北京信息科技大学

毕业设计(论文)

题　　目： 民航订票管理系统

学　　院： 计算机学院

专　　业： 计算机科学与技术

学生姓名： **班级/学号**

指导老师/督导老师： 李艳平

起止时间： 2009年2月23日 至2009年6月19日

摘　要

民航订票管理系统的开发主要包括后台数据库的建立和维护以及前台界面的开发两个方面。民航订票管理系统可以实现用户注册、登录，航班信息查询、订票和退票，用户管理和航班管理等功能。为了使最终设计的网站更加贴近实际，本系统参考了网上一些专业的飞机订票网站的基本框架设计完成。设计该系统的目的是即使你是一个不懂计算机专业知识的人士也可以很轻松地管理网站，同时用户可以快捷、方便的订票，使得飞机订票公司以较少的人力取得更多的效益。

该系统使用B/S结构,由前台和后台管理两个部分组成。前台作为与用户直接交流、联系的可视化界面，将系统的各个功能提供给用户，后台管理主要由数据库系统构成，完成对各个数据库表单的增、删、改操作。系统采用ASP.NET和C#作为前台开发工具,SQL Server 2000为后台数据库管理系统,其数据库连接主要运用了ASP.NET中的ADO.NET控件。在论文的最后，讲述了我在软件编码过程中遇到的问题及解决方法。

关键词： 航班；订票； 管理； ASP.NET

Abstract

The exploitation of the management system of booking flights mainly includes two aspects: The background database's establishment and maintenance and the foreground interface designment. The management system of booking flights can carry out functions of register and login of users, inquiring about the flights information , booking and canceling tickets, the management of users and the management of flights. In order to make the final designing website more truthful, the system has come true with consulting the basic frame of some professional website of booking flights. The purpose of designing the system is that even if you are a person who doesn't know computer professional knowledge entirely , you can manage the website easily. At the same time users can book flights rapidly and conveniently. The company of booking flights can engage less employees and acquire more benefit.

The system uses B/S structure and is composed of foreground management and background management. Foreground as a visualization interface to communicate with users, provides various functions to users, Background makes up of database management system and completes the addition, deletion and alteration of database tables. The system uses C # and ASP.NET as the foreground development tools, SQL Server 2000 database as the background database management system. The connection of database uses ADO.NET Controls in ASP.NET .In the end, the thesis narrates the problems while I am compiling codes and the ways of solving the problems.

Keywords: flight; booking flights; manage ; ASP.NET

目录

第1章　概述

1.1 背景分析

随着科学技术的迅猛发展和计算机成本的不断降低，计算机早已像电视、洗衣机、电冰箱一样变成了千家万户的家庭必备用品。人们喜欢在闲时上上网，看看网络小说，浏览浏览新闻，或是跟素未蒙面的网友聊聊天，网络将人们之间的距离拉的越来越近了，世界俨然已是一个地球村了。上网已经成为很多人享受生活，放松心情的一种方式。我国的网民数量在2007年已达1.62亿，因此给互联网带来了巨大的商机，同时也促使我国网络产业需要更好更快的向前发展。从而电子商务这种以网络作为媒介的应用方式这些年得到了越来越多人的认可，取得了飞速的发展，电子商务网站以雨后春笋般的速度出现在互联网上。

电子商务网站不同于以往的商贸活动，具有开放性、全球性、低成本、高效率的特点。可以毫不夸张地说，电子商务的出现颠覆了以往人们对传统商贸活动的思维定势，使商贸活动有了一个质的飞跃。在以前你可能很难想象一个盲人可以开店卖东西，依靠自己的力量养活自己，但是在互联网时代的今天，一个盲人也可以借助语音识别技术实现在网上开店卖东西。电子商务较之传统的商贸活动有很多的便利。用户可以不受时间的限制，一天二十四小时电子商务网站都向你敞开大门；不受空间的限制，无论你走到哪里只需拥有一台可以上网的电脑，在浏览器地址栏中输入网址便可轻松购物，买到自己心仪的东西，同时又可以免去逛商场走路带来的劳累之苦，尽享购物的乐趣。电子商务对于商家和消费者来讲是一种双赢的方式。一个商家在网上开一家电子商务网站面对的将是全球七个多亿的潜在消费者，这样自己的商品会更快的销售出去，从而商品变现的速度会加快，商家也可以用这些钱重新投入生产，产生更多的价值，达到事半功倍的效果。受益的不仅仅是商家，消费者也可以从电子商务中得实惠，消费者可以足不出户在全世界任何一家电子商务网站购物，货比三家，不！更确切地说是货比千家甚至万家，以较低廉的价格买到自己想要的东西，选择的余地便大大增加了。同时商家开设自己的商务网站，推销自己的商品，减少了中间商这一环节，可以以更大的优惠幅度来回报消费者，获得更多的消费者的芳心。 如果一个商家没有在网上设立自己的商务网站的话，那它将会失去很大的市场份额，利润也会大大缩水。电子商务作为21世纪的主要经济增长方式之一，已经成为衡量一个国家的综合国力、科技水平和社会信息化的重要标志之一。

随着我国的民航事业的发展，航班也日益频繁，全国各地的飞机订票公司也越来越多。这些飞机订票公司看到了电子商务相对于传统售票方式更快捷、更便利的优势，纷纷采取了设立电子商务网站的这种营销手段，通过网站，可为旅客提供飞机机票的查询，预定机票，缴费等功能，方便了旅客的同时，给公司也带来了丰厚的效益。

针对这一现状，我的毕业设计题目选为“民航订票管理系统”，利用ASP.NET创建一个“民航订票管理系统”的动态网站。论文中详细介绍了我在创建该网站的过程中所使用的B/S结构，IIS、SQL Server2000、ASP.NET和ADO.NET等开发工具和平台以及网站创建的思路和各个界面的功能。

1.2 系统使用的相关技术

1.2.1 网络计算模式

C/S模式与B/S模式是网络计算模式中运用最多的两种模式。

C/S（Client/Server）结构即客户机和服务器结构。它的工作分别由服务器和客户机完成。服务器负责管理数据库的访问，为多个客户程序管理数据，对数据库进行检索和排序，此外还要对客户机/服务器网络结构中的数据库安全层层加锁，进行保护。客户机负责与用户的交互，收集用户信息，通过网络向服务器请求对诸如数据库，电子表格或文字处理文档等信息处理工作。

B/S（Browser/Server）结构即浏览器和服务器结构。它是随着 Internet 技术的兴起，对 C/S 结构的一种变化或者改进的结构。这种模式统一了客户端，将系统功能实现的核心部分集中到服务器上，简化了系统的开发、维护和使用。 客户机上只要安装一个浏览器（Browser），如 Netscape Navigator 或 Internet Explorer，服务器安装 Oracle、Sybase、Informix 或 SQL Server 等数据库。浏览器通过 Web Server 同数据库进行数据交互。

1、B/S 结构的优点

B/S 结构最大的优点就是可以在任何地方进行操作而不用安装任何专门的软件。只要有一台能上网的电脑就能使用，客户端零维护。系统的扩展非常容易，只要能上网，再由系统管理员分配一个用户名和密码，就可以使用了。甚至可以在线申请，通过公司内部的安全认证（如ＣＡ证书）后，不需要人的参与，系统可以自动分配给用户一个账号进入系统。

2、B/S 架构软件的优势与劣势

（1）维护和升级方式简单。目前，软件系统的改进和升级越来越频繁，B/S 架构的产品明显体现着更为方便的特性。对一个稍微大一点单位来说，系统管理人员如果需要在几百甚至上千部电脑之间来回奔跑，效率和工作量是可想而知的，但 B/S 架构的软件只需要管理服务器就行了，所有的客户端只是浏览器，根本不需要做任何的维护。无论用户的规模有多大，有多少分支机构都不会增加任何维护升级的工作量，所有的操作只需要针对服务器进行；如果是异地，只需要把服务器连接专网即可，实现远程维护、升级和共享。所以客户机越来越“瘦”，而服务器越来越“胖”是将来信息化发展的主流方向。今后，软件升级和维护会越来越容易，而使用起来会越来越简单，这对用户人力、物力、时间、费用的节省是显而易见的，惊人的。因此，维护和升级革命的方式是“瘦”客户机，“胖”服务器。

