
第 26 章 投票模块设计

在一些网站的应用中，为了加强用户和网站之间的交互，常常开发投票模块让用户能够参与到网站的活动，网站还能够通过投票模块进行用户信息的统计和调查，使用投票模块能够更好的让网站与用户进行交互。

26.1 学习要点

投票模块需要涉及到一些 ASP.NET 3.5 的基本知识，如果要仔细学习投票模块的开发，需要详细了解本书的一些章节知识，这些章节如下所示：

- ☐ Web 窗体基本控件。
- ☐ 数据库基础。
- ☐ ADO.NET 常用对象。
- ☐ Web 窗体数据控件。
- ☐ ASP.NET 内置对象。
- ☐ 生成静态的概念
- ☐ 自定义控件和用户控件。

基本了解了以上章节的知识点后，就能够熟练学习和开发此模块。

26.2 系统设计

投票模块开发起来对技术的要求并不是很高，但是投票的表设计和样式呈现都是有技巧的。在网站应用中，有些应用就需要使用投票模块，例如网站信息统计和网站信息调查等，投票模块还能够进行热点调查等。

26.2.1 模块功能描述

投票模块能够加强用户与网站之间的交互，投票模块能够加强用户和网站信息之间的互动，网站可以使用投票功能进行网站内容的调查，例如调查用户是否满意网站的设计或者是否满意网站改版等等。同时投票模块还能够进行热点事件的调查，例如“你怎么看待 XX 事件”等等，都可以使用投票模块进行统计。

投票模块在设计上来说比较的简单，在后台的投票发布中，只需要进行相应的投票项目的发布即可，而在呈现时，需要调用多个表进行投票的呈现。例如在投票表的设计中，不能够将一个投票和选项设计在一起，那么一个投票有可能有单个选项或多个选项，不同的投票之间可能选项不同，如果将这些字段设计在一起，那么就会造成很大的数据浪费。投票模块的功能模块比较容易和简单，模块图如图 26-1 所示。

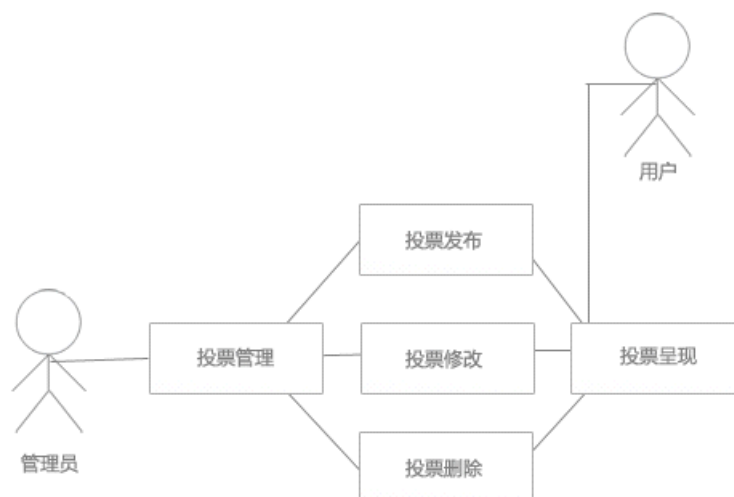


图 26-1 投票模块功能描述

正如图 26-1 所示，投票模块的功能大概可以描述为投票管理和投票呈现，管理员可以在后台进行投票管理，包括投票发布、投票修改和投票删除。投票发布或修改完毕后，用户就能够在前台进行相应的操作，包括投票和查看结果等。在前台的用户投票页面中，可以使用 ASP.NET 3.5 AJAX 进行无刷新操作实现用户的无刷新进行投票和结果统计。从上述模块功能描述中可以规划成以下几个页面和控件：

- ❑ 登录页面：管理员登录页面，为管理员提供身份验证。
- ❑ 后台框架集：用于管理员的管理操作。
- ❑ 投票发布页面：管理员用于投票的发布。
- ❑ 投票删除页面：管理员用于删除投票。
- ❑ 投票修改页面：管理员用于投票的修改。

其中登录页面用于制作后台的登录窗口，其作用同新闻模块中的登录一样。框架集用于制作后台管理界面，能够方便管理员在不同的页面和功能之间进行切换。

管理员首先需要进行身份验证，当身份验证通过后管理员就能够在后台进行投票的发布、删除和修改，管理员在后台进行投票操作后，用户能够在前台查看相应的投票并进行操作，投票控件可以用用户控件制作，这样能够为用户投票提供操作。而投票的查看可以使用自定义控件制作，使用自定义控件能够呈现更多的投票效果，这些效果包括统计、百分比等。

26.2.2 模块流程分析

虽然投票模块不是网站开发过程中最重要的模块，但是投票模块在网站开发过程中也非常的实用。现在可以看到在各个大型网站中都会有投票模块的存在，因为投票模块可以提高用于和网站之间的交互，也能够提高用户与用户之间的交互。

当出现了一个热门话题时，例如“您对番茄花园作者被抓有什么看法”时，使用新闻模块能够进行网站和用户之间的交流，但是却很难明显的看出用户的意愿，也无法统计用户的观点的信息。使用投票功能能够良好的解决这个问题，投票能够很好的进行统计，非常直观的呈现百分比，如图 26-2 所示。

