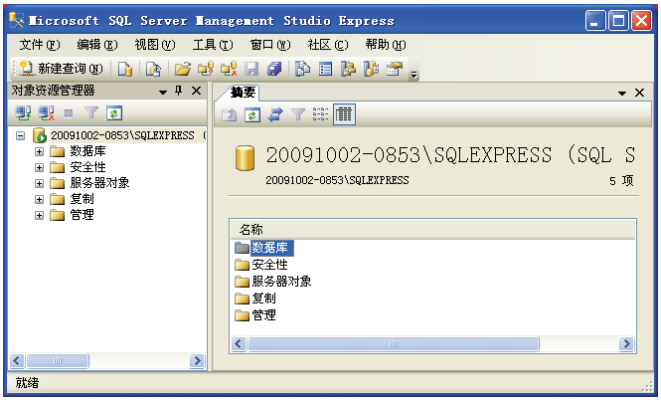

图 5-3　SSMS 窗口

（3）在左侧的“对象资源管理器”窗格中，右击“数据库”，在快捷菜单中单击“新建数据库”命令，显示“新建数据库”窗口，如图 5-4 所示。

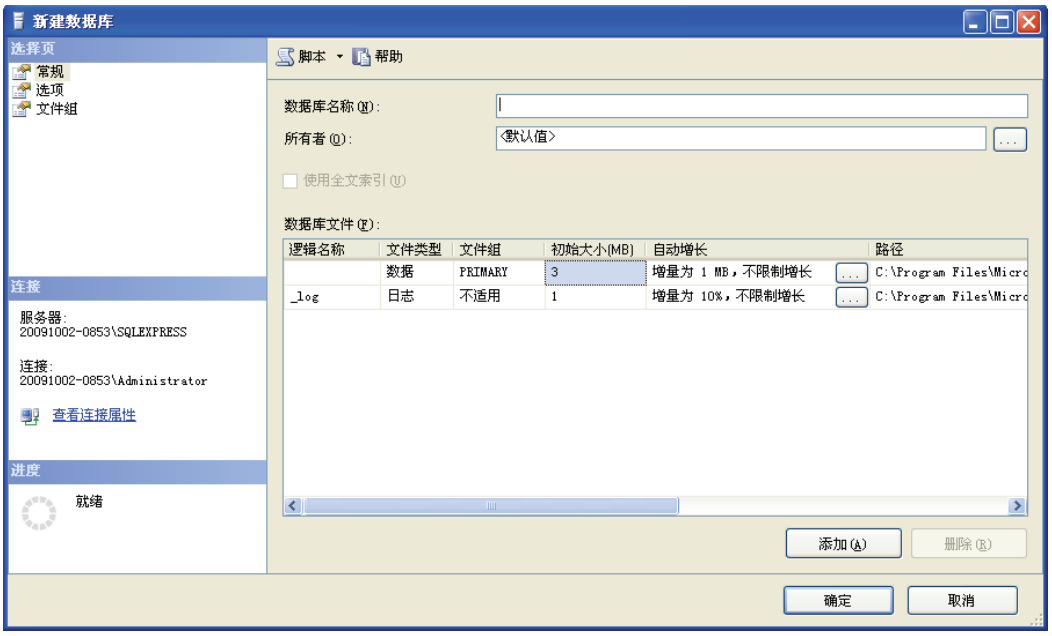

图 5-4 “新建数据库” 窗口

（4）在“数据库名称”文本框中输入要创建的数据库的名称“hospital”，然后单击“确定”按钮，创建完成。这时在“对象资源管理器”窗格中多了一个数据库“hospital”（见图 5-5）。同时在盘上多了两个文件，一个是数据库主文件 hospital.mdf，另一个是日志文件 hospital_log.ldf。

2. 在数据库中添加 userInfo 表

（1）在“对象资源管理器”窗格中展开数据库 hospital，右击其中的表对象，在快捷菜单中单击“新建表”命令，窗口中间显示设计表结构的窗格，如图 5-6 所示。

图 5-5 新建数据库

图 5-6 表结构设计窗格

（2）在窗格中输入 hospital 表各个字段的名称、类型、属性，该表结构的具体内容如表 5-1 所示（说明一列不需要输入）。

表 5-1　hospital 表各个字段的名称、类型、属性

列名（字段名）	数 据 类 型	允　许　空	说　　明
userId	int		主键
username	Nvarchar(50)	允许	
loginName	Nvarchar(50)	允许	
loginPwd	Nvarchar(20)	允许	
Sex	Nvarchar(2)	允许	
Address	Nvarchar(50)	允许	
Phone	Nvarchar(13)	允许	
SectionRoom	Nvarchar(10)	允许	科室

输入 userId 后，单击工具栏上的主键按钮，将该字段设置为主键，设计后的表结构如图 5-7 所示。

（3）表结构信息输入后，单击窗格右上角的“关闭”按钮，显示提示对话框，询问是否保存所做的更改，单击“是”按钮，显示“选择名称”对话框，如图 5-8 所示。

图 5-7　表结构信息

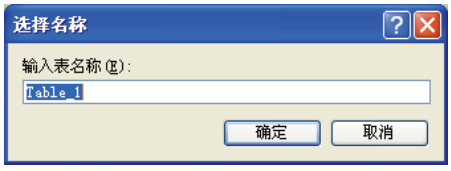

图 5-8　“选择名称”对话框

（4）向对话框中输入表名称“userInfo”，然后单击“确定”按钮，关闭设计窗格。

（5）在“对象资源管理器”窗格中右击 userInfo 表，在快捷菜单中单击“打开表”命令，窗口中间显示编辑表记录的窗格。

（6）向窗格输入表的记录，输入后显示内容如图 5-9 所示，单击工具栏上的保存按钮。

userId	userName	loginName	loginPwd	Sex	Address	Phone	SectionRoom
1	admin	admin	admin	男	交大医学院	13000000000	外科
2	张三	张三	123456	男	交大计教中心	13000001111	外科
3	李四	李四	111111	女	交大医学院	13000002222	外科
4	王五	王五	222222	女	交大电信学院	13000004444	内科

图 5-9　userInfo 表的内容

3. 备份数据库

（1）在“对象资源管理器”窗格中右击 hospital 数据库，在快捷菜单中单击“任务”→“备份”命令，打开“备份数据库”窗口，如图 5-10 所示。

（2）选择数据库，在对话框中，从“数据库”下拉列表框中选择要备份的数据库，“备份类型”下拉列表框中选择“完整”。

（3）指定备份位置，单击“删除”按钮，删除已经存在的目录，然后单击“添加”按钮，打开“选择备份目标”对话框，在对话框中指定备份的位置，单击“确定”按钮返回

到“备份数据库”对话框，再次单击“确定”按钮对话框完成备份，备份文件扩展名为 bak。

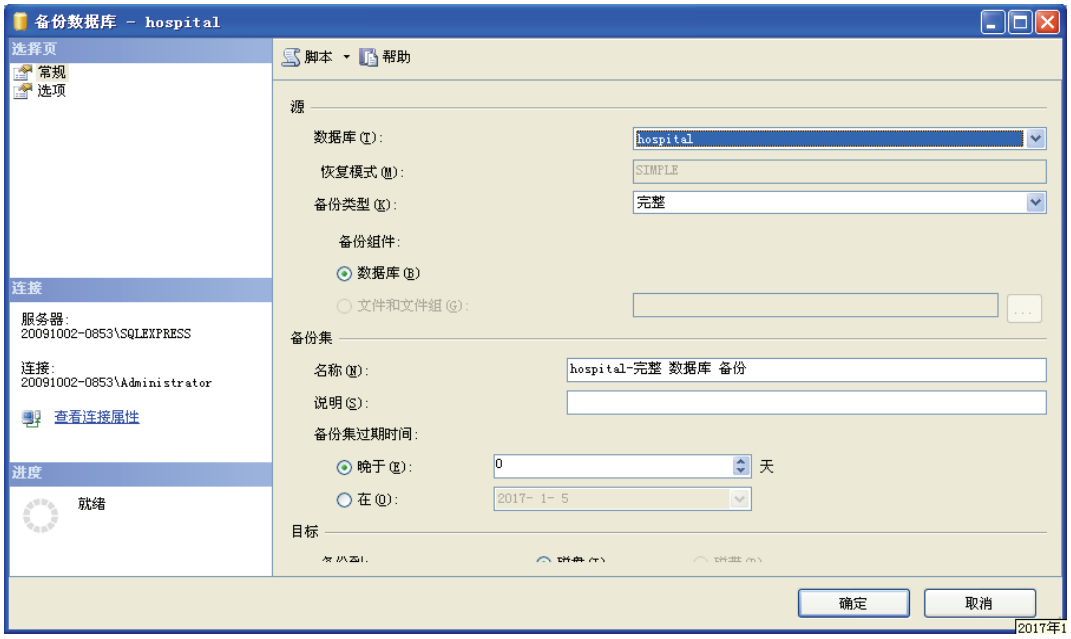

图 5-10 “备份数据库”窗口

4. 恢复数据库

（1）在“对象资源管理器”窗格中右击 hospital 数据库，在快捷菜单中选择“任务”→“还原”→“数据库”，打开“还原数据库”窗口，如图 5-11 所示。

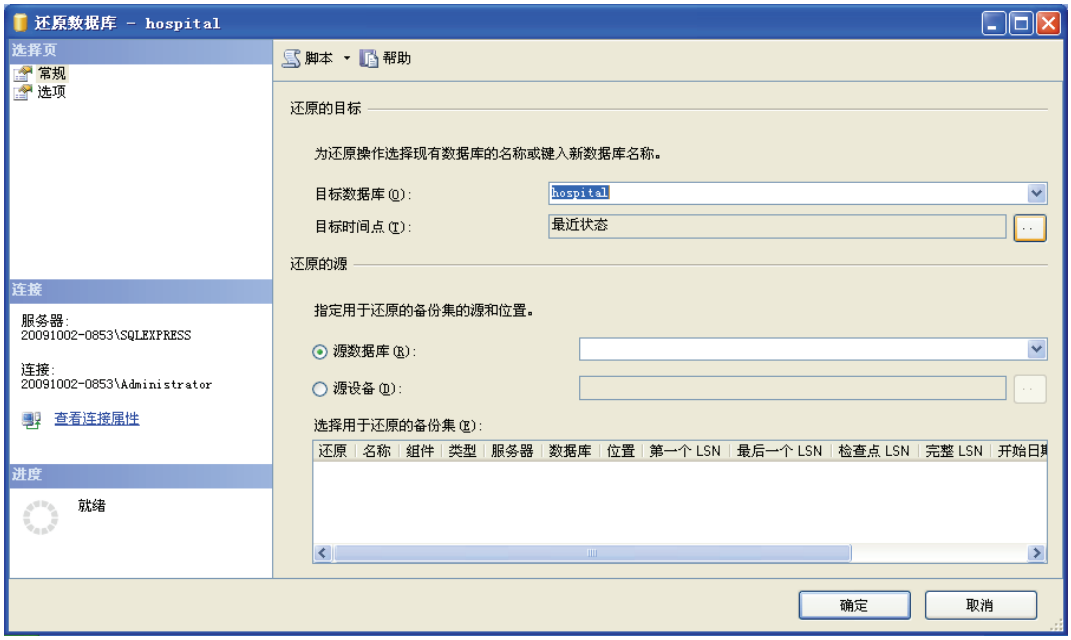

图 5-11 “还原数据库”窗口

（2）在图 5-11 中，选择“源设备”，单击右侧的“…”按钮，弹出“指定备份”界面。

（3）在“指定备份”界面中选择“备份”设置，单击“添加”按钮，在打开的界面中选择以前备份的数据库备份文件，然后连续单击“确定”按钮，依次关闭每个界面，完成数据库的还原。

5.1.3　SQL 命令的使用

SQL（Structured Query Language，结构化查询语言）是用于关系数据库的标准语言，该语言使用方便、功能丰富、语言简洁易学，因而很快得到推广和应用，目前大多数数据库产品都支持 SQL。

SQL 的主要功能包括数据定义、数据查询、数据操纵和数据控制 4 个方面，每个功能都由具体的命令实现。

对表中记录进行的基本操作有增加、删除、修改和查询，对应的命令分别是 insert、delete、update 和 select。

在 SQL Server 2005 中使用和练习 SQL 命令，单击工具栏中的“新建查询”按钮，在窗口中间显示查询窗格，可以在窗格上方输入 SQL 命令，单击执行按钮后，在窗格的下方显示命令的执行结果。

1. 查询记录

查询是 SQL 中非常重要的操作，它能够完成多种查询任务，如查询满足条件的记录、查询时进行统计计算、同时对多表查询、对记录排序等。当结合函数进行查询时，可完成更多的诸如计算的功能。

SQL 的所有查询都是利用 SELECT 命令实现的，该命令的完整格式比较复杂，其中最为常用的是下面的简化格式：

```
select…from …where
```

其中：select 之后指出要输出的字段，from 之后指出查询的数据源，where 之后则指出查询的条件。

操作 1：显示 userInfo 表中所有记录的每个字段的值。

在查询窗格上方输入如下的命令：

```
select * from userInfo
```

单击执行按钮后，在窗格的下方显示命令的执行结果，如图 5-12 所示。

	userId	userName	loginName	loginPwd	Sex	Address	Phone	SectionRoom
1	1	admin	admin	admin	男	交大医学院	13000000000	外科
2	2	张三	张三	123456	男	交大计数中心	13000001111	外科
3	3	李四	李四	111111	女	交大医学院	13000002222	外科
4	4	王五	王五	222222	女	交大电信学院	13000004444	内科

图 5-12　查询命令的执行结果

在指定查询的输出字段时，如果要显示表中所有的字段，并且字段的顺序与表中顺序一致时，可以用“*”代替所有的字段名，否则要指出具体的字段名。

操作 2：显示 userInfo 表中每条记录的 userName、Address 和 Phone 三个字段，命令如下：

```
select userName,Address,Phone from userInfo
```

操作 3：查询 userInfo 表中 userId 为 1 的记录。

```
select * from userInfo where userId=1
```

操作 4：查询 userInfo 表中 userName 为 admin 的记录。

```
select * from userInfo where userName='admin'
```

由于 userName 字段的类型为字符型，其字段的值 admin 在 SQL 命令中要放在一对引号中，SQL Server 要求使用一对单引号。

输入 SQL 命令时，要注意：

- 命令中的关键字不区分大小写；
- 命令中出现的字符型字段的值要放在一对单引号中；
- 命令中出现的数值型字段的值可以放在一对单引号中，也可以不用单引号；
- 命令中出现的逗号、引号、空格之类的符号一定要在英文状态下输入。

2. 添加记录

在 SQL 中，添加记录使用 INSERT 命令，添加的记录被加到表的末尾。其格式如下：

```
insert into  <表名>  [(<字段名 1>[, <字段名 2>[, …]])]
                              values (<表达式 1>[, <表达式 2>[, …]])
```

使用该命令时，命令中的字段名与 values 值的个数应相同，并且类型一一对应。如果 values 值的个数、顺序和类型与定义表结构时各个字段一致，则可以省略字段名。

操作 5：将以下的数据添加到 userInfo 表中。

```
5,'guest','guest','33333','男','交大理学院','13000005555','皮肤科'
```

该记录的数据完整，且顺序与表中字段顺序一致，可以省略字段名，使用的 SQL 命令如下：

```
insert into userInfo values(5,'guest','guest','33333','男','交大理学院',
'13000005555','皮肤科')
```

操作 6：将以下的数据添加到 userInfo 表中。

```
6,'guest1','guest1','999999','男','交大理学院'
```

该记录的数据不全，必须在命令中指出字段名，使用的 SQL 命令如下：

```
insert into userInfo (userId,userName,loginName,loginPwd,sex,address)
values(6,'guest1','guest1','999999','男','交大理学院')
```