（2）成本降低，选择更多。大家都知道 windows 在桌面电脑上几乎一统天下，浏览器成为了标准配置，但在服务器操作系统上 windows 并不是处于绝对的统治地位。 现在的趋势是凡使用 B/S 架构的应用管理软件，只需安装在 Linux 服务器上即可，而且安全性高。所以服务器操作系统的选择是很多的，不管选用那种操作系统都可以让大部分人使用 windows 作为桌面操作系统电脑不受影响，这就使的最流行免费的 Linux 操作系统快速发展起来，Linux 除了操作系统是免费的以外，连数据库也是免费的，这种选择非常盛行。

比如说很多人每天上“新浪”网，只要安装了浏览器就可以了，并不需要了解“新浪”的服务器用的是什么操作系统，而事实上大部分网站确实没有使用 windows 操作系统，但用户的电脑本身安装的大部分是 windows 操作系统。

（3）应用服务器运行数据负荷较重。由于 B/S 架构管理软件只安装在服务器端（Server）上，网络管理人员只需要管理服务器就行了，用户界面主要事务逻辑在服务器（Server）端完全通过 WWW 浏览器实现，极少部分事务逻辑在前端（Browser）实现，所有的客户端只有浏览器，网络管理人员只需要做硬件维护。但是，应用服务器运行数据负荷较重，一旦发生服务器“崩溃”等问题，后果不堪设想。因此，许多单位都备有数据库存储服务器，以防万一。

1.2.2 IIS

IIS 是 Internet Information Server（互联网信息服务）的缩写，它是一种 Web（网页）服务组件，其中包括 Web 服务器、FTP 服务器、NNTP 服务器和 SMTP 服务器，分别用于网页浏览、文件传输、新闻服务和邮件发送等方面，它使得在网络（包括互联网和局域网）上发布信息成了一件很容易的事。IIS 响应极高，同时系统资源的消耗也最少。它的安装、管理、和配置都很容易。本系统采用的是

IIS5.1 的版本。

IIS 5.1 在网络安全性、可编程性和管理方面做出了相当大的改进，并能支持更多的 Internet 标准的支持，这些可以帮助用户轻松创建和管理站点，并制作易于升级、灵活性更高的 Web 应用程序。为了提高安全性，IIS 5.1 改进了自己安全验证方法，

IIS 5.1 采用分级验证，能够安全地可靠地通过代理服务器和防火墙验证用户，此外使用 Anonymous 和 Windows 验证。IIS 的管理工具使用 Microsoft 管理控制台（MMC），有利于进行集中管理。在管理过程中，用户可以在不重新启动计算机的情况下重新启动 Internet 服务；也可备份和保存 Internet 信息服务的设置以便出现问题后返回到安全、已知状态。另外，用户可以在站点、目录或文件等不同位置来设置信息服务的安全性，减少了用户的安全管理工作。

1.2.3 数据库

数据库是“按照数据结构来组织、存储和管理数据的仓库”。在经济管理的日常工作中，常常需要把某些相关的数据放进这样的“仓库”，并根据管理的需要进行相应的处理。例如，学校的教务处常常要把该学校学生的基本情况(学号、姓名、年龄、性别、生源地、专业等)存放在表中，这张表就可以看成是一个数据库。有了这个“数据仓库”我们就可以根据需要随时查询指定学生的基本情况，也可以查询年龄在某个范围内的学生人数或是统计该专业生源地是某地的人数等等。这些工作如果都能在计算机上自动进行，那将大大减少教务处工作人员的劳动强度。使用数据库可以减少了数据的冗余度，从而大大地节省了数据的存储空间；可以实现数据资源的充分共享等等。此外，数据库技术还为用户提供了非常简便的使用手段使用户易于编写有关数据库应用程序。数据库在我国正得到愈来愈广泛的应用。本系统采用的是 Microsoft 公司的 SQL Sever 2000 数据库。

SQL Server 2000 是 Microsoft 公司推出的大型关系数据库管理系统。它功能强大、操作简便，广泛应用于数据库后台系统。它在电子商务、数据仓库和数据库解决方案等应用中起着重要的核心作用。

SQL Server 2000 数据库的规模大，它可以将用户的数据存储在多个服务器上，并利用复制技术跨越多个服务器进行分布式处理，实现真正意义上的分布式数据库。SQL Server 2000 的安全性好，它可以对登录用户的身份进行认证，并对用户的操作权限进行控制。SQL Server 2000 的故障恢复功能强，它提供了强大的数据库备份和恢复功能，当故障发生时，能根据备份和日志迅速恢复到某一正确时刻。同时，SQL Server 2000 还提供了很好的并发控制功能以及大量的监控和管理数据库系统的工具。SQL Server 2000 适合于投入实际运营的较大规模的网站的后台数据库。

SQL 可以进行四个基本操作：

● Select——读取数据，从数据库中选择读取相应的数据，要从数据库中读取数据，就要指定字段列表，表格列表，要排序的字段列表和排序列表。SQL 的各个部分称为从句。基本 SELECT 语句最多有四个从句。语法如下：

```
Select (field1,field2,etc) from (table list) where (condition) order by (field1,field2)
```

Where 和 Order by 从句是可选的。

● Insert——增加数据，它向数据库表中插入一行或几行信息。语法如下：

```
insert into table name (field list) Values (Values list)
```

● Update——更新数据，它将一列或几列和一行或几行的数据更新。Update 语句是危险的，如果没有指定条件，则可能改变表中的所有行。更新数据时，一定要指定 Where 条件。语法如下：

UPDATE (table name) set field1=(value/expression),field2=(value/expression),…
From (table/query source) where (condition)

● Delete——删除数据，它时最简单而又最强大的语句。可以用 Delete 语句删除一个或几个表中的一行或几行。它和 Update 一样时危险的，因为它会毫无提示的删除数据。如果不慎运行了 Delete 语句，则很难恢复数据。通常不能用不带 Where 从句的 Delete 语句。语法如下：

Delete From (table name) where (condition)

1.2.4 ASP.NET

ASP.NET 是一种用于创建基于 Web 的应用程序的编程模型。从本质上来说，运行时和.NET Framework 类库集可以用于创建动态 Web 页。它需要在 Web 服务器的环境中运行，例如 Microsoft Internet Information Server(Microsoft 互联网信息服务器，IIS)，并且根据服务浏览器请求指示在服务器上执行程序。与直接由 Web 服务器提供的静态 HTML 不同的是，ASP.NET 页面实际上是在服务器上执行以后再产生结果的。页面的最后生成也许是由许多不同的指令和/或数据源构造的。

ASP.NET 页面以.aspx 扩展名存储。页面由程序员将文本、标记(例如 HTML)以及 ASP.NET 特定服务器标记和脚本组合在一起，然后存储在 Web 服务器上。可以将存储后的 ASP.NET 页面看成是一套描述如何创建一个 HTML 页面的指令。当该页面被请求浏览时，服务器端程序将会用纯标记来创建一个客户端浏览器可以读懂并能呈现(render)的页面。因为呈现后的输出是纯标记，所以任何浏览器都能够读懂；所有的动态过程都发生在 Web 服务器端。ASP.NET 特定服务器标记非常强大，例如，它可以对用户的动作作出反应，连接至数据存储以及自动创建非常复杂的 HTML 结构。

正像前面提到的那样，ASP.NET 只是.NET Framework 的一部分，所以 ASP.NET 页面可以利用这个框架提供的所有服务，包括连网、数据访问、安全以及更多其他服务。因为 ASP.NET 可以使用所有这些服务，所以相比以前，能够创建更加丰富的 Web 应用程序。只需花少量的时间来构建所有应用程序所需的构建块，而将大多数时间用在应用程序独有的特殊逻辑上。