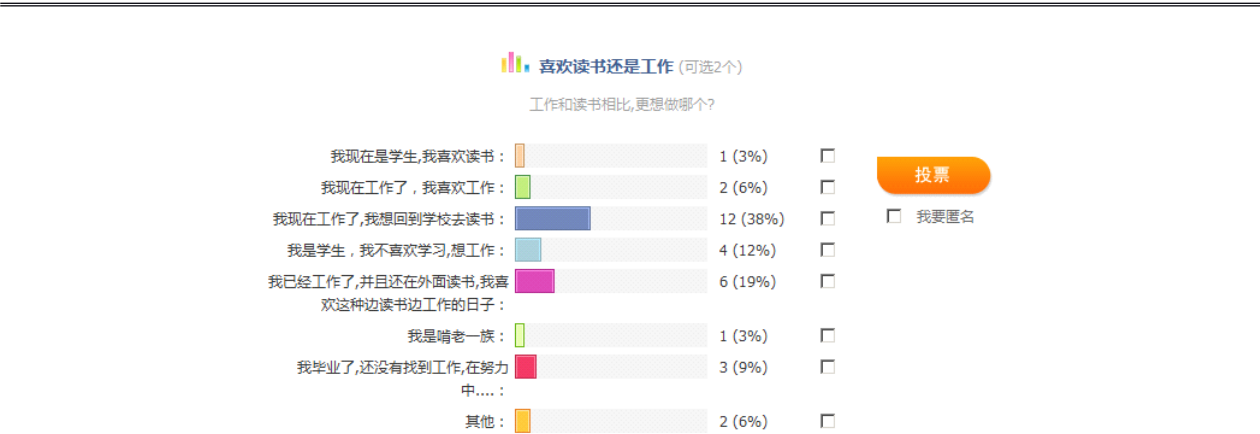


图 26-2 投票模块

从图 26-2 中可以看出不同的用户的不同的观点，也能够快速的统计出大流的用户管理，例如“喜欢读书还是工作”这个问题上，可以看出很多人选择了“我现在工作了,我想回到学校去读书”这一选项。在新闻模块中虽然能够通过评论来进行用户交互，但是很多时候的效果并没有投票好。投票可以快速的统计群体对于某观点的意向，但是在投票模块中，管理员和用户进行的操作流程都是不同的，投票模块相应的流程如图 26-3 和图 26-4 所示所示。

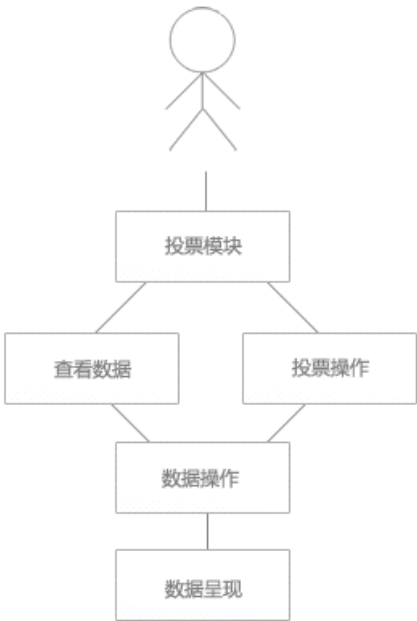


图 26-3 用户操作流程图



图 26-4 管理员操作流程图

对于用户而言，用户需要访问投票模块，在访问投票模块时可以查看投票数据然后进行投票操作，当用户进行了投票操作就会更新投票数据和相应的统计信息，系统在对数据进行重更新后再呈现在页面。而对于管理员而言，需要在后台进行投票发布和修改，在发布和修改后系统根据投票进行数据的初始化或统计并呈现在相应的页面中。

26.3 数据库设计

在投票模块的数据库设计中，需要多个表进行投票数据的描述。同样为了系统运行的安全，在后台同样需要登录操作，只有管理员进行登录后才能够发布投票。而对于用户，进行不同的操作时，还需要对数据库中的相应字段进行更新。

26.3.1 数据库设计

在投票模块中，需要多个表进行投票模块的描述，其中最重要的数据库就是投票问题表和投票选项表。投票问题表用于存放和投票相关的问题的数据，而投票选项表用于存放和投票选项相关的数据，在对投票模块和流程分析后，可以为几个表进行字段规划，其中投票表字段可以归纳如下。

- ❑ 投票编号：用于标识投票，为自动增长的主键。
- ❑ 投票标题：用于显示标题，作为投票模块的标题。
- ❑ 投票内容：用于解释投票信息的一些内容。

投票问题表的字段非常的少，其主要是用于索引，而投票选项表需要同投票问题表一起整合使用，一同描述一个投票模块。

- ❑ 投票选项编号：用于标识投票选项，为自动增长的主键。
- ❑ 投票选项统计：用于标识该投票被选择的次数。
- ❑ 投票描述：用于描述投票中的一个选项。
- ❑ 投票选项 ID：用于标识该投票选项是隶属于哪个投票问题，为投票问题表的外键。

管理员表用于管理员登录和验证操作，其作用同登录模块和新闻模块中的登录模块基本相同，其字段如下所示。

- ❑ 管理员编号：用于标识管理员信息，为自动增长的主键。
- ❑ 管理员用户名：用于标识管理员用户名。
- ❑ 管理员密码：用于标识管理员的密码，通常情况下和管理员用户名一起进行身份验证。

其中投票问题表和投票选项表一同描述投票模块，如图 26-5 所示。

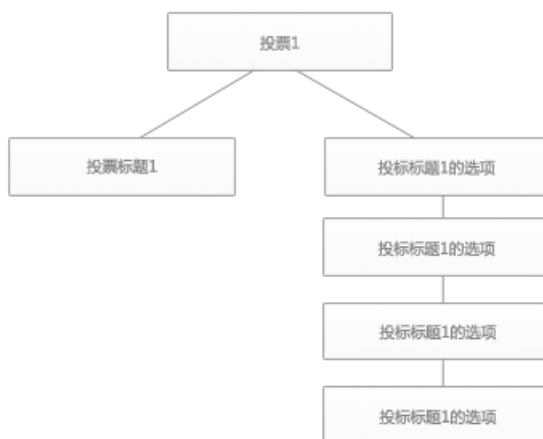


图 26-5 投票模块的形成

投票问题表和投票选项表一同描述投票模块，在呈现一个投票项目时，在投票问题表中只需要显示

投票的问题，而投票的选项则需要通过投票问题的标识（ID）号来进行筛选，在投票选项表中有一个投票选项 ID 字段，该字段就是用于标识这个选项是属于哪个投票问题的。只有将这两个表中的数据进行整合才能够完整的实现一个投票所需要的数据表。投票模块的数据设计并不复杂，而投票模块中将不同的表的数据进行呈现却比数据设计更加复杂的。

26.3.2 数据表的创建

创建表可以通过 SQL Server Management Studio 视图进行创建也可以通过 SQL Server Management Studio 查询使用 SQL 语句进行创建。在创建表之前首先需要创建数据库 post，在 post 数据库中只需要创建 3 个表就能够实现投票项目的描述，其中投票表（posttitle）的字段如图 26-6 所示。