对表中记录进行增、删、改之后，可以使用 select 命令验证执行的效果。

3. 修改记录

修改已输入记录的字段的值，可以使用 update 命令，其格式如下：

```
update  <表名>  SET <字段名 1>=<表达式 1>,[<字段名 2>=<表达式 2>…]
 [where  <条件>]
```

该命令的功能是按给定的表达式的值，修改满足条件的记录的各字段值，其中 where <条件>是表示满足条件的记录。命令中如果没有使用 where 指定条件时，表中所有的记录都被修改。

操作 7：修改 userInfo 表中 userId 字段值为 6 的记录，将其 address 字段的值改为'药学院'，命令如下：

```
update userInfo set address='药学院' where userId=6
```

4. 删除记录

删除记录使用 delete 命令，其格式如下：

```
delete  from  <表名>  [where  <条件>]
```

该命令的功能是将满足条件的记录删除，省略 where 子句时，表中所有的记录都将被删除。

操作 8：将 userInfo 表中 userId 为 5 的记录删除，SQL 命令如下：

```
delete from userInfo where userId=5
```

操作 9：删除 userInfo 表中所有男性用户的记录，SQL 命令如下：

```
delete from userInfo where sex='男'
```

5.1.4　数据库的设计

在使用具体的 DBMS 创建数据库之前，应根据用户的需求对数据库应用系统进行分析和研究，然后再按照一定的原则设计数据库中的具体内容。

1. 数据库设计的一般方法

数据库的设计一般要经过分析建立数据库的目的、确定数据库中的表、确定表中的字段、确定主关键字以及确定表间的关系等过程，如图 5-13 所示。

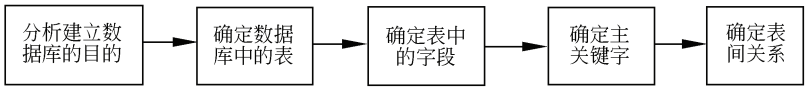

图 5-13　数据库的设计步骤

（1）分析建立数据库的目的

在分析过程中，应与数据库的最终用户进行交流，了解用户的需求和现行工作的处理过程，共同讨论使用数据库应该解决的问题和完成的任务，同时尽量收集与当前处理有关的各种表格。

在需求分析中，要从以下三个方面进行。

- 信息需求：定义数据库应用系统应该提供的所有信息。
- 处理需求：表示对数据需要完成什么样的处理及处理的方式，也就是系统中数据处理的操作，应注意操作执行的场合、操作进行的频率和对数据的影响等。
- 安全性和完整性需求：例如实体完整性约束。

本章设计 hospital 数据库的目的是对医院信息的组织和管理，主要包括用户信息管理、药品信息管理、开药信息管理、缴费信息管理等。

（2）确定数据库中的表

一个数据库中要处理的数据很多，不可能将所有的数据放在一个表中，需要分析将收集到的信息使用几个表进行保存。

应保证每个表中只包含关于一个主题的信息，这样，每个主题的信息可以独立地维护。例如，分别将用户信息、药品信息、开药信息、缴费信息放在不同的表中，这样对某一类信息的修改不会影响到其他的信息。

通过将不同的信息分散在不同的表中，可以使数据的组织和维护变得简单，同时也可以保证在此基础上建立的应用程序具有较高的独立性。

根据上面的原则，确定在 hospital 数据库中创建以下若干张表，分别是 caseInfo（病历）表、DrugInfo（药品）表、houseRegist（病房）表、prescribe（处方）表、register（挂号）表等。

（3）确定表中的字段

确定每个表中包括的字段应遵循下面的原则：

- 保证一个表中的每个字段都是围绕着一个主题，例如 userId、userName、loginName、loginPwd 等字段都是与用户信息有关的字段。
- 避免在表和表之间出现重复的字段，在表中除了为建立表间关系而保留的，尽量避免在多个表之中同时存在重复的字段，这样做的目的一是为了减少数据的冗余，同时也是防止因插入、删除和更新数据时造成的数据不一致。
- 表中的字段所表示的数据应该是最原始和最基本的，不应包括可以推导或计算出的数据，也不应包括可以由基本数据组合得到的字段，例如看病总费用字段可以通过各项收费（挂号费、治疗费、检查费等）之和得到，这些数据不要设计在表中，可以使用查询的方法进行计算。

（4）确定主键

在一个表中确定主键，目的之一是保证实体的完整性，即主键的值不允许是空值或重复值，另一个目的是在不同的表之间建立联系。

例如在 userInfo 表中 userId 定义为主键，caseInfo 表中的主键是 caseId。

（5）确定表间的关系

表间的关系要根据具体的问题来确定，不能随意建立，例如 caseInfo（病历）表和 houseRegist（病房）表之间可以通过同名字段 registerNo（病历号）建立联系。

如果确认设计符合要求，就可以在 DBMS 中创建数据库和各张表了。

2. hospital 数据库中各个表的结构

（1）caseInfo（病历）表（表 5-2）

表 5-2　CaseInfo 表

字　段　名	数 据 类 型	说　　明
caseId	整数	主键
registerNo	整数	病历号
Mainsuit	字符	主诉
medicalHistory	字符	用药史
inspectItem	字符	检查
suggest	字符	建议

（2）DrugInfo（药品）表（表 5-3）

表 5-3　DrugInfo 表

字　段　名	数 据 类 型	说　　明
drugId	整数	主键
drugName	变长字符（100）	药品名称
drugType	变长字符（10）	药品类型
Spec	变长字符（50）	规格
price	货币	单价
manufacturer	变长字符（100）	制造商
manuDate	日期时间	生产日期
AddDate	日期时间	进库日期

（3）houseRegist（病房）表（表 5-4）

表 5-4　houseRegist 表

字　段　名	数 据 类 型	说　　明
registId	整数	主键
registerNo	整数	病历号
sickroomId	整数	病房 Id
bedNo	变长字符（10）	病床号
expenses	货币	费用
status	比特	状态

（4）prescribe（处方）表（表 5-5）

表 5-5　prescribe 表

字　段　名	数 据 类 型	说　　明
prescribeId	整数	处方 Id,主键
drugId	整数	药品 Id
registerNo	整数	病历号
number	整数	数量
doctor	变长字符（50）	开药医生
prescribeDate	日期	开药日期
takeState	比特	取药状态
payStatus	比特	付费状态

（5）register（挂号）表（表 5-6）

表 5-6 register 表

字 段 名	数 据 类 型	说 明
registerNo	整数	病历号 Id,主键
Name	变长字符（50）	姓名
sectionRoom	变长字符（15）	诊室
cardNo	变长字符（50）	卡号
Address	变长字符（25）	地址
Phone	变长字符（13）	电话
age	int	年龄
sex	字符（2）	性别
isMarrage	比特	婚否
isVisit	比特	是否初诊
registerDate	日期时间	挂号日期
treatmentDate	日期时间	治疗日期

（6）sectionRoom（诊室）（表 5-7）

表 5-7 sectionRoom 表

字 段 名	数 据 类 型	说 明
SectionRoomId	整数	诊室 Id,主键
SectionRoom	变长字符（20）	诊室

（7）sickroomId（病房）（表 5-8）

表 5-8 sickroomId 表

字 段 名	数 据 类 型	说 明
SickRoomId	整数	病房 Id,主键
sickRoomNo	变长字符（20）	病房编号
Type	变长字符（20）	病房类型
Bednum	整数	床位数
Price	货币	收费

创建了以上各张表后，可以向表中输入若干条具体的记录。

5.2 创建简单的 Web 应用程序

本章使用的开发工具是 Visual Studio 2008（以下简称 VS2008）。使用 VS2008 可以开发多种应用程序，常用的有控制台应用程序、窗体应用程序和 ASP.NET Web 应用程序等，本章创建的是 ASP.NET Web 应用程序，使用的语言是 C#。

本节通过几个实例说明 VS2008 创建程序的过程、常用控件的使用等，每个例题由功能要求、操作过程和编程归纳三部分组成，个别题目还有编程分析，其中编程归纳对本题目中涉及的语法问题和关键点进行归纳。

5.2.1　Web 应用程序的创建过程

例 5-1　创建 Web 应用程序。

1. 功能要求

在浏览器中显示信息“欢迎使用医院信息管理系统”。

2. 操作过程

（1）在 E 盘下创建文件夹“his 系统的开发\C#例题”，保存后面创建的例题。

（2）选择菜单“开始”→“所有程序”→Microsoft Visual Studio 2008→Microsoft Visual Studio 2008，启动后的窗口如图 5-14 所示。

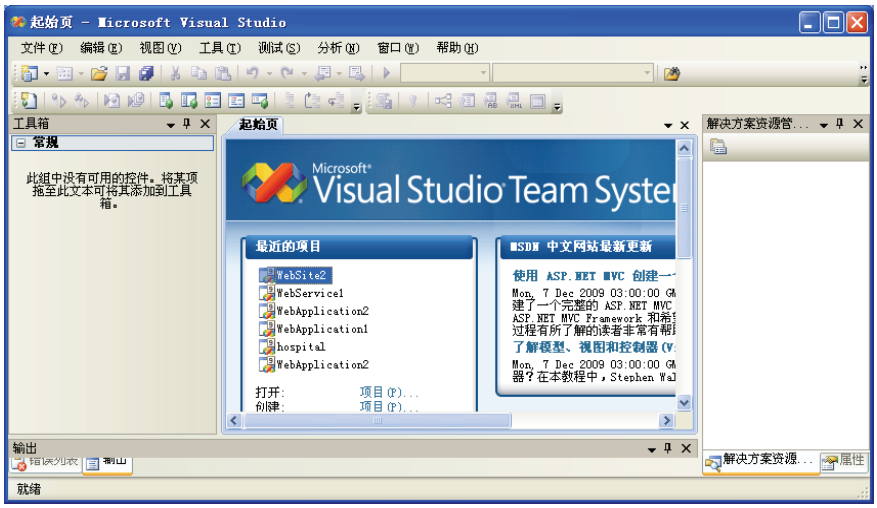

图 5-14　VS2008 启动窗口

（3）在 VS2008 窗口中，执行“文件”→“新建”→“网站”命令，显示“新建网站”对话框，如图 5-15 所示。

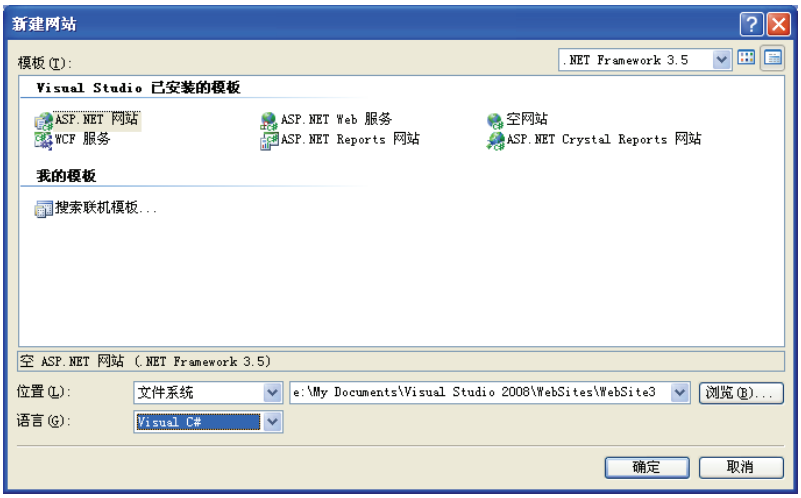

图 5-15　“新建网站”对话框

（4）在对话框中，选择“ASP.NET 网站”，“语言”选择 Visual C#，在位置文本框中输入“E:\his 系统的开发\C#例题\example1”，其中的 example1 是项目名称，然后单击“确定”按钮。这时，显示程序设计界面，如图 5-16 所示。

同时，“E:\his 系统的开发\C#例题”文件夹中新增名为 example1 的文件夹，本程序中生成的所有文件都保存在该文件夹中。

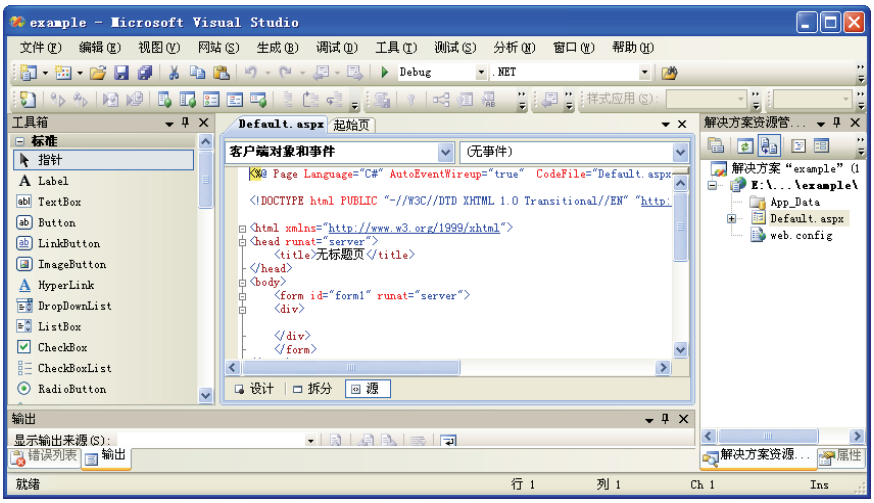

图 5-16　程序设计界面

设计界面窗口中，左侧是控件工具箱窗格，显示出可以使用的各个控件，右侧是“解决方案资源管理器”任务窗格，中间是设计区。

（5）双击 Default.aspx，打开设计视图，窗口中间显示 Default.aspx 选项卡。

该选项卡下方有三个视图按钮，“设计”“拆分”和“源”。其中“设计”视图以可视化方式显示设计的网页即设计的效果，“源”视图显示该网页的 HTML 代码，“拆分”视图则将选项卡分为上下两部分分别显示 HTML 源代码和网页效果。

（6）在工具箱中选择 Label（标签）控件，将其拖动到 div 区域，然后在窗口右侧的“属性”窗格的 Text 属性中输入“欢迎使用医院信息管理系统”，在 Size 属性中选择××-Large，此时选项卡的内容如图 5-17 所示。

图 5-17　设计好的选项卡

（7）在“解决方案资源管理器”任务窗格中右击 Default.aspx，在快捷菜单中选择“设为起始页”。

（8）按组合键 Ctrl+F5，在浏览器中显示程序的运行结果，如图 5-18 所示。

图 5-18　程序运行结果

3. 编程归纳

（1）本程序完整说明了 Web 应用程序的设计和运行过程。

（2）使用 VS 开发的 ASP.NET 网站通常包含以下主要文件：

- 一个或多个扩展名为.aspx 网页文件；
- 一个或多个 web.config 配置文件；
- App_Code 代码共享目录：存放应用程序中所有网页都可以使用的共享文件；
- App_Data 目录：存放数据库文件。

（3）本程序中使用了一个控件 Label（标签）A Label，该控件的功能是为控件和窗体的其他组成部分提供标识。

5.2.2　创建欢迎页面

例 5-2　创建 Web 应用程序，程序界面如图 5-19 所示。

1. 功能要求

向文本框中输入用户名，单击“确定”按钮后，如果输入了用户名，则弹出消息框，显示“欢迎使用本系统”的信息，如果没有输入用户名，则消息框中显示“请先输入用户名”的信息。

图 5-19　程序界面

2. 操作过程

（1）启动 VS2008。

（2）执行“文件”→“新建”→“网站”命令，创建名为 example2 的项目。

（3）双击 Default.aspx 打开设计视图。

（4）向网页中依次添加 Label、TextBox 和 Button 三个控件，其名称分别是 Label1、TextBox1 和 Button1，然后将 Label1 的 Text 属性设置为“请输入用户名”，将 Button1 的 Text 属性设置为“确定”。