ASP.NET 还在 Web 编程中引入了一些独特的新技术，可以在典型的动态服务器页面(Active Server Pages，ASP)上极大地改善开发模式：

● 语言独立性——因为 ASP.NET 是.NET Framework 的一部分，所以可以使用您自己选择的语言来构建 ASP.NET 应用程序，例如 C#、VB 或 J#。而典型的 ASP 则仅限于 JScript 或者 VBScript 页面。

● 编译而不是解释——与典型的 ASP 在每一次页面请求时都解释编程结构不同，ASP.NET 在服务器端动态地将页面编译成可以运行得非常快的本机编程指令。可以很明显地看到典型的 ASP 页面的性能与相同 ASP.NET 页面的性能之间相差的数量级别。

● 事件驱动编程模式——在典型的 ASP 中，页面总是以自顶向下的线性方式执行，并且 HTML 标记常常与程序指令混合在一起。任何一个有一定 ASP 经验的人都知道这样会使得页面难以阅读，甚至更加难以维护。ASP.NET 引入了事件驱动模型，这个模型允许您将代码与标记内容分离，将代码并入处理专门任务的有意义的单元中，例如响应客户端的按钮单击动作。这个类似 VB 的事件模型极大地提高了页面的可读性和可维护性。

● 服务器控件——典型的 ASP 需要动态地将 HTML 片断代码接合在一起呈现，这样做的结果就是在应用程序中一遍又一遍地编写相同的代码(您需要多少次才能从数据库查询中构建一张表格)。ASP.NET 带给 Web 编程的一个最大的好处就是能够将公共的呈现和行为封装成服务器控件(server control)，可以在应用程序中很方便地重复使用。就像 HTML 标记一样，服务器控件以声明的形式创建，但是表现为一个位于服务器端的可编程对象，它可以与代码进行交互并输出定制的动态 HTML

呈现。ASP.NET 包含了大约 80 多个服务器控件，这些控件封装了从标准表单元素到复杂控件(如网格和菜单)的所有内容。

● 控件设计时间的改善(当使用 Visual Web Developer 时)—— 开发人员通过使用设计时间界面可以减少花费在开发复杂页面上的时间，这些界面包括敏捷任务面板、标签级导航栏和可以设置控件属性的向导。

1.2.5 ADO.NET

ADO.NET 是.NET Framework 中的一套类库，它将会让您更加方便地在应用程序中使用数据。Microsoft 收集了过去几十年中最佳的数据连接的实践操作，并编写代码实现这些实践。这些代码被包装进了一些对象中，以便其他软件可以方便地使用。

ADO.NET 中的代码处理了大量的数据库特有的复杂情况，所以当 ASP.NET 页面设计人员想读取或者写入数据时，他们只需编写少量的代码，并且这些代码都是标准化的。就像 ASP.NET 一样，ADO.NET 不是一种语言。它是对象(类)的集合，在对象(类)中包含了由 Microsoft 编写的代码。可以使用诸如 Visual Basic 或者 C#等编程语言来在对象外部运行这些代码。

ADO.NET 里包括了许多专门用于和数据打交道的对象。这些对象是学习 ADO.NET 必须了解的。掌握它们后你将了解使用 ADO.NET 和数据打交道会需要考虑哪些事情。下面以 SQL Server 数据源为例介绍

The Sq1Connection Object

要访问一个数据源，你必须先建立一个到它的连接。这个连接里描述了数据库服务器类型、数据库名字、用户名、密码，和连接数据库所需要的其它参数。command 对象通过使用 connection 对象来知道是在哪个数据库上面执行 SQL 命令。

The Sq1Command Object

连接数据库后就可以开始想要执行的数据库操作，这个是通过 command 对象完成，command 对象一般被用来发送 SQL 语句给数据库。command 对象通过 connection 对象得知道应该与哪个数据库进行连接。我们既可以用 command 对象来直接执行 SQL 命令，也可以将一个 command 对象的引用传递给 Sq1DataAdapter，Sq1DataAdapter 能包含一系列的 command 对象，可以处理大量数据。

The Sq1DataReader Object

根据经验，许多数据库操作要求我们仅仅只是需要读取一组数据。这时候就用到了 data reader 对象。通过 data reader 对象，我们可以获得从 command 对象的 SELECT 语句得到的结果。考虑到性能方面的因素，data reader 返回的数据流被设计为只读的、单向的，这将意味着你只能按照一定的顺序从数据流中取出数据。虽然你在这里也获得了性能上的提升，但是缺点也是明显的，不能够操作取回数据，如果需要操作编辑数据，解决的办法是使用 DataSet。

The DataSet Object

DataSet 对象用于表示那些储存在内存中的数据。它包括多个 DataTable 对象，DataTable 就象一个普通的数据库中的表一样，也有行和列，我们甚至能够通过定义表和表之间的关系来创建从属关系。DataSet 主要用于管理存储在内存中的数据以及对数据的断开操作。

The Sq1DataAdapter Object

某些时候我们只需要读数据，并且你不需要修改它们把更改写回数据源。但是还有这样一些情况为了减少数据库调用的次数，我们把数据缓存在内存中。Data adapter 通过断开模型来轻松的实现了后面这种情况的处理。当批量完成的对数据库的读写操作的并将改变写回数据库的时候，Data

adapter 会填充（fill）DataSet 对象。Data adapter 里包含了 connection 对象,当对数据源进行读取或者写入的时候，Data adapter 会自动的打开或者关闭连接。此外，Data adapter 还包含对数据的 SELECT,INSERT,UPDATE 和 DELETE 操作的 command 对象引用。如果我们为 DataSet 中的每一个 table 都指定 Data adapter，它将会帮你处理好所有与连接处理数据库的操作，我们所需要做的仅仅就是告诉 data adapter 什么时候读取或者写入到数据库。

1.3 开发环境：

硬件环境：Pentium III 以上 PC 机一台

软件开发环境：Windows XP，IIS

Microsoft Visual Studio.NET 2003

SQL Server 2000

第二章 需求分析

2.1 项目介绍

该民航订票管理系统是一个 B/S 结构的民航订票网站，用户可以通过登录网站，输入出发地点、

到达地点，选择座位类型并选择是单程还是往返，填入出发日期和返回日期查询满足条件的航班信息。用户可以从查询到的航班中选择您要预定的航班生成订单，进入订单界面进行最后的确认或是取消订单的操作。

2.2 项目目标

设计本系统的目的是让民航订票走向电子化，使民航订票公司的员工提高工作效率，这样就可以更方便、更快捷的为旅客提供优质的服务，同时为公司降低了运营成本。在本系统中有两种用户身份，分别为普通用户与系统管理员，他们实现的功能也是不同的，因此需要进行身份验证。普通用户实现查询航班、预定航班、发表留言、修改密码和用户信息的功能；而系统管理员可以进行管理航班信息、管理用户信息、管理订单、管理留言等操作。

系统所要实现的功能如下：

用户信息管理：注册新用户、用户登录、用户密码修改，用户信息修改等。

航班信息管理：添加新航班、更改航班信息，删除航班等

订单管理：查看订单信息、删除订单等

留言管理：查看留言、回复留言等

航班查询：输入出发地点和到达地点，选择座位类型，如果是单程则选择单程按钮，输入出发时间点击查询即可；如果是往返则选择往返按钮，输入出发时间和返回时间点击查询即可。