	列名	数据类型	允许空
	id	int	<input type="checkbox"/>
	title	nvarchar(50)	<input checked="" type="checkbox"/>
	[content]	nvarchar(MAX)	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

图 26-6 投票表结构图

从投票表的结构图可以看出只需要创建三个字段就能够表述投票问题，这三个字段的意义如下所示。

- ❑ id: 用于标识投票，为自动增长的主键。
- ❑ title: 用于显示标题，作为投票模块的标题显示。
- ❑ content: 用于解释投票信息的一些内容。

创建表的 SQL 语句如下所示。

```
USE [post]
GO
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[posttitle](
    [id] [int] IDENTITY(1,1) NOT NULL,
    [title] [nvarchar](50) COLLATE Chinese_PRC_CI_AS NULL,
    [content] [nvarchar](max) COLLATE Chinese_PRC_CI_AS NULL,
    CONSTRAINT [PK_posttitle] PRIMARY KEY CLUSTERED
(
    [id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
//创建 posttitle 表
```

上述代码创建了投票表并创建了相应的字段，为了配合投票问题表，还需要创建投票选项表，投票选项表的字段如下所示。

- ❑ id: 用于标识投票选项，为自动增长的主键。
- ❑ hits: 用于标识该投票被选择的次数。
- ❑ content: 用于标识投票项及其描述。
- ❑ askid: 用于标识该投票选项是隶属于哪个投票问题，为投票问题表的外键。

这里值得注意的是，选项表中的 askid 字段同投票表一同描述投票项，投票选项表创建的 SQL 语句

如下所示。

```
USE [post]
GO
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[postchoose](                                //创建 postchoose 表
    [id] [int] IDENTITY(1,1) NOT NULL,
    [hits] [int] NULL,
    [content] [nvarchar](max) COLLATE Chinese_PRC_CI_AS NULL,
    [askid] [int] NULL,
    CONSTRAINT [PK_postchoose] PRIMARY KEY CLUSTERED
(
    [id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
    ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
```

上述代码创建了 postchoose 表用于描述投票的选项，其中 askid 为投票问题表的 id，该键为外键，用于筛选和整合两个表中的数据。在投票的发布中，为了系统的安全性，还需要创建系统表用于管理员登录，admin 表只需要保存管理员的用户名和密码即可，则其字段可以描述为如下所示。

- ❑ id: 用于标识管理员信息，为自动增长的主键。
- ❑ admin: 用于标识管理员用户名。
- ❑ password: 用于标识管理员的密码，通常情况下和管理员用户名一起进行身份验证。

上述字段描述了 admin 表中需要使用的字段，可以使用 SQL 语句进行表和字段的创建，创建 newsclass 表的 SQL 语句如下所示。

```
USE [post]
GO
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[admin](                                    //创建 admin 表
    [id] [int] IDENTITY(1,1) NOT NULL,
    [admin] [nvarchar](50) COLLATE Chinese_PRC_CI_AS NULL,
    [password] [nvarchar](50) COLLATE Chinese_PRC_CI_AS NULL,
    CONSTRAINT [PK_admin] PRIMARY KEY CLUSTERED
(
    [id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
    ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
```

上述代码创建了 admin 表，用于进行管理员的身份验证，在后台的管理员登录中需要使用到该表，在表创建完成后，需要向数据库中添加管理员，添加管理员代码如下所示。

```
INSERT INTO admin (admin,password) VALUES ('guojing','0123456')
```

执行上述代码就能够进行 admin 表的数据插入，插入一个新管理员之后，就能够在后面的登录操作中使用该表的管理员信息进行登录和投票操作。

26.4 界面设计

投票模块包括众多的页面，这些页面包括发布投票页面、修改投票页面和删除投票页面，这其中还包括投票管理页面。在投票模块中，不适用进行 ASP.NET 自带的控件进行操作，在投票模块的页面设计中需要自行开发自定义页面。

26.4.1 后台框架集

在 Microsoft Expression Web 2 中，选择菜单栏中的【文件】选项，在下拉菜单中选择【新建】选项，单击【新建】选项中的【网站】选项创建网站，在弹出窗口中选择框架集，如图 26-7 所示。

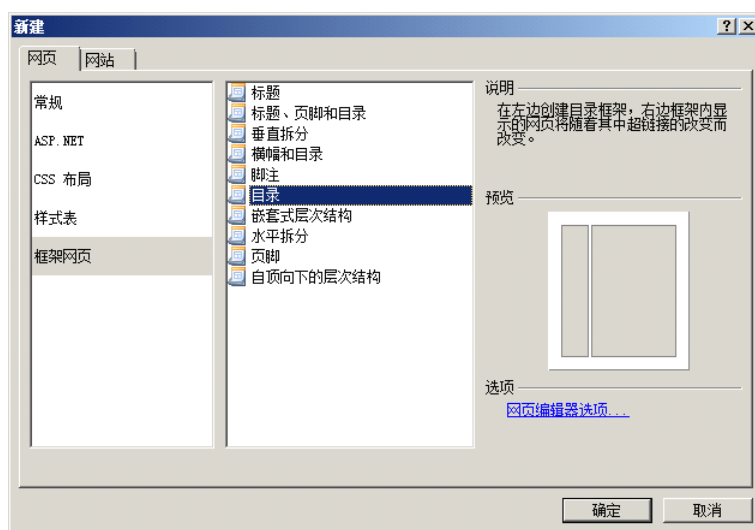


图 26-7 创建框架集

框架集可以将多个页面放置在同一个页面，在 Microsoft Expression Web 2 中可以创建框架集并为框架集中的页面进行指定或新建，这里可以创建一个目录类型的框架集，创建后框架集代码见光盘中源代码\第 26 章\26-1\26-1\admin\default.aspx。

开发人员可以在框架集中创建网页或选择设置初始网页，这里创建两个网页，一个用于显示导航，此页面为 sidebar.aspx，管理员能够在该页面进行导航处理。另一个则是主工作区，用于管理员操作，导航栏初始化代码见光盘中源代码\第 26 章\26-1\26-1\admin\sidebar.aspx。