（5）输入事件代码，双击 Button1，窗口中出现 Default.aspx.cs 选项卡，向该选项卡中输入代码，输入后的选项卡如图 5-20 所示。

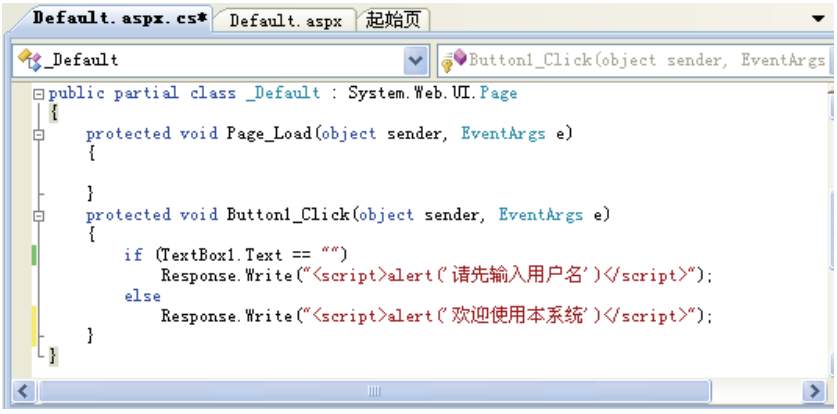

图 5-20　程序代码

（6）在“解决方案资源管理器”任务窗格中右击 Default.aspx，在快捷菜单中选择“设为起始页”。

（7）输入用户名和不输入用户名的两次运行结果如图 5-21 所示。

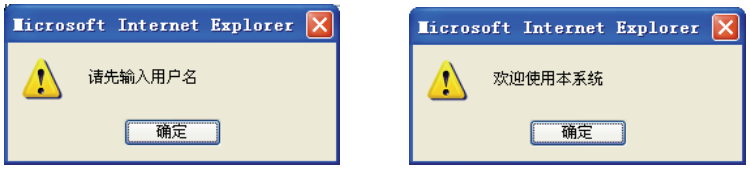

图 5-21　两次不同的运行结果

3. 编程归纳

（1）本题中用到了两个新的控件，文本框和命令按钮：

- TextBox（文本框）控件 abl TextBox：显示文本信息。与 Label 控件不同的是，文本框中的文本可以被编辑，标签中的文本不能被编辑。
- Button（按钮）控件 ab Button：用户可以单击按钮控件触发程序动作。

（2）和例 5-1 一样，本程序中的网页使用了默认的名称 default，系统创建了与本网页有关的两个文件，分别是 Default.aspx.cs 和 Default.aspx。Default.aspx 描述的是页面的内容，Default.aspx.cs 则是代码部分，这就是 ASP.NET 中页面文件与代码文件的分离。

由于例 5-1 中没有编写事件代码，所以只有一个页面文件 Default.aspx。

（3）通常程序执行时，单击窗口中的命令按钮可以完成某个动作，对按钮来说是发生了一次单击事件。不同的控件还有不同的其他事件，在编程时要为每个控件的事件编写不同的代码，本程序是在下面的一对花括号中输入事件代码，花括号之前的代码名称

Button1_Click 表示该段代码是控件 Button1 的 Click 事件的代码。

```
protected void Button1_Click(object sender, EventArgs e)
    {
}
```

（4）关于代码中下面语句的解释：

```
Response.Write("<script>alert('hello')</script>");
```

语句中的 Response.Write()称为一个方法，这是 ASP.NET 中的一个方法，它的作用是向客户端输出信息，输出的信息被放在括号里。

语句中的 alert()表示要在客户端弹出一个消息框，消息框中要显示的内容'hello'放在括号中。

5.2.3　创建收集信息的页面

标签、文本框和命令按钮是程序中最常用的三种控件，下面通过几个例题介绍其他控件的使用。

例 5-3　创建 Web 应用程序用于收集信息，其页面如图 5-22 所示。

图 5-22　信息采集页面

1. 功能要求

向页面中输入各项的内容，输入后单击“提交”按钮，在下方的“基本信息汇总”文本框中显示输入的信息，单击“清除”按钮时，清除所有文本框中的内容并取消复选框中的已选各项。

2. 设计分析

本程序中输入学号、姓名、年龄使用的控件是文本框，输入性别使用的控件是 RadioButtonList(单选按钮列表)，输入政治面貌使用的控件是 DropDownList(下拉列表框)，输入经常阅读的图书使用的控件是 CheckBoxList（复选列表框），最后汇总信息显示在文本框控件中。

程序中还要为“提交”按钮和“清除”按钮编写单击事件代码。

3. 操作过程

（1）启动 VS2008，创建名为 example3 的项目。

（2）向 Default.aspx 网页中从上到下依次添加以下控件：

① 添加 Label 控件 Label1，其 Text 属性设置为“学生基本信息采集”，Size 属性设置为××-Large。

② 添加 Label 和 TextBox 控件，其名称分别是 Label2、TextBox1，Label2 的 Text 属性设置为“学号”。

③ 添加 Label 和 TextBox 控件，其名称分别是 Label3、TextBox2，Label2 的 Text 属性设置为“姓名”。

④ 添加 Label 和 RadioButtonList 控件，其名称分别是 Label4、RadioButtonList1，Label4 的 Text 属性设置为“性别”。在添加 RadioButtonList1 控件时，屏幕上会自动显示 RadioButtonList 任务，如图 5-23 所示。单击其中的“编辑项”，显示“ListItem 集合编辑器”对话框，如图 5-24 所示。

图 5-23 RadioButtonList 任务

图 5-24 “ListItem 集合编辑器”对话框

在对话框中单击“添加”按钮添加各个选项（成员），在右侧的 Text 框中输入“男”，同时在 Selected 中选择 True，然后添加第 2 项“女”。最后将 RadioButtonList1 的 RepeatDirection 属性设置为 Horizontal（水平方向）。

⑤ 添加 Label 和 TextBox 控件，其名称分别是 Label5、TextBox3，Label5 的 Text 属性设置为“年龄”。

⑥ 添加 Label 和 DropDownList 控件，其名称分别是 Label6、DropDownList1，Label6 的 Text 属性设置为“政治面貌”。

在添加 DropDownList1 控件时，屏幕上会自动显示 DropDownList 任务。单击其中的“编辑项”，显示“ListItem 集合编辑器”对话框，在对话框中分别添加“党员”“团员”和“其他”各个选项，其中“团员”选项的 Selected 选择 True。

⑦ 添加 Label 和 CheckBoxList 控件，其名称分别是 Label7、CheckBoxList1，Label7 的 Text 属性设置为“经常阅读的图书”，向 CheckBoxList1 添加 4 个选项，分别是“政治”“外语”“文学”和“其他”。

⑧ 添加 Button 控件 Button1 和 Button2，Button1 的 Text 属性设置为“提交”，Button2 的 Text 属性设置为“清除”。

⑨ 添加 Label 和 TextBox 控件，其名称分别是 Label8、TextBox4，Label8 的 Text 属性设置为“基本信息汇总”。

（3）输入“提交”按钮的事件代码，双击 Button1，窗口中出现 Default.aspx.cs 选项卡，

向该选项卡中 Button1_Click()中输入如下的代码：

```
protected void Buttonl_Click(object sender, EventArgs e)
{
    string str = "";
    int n = 0;
    str += TextBox1.Text+" ";
    str += TextBox2.Text+" ";
    str += RadioButtonList1.SelectedItem.Text + " ";
    str += TextBox3.Text+" ";
    str += DropDownList1.SelectedValue + " ";
    for (int i = 0; i < CheckBoxList1.Items.Count; i++)
       if (CheckBoxList1.Items[i].Selected)
       {
          str += CheckBoxList1.Items[i].Text + ",";
          n++;
       }
    if (n == 0)
       str += "没有选择喜爱的图书";
    TextBox4.Text = str;
}
```

（4）输入“清除”按钮的事件代码，双击 Button2，向该选项卡中 Button2_Click()中输入如下的代码：

```
protected void Button2_Click(object sender, EventArgs e)
{
    TextBox1.Text = "";
    TextBox2.Text = "";
    TextBox3.Text = "";
    TextBox4.Text = "";
    for (int i = 0; i < CheckBoxList1.Items.Count; i++)
       CheckBoxList1.Items[i].Selected = false;
}
```

（5）将 Default.aspx 设置为“设为起始页”，该程序的两次运行结果如图 5-25 所示。

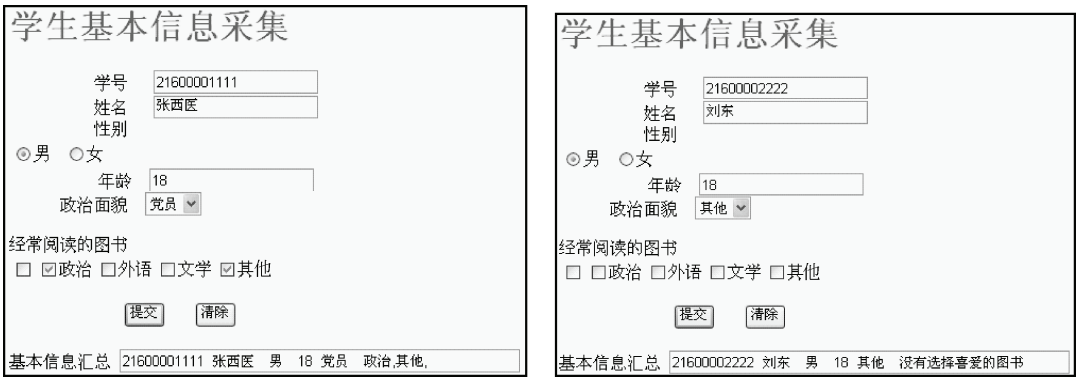

图 5-25　程序的两次运行结果

4. 编程归纳

（1）本题分别为两个按钮编写了事件代码 Button1_Click()和 Button2_Click()。

（2）RadioButtonList（单选按钮列表）控件。如果在多个互斥的选项中必须选中一项并且也只能选中一项，可以使用一组 RadioButton 控件或使用一个 RadioButtonList 控件。本题使用的是后者，控件中的各个选项可以添加控件后在弹出的“ListItem 集合编辑器”中输入，被选中选项可以用控件名.SelectedItem.Text 格式来表示，例如本例中的 RadioButtonList1.SelectedItem.Text。

（3）DropDownList（下拉列表框）控件。该控件用来实现下拉式列表，可以从列表框中选择一项，控件中的各个选项也可以在“ListItem 集合编辑器”中输入，被选中选项同样可以使用控件名.SelectedItem. Value 格式来表示，例如本例中的 DropDownList1.SelectedValue。

（4）CheckBoxList（复选列表框）控件。该控件可以包含多个复选框，每一个复选框都可以分别进行选中或取消选中，各个选项同样可以在“ListItem 集合编辑器”中输入，每个选项在控件中的顺序通过下标（或称为索引）实现，下标从 0 开始。每一个选项的选中状态要分别进行判断，本例中使用了下面的循环语句来判断每个选项的选中状态：

```
for (int i = 0; i < CheckBoxList1.Items.Count; i++)
```

其中的 CheckBoxList1.Items.Count 表示控件中复选框的数目，每一项的选中状态通过属性的值为 true 或 false 来判断：

```
CheckBoxList1.Items[i].Selected
```

在 Button2_Click()的代码中，通过下面的语句对每个选项进行取消选中的操作：

```
for (int i = 0; i < CheckBoxList1.Items.Count; i++)
CheckBoxList1.Items[i].Selected = false
```

5.2.4 网页之间的跳转和数据的传递

例 5-4 创建含有两个网页的网站。

1. 功能要求

第一个网页 default.aspx 显示的内容与例 5-3 相似，只是没有最后一行的标签和文本框（图 5-26），第二个网页 StuInfo.aspx 显示内容如图 5-27 所示。程序运行时，向第一个网页中输入学生信息，单击“提交”按钮后，跳转到第二个网页显示来自第一个网页的学生信息，单击“返回上一页”按钮时，可以返回到第一个网页，从而实现网页之间的跳转和数据的传递。

2. 操作过程

（1）启动 VS2008，创建名为 example4 的项目。

（2）向 Default.aspx 网页中依次添加图 5-26 中的各个控件。

（3）输入“提交”按钮的事件代码，向 Default.aspx.cs 选项卡中的 Button1_Click()中输入如下的代码：

```
protected void Button1_Click(object sender, EventArgs e)
{
    int n = 0;
    string str;
    string id = TextBox1.Text;
    string name = TextBox2.Text;
    string xb = RadioButtonList1.SelectedItem.Text;
    string age = TextBox3.Text;
    string zzmm = DropDownList1.SelectedValue;
    string book="";
    for (int i = 0; i < CheckBoxList1.Items.Count; i++)
        if (CheckBoxList1.Items[i].Selected)
        {
            book+= CheckBoxList1.Items[i].Text + ",";
            n++;
        }
    if (n == 0)
        book= "没有选择喜爱的图书";
    str = "id=" + id + "&name=" + name + "&xb=" + xb + "&age=" + age + "&zzmm="
    + zzmm + "&book=" + book;
    Response.Redirect("StuInfo.aspx?"+str);
}
```

图 5-26　第一张网页

图 5-27　第二张网页

（4）输入“清除”按钮的事件代码，向 Button2_Click()中输入如下的代码：

```
protected void Button2_Click(object sender, EventArgs e)
{
    TextBox1.Text = "";
    TextBox2.Text = "";
    TextBox3.Text = "";
    for (int i = 0; i < CheckBoxList1.Items.Count; i++)
        CheckBoxList1.Items[i].Selected = false;
}
```

（5）将 Default.aspx 设置为“设为起始页”。

（6）创建第二个网页，在“解决方案资源管理器”窗格中右击项目 example4，在弹出的快捷菜单中执行“添加新项…”命令，显示“添加新项”对话框，如图 5-28 所示。

图 5-28 “添加新项”对话框

（7）在对话框中，在已安装的模板中选择“Web 窗体”，在“名称”文本框中输入网页名称“StuInfo”，然后单击“确定”按钮。

（8）向 StuInfo.aspx 网页中依次添加图 5-27 中的各个控件。

（9）输入页面加载事件代码，向 StuInfo.aspx.cs 选项卡中的 Page_Load()中输入如下的代码：

```
protected void Page_Load(object sender, EventArgs e)
{
    TextBox1.Text = Request.QueryString["Id"];
    TextBox2.Text = Request.QueryString["name"];
    TextBox3.Text = Request.QueryString["xb"];
    TextBox4.Text = Request.QueryString["age"];
    TextBox5.Text = Request.QueryString["zzmm"];
    TextBox6.Text = Request.QueryString["book"];
}
```

（10）输入“返回上一页”按钮的事件代码，向 StuInfo.aspx.cs 选项卡中的 Button1_Click()中输入如下的代码：

```
protected void Button1_Click(object sender, EventArgs e)
{
    Response.Redirect("default.aspx");
}
```