订单查询：用户可以查看自己的订单信息

预定航班：实现用户定制机票

2.3 分析建模

为了更好地理解民航订票管理系统，现将系统功能抽象，更加确切地分析实际需求，构造系统的模型。

2.3.1 系统功能模型

基于上述分析将系统划分为四大模块：用户信息管理模块，订单管理模块，航班信息管理模块留言信息管理模块。

其结构如图 2-1 所示：

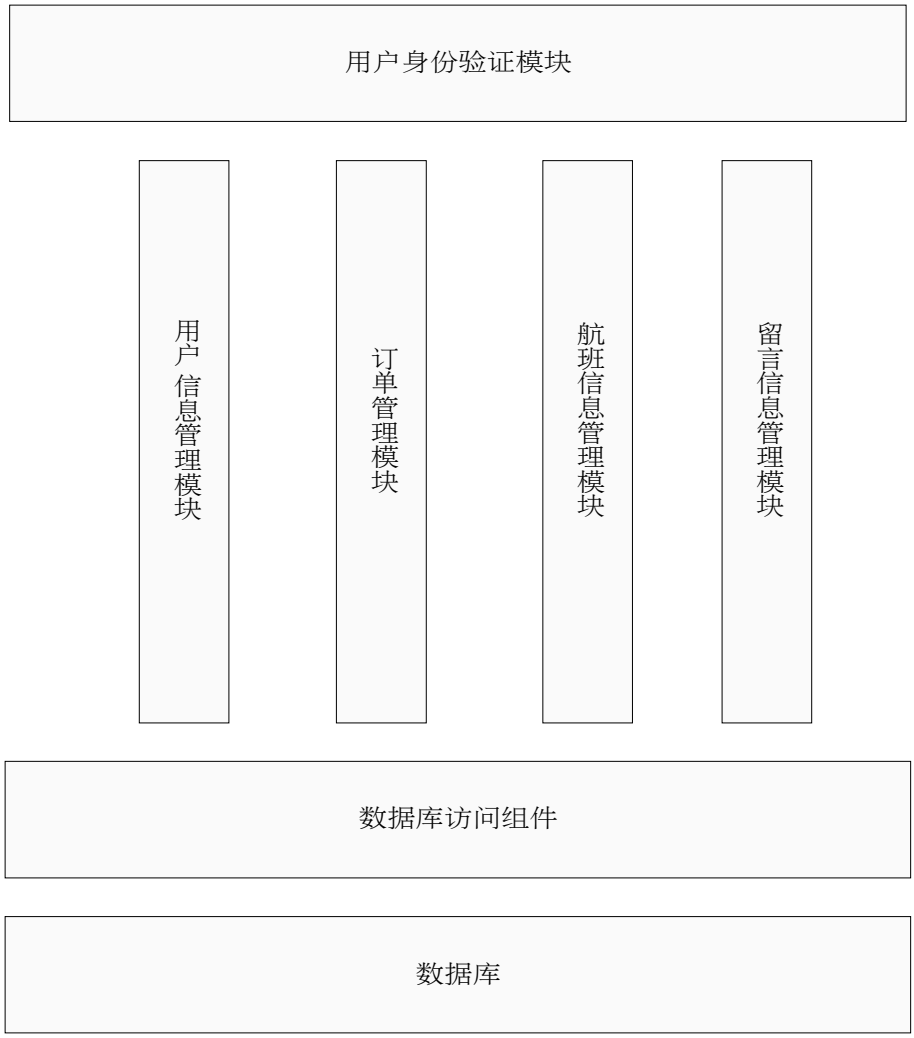

图 2-1　模块划分图

其具体功能需求分析如下：

1．用户信息管理

系统管理员对指定用户进行删除和更改用户信息等操作；普通用户更改用户基本信息和更改用户密码操作

2．订单管理

系统管理员可以查看所有用户订单的详情并可进行删除订单操作；普通用户进行订单的预定和删除操作

3.航班信息管理

系统管理员进行航班的添加，对指定航班信息的修改和删除操作；普通用户查询满足条件的航班，查询所有航班实时信息和查询热门航班操作。

4．留言信息管理

系统管理员对用户留言进行回复和删除操作；普通进行用户查看留言和发表留言操作。

2.3.2 系统功能框架图

系统的功能框架图如图 2-2 所示

图 2-2 系统的功能框架图

2.3.3 数据模型

1．数据对象

(1) 角色：管理员，普通用户

(2) 事物：航班，订单，留言，飞机座位类型

(3) 行为：管理订单，管理航班，管理用户，管理留言

2．数据属性：

(1) 管理员：用户 ID，用户类型，用户名，密码，姓名，性别，身份证号，Email 地址，家庭住址

(2) 普通用户：用户 ID，用户类型，用户名，密码，姓名，性别，身份证号，Email 地址，家庭住址

(3) 航班：航班 ID，航班号，航空公司，飞机类型，起飞地点，到达时间，起飞时间，到达时间

(4) 飞机座位类型：航班 ID，座位类型，飞机票价，座位数量

(5) 留言：留言 ID，主题，内容，留言时间，用户 ID，回复信息，回复时间

3．民航订票管理系统 E-R 图：

对应关系如图 2-3 所示：

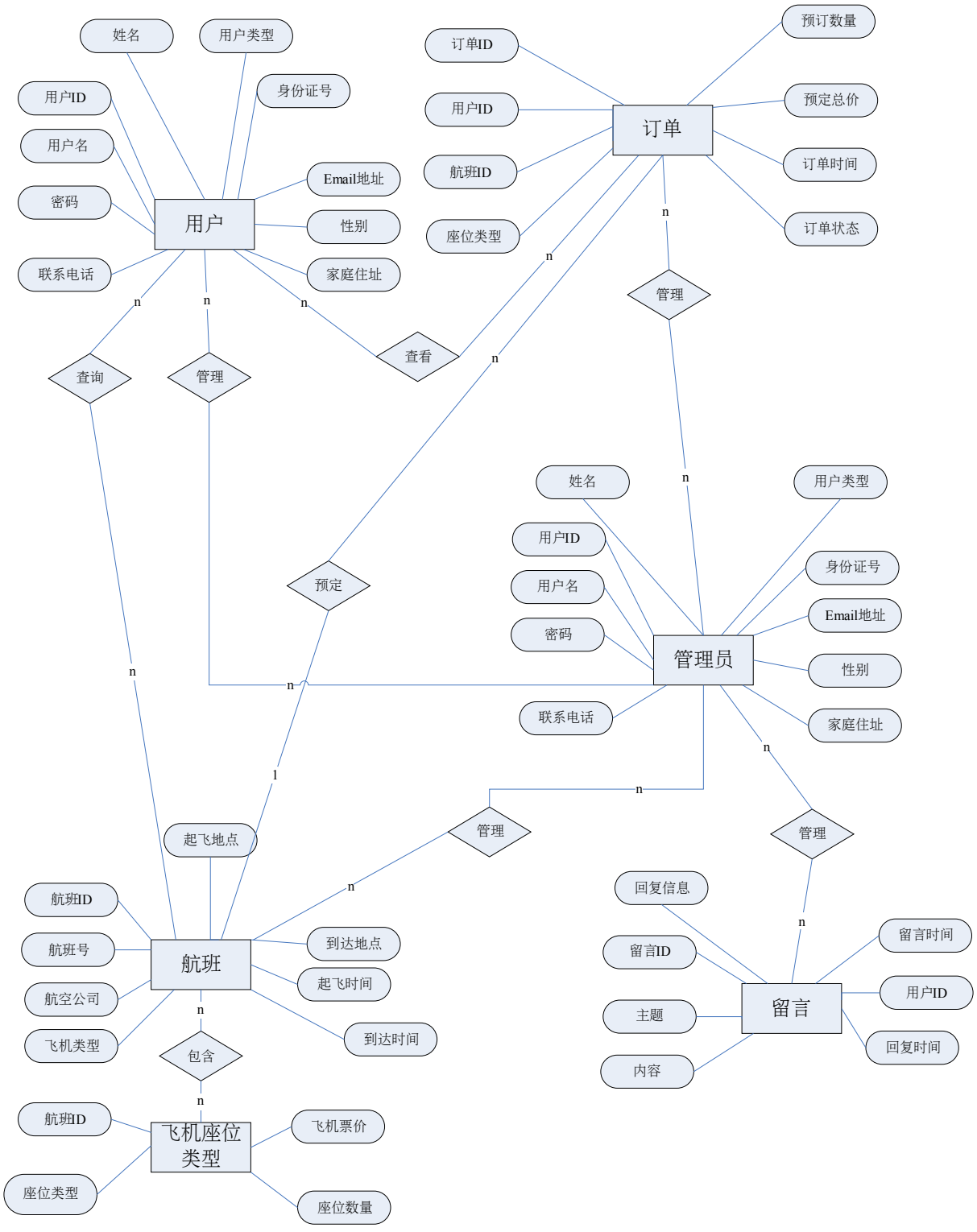

图 2-3 民航订票管理系统 E-R 图

第三章 概要设计

3.1 系统分析

经过需求分析阶段，对整个系统已经有了初步的认识，现在要进一步细化系统功能，设计合理的方案，重点是要确定系统的界面风格。

为了保证此系统能够长期、安全、稳定、可靠、高效的运行，系统应该满足以下的性能需求：

1.系统处理的准确性和实时性：系统处理的准确性和实时性是系统的必要性能。在系统设计和开发过程中，要充分考虑系统当前和将来可能承受的负荷。

2.系统的易用性和易维护性:要实现这一点就应该尽量使用用户熟悉的用语及友好界面。

3.系统数据要求：(1)数据的录入和处理的准确性和实时性 (2)数据的一致性和完整性。(3)数据的共享与独立性。

3.2 系统界面设计

1． 民航订票管理系统首页页面如图 3-1 所示

图 3-1 民航订票管理系统首页页面图

本系统的起始页面是查询和预定页面，输入出发地点和到达地点，选择座位类型，如果是单程则选择单程按钮，输入出发时间点击查询即可；如果是往返则选择往返按钮，输入出发时间和返回时间点击查询即可。点击查询后显示满足条件的航班信息（包括航班号、航空公司、飞机类型、起飞地点、到达地点、起飞时间、到达时间、座位类型、剩余票数和飞机票价），用户只需在要预定的航班的预订数量一列输入数量并点击预定按钮，便会生成新的订单并跳转至订单界面。（注：如果用户未登录，便会跳转至登录界面。如果用户未注册可自行点击注册界面进行注册。用户登录后除了实现上述功能外，还可以进行查看订单和取消订单，查看留言和发表留言，更改用户基本信息和更改用户名和密码等操作。）

2.民航订票管理系统的新用户注册页面如图 3-2 所示

图 3-2　新用户注册界面

注册界面规定用户名必须唯一，否则系统会弹出窗口进行提示，密码由 6-16 位数字或字母组成，联系电话可以是手机号也可是固定电话，Email 满足 Internet 电子邮件地址的格式，身份证号码符合中华人民共和国身份证号码编码规则，其他项也需填写完整才能完成注册。注册后自动跳转至登录界面。

3．民航订票管理系统的用户登录页面如图 3-3 所示

图 3-3　用户登录界面

注册的普通用户可以在这里进行登录，登录之后系统会自动跳转到如图 3-1 所示的民航订票管理系统的起始页，进行查询航班，预定航班，发表留言等操作。管理员登录之后系统会自动跳转到后台管理员的主界面。