投票管理的操作并不多，所以在后台中管理员导航的节点也并不多，管理员在后台主要执行投票管理和添加投票两个操作，投票管理操作包括了投票的修改和删除。

26.4.2 投票管理页面

投票管理页面的数据显示可以使用 ASP.NET 3.5 提供的 GridView 控件，由于投票模块其功能的复杂性，这里并不能使用 ASP.NET 3.5 自带的控件进行修改，所以在投票管理页面只能使用 GridView 控件进行数据罗列，并添加相应的超链接，示例代码见光盘中源代码\第 26 章\26-1\26-1\admin\manage.aspx。

在管理页面中，其代码使用了 GridView 控件进行数据罗列，但在 GridView 控件中添加的是两个数

据绑定的超链接分别用于实现修改和删除功能，如图 26-8 和图 26-9 所示。

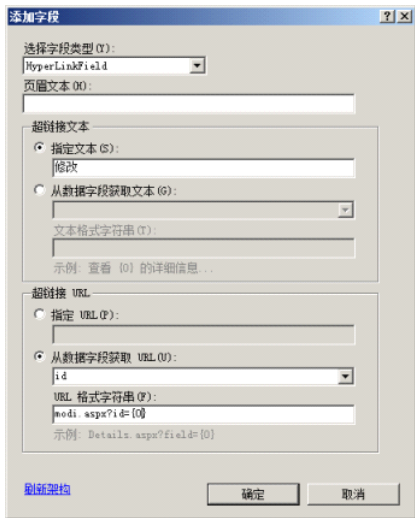


图 26-8 添加修改超链接

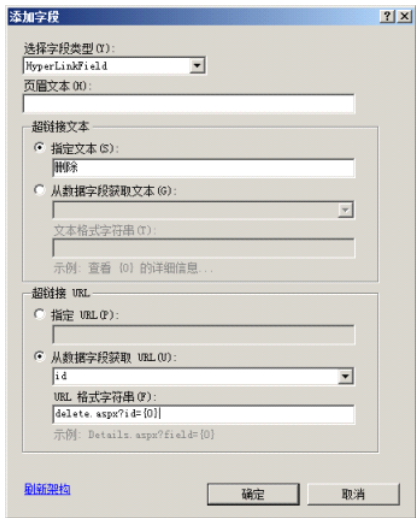


图 26-9 添加删除超链接

可以看出修改和删除超链接都会通过参数传递跳转到另一个页面，其中修改超链接跳转的是 `modi.aspx?id=编号`，而删除超链接跳转的是 `delete.aspx?id=编号`。通过参数的传递，开发人员能够在相应的页面进行业务逻辑处理。

注意：这种情况很像 ASP 的开发过程，为了能够让开发人员更好的理解控件的制作，可以使用类似 ASP 的开发流程进行过程开发。

由于页面只需要进行数据的呈现，所以数据源并不需要支持数据操作，数据源代码如下所示。

```
<asp:SqlDataSource ID="SqlDataSource1" runat="server"
    ConnectionString="<%%$ ConnectionStrings:postConnectionString %>"
    SelectCommand="SELECT * FROM [posttitle] ORDER BY [id] DESC">
</asp:SqlDataSource>
```

从上述代码可以看出，数据源中并没有自动支持插入、更新、删除等数据操作，因为对于投票管理页面的 GridView 控件而言，其作用只是用于数据的呈现，而无需进行数据操作。

26.4.3 投票发布页面

投票发布页面用于管理员的投票发布，管理员可以在投票发布页面进行投票信息的填写，在投票发布页面，管理员需要填写投票信息和投票选项信息，这就涉及到多个表的数据操作。在管理员填写投票选项信息时，需要向投票选项表中重复插入数据，因为投票选项往往是多项，而投票问题只是一项。投票发布页面涉及到两个表和程序筛选，将在后面的章节中讲到，而投票发布页面的设计只需要进行基础控件布局即可，示例代码见光盘源代码\第 26 章\26-1\26-1\admin\post.aspx。

投票代码使用了 TextBox 控件用于管理员的信息输入，管理员能够在相应的控件中填写投票信息并通过按钮控件提交数据。在投票发布页面中，投票选项同样使用的是 TextBox 控件。再次回到流程分析中，用户在前台进行投票。投票同样包含多个选项，有的投票包含 1 个或 3 个选项，有的投票包含 2 个或 4 个选项，那么在投票功能中就不能规定死投票选项的个数。

在投票发布时，管理员需要按照投票的选项个数进行投票发布，同时在投票中又需要降低数据库的使用率，这里就需要智能筛选数据。这里使用的是回车筛选，即一个回车就是一个选项。为投票功能设计页面以便管理员添加投票项目，页面设计后如图 26-10 所示。

图 26-10 投票页面设计

26.4.4 投票修改页面

投票修改页面同投票发布页面相同，在投票页面被加载时，投票的基本信息需要载入并存储在投票修改页面中的控件里，当管理员单击【修改投票】按钮时，就能够进行数据筛选和修改，投票修改页面示例代码见光盘中源代码\第 26 章\26-1\26-1\admin\modi.aspx。

投票修改页面在加载时接受传递过来的参数，使用传递的参数获取数据库中投票的相应信息进行页面中控件的文本填充。管理员可以通过修改页面中相应的信息进行数据更改，当更改完毕后管理员可以进行数据操作更新相应的数据选项。

26.4.5 投票删除页面

投票删除页面可以不进行页面布局的处理，因为投票删除页面主要作用为删除数据。当管理员在投票管理页面进行投票删除选择时，会跳转到投票删除页面，投票删除页面通过获取传递过来的参数进行投票的删除，删除完毕后再次返回投票管理页面，所以在投票删除页面中只需要进行事务的处理而不需要进行页面布局和控件处理。

26.5 代码实现

投票模块的页面整体设计比较简单，但是投票功能的实现还需要掌握一些难点。投票模块的难点就在于一个投票的问题和多个投票选项是如何整合，并且在发布投票时如何进行多项投票发布，在修改投票时同样修改选项和筛选选项都是一个难题。

26.5.1 添加投票代码实现

添加投票时，管理员在投票选项中可以用回车进行分割，系统能够使用回车进行多个选项筛选和数据插入。在代码实现中，首先要将投票选项控件中的文本进行分割，这里使用回车进行分割，每个回车就会创建一个新的投票项。添加投票的具体实现思路为：