（11）运行程序，在 default.aspx 中输入学生的信息，单击“提交”按钮后，该信息被

传递到 stuinfo.aspx 中并且在窗口中显示出来，如图 5-29 所示，实现了数据从一个网页传递到另一个网页。

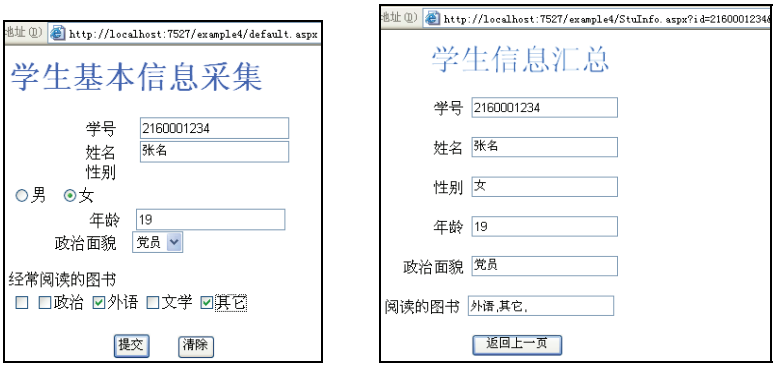

图 5-29　程序的运行结果

3. 编程归纳

（1）在 ASP.NET 中，可以通过方法 Response.Redirect()网页之间的跳转，使用时在括号中添加目标 URL 地址即可。

例如 StuInfo.aspx.cs 程序中的下列语句实现返回 default.aspx 网页：

```
Response.Redirect("default.aspx");
```

（2）在使用 Response.Redirect()方法跳转网页时，还可以向目标网页传递数据，方法是在参数中的 URL 后面加上查询字符串，查询字符串就是要传递的数据，它由多个属性构成，每个属性的格式如下：

```
属性名＝属性值
例如，属性 name 的值为“张三”，则该属性表示为：name="张三"
如果有多个属性，属性之间用“&”连接起来，例如本题中的查询字符串：
str = "id=" + id + "&name=" + name + "&xb=" + xb + "&age=" + age + "&zzmm="
+ zzmm + "&book=" + book;
   Response.Redirect("StuInfo.aspx?"+str);
```

再如下面的例子：

```
String URL="StuInfo.aspx?name=张三&id=2160001";
```

该查询字符串中有两个属性（变量）name 和 id，其属性值分别是“张三”和“2160001”。

（3）目标网页在接收数据时，可以通过 Request.QueryString[属性名]来获得属性的值，例如 Request.QueryString[name]的值为“张三”，Request.QueryString[id]的值为“2160001”。

5.2.5　使用表格进行页面布局

上述几个例题向页面中添加若干个控件，没有涉及这些控件在页面上的布局情况，也就是组成一个页面的若干个板块的显示位置和显示方式，对于页面结构不太复杂的情况，可以采用表格的方式布局页面。方法是将整个页面不同内容规划到一个或多个表格中。

在 VS2008 中，可以通过执行“表”→“插入表”命令在页面上添加表格，然后向表的各个单元格中添加不同的控件等内容。

例 5-5 设计登录页面，见图 5-30。

1. 功能要求

- 使用非空数据验证控件（RequiredFieldValidator）保证用户名和密码不为空，如果没有输入用户名或密码就单击“登录”按钮，在文本框右侧会显示提示信息。
- 使用表格进行页面布局。
- 根据输入的内容在消息框中显示不同的内容，假定正确的用户名为 student，密码为 123456，输入用户名不正确时显示“该用户名不正确”，用户名正确但密码不对时显示“密码不正确”，两者都正确时显示“欢迎使用”。

2. 设计分析

该页面中有 4 行内容，第 2 行和第 3 行各有 3 个部分，其中“登录”按钮和密码所在行之间有一空行。设计布局时，先插入一个 5 行 3 列的表格，然后将第 1 行和最后一行的 3 个单元格分别合并为一个单元格，再向每个单元格输入内容，如图 5-31 所示。

图 5-30 登录界面

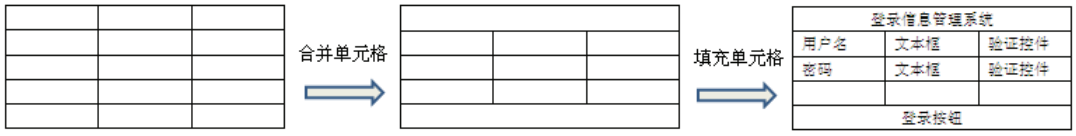

图 5-31 使用表格布局过程

3. 创建过程

（1）启动 VS2008，创建名为 example5 的项目。

（2）在 Default.aspx 网页中，执行“表”→“插入表”命令，显示“插入表格”对话框，如图 5-32 所示。

（3）在对话框的“行数”框中输入 5，“列数”框中输入 3，在“指定宽度”中选择“百分比”，然后输入“50”，表示表格占页面宽度的 50%，输入后单击“确定”按钮关闭对话框。

（4）选中第一行的三个单元格，右击，在弹出的快捷菜单中执行“修改”→“合并单元格”命令，将这一行的三个单元格合并为一个，然后向单元格中输入“登录信息管理系统”，加粗，执行“格式”→“两端对齐”→“居中”命令，将该文本设置为居中对齐。

（5）在第二行中，向第一个单元格输入“用户名”，第二个单元格插入文本框 TextBox1，第三个单元格插入非空数据验证控件 RequiredFieldValidator1。在属性窗格中，单击 ControlToValidate 属性，在下拉列表框中选择 TextBox1 表示该控件对 TextBox1 的内容进行

验证，将 ErrorMessage 设置为“用户名不能为空”。

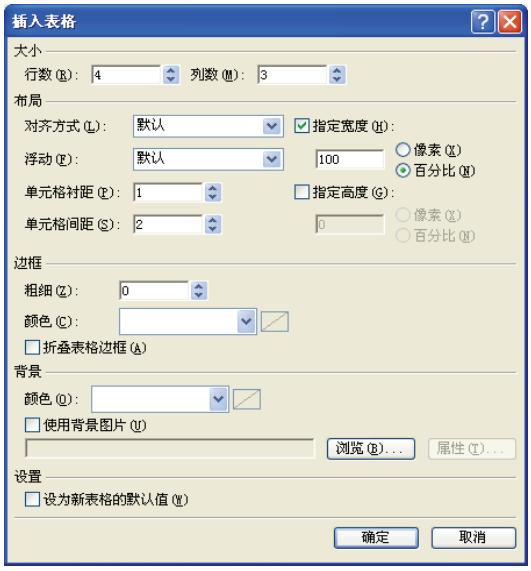

图 5-32　“插入表格”对话框

（6）在第三行中，向第一个单元格输入“密码”，第二个单元格插入文本框 TextBox2，将文本框的 TextMode 属性设置为 Password，向第三个单元格插入非空数据验证控件 RequiredFieldValidator2，并将其 ControlToValidate 属性设置为“TextBox2”，将 ErrorMessage 设置为“密码不能为空”。

（7）合并第五行中的三个单元格，然后插入命令按钮 Button1，并将其 Text 属性设置为“登录”，居中对齐，设计后的页面如图 5-33 所示。

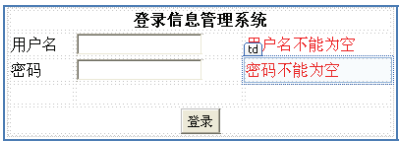

图 5-33　设计好的页面

（8）输入“登录”按钮的事件代码，向 default.aspx.cs 选项卡中的 Button1_Click()中输入如下的代码：

```
protected void Button1_Click(object sender, EventArgs e)
{
    string username = TextBox1.Text;
    string pwd = TextBox2.Text;
    if (username!="student")
        Response.Write("<script>alert('用户名不正确')</script>");
    else
        if(pwd!="123456")
            Response.Write("<script>alert('密码不正确')</script>");
```

```
        else
            Response.Write("<script>alert('欢迎使用')</script>");
}
```

（9）将 Default.aspx 设置为“设为起始页”，程序的某次运行结果如图 5-34 所示。

图 5-34　程序的运行结果

4. 编程归纳

（1）使用表格进行页面的布局并不是唯一的布局技术，还可以使用 CSS+DIV 的方法。

（2）程序中使用了非空数据验证控件（RequiredFieldValidator），除此之外，C#.NET 中还有其他验证类型的控件。例如数据类型和数据比较验证（CompareValidator）、数据范围验证（RangeValidator）等。验证控件验证的是文本框中的 Text 属性值和 ListBox、RadioButtonList 等 SelectedItem.Value 的值。

当网页有提交发生时，首先启动这些验证控件的验证功能，只有当页面上所有验证都通过时，网页才会被提交到服务器进行处理。

（3）文本框 TextBox2 是用来输入密码的，所以要将其 TextMode 属性设置为“Password”，这样在输入密码时不会显示密码的具体内容，而是用“●●●●●●”来代替。

（4）本程序中输入的用户名“student”和密码“123456”是固定不变的，如果要更改用户名和密码，就要修改源程序，而且只能处理一个用户。在学习 SQL Server 时本书曾将每个用户的信息保存在 hospital 数据库的 UserInfo 表中，所以应将程序修改为在 UserInfo 表中查找输入的用户名和密码是否存在，这就要用到数据库访问技术。

5.3　数据库访问技术

将数据库中的数据显示到网页上、将页面上的信息保存到数据库中、用网页上的数据更新数据库中的数据或通过网页删除数据库中表的记录，这就是对表中记录进行的查询、增加、修改和删除操作，这些操作都需要通过网页访问到数据库。首先是在网页中连接数据库，然后通过在网页代码中添加 SQL 命令完成。

5.3.1　C#中访问数据库的一般过程

对数据库进行访问时，首先要连接到数据库，然后将数据库中的数据保存到称为数据集（DataSet）的对象，最后在 DataSet 中对数据进行操作。访问过程中要用到 SqlConnection、SqlDataAdapter 和 DataSet 对象。

这三个对象的作用如下：

- SqlConnection：连接到数据库。

- DataSet：保存来自数据库的数据。
- SqlDataAdapter：称为数据适配器，是 SqlConnection 和 DataSet 之间的桥梁，作用是将数据填充到数据集 DataSet 中，也可以将数据集中的数据更新到数据库中。

要连接到数据库，可以使用代码的方法，也可以在 VS 中使用数据连接向导的方法。其中，使用程序代码连接数据库并将数据提取到数据集 DataSet 的过程如下：

（1）设置连接字符串，该字符串指定要连接数据库的服务器名、数据库名、用户名和口令。例如，下面的语句定义了一个连接字符串（两行合起来是一个完整的字符串）：

```
string strcon = "Data Source=A227\\SQLEXPRESS;Initial Catalog=hospital;
User ID=sa;Password=123456";
```

语句中：strcon 是连接字符串的名称，字符串中包含 4 个部分的内容，要连接的数据库所在的服务器名、数据库名、用户名和口令，各部分之间使用分号隔开。

其中的服务器名指出要连接的 SQL Server 实例名或网络地址，如果要连接到本地主机的服务器，可以用 localhost 或句点“.”即可。

用户名和口令是数据库用户账号的用户名和口令。

（2）建立 SqlConnection 的连接对象，其格式如下：

```
SqlConnection 连接对象名 = new SqlConnection(连接字符串);
```

例如，下面的语句创建的连接对象为 con：

```
SqlConnection con = new SqlConnection(strcon);
```

（3）建立 SqlCommand 的命令对象，方法是调用连接对象的 CreateCommand()方法，格式是：

```
SqlCommand 命令对象名＝连接对象名. CreateCommand()
```

例如，下面的语句创建了名为 com 的命令对象：

```
SqlCommand com＝con. CreateCommand();
```

该操作也可以由下面两条语句完成：

```
SqlCommand com = new SqlCommand();
com.Connection = con;
```

（4）设置 SqlCommand 对象的 CommandText 属性（即命令文本）：

CommandText 属性就是相应的 SQL 语句，例如下面的语句指定 com 对象的 CommandText 属性，提取 userInfo 表中的所有记录：

```
com.CommandText = "select * from userInfo";
```

（5）创建 SqlDataAdapter 数据适配器对象，使用格式为：

```
SqlDataAdapter 适配器对象名 = new SqlDataAdapter(命令对象)
```

例如，下面的语句创建了名为 sda 的适配器对象：

```
SqlDataAdapter sda = new SqlDataAdapter(com);
```

（6）创建数据库 DataSet 对象，下面语句创建了名为 ds 的对象：

```
DataSet ds = new DataSet();
```

（7）打开 SqlConnection 连接，方法是：

```
连接对象名.open();
```

（8）对数据库的操作，可以是将数据填充到数据集，也可以是将数据集更新到数据库，前者使用适配器的 Fill()方法，后者使用 Update()方法，也可以添加和删除记录。

例如，下面的语句将查询的记录内容填充到数据集 ds 中：

```
sda.Fill(ds);
```

（9）关闭对数据库的连接，方法是：

```
连接对象名.close();
```

为了使用与 SqlServer 数据库操作有关的类例如 SqlConnection，SqlCommand，SqlDateAdapter，要在程序开始加上下面的语句：

```
using System.Data.SqlClient;
```

将以上各部分的语句按顺序排列起来，就是访问 SQL 数据库的基本程序段。虽然步骤比较多，但操作过程基本固定。通过一个完整例题的学习，在访问其他数据库时，只需要修改连接字符（修改要访问的数据库的信息，第（4）步）和 SQL 的查询命令（第（6）步）即可。

以上的操作过程称为 ADO.NET 数据库访问技术。访问成功后，可以使用 Repeater 控件、GridView 控件、FormView 控件、DataGrid 控件等将 DataSet 中的数据展示在网页上。

5.3.2 使用 Repeater 控件显示记录

例 5-6 设计页面，程序运行后，使用 Repeater 控件显示数据集的记录。

1. 功能要求

将 hospital 数据库中 UserInfo 表的所有记录显示在页面上。

2. 操作过程

（1）启动 VS2008，创建名为 example6 的项目。

（2）将已创建好的数据库文件 hospital.mdf 和对应的日志文件 hospital_log.ldf 复制到 example6\App_Data 文件夹中。

（3）向 default.aspx 页面中添加控件 Repeater1，该控件在工具箱“数据”分类中。

（4）在源视图下，向 default.aspx 页面中 和 之间添加如下的代码：

```
<table style="width:60%;" border="1" bgcolor="#00FF00">
    <tr>
    <td>用户 ID</td>
```

```
        <td>用户名</td>
        <td>性别</td>
        <td>地址</td>
        <td>电话</td>
        <td>科室</td>
    </tr>
    <asp:Repeater ID="Repeater1" runat="server">
    <ItemTemplate>
    <tr>
        <td><%#Eval("userId") %> </td>
        <td><%#Eval("userName") %></td>
        <td><%#Eval("Sex") %></td>
        <td><%#Eval("Address") %></td>
        <td><%#Eval("Phnoe") %></td>
        <td><%#Eval("SectionRoom") %></td>
    </tr>
    </ItemTemplate>
    </asp:Repeater>
   </table>
```

切换到设计视图，repeater 控件显示成如图 5-35 所示形式。

用户ID	用户名	性别	地址	电话	科室
数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定
数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定
数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定
数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定
数据绑定	数据绑定	数据绑定	数据绑定	数据绑定	数据绑定

图 5-35　设置 ItemTemplate 模板后的 Repeater

（5）在设计视图下，双击 default.aspx 页面，在程序上方添加如下的语句：

```
using System.Data.SqlClient;
```

然后向 Page_Load()过程中输入如下的代码：

```
using System.Data.SqlClient;
public partial class_Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        string strcon = "Data Source=.\\SQLEXPRESS;AttachDbFilename=";
        strcon +="E:\\his 系统的开发\\C#例题\\example6\\App_Data\\hospital.
         mdf: ";
        strcon +="Integrated Security=True;Connect Timeout=30;User Instance=
        True";
        string sql = "select * from userInfo";
```

```
        SqlConnection con = new SqlConnection(strcon);
        if (con.State != ConnectionState.Open)
        {
            con.Open();
        }
        SqlCommand com = new SqlCommand();
        com.Connection = con;
        com.CommandType = CommandType.Text;
        com.CommandText = sql;
        DataSet ds = new DataSet();
        SqlDataAdapter sda = new SqlDataAdapter(com);
        sda.Fill(ds);
        con.Close();
        Repeater1.DataSource = ds;    //指定数据源
        Repeater1.DataBind();         //绑定数据源
    }
```