4．系统后台主页面如图 3-4 所示

图 3-4　系统后台首页

管理员登录后可以选择该界面上的航班管理、用户管理、添加管理员、留言管理、订单管理、添加新航班等超级链接按钮进入相应的界面，实现相应的功能。

5．添加新航班界面如图 3-5、3-6 所示

图 3-5 添加新航班界面（1）

根据下拉菜单选择航班公司、起飞地点、降落地点和飞机型号，输入唯一的航班号，根据提示的格式输入起飞和降落时间，点击下一步按钮，即弹出图 3-6 所示界面：

图 3-6 添加新航班界面（2）

点击下一步按钮后设置该航班的三种座位类型的数量和票价并点击提交按钮，便会显示该航班的信息。

6.航班管理界面如图 3-7 所示

图 3-7 航班管理界面

上面的表显示所有航班信息，根据该表查到的航班号管理员可对指定航班进行更改航班信息和删除航班的操作。

7.用户管理界面如图 3-8 所示

图 3-8 用户管理界面

在该界面上管理员可根据用户名查询用户信息，并且可对用户信息进行更改或删除用户的操作。

8.留言界面如图 3-9 所示

图 3-9 留言界面

用户登录后可以进行留言和查看留言操作

9.订单界面如图 3-10 所示

图 3-10 订单界面

用户登录后方可进入订单界面，订单界面显示该用户预定的所有订单信息。用户可以选择订单列表的最后两列按钮，选择是取消订单还是最终确认订单。如果选择取消订单按钮，数据库中该订单信息将删除，如果选择提交订单按钮，将会显示提交的当前时间，订单状态也将相应从未确认订单改成已确认订单。提交订单后，管理员会尽快与您取得联系。

10.更改用户信息界面如图 3-11、3-12 所示

图 3-11 更改用户信息界面（1）

图 3-12 更改用户信息界面（2）

图 3-11 是修改个人基本信息界面，用户可随时更改用户基本信息，更改后跳转至系统首页，图 3-12 是修改用户密码界面，修改用户密码后会跳转至登录界面，需要重新登录。

如果用户想查看实时的航班信息可以选择“所有航班”链接，该界面会显示所有航班的实时信息；如果用户想了解哪架航班预定人数比较多，可以点击“热门航班”链接，该页面将会列出预定数量排名前十航班的航班号、航空公司、起飞地点、降落地点等信息。

第四章　详细设计

4.1 系统功能概述

根据系统需求分析中的系统功能模型，对其中的每个模块进行详细设计，在设计各个功能模块之前，首先需要对民航订票管理系统设计一个操作流程，有了操作流程开发人员就有了明确的前进方向，可以缩短开发时间，避免编码时产生逻辑错误。

4.1.1 系统工作流程简述

该系统允许任何人进行查询满足条件的航班信息，查询所有航班的实时信息和热门航班等操作，但是如果你想预定航班的话，就必须先注册才可预定航班。同时登录界面会对管理员和普通用户身份进行验证，如果是普通用户，登录后跳转至系统首页，如果是管理员用户，登录后跳转至后台管理界面，进行用户管理，航班管理，留言管理，订单管理等操作。

4.1.2 系统主要功能组件

通过角色权限的划分，明确各部分的功能结构，现在该考虑如何实现各部分功能，并将其有序的组织在一起。模块主要功能介绍:

1.前台用户管理:

(1) default.aspx 首页（查询及预定航班界面）

(2) login.aspx 登录页面

(3) register.aspx 注册界面

(4) updateuserinfo.aspx 更改用户信息界面

(5) updatepassword.aspx 更改用户密码界面

(6) order.aspx 订单界面

(7) allflight.aspx 所有航班界面

(8) hotplane.aspx 热门航班界面

(9) message.aspx 留言界面

(10) messagedetail.aspx 留言详情界面

2.后台管理员管理:

(1) admin.aspx 后台管理首页

(2) addnewplane.aspx 添加新航班界面

(3) adminPlane.aspx 航班管理界面

(4) adminuser.aspx 用户管理界面

(5) addadminuser.aspx 添加管理员界面

(6) adminorder.aspx 订单管理界面

(7) adminmessage,aspx 留言管理界面

4.2 系统模块的流程设计

4.2.1 前台用户管理模块执行流程

前台用户管理模块的流程图显示了本系统在系统前台的全部功能和使用方法。本系统前台主要有三个功能模块，包括留言信息，用户信息，以及最主要的订票模块，前台用户管理模块的流程图