- ❑ 管理员在投票控件中按回车进行多个选项添加。
- ❑ 管理员单击按钮控件进行数据添加。
- ❑ 数据添加时按照回车进行选项分割，当有多个回车时说明有多个选项。
- ❑ 添加投票问题表，添加完成后获取刚刚添加的投票的编号。
- ❑ 循环添加投票选项，并为选项添加相应的 ID。

从上述实现思路可以编写相应的代码，示例代码如下所示。

```
protected void Button1_Click(object sender, EventArgs e)
{
    SqlConnection con = new SqlConnection("Data Source=(local);
        Initial Catalog=post;Integrated Security=True");           //创建连接
    con.Open();                                                     //打开连接
    string trim = TextBox3.Text.Replace("\n", "|");                 //回车替换成字符
    string[] count = trim.Split("|");                               //替换后分拆
    string strsql = "insert into posttitle (title,content) values ('"+TextBox1.Text+"','"+TextBox2.Text+"')
        SELECT @@IDENTITY as 'bh' ";                               //编写 SQL 语句
    SqlCommand cmd = new SqlCommand(strsql, con);                  //创建执行对象
    int id=Convert.ToInt32(cmd.ExecuteScalar());                    //执行数据插入
    for (int i = 0; i < count.Length; i++)                          //循环添加
    {
        string contentinsert = "insert into postchoose (hits,content,askid) values
            ('0','"+ count[i].ToString() + "','"+ id + "')";       //遍历插入项目
        SqlCommand command = new SqlCommand(contentinsert, con);   //创建执行对象
        command.ExecuteNonQuery();                                  //执行遍历
    }
    con.Close();                                                    //关闭连接
}
```

上述代码中将管理员输入的投票选项通过回车分割，添加选项方法如图 26-11 所示。

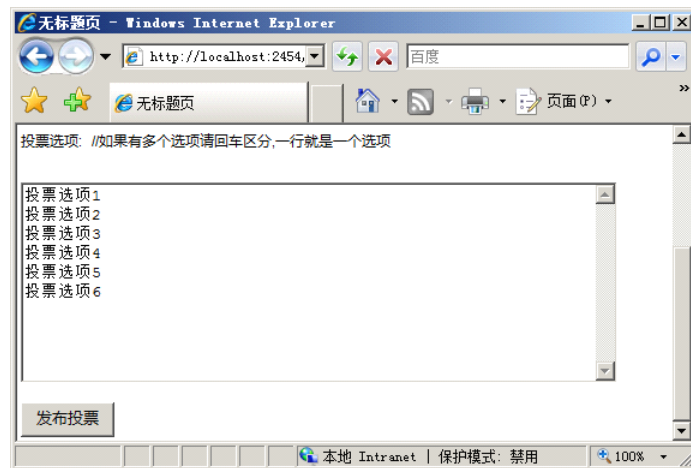


图 26-11 输入投票选项

管理员输入投票选项时，可以以回车的形式进行投票选项的分割，如图 26-11 所示，其中管理员添加了 6 个选项，在数据插入前，首先需要将不同的投票选项之间分割，示例代码如下所示。

```
string trim = TextBox3.Text.Replace("\n", "|");                 //替换字符
string[] count = trim.Split("|");                               //分割字符串
```

上述代码将回车字符串替换成为“|”符号，然后通过 Split 将字符串进行分割，在图 26-11 中输入

的字符串会被分割为“投票选项 1|投票选项 2|投票选项 3..”，Split 函数能够将这个字符串进行分割并放置在数组中，如“count[0]=投票选项 1，count[1]=投票选项 2”，这样就能够通过数组循环进行数据插入，示例代码如下所示。

```
for (int i = 0; i < count.Length; i++) //遍历执行
{
    string contentinsert =
        "insert into postchoose (hits,content,askid) values ('0','" + count[i].ToString() + "','" + id + "')";
    SqlCommand command = new SqlCommand(contentinsert, con); //SQL 语句
    command.ExecuteNonQuery(); //执行 SQL
}
```

使用 ADO.NET 中数据操作的 ExecuteScalar 方法和 SQL 语句的“SELECT @@IDENTITY as”语法能够返回刚才数据插入的值，在进行选项遍历插入之前，首先需要得到刚才插入的投票问题的编号。使用 ExecuteScalar 方法能够获取刚刚插入的数据的编号，获取后就需要在遍历操作中为相应的选项指定其问题的 ID 号。执行一次操作后其表之间的关系如图 26-12 和图 26-13 所示。

	id	title	content
▶	1	投票1	投票简介
✱	NULL	NULL	NULL

图 26-12 投票问题表

	id	hits	content	askid
	1	0	投票选项1	1
	2	0	投票选项2	1
	3	0	投票选项3	1
	4	0	投票选项4	1
	5	0	投票选项5	1
▶	6	0	投票选项6	1
✱	NULL	NULL	NULL	NULL

图 26-13 投票选项表

从图 26-12，26-13 可以看出投票问题表和投票选项表之间一起描述了一个投票项目，通过回车的方式能够使不同的投票项目中的投票问题和投票选项之间分离开，这样投票项目与投票项目之间就没有选项等约束，一个投票可以有多个选项，不同的投票之间投票选项个数可以不同，其统计和描述也可以不同。但是这样进行数据库设计同样也会有一些缺点，其缺点就是在投票选项表中可能会占用过多的数据。

26.5.2 修改投票代码实现

在修改投票代码页面载入时，首先同样需要和投票插入一样进行数据筛选。一个投票项目需要两个数据表进行描述，在页面加载时需要加载多个表中的数据，示例代码如下所示。

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack) //判断是否第一次加载
    {
        TextBox3.Text = ""; //清空控件值
        Label1.Text = ""; //清空控件值
        SqlConnection con = new SqlConnection("Data Source=(local);
            Initial Catalog=post;Integrated Security=True"); //创建连接
        con.Open(); //打开连接
        string strsql = "select * from posttitle where id='" + Request.QueryString["id"] + "'";
        SqlDataAdapter da = new SqlDataAdapter(strsql, con);
        DataSet ds = new DataSet(); //创建数据集
        da.Fill(ds, "table"); //填充投票数据
        string strchoose = "select * from postchoose where askid='" + Request.QueryString["id"] + "'";
        SqlDataAdapter ch = new SqlDataAdapter(strchoose, con);
```