（6）该网页的运行结果如图 5-36 所示，不断地改变浏览器窗口的大小，该表格的宽度始终是窗口的 60%。

用户ID	用户名	性别	地址	电话	科室
1	admin	男	交大医学院	13000000000	外科
2	张三	男	交大计教中心	13000001111	外科
3	李四	女	交大医学院	13000002222	外科
4	王五	女	交大电信学院	13000004444	内科
6	guest	男	药学院		

图 5-36　运行结果

3. 编程归纳

（1）在编写访问 SQL 数据库的程序时，在程序开始写上如下的语句：

```
using System.Data.SqlClient;
```

（2）在 Default.aspx 中编写代码。使用 Repeater 控件时，其所有代码必须在 Web 页面的源视图中手工添加，这要使用到 HTML 标记语言。

本题中使用表格的形式显示数据源中的数据，添加的第一行是表格标记命令：

```
<table style="width:60%;" border="1" bgcolor="#00FF00">
```

其中的“width:60%”表示表格占窗口宽度的 60%，border="1"是表格的框线宽度，bgcolor="#00FF00"是表格单元格内的填充颜色，#00FF00 表示按 RGB（红绿蓝）模式并用 6 位十六进制设计，红、绿、蓝各用两位。本表格的线条颜色为蓝色。

接下来的一对<tr>和</tr>为表格设置显示各个字段名称，例如<td>用户名</td>等。从表格的第 2 行开始用来显示各条记录，每行显示一条，对于每一行中各个单元格的内容设置，这里使用了 Repeater 控件中的 ItemTemplate 即条目模板。在该模板中，可以定义各个单元格中与字段有关的数据，使用数据绑定表达式完成，每个表达式必须放在“<%#”和

“%>”之间，例如第 1 个单元格显示 userId 的值，该单元格设置如下：

```
<td><%#Eval("userId") %> </td>
```

其中的 Eval("userId")用来计算绑定的表达式，并将结果格式化为字符串，其他各单元格以此类推。

（3）本例题中的 Default.aspx.cs 的代码不是 Button 的 Click 事件代码，而是 Page_Load 即页面加载事件的代码。

（4）关于数据库连接串的设计，由于该字符串比较长，直接书写不太方便，可以通过数据源配置向导进行查询，方法如下。

① 在 default.aspx 的设计视图中，单击 Repeater1 控件右上角的符号，然后单击右侧的下拉箭头，在弹出的列表框中选择新建数据源（图 5-37）打开“数据源配置向导”对话框（图 5-38）。

图 5-37　Repeater 任务

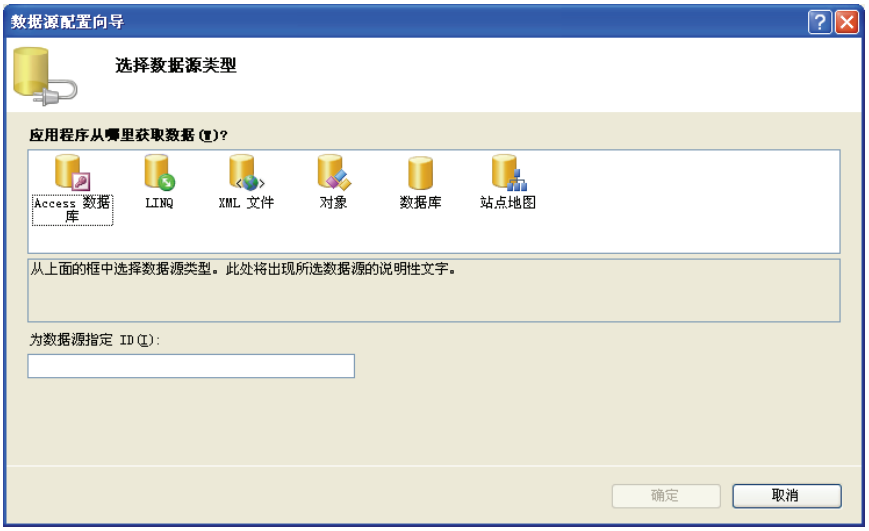

图 5-38　“数据源配置向导”对话框

② 在“应用程序从哪里获取数据”列表框中选择“数据库”，然后单击“确定”按钮，打开“配置数据源”对话框，在对话框中单击“新建连接”按钮，显示“添加连接”对话框，如图 5-39 所示。

③ 在“添加连接”对话框中，单击“浏览”按钮，打开“选择 SQL Server 数据库文件”对话框，如图 5-40 所示。在对话框中选择复制到 example6\App_Data 文件夹中的 hospital.mdf 文件，选择后单击“打开”按钮，返回到“添加连接”对话框。

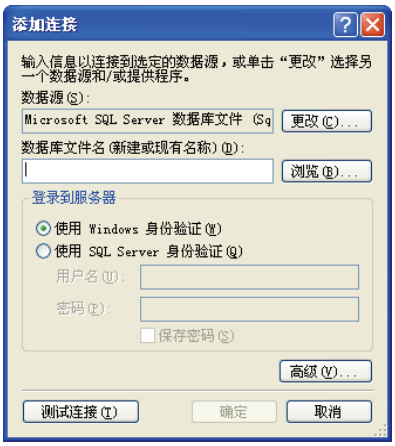

图 5-39 “添加连接”对话框

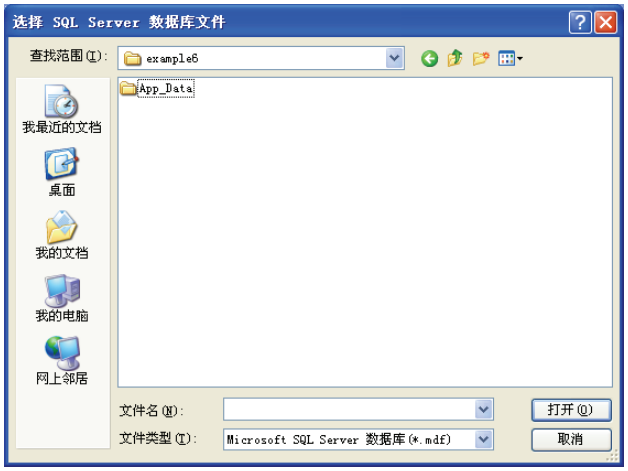

图 5-40 选择数据库文件

④ 单击“测试连接”按钮，如果弹出消息框显示“测试连接成功”表示连接正确，关闭消息框。

⑤ 在“添加连接”对话框中单击“确定”按钮，返回到“配置数据源”对话框，对话框下方显示了连接字符串，如图 5-41 所示。可以将该字符串粘贴到程序代码中，粘贴后将字符串中所有的“\”都改为“\\”。

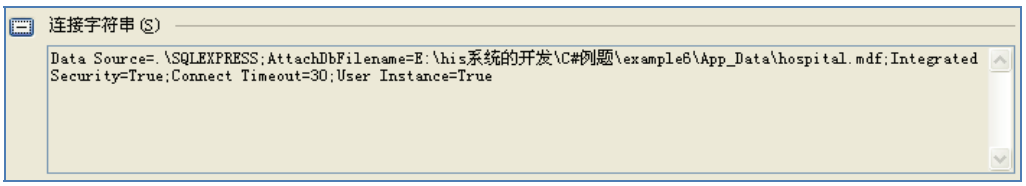

图 5-41 数据库连接字符串

5.3.3 非空数据验证控件的使用

例 5-7 重新编写例 5-5 的程序。

1. 功能要求

- 设计的页面界面不变，包括使用非空数据验证控件（RequiredFieldValidator）保证用户名和密码不为空，并且用表格进行页面布局。见图 5-30。
- 对于用户输入的用户名和密码在 hospital.mdf 数据库的 UserInfo 表中进行查询，输入的用户名不存在时显示“该用户名不存在”，用户名存在但密码不对时显示“密码不正确”，两者都正确时进入新页面，页面显示“×××你好，欢迎使用本系统”，其中的×××是用户名。

2. 创建过程

（1）创建名为 example7 的项目。

（2）将已创建好的数据库文件 hospital.mdf 和对应的日志文件 hospital_log.ldf 复制到 example7\App_Data 文件夹中。

（3）在 Default.aspx 网页中，插入 5 行 3 列的表格，表格占页面宽度的 50%。

（4）将表格第 1 行的 3 个单元格合并为一个，然后向单元格中输入“登录信息管理系统”，并设置加粗和居中对齐。

（5）在第 2 行中，向第 1 个单元格输入“用户名”，第 2 个单元格插入文本框 TextBox1，第 3 个单元格插入非空数据验证控件 RequiredFieldValidator1，将其 ControlToValidate 属性设置为 TextBox1，将 ErrorMessage 设置为“用户名不能为空”。

（6）在第 3 行中，向第 1 个单元格输入“密码”，第 2 个单元格插入文本框 TextBox2，将其 TextMode 属性设置为 Password，向第 3 个单元格插入非空数据验证控件 RequiredFieldValidator2，将其 ControlToValidate 属性设置为 TextBox2，将 ErrorMessage 设置为“密码不能为空”。

（7）合并第 5 行中的 3 个单元格，然后插入命令按钮 Button1，并将其 Text 属性设计为“登录”，居中对齐，设计后的页面如图 5-33 所示。

（8）输入“登录”按钮的事件代码，在 default.aspx.cs 选项卡中，在程序开始输入如下的语句：

```
using System.Data.SqlClient;
```

然后在 Button1_Click()中输入如下的代码：

```
protected void Button1_Click(object sender, EventArgs e)
    {
        string strcon = "Data Source=.\\SQLEXPRESS;AttachDbFilename=";
        strcon += "E:\\his 系统的开发\\C#例题\\example7\\ App_Data\\ hospital. mdf;";
        strcon+="Integrated Security=True;Connect Timeout=30;User Instance=
         True";
        string useName = TextBox1.Text;
        string userPwd = TextBox2.Text;
        string sql = "select * from userInfo where loginName='" + useName+"'" ;
        SqlConnection con = new SqlConnection(strcon);
        if (con.State != ConnectionState.Open)
        {
            con.Open();
        }
        SqlCommand com = new SqlCommand();
        DataSet ds = new DataSet();
        com.Connection = con;
        com.CommandType = CommandType.Text;
        com.CommandText = sql;
        SqlDataAdapter sda = new SqlDataAdapter(com);
        sda.Fill(ds);
        con.Close();
```

```
        if (ds.Tables[0].Rows.Count == 0)
        {
            Response.Write("<script>alert('该用户名不存在')</script>");
        }
        else
        {
            sql += " and loginPwd='" + userPwd + "'";
            SqlConnection con1 = new SqlConnection(strcon);
            if (con1.State != ConnectionState.Open)
            {
                con1.Open();
            }
            SqlCommand com1 = new SqlCommand();
            DataSet ds1 = new DataSet();
            com1.Connection = con;
            com1.CommandType = CommandType.Text;
            com1.CommandText = sql;
            SqlDataAdapter sda1 = new SqlDataAdapter(com1);
            sda1.Fill(ds1);
            con1.Close();
            if (ds1.Tables[0].Rows.Count == 0)
            {
                Response.Write("<script>alert('密码不正确')</script>");
            }
            else
            {  Session["username"] = useName;
               Response.Redirect("WebForm1.aspx");
            }
        }
    }
```

（9）设计一个名为 webform1.aspx 的简易网页，向网页中添加一个 Label1 控件，然后向该网页的 Page_Load 过程中输入如下的代码：

```
public partial class webforml : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        Label1.Text Session["username"].ToString();
        Label1.Text += "你好，欢迎使用本系统";
    }
}
```

（10）将 Default.aspx 设置为“设为起始页”，用户名和密码都正确时程序的运行结果

如图 5-42 所示。

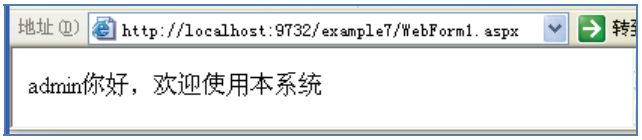

图 5-42　运行结果

3. 编程归纳

（1）网页之间数据的传递，除了使用例 5-4 构造查询字符串的方法，也可以使用本例中的 Session。Session 是 ASP.NET 中的一个对象，它可以存储特定的数据，当应用程序在页面之间跳转时，保存在 Session 中的变量其值保持不变，从而实现了在页面之间传递数据。

该对象的使用格式如下：

```
Session["变量名"] = 表达式;
```

例如本例在 default.aspx 中，下面的语句将 useName 的值保存到 Session 的变量 username 中：

```
Session["username"] = useName;
```

而在 webform1.aspx 页面中，下面的语句使用了来自 default.aspx 页面中的数据（用户名）：

```
Label1.Text = Session["username"].ToString();
```

其中的 ToString()是用来将 Session["username"]值的类型转换为 string 的方法。

（2）程序中下面的语句用来构成查询字符串：

```
string sql = "select * from userInfo where loginName='" + useName+"'" ;
```

如果用户输入的用户名是 admin 则构成的 SQL 语句为：

```
string sql = "select * from userInfo where loginName='admin'"
```

注意语句中单、双引号的使用。

（3）default.aspx 中分别验证用户输入的用户名和密码，对数据库进行了两次访问，这两次访问的代码除了查询字符串不同，其他代码都是相似的。对比例 5-6，除了 SQL 语句之外，前面若干行也都是相同的，这样的操作还会在不同的程序中重复多次。为简化程序，在开发系统时，可以将其编写成一个独立的方法（过程），将查询字符串作为方法的一个参数，这样可以反复地调用，这个独立过程的编写见例 5-8。

（4）数据集中的记录个数保存在 ds.Tables[0].Rows.Count 属性中，该值为零时表示数据集中没有记录，也就是没有查询到满足条件的记录。程序中使用下面的条件语句判断是否查询到记录：

```
if (ds.Tables[0].Rows.Count == 0)
        {
            Response.Write("<script>alert('该用户名不存在')</script>");
```

```
}
```

例 5-6 和例 5-7 都是对数据库中的表进行查询操作，下面通过几个例题介绍对记录的增加、删除和修改操作。共同的特点是将查询数据填充到数据集后，对数据集进行不同的操作，然后用数据集更新表中的数据。

5.3.4 向表中添加新的记录

例 5-8 向 userInfor 表中添加新的记录。

1. 功能要求

设计的页面如图 5-43 所示，其中的验证控件保证输入的用户名、密码不能为空，同时两次输入的密码要相同。单击“保存”按钮后，先检查输入的用户名在表中是否已经存在，如果存在，则显示“无法输入”的提示信息；不存在则将该用户信息输入到表中，输入后在另一个页面中显示表中的所有记录。

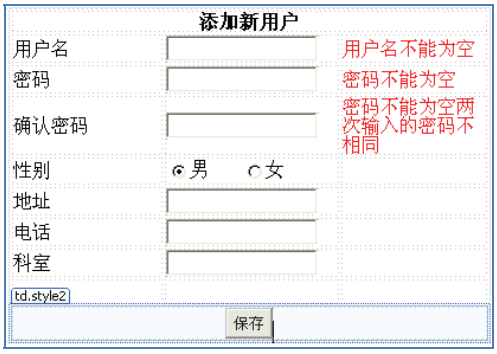

图 5-43 输入新记录页面

2. 操作过程

（1）创建名为 example8 的项目。

（2）将已创建好的数据库文件 hospital.mdf 和对应的日志文件 hospital_log.ldf 复制到 example8\App_Data 文件夹中。