如图 4-1 所示

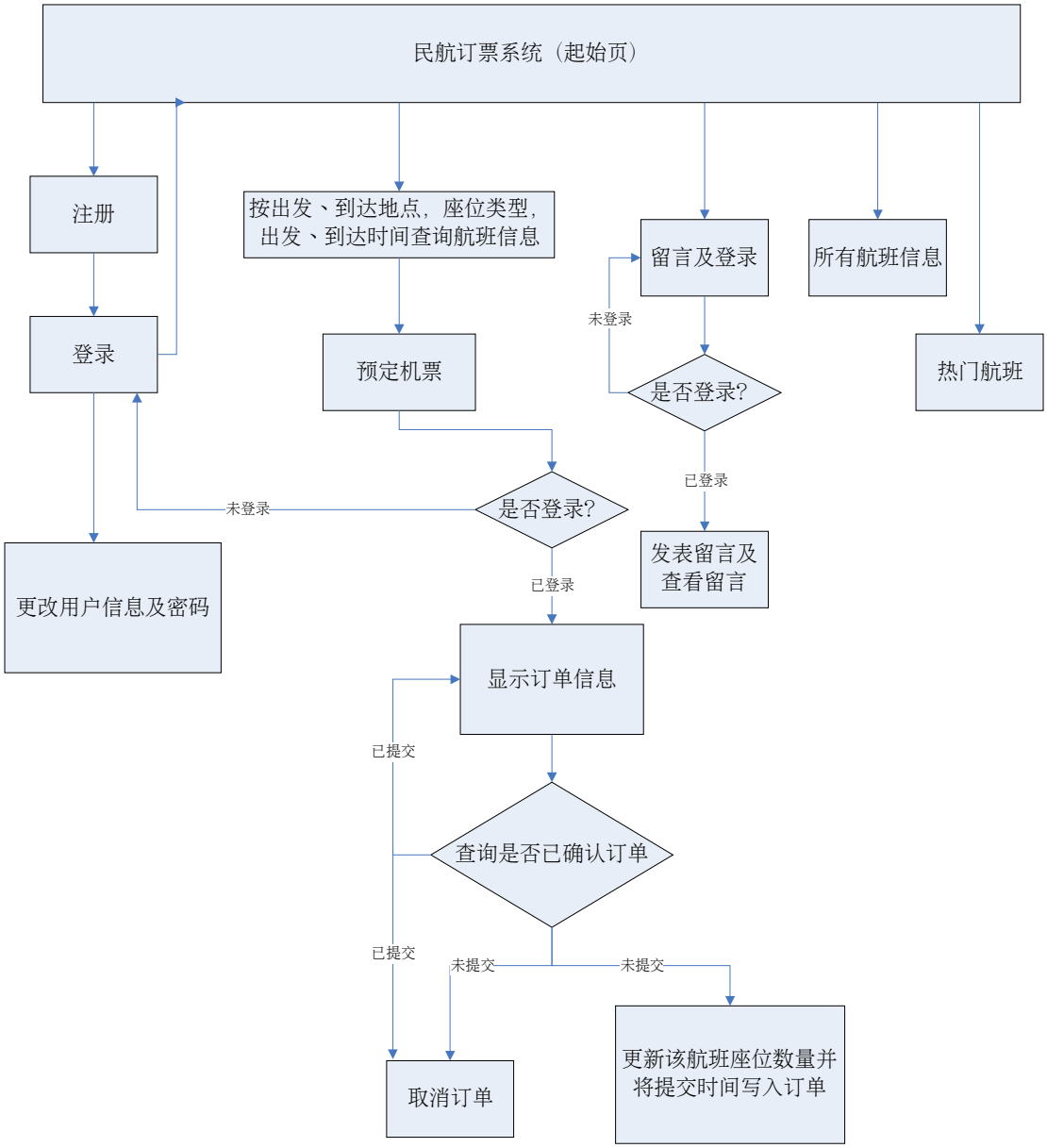

图 4-1 前台用户管理模块的流程图

4.2.2 后台管理员管理模块执行流程

后台管理员管理模块的流程图显示了本系统在系统后台的全部功能和使用方法。用户在登录界面进行登录验证，验证成功后如果是管理员账户，就进入管理员界面进行航班管理，用户管理，订单管理，留言管理，如果是普通用户跳转至前台界面。后台管理员管理模块的流程图如图 4-2 所示：

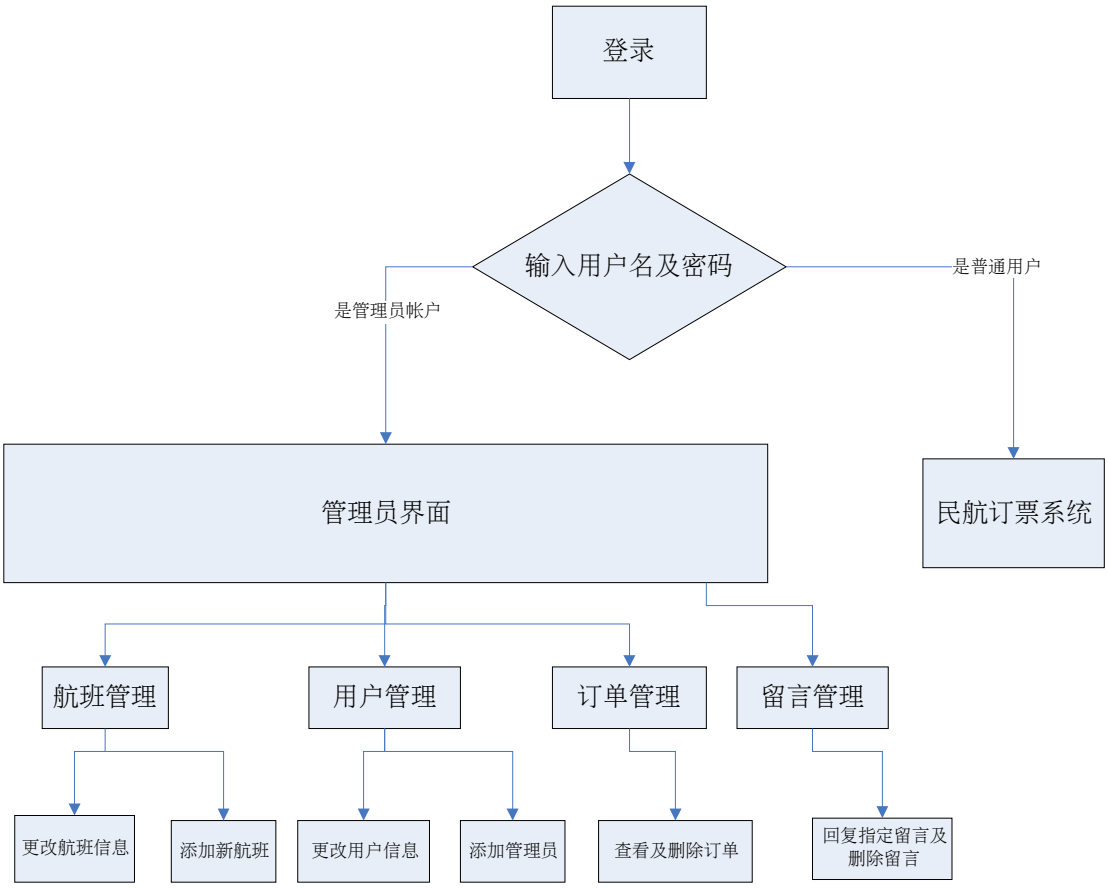

图 4-2 后台管理员管理模块的流程图

4.3 数据库设计

民航订票管理系统的数据库使用 SQL Server 2000，共建立了六张表，如表 4-1 所示，分别是 users、plane、planeseattype、orders、message、counttotal。

users 表包括了用户编号、用户名、密码、姓名、性别、联系电话、家庭住址、Email、身份证号码等信息。

plane 表包括了航班编号、航班号、航空公司、起/降地点、起/降时间、等信息。

planeseattype 表使用了外键 planeid 与 plane 表关联，包括了与指定飞机相关联的飞机座位类型、座位数、飞机票价、飞机类型等信息。

orders 表包括了订单号、用户编号、航班编号、座位类型、预订数量、预定总价、预定时间、订单状态，其中外键用户编号（userid）与 users 表关联，外键航班编号（planeid）与 plane 表关联。

message 表包括了留言编号、主题、详细内容、留言时间、用户编号、回复信息、回复时间，其中外键用户编号（userid）与 users 表关联。

counttotal 表只包括了访问本网站的总人数。

表 4-1　　系统中所用到的表及用途

表名	作用
users	用户信息表
plane	航班信息表
planeseattype	飞机座位类型表
message	留言信息表
orders	订单信息表

counttotal	历史在线人数表

1．用户信息表（users）

表 4-2　　users 表

列名	数据类型	作用	是否为空
userid	int	用户 ID（自增字段（主键））	否
username	nvarchar	用户名	否
userpassword	nvarchar	用户密码	否
usertype	int	用户类型	否
usertel	nvarchar	联系电话	否
identityid	nvarchar	身份证号码	否
identityname	nvarchar	姓名	否
usersex	nvarchar	性别	否
useremail	nvarchar	Email 地址	否
useraddress	nvarchar	家庭地址	否