```

int count = ch.Fill(ds, "choosetable"); //填充投票选项
TextBox1.Text = ds.Tables["table"].Rows[0]["title"].ToString(); //填充控件
TextBox2.Text = ds.Tables["table"].Rows[0]["content"].ToString(); //填充控件
for (int i = 0; i < count; i++) //循环获取数据
{
    if (i == count - 1) //判断是否循环完
    {
        TextBox3.Text += ds.Tables["choosetable"].Rows[i]["content"].ToString().Replace("\r", "");
        Label1.Text += ds.Tables["choosetable"].Rows[i]["id"].ToString(); //填充数据
    }
    else
    {
        TextBox3.Text += ds.Tables["choosetable"].Rows[i]["content"].ToString() + "\n";
        Label1.Text += ds.Tables["choosetable"].Rows[i]["id"].ToString() + "|"; //填充数据
    }
}
con.Close(); //关闭连接
}

```

上述代码实现了当页面加载时其相应的数据被加载的相应的控件的功能，其中比较复杂的就在于两个数据表中的数据如何整合在一起。在投票选项中，其选项需要整合在一起放置在控件中，同时还需要像添加投票一样一行一个投票项目。

不仅如此，还需要将这些投票项目的编号进行统计，如果不进行统计则不能够使用编号进行相应的数据字段的更新，只有保存相应的选项的数据的编号才能够循环更新。在添加投票时，将选项与选项之间进行回车分割，在载入页面时，同样需要将数据呈现为选项与选项之间的回车形式，示例代码如下所示。

```

for (int i = 0; i < count; i++) //循环获取数据
{
    if (i == count - 1) //判断是否循环完
    {
        TextBox3.Text += ds.Tables["choosetable"].Rows[i]["content"].ToString().Replace("\r", "");
        Label1.Text += ds.Tables["choosetable"].Rows[i]["id"].ToString(); //填充数据
    }
    else
    {
        TextBox3.Text += ds.Tables["choosetable"].Rows[i]["content"].ToString() + "\n";
        Label1.Text += ds.Tables["choosetable"].Rows[i]["id"].ToString() + "|"; //填充数据
    }
}

```

上述代码将投票选项的数据和相应的编号进行填充，在进行更新时同样可以使用控件中的数据进行数据分割操作然后进行数据更新，更新代码如下所示。

```

protected void Button1_Click(object sender, EventArgs e)
{
    SqlConnection con = new SqlConnection("Data Source=(local);
        Initial Catalog=post;Integrated Security=True"); //创建连接
    con.Open(); //打开连接
    string trim = TextBox3.Text.Replace("\n", "|"); //替换字符
    string[] count = trim.Split('|'); //分割字符
}

```

```

string[] count2 = Label1.Text.Split('|');           //计算分割
if (count != count2)                               //判断项数
{
    Label2.Text = "修改不能修改投票项数";
}
else
{
    string strsql = "update posttitle set title=" + TextBox1.Text + ",content=" + TextBox2.Text + "
                    where bh=" + Request.QueryString["id"] + ""';           //编写更新语句
    SqlCommand cmd = new SqlCommand(strsql, con);           //创建执行对象
    cmd.ExecuteNonQuery();           //执行对象
    for (int i = 0; i < count.Length; i++)           //遍历更新
    {
        strsql = "update postchoose set content=" + count[i].ToString() + " where id=" +
                count2[i].ToString() + ""';           //生成 SQL 语句
        SqlCommand cmd1 = new SqlCommand(strsql, con);           //更新数据
        cmd1.ExecuteNonQuery();           //执行更新
    }
    con.Close();           //关闭连接
}
}

```

在 Page_Load 代码中，将选项的编号都存放到 Label 控件中，在执行更新时，需要分别循环遍历选项控件和 Label 控件进行数据更新。在数据更新中，并不能修改投票的项数，这也就是说在发布投票时有多少选项则在修改时只能修改相应的选项而不能增加选项或删除选项。

26.5.3 删除投票代码实现

在投票项目删除时，并不能像前面的模块一样只对模块问题项目进行删除。在删除模块问题表时，同样需要删除投票的选项，这样就能够保证数据库的完整性。因为如果对投票的问题进行删除而不对投票的选项进行删除的话，会造成大部分的垃圾数据，同样也会影响到程序的性能。删除投票通过传递的参数进行两个表中相应的数据的删除，示例代码如下所示。

```

protected void Page_Load(object sender, EventArgs e)
{
    SqlConnection con = new SqlConnection("Data Source=(local);
        Initial Catalog=post;Integrated Security=True");           //创建连接
    con.Open();           //打开连接
    string deletetitle = "delete posttitle where id=" + Request.QueryString["id"] + ""';           //删除
    string deletechoose = "delete postchoose where askid=" + Request.QueryString["id"] + ""';           //删除
    SqlCommand cmd = new SqlCommand(deletetitle, con);
    cmd.ExecuteNonQuery();           //执行删除
    SqlCommand cmd1 = new SqlCommand(deletechoose, con);
    cmd1.ExecuteNonQuery();           //执行删除
    Response.Redirect("manage.aspx");           //页面跳转
}

```

上述代码分别删除了两个表中的数据，在进行删除操作时首先需要对数据库中的投票问题表中的数据删除，删除后还需要删除与投票问题表相关的投票选项，其中投票选项中的 askid 字段用于标识选项所对应的问题，只有执行了两个表中相应的数据的删除才能够真正删除一个投票项目。

26.5.4 显示投票代码实现

当管理员发布代码后,就需要进行代码的显示,代码显示同样需要获取两个表中相应的投票项的信息,在显示过程,还需要对投票中的数据进行显示和图表设计。

在投票显示之前,首先需要遍历投票的选项并且计算投票选项的总和,计算完成投票的总和后,再分别计算每个选项被用户选择的次数。例如有一个投票项目名为“你怎么看待上大学”,其中选项有“上学好,比工作轻松”和“上学没自由”等等选项,首先就需要统计所有的选数的总和,假设是 100 票,然后再统计选择“上学好,比工作轻松”的票数,进行相应的统计。在统计时,需要遍历数据库中的数据进行计算,计算完成后可以通过 HTML 呈现给用户,示例代码如下所示。