（3）在 Default.aspx 网页中，插入 10 行 3 列的表格，表格占页面宽度的 50%。

（4）将表格第 1 行的 3 个单元格合并为一个，然后向单元格中输入“添加新用户”，并设置加粗和居中对齐。

（5）在第 2 行中，向第 1 个单元格输入“用户名”，第 2 个单元格插入文本框 TextBox1，第 3 个单元格插入非空数据验证控件 RequiredFieldValidator1，将其 ControlToValidate 属性设置为 TextBox1，将 ErrorMessage 设置为“用户名不能为空”。

（6）在第 3 行中，向第 1 个单元格输入“密码”，第 2 个单元格插入文本框 TextBox2，将其 TextMode 属性设置为 Password，向第 3 个单元格插入非空数据验证控件 RequiredFieldValidator2，将其 ControlToValidate 属性设置为 TextBox2，将 ErrorMessage 设置为“密码不能为空”。

（7）在第 4 行中，向第 1 个单元格输入“确认密码”，第 2 个单元格插入文本框 TextBox3，将其 TextMode 属性设置为 Password，向第 3 个单元格插入两个验证控件，一

个是非空数据验证控件 RequiredFieldValidator3，将其 ControlToValidate 属性设置为 TextBox3，将 ErrorMessage 设置为“密码不能为空”。另一个是比较数据验证控件 CompareValidator1，将其 ControlToValidate 属性设置为 TextBox3，ControlToCompare 属性设置为 TextBox2，将 Operator 设置为 Equal，将 Type 设置为 String，将 ErrorMessage 设置为“两次输入的密码不相同”。

（8）在第 5 行中，向第 1 个单元格输入“性别”，第 2 个单元格插入两个单选按钮 RadioButton1 和 RadioButton2，其 Text 属性分别是“男”和“女”，RadioButton1 的 Checked 属性设置为 True，两个控件的 GroupName 属性都设置为“性别”。

（9）在第 6 行中，向第 1 个单元格输入“地址”，第 2 个单元格插入文本框 TextBox4。

（10）在第 7 行中，向第 1 个单元格输入“电话”，第 2 个单元格插入文本框 TextBox5。

（11）在第 8 行中，向第 1 个单元格输入“科室”，第 2 个单元格插入文本框 TextBox6。

（12）合并第 10 行中的 3 个单元格，然后插入命令按钮 Button1，并将其 Text 属性设置为“保存”，居中对齐。

（13）编写两个独立的过程 GetData()和 ExecuteNonQuery()。

首先在 default.aspx.cs 选项卡中，在程序开始添加如下的语句：

```
using System.Data.SqlClient;
```

为避免代码段的重复编写，本例中把对表的查询操作编写为一个独立的过程，过程名为 GetData()，参数为查询字符串，过程的返回结果为数据集。

```
public DataSet GetData(string sql)
{
    string strcon = "Data Source=.\\SQLEXPRESS;AttachDbFilename=";
    strcon += "E:\\his 系统的开发\\C#例题\\example8\\App_Data\\hospital. mdf;";
    strcon+="Integrated Security=True;Connect Timeout=30;User Instance= True";
    SqlConnection con = new SqlConnection(strcon);
    if (con.State != ConnectionState.Open)
    {
        con.Open();
    }
    SqlCommand com = new SqlCommand();
    com.Connection = con;
    com.CommandType = CommandType.Text;
    com.CommandText = sql;
    DataSet ds = new DataSet();
    SqlDataAdapter sda = new SqlDataAdapter(com);
    sda.Fill(ds);
    con.Close();
    return ds;
}
```

对记录进行增加、删除和修改的操作编写成另一个独立的过程 ExecuteNonQuery()，参数是 SQL 命令，返回结果是一个整数，表示命令影响到的记录条数，过程内容如下：

```
public int ExecuteNonQuery(string sql)
    {
        string strcon = "Data Source=.\\SQLEXPRESS;AttachDbFilename=";
        strcon += "E:\\his 系统的开发\\C#例题\\example8\\App_Data\\hospital.
         mdf;";
        strcon += "Integrated Security=True;Connect Timeout=30;User Instance=
        True";
        SqlConnection con = new SqlConnection(strcon);
        con.Open();
        SqlCommand com = new SqlCommand();
        com.Connection = con;
        com.CommandType = CommandType.Text;
        com.CommandText = sql;
        int count = com.ExecuteNonQuery();
        con.Close();
        return count;
    }
```

（14）输入“保存”按钮的事件代码。

在 Button1_Click()中输入如下的代码：

```
protected void Button1_Click(object sender, EventArgs e)
    {
        string userName = TextBox1.Text;
        string sql = "select * from userInfo where userName='" + userName + "'";
        DataSet ds = GetData(sql);
        if (ds.Tables[0].Rows.Count != 0)
        {
            Response.Write("<script>alert('该用户名已经存在')</script>");
        }
        else
        {
            sql = "select * from userInfo " ;
            ds = GetData(sql);
            int count = ds.Tables[0].Rows.Count;
            string Id = ds.Tables[0].Rows[count - 1]["userId"].ToString ();
            int  userId =  Convert.ToInt32 (Id);
            userId++;
            string pwd = TextBox2.Text;
            string sex = "";
            if (RadioButton1.Checked)
            {
                sex = "男";
```

```
            }
            else
            {
                sex = "女";
            }
            string address = TextBox4.Text;
            string tel = TextBox5.Text;
            string ks = TextBox6.Text;
            sql = "insert into userInfo(userId,userName,loginName,loginPwd,
             Sex,Address,Phone,SectionRoom)"
                  + "values("+ userId+",'" +userName +"','"  + userName + "',
                  '" + pwd + "','" + sex + "','" + address + "','" + tel + "','"
                  + ks + "')";
            count = ExecuteNonQuery(sql);
            if (count == 1)
            {
                Response.Redirect("userList.aspx");
            }
        }
```

（15）添加新的页面 userList.aspx，该页面用来显示添加记录后表中的信息。在网页中添加文本“用户信息清单”，设置格式为加粗、大小为“xx-large”，然后向页面添加一个 repeater 控件。在源视图下，向 userList.aspx 页面中 和 之间添加如下的代码：

```
<table style="width:60%;" border="1" >
 <tr>
    <td>用户名</td> <td>性别</td> <td>地址</td>
    <td>电话</td>  <td>科室</td>
 </tr>
 <asp:Repeater ID="Repeater1" runat="server" >
 <ItemTemplate>
 <tr>
    <td><%#Eval("userName") %></td>
    <td><%#Eval("Sex")%></td>
    <td><%#Eval("Address")%></td>
    <td><%#Eval("Phone")%></td>
    <td><%#Eval("SectionRoom")%></td>
 </tr>
 </ItemTemplate>
 </asp:Repeater>
</table>
```

（16）将 default.aspx 设置为起始页，运行该页面。

3. 编程归纳

（1）将查询编写为一个独立的过程 GetData(string sql)，本程序中三次调用了该过程。

（2）CompareValidator 控件用于比较两个控件输入的内容是否满足程序的要求。通常是对两个文本框中输入的内容进行比较，比较时可以按字符串、整数、浮点数、日期等进行，这可以通过 Type 属性进行设置；比较的关系有相等、不等、大于、大于等于、小于、小于等于，这可以通过 Operator 属性进行设置。

（3）userId 的处理。userId 是 userInfo 表中的一个字段，该字段被设置为主键，所以在表中该字段的值既不允许为空值，也不允许出现重复。本程序中该字段的值不是用户输入而是通过程序自动添加，添加的方法是对表中已存在的该字段最大值加 1 后作为新记录的 userId。

（4）本程序三次调用过程 GetData(string sql)，第一次调用是查询新的用户名在表中是否已经存在，如果查询有结果，说明该用户名与已有的重名，需要重新输入，不重复时才将新记录添加到表中。

第二次调用是查询表中所有记录，因为表中记录是按主键的顺序排列的，所以数据集中最后一条记录中的 userId 值就是最大的，通过下列的语句获取了这个最大值：

```
string Id = ds.Tables[0].Rows[count - 1]["userId"].ToString ();
```

上一条语句中的表达式 ds.Tables[0].Rows.Count 表示表中记录的个数。

第三次调用是在插入新记录后查询表中所有记录（包括新输入的记录），然后在 userList 页面中显示出来，三次调用只需要修改查询字符串即可。

5.3.5 删除表中记录

例 5-9 删除 userInfo 表中的记录。

1. 功能要求

页面中使用 repeater 控件显示表中的记录，每条记录前面加上一个复选框，如图 5-44 所示。单击“删除”按钮后，将选中的记录删除，删除完成后在第 2 个页面中显示表中其他的所有记录。

选择	用户名	性别	地址	电话	科室
☐	admin	男	交大医学院	13000000000	外科
☐	张三	男	交大计教中心	13000001111	外科
☐	李四	女	交大医学院	13000002222	外科
☐	王五	女	交大电信学院	13000004444	内科
☐	gxyhy11	男	0000	0111	1111
☐	hy11	男	0000	0111	1111
☐	yq11	男	22	222	2222

删除

图 5-44 页面设计

2. 操作过程

（1）创建名为 example9 的项目。

（2）将已创建好的数据库文件 hospital.mdf 和对应的日志文件 hospital_log.ldf 复制到 example9\App_Data 文件夹中。

（3）创建专门用于访问数据库的类，在“解决方案资源管理器”窗格中右击 example9，在弹出的快捷菜单中执行“添加新项”命令。打开“添加新项”对话框，在对话框中的“模

板”列表框中选择“类”，“名称”框中输入“DBHelper.cs”，然后单击“添加”按钮，弹出如图 5-45 所示的对话框，单击“是”按钮，这时新建的类 DBHelper.cs 被保存在项目的 App_Code 文件夹中。保存在该文件夹中的程序可以被本项目的其他程序使用。

图 5-45　提示对话框

（4）在 DBHelper.cs 的 public class DBHelper 大括号中输入两个过程的内容，就是在例 5-8 中创建的两个过程。

（5）在页面 default.aspx 中添加一个 repeater 控件，在源视图下，向页面中 和 之间添加如下的代码：

```
<table style="width:60%" border="1" >
    <tr>
    <td>选择</td>
        <td>用户名</td>
        <td>性别</td>
        <td>地址</td>
        <td>电话</td>
        <td>科室</td>
    </tr>
    <asp:Repeater ID="Repeater1" runat="server">
  <ItemTemplate>
    <tr>
    <td>
        <asp:CheckBox ID="CheckBox1"  runat="server" />
        <asp:HiddenField ID="HiddenField1" runat="server" Value='<%#Eval
        ("userId") %>'  />
          </td>
        <td> <%#Eval("userName") %></td>
        <td> <%#Eval("Sex")%></td>
        <td> <%#Eval("Address")%></td>
        <td> <%#Eval("Phone")%></td>
        <td> <%#Eval("SectionRoom")%></td>
    </tr>
    </ItemTemplate>
      </asp:Repeater>
</table>
```

（6）在设计视图下，向页面添加一个 Button 控件 Button1，将其 Text 属性设置为“删除”，此时页面布局如图 5-46 所示。

图 5-46　页面布局

（7）在页面加载事件中，输入如下的代码：

```
protected void Page_Load(object sender, EventArgs e)
    {
        if (Page.IsPostBack)
            return;
        string sql = "select * from userInfo";
        DBHelper db = new DBHelper();     //实例化对象
        DataSet ds = db.GetData(sql);
        Repeater1.DataSource = ds;        //指定数据源
        Repeater1.DataBind();             //绑定数据源
    }
```

（8）向 Button1_Click()中输入如下的删除记录代码：

```
DBHelper db = new DBHelper();
       int count = 0;
       foreach (RepeaterItem item in Repeater1.Items)//循环遍历Repeater1中每一行数据
       {
           CheckBox cb = (CheckBox)item.FindControl("CheckBox1");
           if (cb.Checked)
           {
               HiddenField hf = item.FindControl("HiddenField1") as HiddenField;
               string id = hf.Value;
               string sql = "delete from userInfo where userId='" + id + "'";
               count += db.ExecuteNonQuery(sql);
           }
       }
       if (count >0)
      {
        Response.Redirect("userList.aspx");
      }
    }
```

（9）添加 userList.aspx 页面，用来显示删除指定记录后表中其他所有的记录。由于例 5-8 中已经创建了该网页，现在只需要将上一题中的网页添加到本网站中即可。方法是在“解决方案资源管理器”窗格中右击 example9，在弹出的快捷菜单中执行“添加现有项”

命令，然后在对话框中选择例 5-8 中创建的网页（userList.aspx 和 userList.aspx.cs 两个文件）。

最后将该页面的 Page_Load 代码改写为与本例 default.aspx 的 Page_Load 相同的代码即可。

（10）将 default.aspx 设置为起始页，运行该页面。

3. 编程归纳

（1）DBHelp.cs 类中包含两个过程。一个是 ExecuteNonQuery(string sql)，该过程用于执行 SQL 的记录增加、删除和修改，其返回值是所影响的记录条数；另一个是 GetData，用于对数据库中表的访问，返回值是记录集，通常是将该记录集填充到数据集中。

（2）对 Repeater1 控件添加的代码中，下面的语句是向表格中添加了一个 HiddenField 控件即隐藏字段控件：

```
<asp:HiddenField ID="HiddenField1" runat="server" Value='<%#Eval("userId") %>' />
```

在 Web 中，隐藏控件常用于保存程序中要用到但不需要显示在页面上的数据，其 Value 属性保存被隐藏字段的值。userInfo 表中的 userId 字段是表中的主键，用来唯一地标识每条记录，通过该字段可以查询到要删除的记录。

本页面的隐藏字段名称是 HiddenField1，保存的是 userInfo 表中 userId 字段的值。

（3）Repeater1 控件添加的代码中，下面的语句是在每条记录的前面添加一个复选框，用来选中记录，最后将选中的每条记录删除：

```
<asp:CheckBox ID="CheckBox1"  runat="server" />
```

（4）在页面加载事件代码中，if 语句中的 IsPostBack 是 Page 页面中的一个属性，用来指明网页是否回传，其值为布尔型，用来判断该网页是第一次生成还是回传的。网页中有些程序段仅仅需要在网页第一次生成时才需要执行，写在 IsPostBack 为 False 的语句块中，本代码中后面的语句都是 IsPostBack 为 False 的情况。

（5）在页面加载事件代码中，以下代码使用了上面定义的 DBHelper.cs 类：

```
DBHelper db = new DBHelper();//实例化对象
       DataSet ds = db.GetData(sql);
```

这样，在加载该页面时，就可以将 userInfo 表中的所有记录显示在页面中。

（6）“删除”控件的代码中，下列语句用来循环遍历 Repeater1 中的每一行数据：

```
foreach (RepeaterItem item in Repeater1.Items)
```

对每一条记录，判断其复选框是否被选定，如果选中，则执行 SQL 命令，将该记录的隐藏字段的值 hf.Value 与表中 userId 字段一致的记录删除，变量 count 保存了被删除的记录个数。

循环（遍历）结束后，如果发生了删除记录的操作，则通过下列的语句启动页面 userList.aspx 显示表中目录所有的记录。

```
Response.Redirect("userList.aspx");
```

5.3.6 修改表中的记录

例 5-10 修改 userInfo 表中的记录。

1. 功能要求

程序中有三个页面。

（1）第一个页面中使用 repeater 控件显示表中记录，每条记录前面加上一个复选框，单击“修改”按钮后，跳转到第二个页面；

（2）在第二个页面显示指定记录的内容，然后修改字段的值，其中 userId 和用户名这两个字段不允许修改，单击“保存”按钮后将修改后的内容保存在表中；

（3）在第三个页面中显示修改完成后的记录。

2. 操作过程

本例题在前面几个例题已经创建的页面基础上完成。

（1）创建名为 example10 的项目。

（2）将已创建好的数据库文件 hospital.mdf 和对应的日志文件 hospital_log.ldf 复制到 example10\App_Data 文件夹中。