2．航班信息表（plane）

表 4-3　　plane 表

列名	数据类型	作用	是否为空
planeid	int	航班 ID（自增字段（主键））	否
planename	nvarchar	航班号	否
planecompany	nvarchar	航空公司	否
planestartplace	nvarchar	起飞地点	否
planeendplace	nvarchar	降落地点	否
planestarttime	datetime	起飞时间	否
planeendtime	datetime	降落时间	否
planetype	nvarchar	飞机类型	否

3．飞机座位类型表（planeseattype）

表 4-4　　planeseattype 表

列名	数据类型	作用	是否为空
planest	nvarchar	飞机座位类型	否
planenumber	int	座位数量	否
planeprice	decimal	机票价格	否
planeid	int	航班 ID(外键)	否

4.留言信息表（message）

表 4-5 message 表

列名	数据类型	作用	是否为空
messageid	int	留言 ID（自增字段（主键））	否
messagetopic	text	主题	否
message	text	内容	否
messagetime	datetime	留言时间	否
userid	int	用户 ID（外键）	否
returnmessage	text	回复信息	是
returntime	datetime	回复时间	是

5．订单信息表（orders）

表 4-6 orders 表

列名	数据类型	作用	是否为空
orderid	int	订单 ID（自增字段（主键））	否
userid	int	用户 ID（外键）	否
planeid	int	航班 ID（外键）	否
planst	nvarchar	飞机座位类型	否
ordernumber	int	预定数量	否
orderprice	decimal	预定总价	否
ordertime	datetime	预定时间	是
orderstate	nvarchar	订单状态	否

6.历史在线人数表（counttotal）

表 4-7 counttotal 表

列名	数据类型	作用	是否为空
total	int	历史访问人数	否

第5章 编码与实现

5.1 程序描述

利用 ASP.NET 编程技术完成查询和预定航班，查询热门航班和实时航班信息，查看和取消订单，查看和发表留言等模块的前台平面设计，使用动态连接数据库技术，实现以上系统模块的后台管理。

5.2 总体结构

5.2.1 系统相关类列表

表 5-1　系统相关类列表

类	功能
default.aspx	首页（查询及预定航班界面）
login.aspx	登录页面
register.aspx	注册界面
updateuserinfo.aspx	更改用户信息界面
updatepassword.aspx	更改用户密码界面
order.aspx	订单界面
allflight.aspx	所有航班界面
hotplane.aspx	热门航班界面
message.aspx	留言界面
messagedetail.aspx	留言详情界面
后台 admin.aspx	后台管理首页
后台 addnewplane.aspx	添加新航班界面
后台 adminPlane.aspx	航班管理界面
后台 adminuser.aspx	用户管理界面
后台 addadminuser.aspx	添加管理员界面
后台 adminorder.aspx	订单管理界面
后台 adminmessage,aspx	留言管理界面

5.2.2 主要代码

1．　用户登录界面

登录界面登录按钮事件的实现代码

```
private void login_Click(object sender, System.EventArgs e)
        {
            if(Page.IsValid)
            {
                string userName=Request.Form.Get("userName").ToString();
                string userPwd=Request.Form.Get("userPwd").ToString();
                SqlConnection con=Class.connection();
                con.Open();
                SqlCommand com=new SqlCommand("select count(*) from users where
username='"+userName+"'and userpassword='"+userPwd+"'",con);
              int count=Convert.ToInt32(com.ExecuteScalar());//判断是否登录成功
                if(count>0)//成功登录
                {
                    SqlCommand com2=new SqlCommand("select usertype from users where
username='"+userName+"' ",con);//判断用户类型
```

```
                int judge=Convert.ToInt32(com2.ExecuteScalar());
                if (judge==1)//管理员登录
                {
                    Session["flag"]=true;
                    UserJudge user=new UserJudge();
                    user.username=userName;
                    Session["admin"]=user;
                    Response.Redirect("admin.aspx");}
                else//普通用户登录

                {
                    Session["flag"]=true;
                    UserJudge user=new UserJudge();
                    user.username=userName;
                    Session["user"]=user;
                    Response.Redirect("default.aspx");

                }
            }
            else
            {

                Response.Write("<script language='javascript'>alert('用户名或密码错
误!');</script>");


            }
        }
    }
```

2． 用户注册界面

注册界面提交按钮事件的代码

```
    private void SubmitButton_Click(object sender, System.EventArgs e)
        {
            SqlConnection con=Class.connection();//连接数据库
            con.Open();
            SqlCommand com1=new SqlCommand("select count(*) from users where
username='"+userName.Text+"'",con);
            int count=Convert.ToInt32(com1.ExecuteScalar());
            if(count>0)//用户名重复
            {

                Response.Write("<script language='javascript'>alert('对不起，用户名不可
用');</script>");


            }
            else
            {
                SqlCommand com2=new SqlCommand("insert into users
(username,userpassword,usertype,usertel,identityid,identityname,usersex,useremail,useraddress)
```

```
values('"+userName.Text+"','"+userPwd1.Text+"','"+0+"','"+userTel.Text+"','"+identityId.Text+"','"+identi
tyName.Text+"','"+this.SexSelected.SelectedValue+"','"+userEmail.Text+"','"+userAddress.Text+"')",con);
                    com2.ExecuteNonQuery();
                    con.Close();
                    Response.Redirect("login.aspx");}
        }
```

3．系统首页（查询航班和预定航班）

绑定出发地点和到达地点的代码

```
private void Page_Load(object sender, System.EventArgs e)
        {

            TotalLabel.Text=Application["total"].ToString();
            OnlineLabel.Text=Application["online"].ToString();
            if(Session["flag"]==null)
            {
                LoginJudgeLabel.Text="未登录";
            }
            else
            {

                UserJudge user=(UserJudge)Session["user"];
                LoginJudgeLabel.Text="欢迎"+user.username+"光临本网站";
            }
            if(!this.IsPostBack)//是第一次，就执行绑定。
            {  //绑定出发地点
                SqlConnection con=Class.connection();//数据库连接
                con.Open();
                SqlCommand com=new SqlCommand("select distinct planestartplace from
plane",con);
                SqlDataReader sdr=com.ExecuteReader();
                this.SPDropDownList.DataSource=sdr;
                this.SPDropDownList.DataTextField="planestartplace";
                this.SPDropDownList.DataValueField="planestartplace";
                this.SPDropDownList.DataBind();
                sdr.Close();
                //绑定到达地点
                SqlCommand com1=new SqlCommand("select distinct planeendplace from plane
",con);
                sdr=com1.ExecuteReader();
                this.EPDropDownList.DataSource=sdr;
                this.EPDropDownList.DataTextField="planeendplace";
                this.EPDropDownList.DataValueField="planeendplace";
                this.EPDropDownList.DataBind();
                sdr.Close();
```

```
                con.Close();

            }
```