```
protected void Page_Load(object sender, EventArgs e)
{
    SqlConnection con = new SqlConnection("Data Source=(local);
                                           Initial Catalog=post;Integrated Security=True"); //创建连接
    con.Open(); //打开连接
    string title; //配置字段
    string id = Request.QueryString["id"]; //获取参数
    string width = "500px"; //设置宽度
    string str = "select * from posttitle where id=" + id + ""; //执行查询
    SqlDataAdapter da = new SqlDataAdapter(str, con); //创建适配器
    DataSet ds = new DataSet(); //创建数据集
    da.Fill(ds, "table"); //填充数据集
    string str2 = "select * from postchoose where askid=" + id + ""; //执行查询
    SqlDataAdapter da2 = new SqlDataAdapter(str2, con); //创建适配器
    DataSet ds2 = new DataSet(); //创建数据集
    da2.Fill(ds2, "table2"); //填充数据集
    title = ds.Tables["table"].Rows[0]["title"].ToString(); //获取标题
    int sum = 0; //获取总和
    for (int i = 0; i < ds2.Tables["table2"].Rows.Count; i++) //遍历集合
    {
        sum += Convert.ToInt32(ds2.Tables["table2"].Rows[i]["hits"].ToString()); //求和
    }
    Response.Write("<table style='width:" + width + "><tr><td style='border: #4a95c9
                    1px solid;background-color: #b7d8ed;padding:5px 5px 5px 5px;>"); //输出 HTML
    Response.Write("<strong>" + title + "</strong><br />"); //输出 HTML
    for (int i = 0; i < ds2.Tables["table2"].Rows.Count; i++) //遍历项目
    {
        Response.Write("<br><span style='font-size:12px;'>" + (i + 1) +
            ".<a href='newvote.aspx?id=" + ds2.Tables["table2"].Rows[i]["id"].ToString()
            + "&askid=" + id.ToString() + "#vote' target='_blank'>" +
            ds2.Tables["table2"].Rows[i]["content"].ToString().Replace("\n", "").Replace("\r", "") +
            "</a>票数:" + ds2.Tables["table2"].Rows[i]["hits"].ToString() + "百分比:"); //输出 HTML
        if (sum != 0) //总和不为 0
        {
            Response.Write(Convert.ToSingle(ds2.Tables["table2"].Rows[i]["hits"].ToString())
                / Convert.ToSingle(sum) * 100 + "%"); //求百分比
        }
        else
        {

```



```

        Response.Write("0%"); //输出 0%
    }
    Response.Write("</span><br><br> <div style='height:10px; background-color:White;
border:1px solid #ccc;'><div style='height:10px; background-color:gray; width:"); //输出边框
    if (sum != 0) //总和不为 0
    {
        Response.Write(Convert.ToInt32(Convert.ToSingle(ds2.Tables["table2"].Rows[i]["hits"].ToString())
        / Convert.ToSingle(sum) * 100) + "%"); //输出百分比
    }
    else
    {
        Response.Write("0"); //输出票数
    }
    Response.Write("</div></div>"); //输出 div
}
Response.Write("<br /><strong>总票数</strong>: " + sum + ""); //输出总票数
Response.Write("</td></tr></table>");
}

```

当用户通过访问 `vote.aspx` 页面时，就能够查看到相应的投票信息，投票信息页面中包括投票名称、投票选项、票数和百分比，这些信息能够方便的让投票者或用户了解到大众的相关信息。上述代码首先读取投票的基本信息，包括投票的标题和投票的描述，读取完毕后再进行投票选项的遍历，遍历过后进行票数的统计和百分比的计算，然后再通过 HTML 代码呈现在页面中。

26.5.5 用户投票代码实现

用户可以看见投票项目和投票统计，同样用户也应该能够进行投票，当用户投票之后就需要记录用户对投票进行的操作，才能够让用户不能重复投票。如果用户重复投票就会造成投票的不真实性，从显示的投票页面可以看出，投票页面使用了参数进行另一个页面的传递，示例代码如下所示。

```

Response.Write("<br><span style='font-size:12px;'>" + (i + 1) +
    ".<a href='newvote.aspx?id=" + ds2.Tables["table2"].Rows[i]["id"].ToString() + "&askid="
    + id.ToString() + "#vote' target='_blank'>" +
    ds2.Tables["table2"].Rows[i]["content"].ToString().Replace("\n", "").Replace("\r", "") + "</a>票数:"
    + ds2.Tables["table2"].Rows[i]["hits"].ToString() + "百分比:");

```

从上述代码中可以看出，当用户单击项目进行投票时，系统会通过参数 `id` 和 `askid` 进行传递跳转到另一个 `newvote.aspx` 页面并在 `newvote.aspx` 页面进行逻辑处理。逻辑处理完毕后再跳转回该页面，`newvote.aspx` 页面代码如下所示。

```

protected void Page_Load(object sender, EventArgs e)
{
    try
    {
        if (Request.Cookies[Request.QueryString["id"]] == null) //判断是否已经投票
        {
            SqlConnection con =
            new SqlConnection("Data Source=(local);Initial Catalog=post;Integrated Security=True");
            con.Open(); //打开连接
            string askid = Request.QueryString["askid"]; //获取 askid
            string id = Request.QueryString["id"]; //获取 id

```



```

string str = "select * from postchoose where id='" + id + "'";           //生成 SQL 语句
SqlDataAdapter da = new SqlDataAdapter(str, con);                       //创建适配器
DataSet ds = new DataSet();                                           //创建数据集
da.Fill(ds, "table");                                                 //填充数据集
int hits = Convert.ToInt32(ds.Tables["table"].Rows[0]["hits"].ToString()); //获取点数
string strsql = "update postchoose set hits='" + (hits + 1) + "' where id='" + id + "'"; //增加点
击