（3）将例 5-9 中创建的访问数据库的类 DBHelper.cs 添加到网站中，方法是在“解决方案资源管理器”窗格中右击 example10，在弹出的快捷菜单中执行“添加现有项”命令，在打开的对话框中选择 DBHelper.cs，然后将其复制到 App_Code 文件夹中，最后将程序中数据库所在位置改为本题的 example10\App_Data 文件夹。

（4）将例 5-9 的第一个页面添加到本网站中，在“解决方案资源管理器”窗格中右击 example10，在弹出的快捷菜单中执行“添加现有项”命令，然后在对话框中选择例 5-9 中创建的网页（Default.aspx 和 Default.aspx.cs 两个文件），在添加过程中会显示图 5-47 所示的对话框，单击“是”按钮即可，然后将 Default.aspx 中 Button1 的 Text 属性改为“修改”。

（5）添加第二个页面，名为 userEdit.aspx，设计页面布局如图 5-48 所示，将用户名对应的文本框 TextBox1 的 ReadOnly 属性设置为 True。

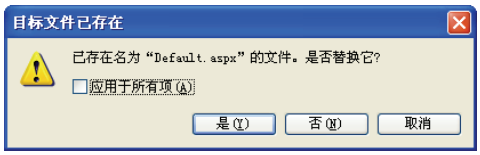

图 5-47 提示对话框

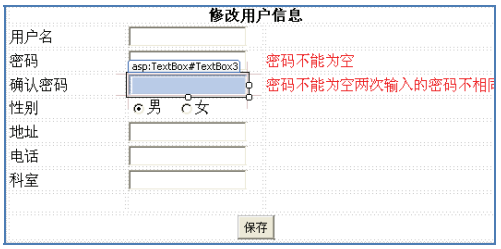

图 5-48 userEdit.aspx 页面布局

（6）输入 userEdit.aspx.cs 的页面加载事件代码：

```
protected void Page_Load(object sender, EventArgs e)
    {
        if (Page.IsPostBack)
            return;
        string userId = Session["Id"].ToString();
```

```
        string sql = "select * from userInfo where userId='" + userId + "'";
        DBHelper db = new DBHelper();
        DataSet ds = db.GetData(sql);
        TextBox1.Text = ds.Tables[0].Rows[0]["userName"].ToString ();
        TextBox2.Text = ds.Tables[0].Rows[0]["loginPwd"].ToString ();
        TextBox3.Text = ds.Tables[0].Rows[0]["loginPwd"].ToString ();
        if (ds.Tables[0].Rows[0]["Sex"].ToString () == "男")
        {
            RadioButton1.Checked = true;
        }
        else
        {
            RadioButton2.Checked = true;
        }
        TextBox4.Text = TextBox3.Text = ds.Tables[0].Rows[0]["Address"].
        ToString ();
        TextBox5.Text = TextBox3.Text = ds.Tables[0].Rows[0]["Phone"].
        ToString ();
        TextBox6.Text = TextBox3.Text = ds.Tables[0].Rows[0]["SectionRoom"].
        ToString ();
}
```

（7）输入 userEdit.aspx.cs 的命令按钮 Button1 单击事件代码：

```
protected void Button1_Click(object sender, EventArgs e)
{
    string userId = Session["Id"].ToString();
    string userName=TextBox1.Text;
    string Pwd = TextBox2.Text;
    string Sex = "";
    if (RadioButton1.Checked)
    {
        Sex = "男";
    }
    else
    {
        Sex = "女";
    }
    string Address = TextBox4.Text;
    string Phone = TextBox5.Text;
    string Room = TextBox6.Text;
    DBHelper db = new DBHelper();
    int update = db.ExecuteNonQuery(sql);
    Response.Redirect("userList.aspx");
}
```

（8）添加第三个页面 userList.aspx，用来显示修改指定记录之后表中所有的记录，该网页的内容与例 5-8 和例 5-9 中相同。

（9）将 default.aspx 设置为起始页，运行该页面。

运行后显示第一个页面如图 5-49 所示，选择某个用户对应的复选框，然后单击“删除”按钮，显示第二个页面，如图 5-50 所示。在这个页面中的各个文本框中直接修改信息，然后单击“保存”按钮，这时第三个页面显示修改后的结果，如图 5-51 所示。

图 5-49 选择用户

图 5-50 修改信息

图 5-51 显示修改后的结果

3. 编程归纳

（1）本题中的前两个页面与例 5-8 和例 5-9 中的页面相似，可以在前面例题基础上修改即可。第三个页面与例 5-9 的 userList.aspx 完全相同，可以将已创建的页面添加到本程序中，不需要重新创建。

（2）userEdit.aspx.cs 页面是将前面选中的用户信息显示出来，这两个页面之间通过 Session["Id"]将用户的 userId 传递过来。

通过前面的多个例题，介绍了开发 HIS 系统的主要技术和 C#的基础，接下来就可以进行信息系统的设计与实施了。

5.4 HIS 系统的框架设计

本节介绍 HIS 系统的框架组合、母版页技术和构成框架的重要控件 TreeView 与 ContentPlaceHolder 控件。

5.4.1 HIS 系统的框架组成

HIS 系统启动后，首先是登录界面，如图 5-52 所示。在正确输入用户名和密码后，单击“登录”按钮，如果用户信息正确，进入 HIS 的首页，如图 5-53 所示。

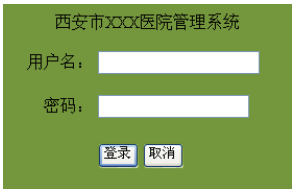

图 5-52 登录界面

图 5-53　系统主界面

单击左侧“门诊管理”下面的“门诊挂号”，界面显示内容如图 5-54 所示。

图 5-54　门诊挂号界面

单击左侧“系统管理”下面的“员工管理”，界面显示内容如图 5-55 所示。

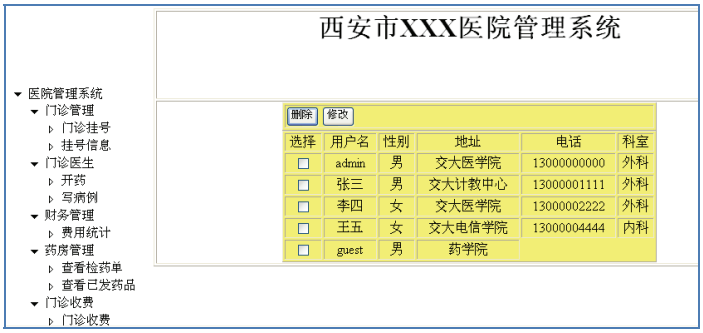

图 5-55　员工管理界面

继续单击其他内容时，可以看出，整个界面由三个部分组成。左侧是菜单选择，右侧上方显示系统的标题（医院名称），右侧正文则是根据单击左侧菜单显示出的不同内容，框架结构如图 5-56 所示。

框架中，左侧和右上方的内容固定不变，这两个部分称为固定区，右侧下方是内容可变区，也就是说，除了登录界面外，其他界面都是统一的外观，即两个固定区和一个可变区（右下方），这样的效果可以通过 ASP.NET 中的母版页技术实现。

就像 PowerPoint 中为幻灯片设置母版一样，ASP.NET 中的母版页是一种重用技术。一个母版页可以为应用程序中所有页或某些页定义统一的外观和标准行为，然后再创建包含

要显示的内容的其他页（称为内容页）。当程序中请求内容页时，这些内容页与母版页合并，然后将母版页的布局和内容页的内容组合在一起输出。

HIS 系统界面左侧是菜单部分（也称为导航区），如图 5-57 所示。可以使用 Menu 控件或 TreeView 控件实现。

图 5-56　系统框架

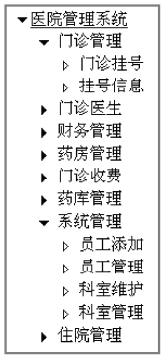

图 5-57　系统菜单

5.4.2　系统界面的开发过程——导航控件、母版页与内容页

下面通过实例说明系统界面开发的部分过程。

例 5-11　开发系统界面。

1. 功能要求

包括登录界面、一个母版页、两个内容页，内容页分别是主界面和显示用户信息。

2. 操作过程

（1）创建名为 hospital 的项目，启动 VS 2008，执行“文件”→“新建”→“项目”命令，显示“新建项目”对话框，如图 5-58 所示。

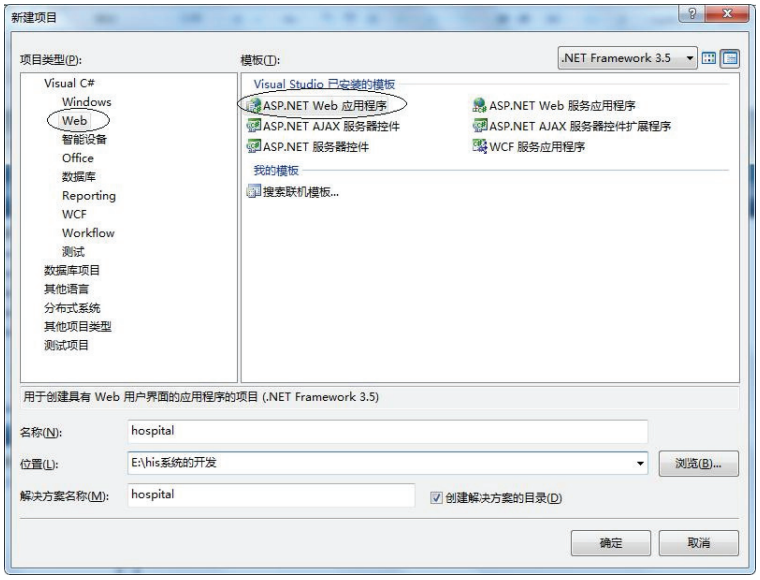

图 5-58　“新建项目”对话框

（2）在对话框中，选择“ASP.NET Web 应用程序”模板，“项目类型”选择 Visual C# 下方的 Web，“位置”文本框中输入“E:\his 系统的开发”，“名称”文本框中输入“hospital”，然后单击“确定”按钮。

（3）将例 5-9 中创建的访问数据库的类 DBHelper.cs 添加到网站中。方法是在“解决方案资源管理器”窗格中右击 hospital，在弹出的快捷菜单中执行“添加现有项”命令，在打开的对话框中选择 DBHelper.cs，然后将其复制到 App_Code 文件夹中，最后将程序中数据库所在位置改为本题的 hospital \App_Data 文件夹。

（4）创建登录界面 login.aspx，创建的内容和操作过程可以参考例 5-7 的内容。

（5）创建母版页并向母版页中添加 TreeView 控件。

① 执行“添加”→“新建项”命令，显示“添加新项”对话框，在对话框中选择“母版页”，“名称”框中输入 web1，然后单击“添加”按钮，窗口显示内容如图 5-59 所示。

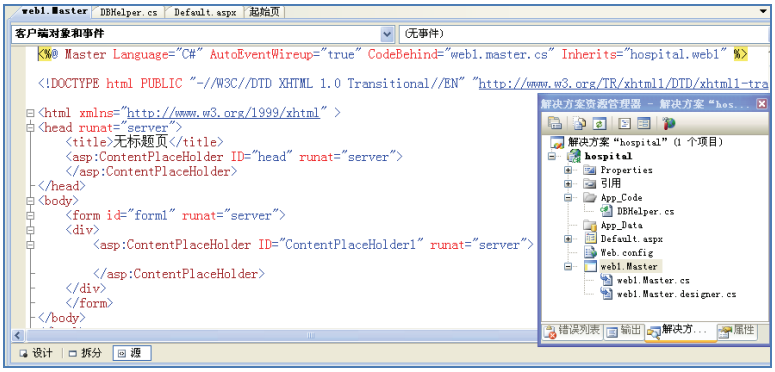

图 5-59　母版页

可以看出，在“解决方案资源管理器”窗格中，多了一个 web1.master 的母版页。

② 母版页中包含两个固定区和一个可变区（由 ContentPlaceHolder 控件来控制），使用表格设计如下的布局：

<table>
<tr><td rowspan="2">固定区 TreeView 控件</td><td>固定内容文本</td></tr>
<tr><td>可变区</td></tr>
</table>

上面的结构可以先插入一行两列的表格，然后在第二个单元格中再插入一个 2 行 1 列的子表格。

③ 左边的固定区添加一个 TreeView 控件设置菜单，切换到 web1.master 的源视图，在<head>区输入如下的格式代码：

```
<style type="text/css">
        .style4
        {
            width: 194px;
            height: 754px;
        }
        .style5
        {
```

```
            width: 90%;
            height: 754px;
        }
    </style>
```

④ 在 和 标记之间输入如下的代码：

```
<table style="width: 100%; border:1 height: 736px;">
              <tr>
                 <td class="style4" >
                    <asp:TreeView ID="TreeView1" runat="server" ImageSet=
                    "Arrows" Height="345px"  Width="165px">
       <HoverNodeStyle Font-Underline="True" ForeColor="" />
       <Nodes>
<asp:TreeNode Text="医院管理系统" Value="医院信息管理系统">
<asp:TreeNode Text="门诊管理" Value="药品信息添加">
<asp:TreeNode Text="门诊挂号" Value="门诊挂号" NavigateUrl="~/gh.aspx">
</asp:TreeNode>
<asp:TreeNode Text="挂号信息" Value="挂号信息" NavigateUrl="~/ghList.aspx">
</asp:TreeNode>
  </asp:TreeNode>
<asp:TreeNode Text="门诊医生" Value="门诊医生">
<asp:TreeNode Text="开药" Value="开药" NavigateUrl="~/ghList1.aspx">
</asp:TreeNode>
<asp:TreeNode NavigateUrl="~/xbl.aspx" Text="写病例" Value="写病例">
</asp:TreeNode>
    </asp:TreeNode>
<asp:TreeNode Text="财务管理" Value="药品类型添加">
<asp:TreeNode Text="费用统计" Value="费用统计" NavigateUrl= "~/CW.aspx">
</asp:TreeNode>
   </asp:TreeNode>
<asp:TreeNode Text="药房管理" Value="药品类型维护">
<asp:TreeNode Text="查看检药单" Value="查看检药单" NavigateUrl="~/jyd.aspx">
</asp:TreeNode>
<asp:TreeNode Text="查看已发药品" Value="查看已发药品" NavigateUrl="~/yfDrug.aspx">
</asp:TreeNode>
      </asp:TreeNode>
<asp:TreeNode Text="门诊收费" Value="门诊收费" NavigateUrl="~/sf.aspx">
</asp:TreeNode>
     </asp:TreeNode>
<asp:TreeNode Text="药库管理" Value="药库管理">
<asp:TreeNode Text="药品添加" Value="药品添加" NavigateUrl="~/drugsEdit.
aspx"></asp:TreeNode>
<asp:TreeNode Text="药品管理" Value="药品管理" NavigateUrl="~/drugList.
aspx"></asp:TreeNode>
     </asp:TreeNode>
```

```
<asp:TreeNode Text="系统管理" Value="系统管理">
<asp:TreeNode Text="员工添加" Value="员工添加" NavigateUrl="~/AddUser.
aspx"></asp:TreeNode>
<asp:TreeNode Text="员工管理" Value="员工管理" NavigateUrl="~/userList.
aspx"></asp:TreeNode>
<asp:TreeNode Text="科室维护" Value="科室维护" NavigateUrl="~/kswh.aspx">
</asp:TreeNode>
<asp:TreeNode NavigateUrl="~/ksgl.aspx" Text="科室管理" Value="科室管理">
</asp:TreeNode>
        </asp:TreeNode>
<asp:TreeNode Text="住院管理" Value="住院管理">
<asp:TreeNode Text="病房添加" Value="病房添加" NavigateUrl="~/BFTJ.aspx">
</asp:TreeNode>
<asp:TreeNode Text="病房查看" Value="病房查看" NavigateUrl="~/zydjList.
aspx"></asp:TreeNode>
<asp:TreeNode Text="住院登记" Value="住院登记" NavigateUrl="~/zydj.aspx">
</asp:TreeNode>
<asp:TreeNode Text="住院查看" Value="住院查看" NavigateUrl="~/Zyck.
aspx"></asp:TreeNode>
            </asp:TreeNode>
           </asp:TreeNode>
     </Nodes>
                        <NodeStyle Font-Names="Tahoma" Font-Size="10pt"
                         ForeColor="Black" HorizontalPadding="5px" NodeSpacing=
                         "0px" VerticalPadding="0px" />
                        <ParentNodeStyle Font-Bold="False" />
                        <SelectedNodeStyle Font-Underline="True"Horizontal
                         Padding= "0px" VerticalPadding="0px" />
                    </asp:TreeView>
                </td>
                <td valign="top" class="style5">
            <table style="width: 23%; height: 131px;" border="1" align=
           "center">
               <tr>
            <td style="width: 100%; text-align: center;background-color:">
              <h1 style="height: 83px; width: 752px; font-size: -20px; ">
              西安市 XXX 医院管理系统</h1>
                        </td>
                    </tr>
                    <tr>
          <td style="width: 100%; text-align: center">
             <asp:ContentPlaceHolder ID="ContentPlaceHolder1" runat="server">
              </asp:ContentPlaceHolder>
                        </td>
                    </tr>
```

```
            </table>
        </td>
    </tr>
</table>
```

（6）创建内容页 WebForm1.aspx 即主界面。

① 在“解决方案资源管理器”窗格中右击 hospital，在弹出的快捷菜单中执行“添加”→“新建项”命令，显示“添加新项”对话框，如图 5-60 所示。

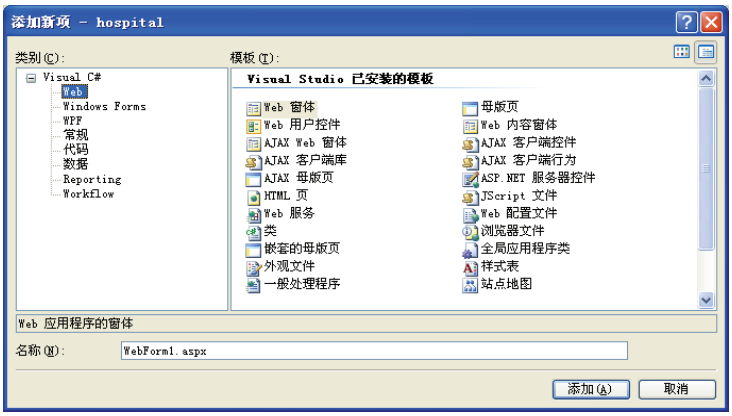

图 5-60 “添加新项”对话框

② 在对话框中，选择“Web 内容窗体”，在名称框中输入“WebForm1.aspx”，然后单击“添加”按钮，显示“选择母版页”对话框，如图 5-61 所示。

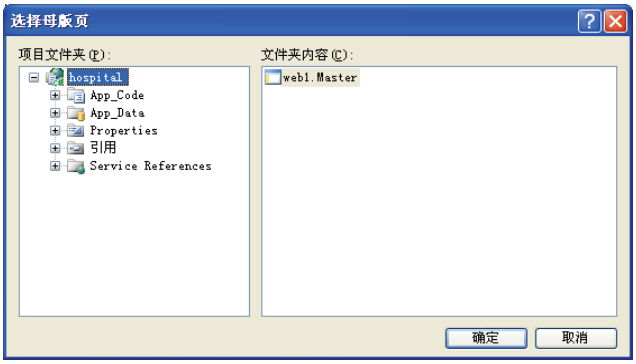

图 5-61 “选择母版页”对话框

③ 在对话框中选择创建的母版页 web1.Master，然后单击“确定”按钮。

④ 在 WebForm1.aspx 页中输入如下的代码：

```
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" runat=
"server">
        <p> <asp:Label ID="Label1" runat="server" ></asp:Label>
        欢迎登录医院管理系统</p>
</asp:Content>
```

⑤ 在 WebForm1.aspx 的 Page_Load 过程中输入如下的代码：

```
protected void Page_Load(object sender, EventArgs e)
        {
            if (Session["username"] != null)
            {
                Label1.Text = Session["username"].ToString();
            }
        }
```

（7）创建内容页 userList.aspx，创建时引用母版页 web1.Master，创建过程可以参考上一个步骤。

（8）将 login.aspx 设置为起始页。

（9）其他的页面可以按同样的方法进行创建。

3. 编程分析

（1）前面例题是直接创建网站，本题创建的是项目。项目中选择的是“ASP.NET Web 应用程序”模板，过程上略有不同，但后面的创建方法和效果是一样的。

（2）在创建母版页时，使用了一个新建的控件 TreeView，该控件可以以树状结构显示分层的数据，也包括了多级层次结构的菜单，每一层的各个数据称为每一个结点（TreeNode）。一个结点之下还可以有多个子结点，控件中各个结点可以动态地添加和删除。本系统中没有进行动态操作，结点结构如图 5-62 所示，图中标有数字的分别表示各级结点。

图 5-62　TreeView 结点结构

在 和 标记之间的代码中，下面的标记“asp:TreeView”表示添加了一个 TreeView 控件。

```
<asp:TreeView ID="TreeView1" runat="server" ImageSet="Arrows" Height= "345px"
Width="165px">
```

接下来的 asp:TreeNode 用来向结构中添加结点，每一对<asp:TreeNode r>和</asp:TreeNode>之间还可以再添加下一级结点。

下面是对某个结点的各个属性的具体描述：

```
<asp:TreeNode Text="门诊收费" Value="门诊收费" NavigateUrl="~/sf.aspx">
```

其中的 NavigateUrl="~/sf.aspx"表示单击该结点时要启动的页面，本句中是启动 sf.aspx 页面。

显然，使用 TreeView 控件，既实现了菜单，也将其他的页面整合到一起形成了一个完整的信息系统。

5.4.3 其他页面的界面设计

各个处理模块的结构与用户界面基本相同，以下给出几个模块的结构，其他的也可以参考这些界面。

（1）门诊挂号和挂号信息（图 5-63）

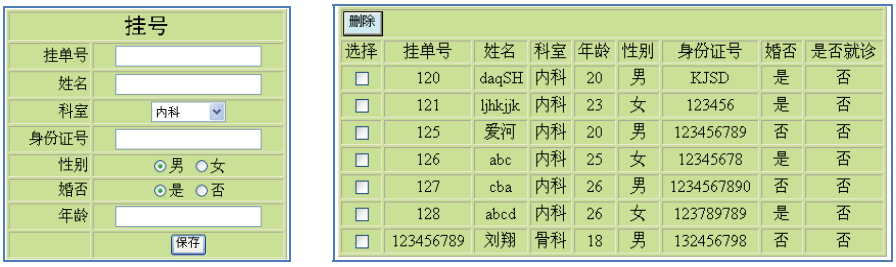

图 5-63 门诊挂号和挂号信息

（2）开药和写病历（图 5-64）

图 5-64 开药和写病历

（3）费用统计（图 5-65）

图 5-65 费用统计

（4）药品添加和药品管理（图 5-66）

图 5-66　药品添加和药品管理

（5）添加用户和用户管理（图 5-67）

图 5-67　添加用户和用户管理

（6）科室管理（图 5-68）

图 5-68　科室管理

（7）住院登记和住院查看（图 5-69）

图 5-69　住院登记和住院查看

5.5　其 他 问 题

5.5.1　调试程序时频繁出现的问题

根据以往经验，现将经常出现的问题归纳如下。

1. SQL 语句的含义及 SQL 字符串的构造

SQL 是关系型数据库的通用命令，在用某个程序设计语言访问数据库时，都是通过 SQL 命令进行的，而对表的操作就是对表中记录进行增加、修改、删除和查询。本章的例 5-6~例 5-10 都是围绕这 4 个操作进行的。

（1）查询使用的命令是 select，下面命令是无条件查询，即查询表中所有记录，通常用于显示表中的所有记录，例如模块中显示所有用户、所有挂号、所有药品等：

```
string sql = "select * from userInfo ";
```

下面命令是条件查询，按用户的 userInfo 字段进行查询，用于对选中的记录进行修改或删除之前进行的查询。

```
string sql = "select * from userInfo where userId='" + userId + "'";
```

（2）添加记录使用 insert 命令，下列命令是向 userInfo 表中添加新的记录，要注意的是命令中字段值的类型和顺序要与字段名中保持一致，特别注意命令中单双引号的混合使用。

```
sql = "insert into userInfo(userId,userName,loginName,loginPwd,Sex,Address,
Phone,SectionRoom)"
                + "values("+ userId+",'" +userName +"','"  + userName + "','"
                + pwd + "','" + sex + "','" + address + "','" + tel + "','"
                + ks + "')";
```

（3）delete 命令用来将满足 where 条件的记录删除，下列的命令删除指定 userId 的记录，特别注意如果没有指定条件，则表中所有记录被删除。

```
string sql = "delete from userInfo where userId='" + id + "'";
```

（4）使用 update 命令修改记录，与 insert 相似的是该 SQL 查询串通常较长，要注意字段名和字段值的区别，在书写时通常要结合表结构对照各个字段的名称。

```
string sql = "update  userInfo  set loginPwd='"  + Pwd + "',Sex='" + Sex + "',
Address='" + Address + "',Phone='" + Phone + "',SectionRoom='" + Room + "'
where userId='" + userId + "'";
```

2. 参数传递

从一个页面跳转到另一个页面时，有时需要将数据进行传递，例如将在一个网页中选中的用户名或 userId 传到另一个页面，常用的方法有使用查询字符串（例 5-4）、Session 对象法（例 5-7），例如下面两条语句的写法：

```
查询字符串：String URL="StuInfo.aspx?name=张三&id=2160001";
Session 对象：Session["username"] = useName;
```

3. 连接字符串的含义

使用 ADO.NET 技术访问数据库时，一定要正确写出连接字符串。从例 5-6 开始，各个例题中都有对数据库的访问，虽然连接字符串比较复杂也比较长，但这些例子中的字符

串几乎一样，只有数据库文件所在位置的区别，主要是区分组成这个字符串的 4 个部分的含义，如果不容易写出，可以参考例 5-6 在“数据源配置向导”中进行查询。

4. 各种符号的输入

程序中使用的各种符号，例如单引号、双引号、逗号、分号甚至是空格，都要在英文状态下输入，否则运行时会出错，并且这种错误通常不太容易发现。

5. 区分大小写

SQL 命令中的命令动词（例如 select）和其他关键词（例如 from、where 等）不区分大小写，但在 C#程序里的变量名必须区分大小写。例如，对于下面的查询字符串：

```
string sql = "update  userInfo  set loginPwd='"  + Pwd + "',Sex='" + Sex + "',
Address='" + Address + "',Phone='" + Phone + "',SectionRoom='" + Room + "'
where userId='" + userId + "'";
```

其中的 SQL 关键字 update、set 等不区分大小写，但其中的变量名（对应的是字段的值）必须区分大小写，例如引号之外的 Address 、Phone 和 Room 等一定要注意大小写的区分。

5.5.2　关于本系统的补充说明

1. 每个例题涉及的语法要点和编程技术小结

在 5.2 节和 5.3 节中介绍了 10 个例题，下面归纳这些例题涉及的主要技术要点供学习时参考，也应特别留意每题结束之前的编程归纳。

例 5-1 的技术要点：创建 Web 应用程序。

- Web 应用程序的设计和运行完整过程；
- Label（标签）控件的使用。

例 5-2 的技术要点：创建欢迎页面。

- TextBox（文本框）控件和 Button（按钮）控件的使用；
- 事件代码的编写；
- 在客户端弹出消息框的方法。

例 5-3 的技术要点：创建收集信息页面。

- 其他控件的使用：RadioButtonList、DropDownList 和 CheckBoxList。

例 5-4 的技术要点：网页之间的跳转和数据的传递。

- 网页之间的跳转 Response.Redirect(URL)；
- 网页之间的数据传递；
- 查询字符串的构成和含义。

例 5-5 的技术要点：使用表格进行页面布局。

- 采用表格布局页面；
- 表格的操作；
- 非空数据验证控件（RequiredFieldValidator）的使用。

例 5-6 的技术要点：使用 Repeater 控件显示记录。

- 数据库访问的完整过程；

- 数据集的概念；
- Repeater 控件的使用，只能在设计视图下输入代码的控件；
- 页面的 Page_Load()事件；
- 用 Eval("userId")计算绑定的字段；
- 连接字符串的查询方法；
- 记录的查询操作。

例 5-7 的技术要点：非空数据验证控件的使用。

- 通过访问数据库获得用户名和密码；
- 使用 Session["username"]对象在页面之间传递用户名（数据）。

例 5-8 的技术要点：向表中添加新的记录。

- 比较数据验证控件 CompareValidator 的使用；
- 独立过程的作用，GetData（string sql）和 ExecuteNonQuery(string sql)的编写；
- 新记录的添加。

例 5-9 的技术要点：删除表中记录。

- 复选框控件的使用；
- DBHelper.cs 类的作用和编写；
- 隐藏字段控件的作用；
- IsPostBack 属性的作用；
- 使用 foreach 循环语句遍历 Repeater1 中每一行数据的方法；
- 删除指定的记录。

例 5-10 的技术要点：修改表中的记录。

- UPDATE 命令的构造；
- 各个页面之间的跳转关系；
- 修改指定记录的字段的值。

2. 网站设计技术的说明

（1）网站设计可以使用的技术很多，本系统采用的只是其中之一。

（2）页面的外观也可以进行修饰和美化，例如页面的背景颜色，页面中字符的字体、字号等。

3. 模块功能的说明

本章的例题仅仅保证了模块基本功能的实现，只是实现了一个简易的管理系统，每个模块中的功能还可以进一步细化，列举如下：

- 每个用户的操作权限等；
- 每个输入数据的有效性检验；
- 模块之间的联系不紧密，例如在删除某个用户时，该用户开的药品信息是否同步被删除；
- 还可以再增加的功能，例如各个检查项目，每个医生的病历统计与分析，各科室就诊人数的统计等。

4. 优化问题

还有没有功能类似的一些程序段可以单独编写成过程或类。

以上问题都需要进一步的完善，距离实用还有很大的距离。因此，本系统只能作为学习网络编程的初步入门。

习　　题

1．基于 Web 的应用系统涉及的主要技术有哪些？

2．结合 HIS 系统说明数据库中主键的作用。

3．简要说明 HIS 系统中各张表之间的关系。

4．本章用到了哪些控件？各自的作用是什么？

5．区分应用程序中窗体、控件、属性、事件、代码和方法的概念。

6．Web 应用程序中如何实现网页之间的跳转？在跳转时如何实现数据的传递？

7．在 Web 应用程序中，如何使用表格进行页面的布局？与表格相关的 HTML 命令有哪些？

8．在 C#中访问数据库一般有几个步骤？每一步分别完成什么操作？

9．访问数据库使用的连接字符串由几部分组成？每部分的含义是什么？在 C#中如何获取连接字符串？

10．在 Web 应用程序中常用的验证控件有哪些？各自有什么作用？

11．说明母版页和内容页的作用和实现方法。

12．在 Web 应用程序中是如何实现导航的？