SqlCommand cmd = new SqlCommand(strsql, con);                         //创建执行对象
cmd.ExecuteNonQuery();                                                //更新数据操作
HttpCookie cookies_votenum = new HttpCookie(id);                     //创建 Cookie
cookies_votenum.Value = id;                                           //设置 Cookie 值
cookies_votenum.Expires = DateTime.Now.AddDays(1);                   //设置持续事件
Response.AppendCookie(cookies_votenum);                               //添加 Cookie
Response.Redirect("vote.aspx?id=" + askid + "");                       //跳转投票
}
else
{
    Label1.Text = "您已经参与投票";                                   //输出用户信息
}
}
catch(Exception ee)
{
    Response.Write(ee.ToString());                                     //抛出异常
}
}

```

上述代码通过传递的参数进行数据更新，该页面进行投票的事务处理，当用户单击投票系统中的项目时，系统会跳转到该页面进行数据更新并跳转回相应的投票项目，返回后用户就能够查看相应的数据信息。

当用户投票后，系统会为用户提供一个 Cookie 对象，用户再次执行投票后首先会检查这个 Cookie 对象，如果存在 Cookie 对象说明用户已经投过票，则不允许用户再次投票，如果没有 Cookie 对象则用户能够进行投票操作。

26.6 实例演示

当使用投票模块进行网站和用户之间的交互时，管理员能够在后台发布投票信息并在前台进行投票信息的呈现，用户能够在前台投票页面中进行投票和数据查看。投票能够清晰明显的显示项目，让用户能够较直观的获取信息。管理员可以在后台页面中进行投票发布，当投票项有多个选项时，管理员可以用回车进行分割，如图 26-14 所示。

管理员添加了一个“你喜欢本书吗”的调查，并通过回车创建了三个选项，这三个选项分别为“喜欢,本书言简意赅”，“还可以,就是知识不够广泛”和“不喜欢,写的太烂了”，创建后这三个选项都会被添加到相应的表中。单击按钮控件，管理员能够创建投票，管理员可以在后台的投票管理页面中进行投票信息的管理，如图 26-15 所示。

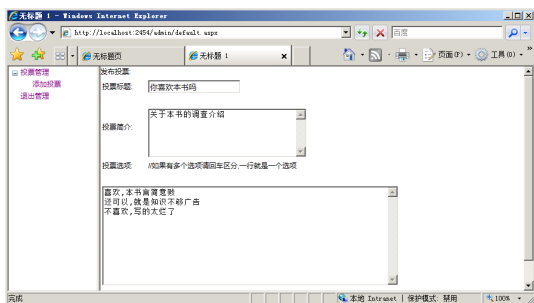


图 26-14 添加投票信息



图 26-15 投票管理

管理员能够在投票管理页面进行投票的修改和删除，修改投票可以修改其中的选项、投票标题和简介。如果管理员发布的投票信息并没有什么错误的话，可以无需修改投票信息就可以进行页面呈现，这里可以访问 `vote.aspx?id=相应 ID` 页面进行投票信息的访问，如图 26-16 所示。

用户能够访问页面进行投票信息的查看并进行投票，当用户对一个项目进行投票后，就不能够再对某个项目进行投票，如果用户投票后，系统会给用户一个 Cookie 对象，当用户投票前首先会检查 Cookie 对象，如果不存在 Cookie 对象则允许用户进行投票。投票后的页面如图 26-17 所示。

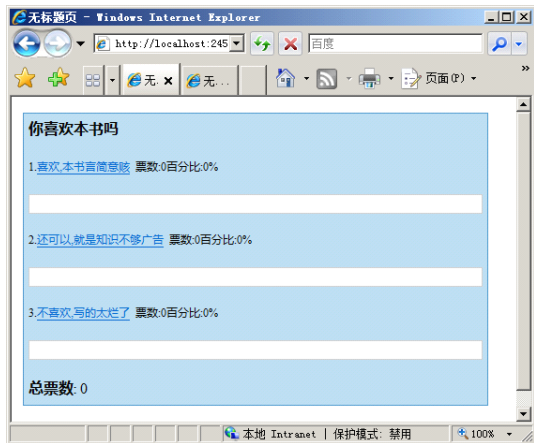


图 26-16 投票信息

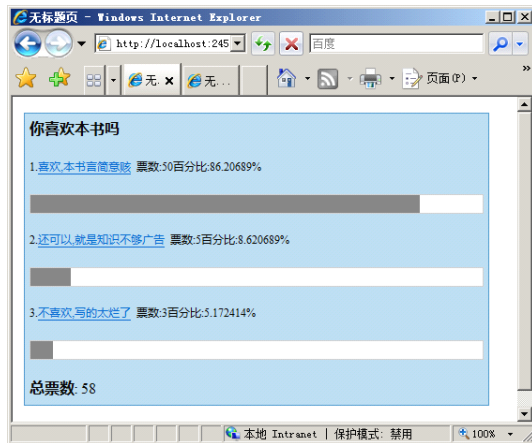


图 26-17 投票统计

当用户进行投票后，投票页面会统计用户投票的信息，并以图表的形式呈现给用户，用户能够非常直观的查看到其他用户的投票信息，例如这里“喜欢,本书言简意赅”选项能够一眼就看出选择这个用户的人数比较多，这样就能够非常清除的了解“你喜欢本书吗”这个问题的结果。

26.7 小结

本章通过投票模块的编写了解了数据库的设计，以及基本的投票设计开发，在投票模块的开发过程中，使用了两个数据库进行投票项目的描述，当对投票模块的数据插入、更新和删除时，都需要考虑到数据的完整性。本章还巩固了：

- ❑ Web 窗体基本控件。
- ❑ 数据库基础。
- ❑ ADO.NET 常用对象。
- ❑ Web 窗体数据控件。

-
- ❑ ASP.NET 内置对象。
 - ❑ 生成静态的概念
 - ❑ 自定义控件和用户控件。

开发人员能够将投票页面更改成为 **JavaScript**，这样就能在不同的页面进行调用，投票是网站开发中一个比较常用的功能，现在的很多的大型网站都包含投票模块用于与用户进行信息交互。

本章还包括了管理员登录等页面，由于前面的章节中详细的讲解了管理员登录的开发方式和登录控件的开发，这里就不再详细的介绍，管理员登录等基本功能在后台开发中是非常重要的，只要在后台包含管理员登录操作，就需要开发验证或后台登录页面。