首页查询按钮事件的代码

```
private void Button1_Click(object sender, System.EventArgs e)
        {
            if(this.SPDropDownList.SelectedValue==this.EPDropDownList.SelectedValue)
            {

            Response.Write("<script language='javascript'>alert('出发地点不能与到达地点相
同！');</script>");
            }
            else
            {
            if(this.RadioButtonList1.SelectedValue=="单程")
            {
                //查询满足条件的航班
                this.DataGrid1.DataKeyField="planeid";
                SqlConnection con=Class.connection();//数据库连接
                con.Open();
                this.DataGrid1.DataSource=Class.ExeSelect("select * from plane join planeseattype
on plane.planeid=planeseattype.planeid where
planestartplace='"+this.SPDropDownList.SelectedValue+"'and
planeendplace='"+this.EPDropDownList.SelectedValue+"' and
planest='"+this.STDropDownList.SelectedValue+"'and convert(varchar,planestarttime,120) like
'%"+this.STTextBox.Text+"%'");
                this.DataGrid1.DataBind();
                this.Datagrid2.Visible=false;
                con.Close();
            }
                if(this.RadioButtonList1.SelectedValue=="往返")//往返
                {
                    DateTime starttime=Convert.ToDateTime(STTextBox.Text);
                    DateTime endtime=Convert.ToDateTime(ETTextBox.Text);
                    if(DateTime.Compare(starttime,endtime)>0)
                    {

                        Response.Write("<script language='javascript'>alert('返回时间小于出发时
间!');</script>");
                    }
                    else
                    {
                        this.DataGrid1.DataKeyField="planeid";
                        SqlConnection con=Class.connection();//数据库连接
                        con.Open();
```

```
                this.DataGrid1.DataSource=Class.ExeSelect("select * from plane join
planeseattype  on plane.planeid=planeseattype.planeid where
planestartplace='"+this.SPDropDownList.SelectedValue+"'and
planeendplace='"+this.EPDropDownList.SelectedValue+"' and
planest='"+this.STDropDownList.SelectedValue+"'and convert(varchar,planestarttime,120) like
'%"+this.STTextBox.Text+"%'");
                this.DataGrid1.DataBind();
                this.Datagrid2.DataKeyField="planeid";
                this.Datagrid2.DataSource=Class.ExeSelect("select * from plane join
planeseattype  on plane.planeid=planeseattype.planeid where
planestartplace='"+this.EPDropDownList.SelectedValue+"'and
planeendplace='"+this.SPDropDownList.SelectedValue+"' and
planest='"+this.STDropDownList.SelectedValue+"'and convert(varchar,planestarttime,120) like
'%"+this.ETTextBox.Text+"%'");
                this.Datagrid2.DataBind();
                this.Datagrid2.Visible=true;
                con.Close();
            }

        }
    }

}
```

查询指定的航班表的预定按钮的响应事件代码

```
private void DataGrid1_ItemCommand(object source,
System.Web.UI.WebControls.DataGridCommandEventArgs e)
    {


        if(e.CommandName=="AddToOrder")
        {
            if(Session["flag"]==null)
            {
                Response.Redirect("login.aspx");
            }
            else
            {
```

```
                int OrderNumber=int.Parse(((TextBox)
(e.Item.Cells[9].FindControl("textbox1"))).Text);
                string PlaneId=this.DataGrid1.DataKeys[e.Item.ItemIndex].ToString();//插入
订单表
                decimal PlanePrice=decimal.Parse(((Label)
(e.Item.Cells[8].FindControl("PlanePrice"))).Text);
                decimal TotalPrice=PlanePrice*OrderNumber;
                UserJudge user=(UserJudge)Session["user"];
                SqlConnection con=Class.connection();//连接数据库
                con.Open();
                SqlCommand com=new SqlCommand("select userid from users where
username='"+user.username+"'",con);
                string UserId=com.ExecuteScalar().ToString();
                string OrderState=Convert.ToString("未确认订单");
                SqlCommand com1=new SqlCommand("insert into orders
(userid,planeid,planest,ordernumber,orderprice,orderstate) values
('"+UserId+"','"+PlaneId+"','"+this.STDropDownList.SelectedValue+"','"+OrderNumber+"','"+TotalPrice+
"','"+OrderState+"')",con);
                com1.ExecuteNonQuery();
                con.Close();
                if(this.RadioButtonList1.SelectedValue=="单程")
            {
                Response.Redirect("order.aspx");
            }
        }
```

实现查询指定的航班表的分页显示的代码

```
private void DataGrid1_PageIndexChanged(object source,
System.Web.UI.WebControls.DataGridPageChangedEventArgs e)
    {
        this.DataGrid1.CurrentPageIndex=e.NewPageIndex;
        if(this.RadioButtonList1.SelectedValue=="单程")
        {
            //查询满足条件的航班
            this.DataGrid1.DataKeyField="planeid";
            SqlConnection con=Class.connection();//数据库连接
            con.Open();
            this.DataGrid1.DataSource=Class.ExeSelect("select * from plane join planeseattype
on plane.planeid=planeseattype.planeid where
planestartplace='"+this.SPDropDownList.SelectedValue+"'and
planeendplace='"+this.EPDropDownList.SelectedValue+"' and
planest='"+this.STDropDownList.SelectedValue+"'and convert(varchar,planestarttime,120) like
'%"+this.STTextBox.Text+"%'");
```

```
                this.DataGrid1.DataBind();
                this.Datagrid2.Visible=false;
                con.Close();
            }
            if(this.RadioButtonList1.SelectedValue=="往返")//往返
            {
                this.DataGrid1.DataKeyField="planeid";
                SqlConnection con=Class.connection();//数据库连接
                con.Open();
                this.DataGrid1.DataSource=Class.ExeSelect("select * from plane join planeseattype
on plane.planeid=planeseattype.planeid where
planestartplace='"+this.SPDropDownList.SelectedValue+"'and
planeendplace='"+this.EPDropDownList.SelectedValue+"' and
planest='"+this.STDropDownList.SelectedValue+"'and convert(varchar,planestarttime,120) like
'%"+this.STTextBox.Text+"%'");
                this.DataGrid1.DataBind();
                this.Datagrid2.DataKeyField="planeid";
                this.Datagrid2.DataSource=Class.ExeSelect("select * from plane join planeseattype
on plane.planeid=planeseattype.planeid where
planestartplace='"+this.EPDropDownList.SelectedValue+"'and
planeendplace='"+this.SPDropDownList.SelectedValue+"' and
planest='"+this.STDropDownList.SelectedValue+"'and convert(varchar,planestarttime,120) like
'%"+this.ETTextBox.Text+"%'");
                this.Datagrid2.DataBind();
                this.Datagrid2.Visible=true;
                con.Close();
            }
```

4.订单界面

用户订单表的取消订单和确认订单按钮事件的代码

```
private void DataGrid1_ItemCommand(object source,
System.Web.UI.WebControls.DataGridCommandEventArgs e)
        {
            if(e.CommandName=="CancelOrder")//选择取消订单

            {
                string OrderId=this.DataGrid1.DataKeys[e.Item.ItemIndex].ToString();
                UserJudge u=(UserJudge)Session["user"];
            SqlConnection con=Class.connection();
                con.Open();
                SqlCommand com=new SqlCommand("select ordernumber from orders join users on
orders.userid=users.userid join plane on  orders.planeid=plane.planeid join planeseattype on
plane.planeid=planeseattype.planeid  where username='"+u.username+"' and
orders.planest=planeseattype.planest and orderid='"+OrderId+"'",con);
                string OrderNumber=com.ExecuteScalar().ToString();
                SqlCommand com1=new SqlCommand("select ordertime from orders join users on
```

```
orders.userid=users.userid join plane on  orders.planeid=plane.planeid join planeseattype on
plane.planeid=planeseattype.planeid  where username='"+u.username+"' and
orders.planest=planeseattype.planest and orderid='"+OrderId+"'",con);
                    string  OrderTime=com1.ExecuteScalar().ToString();
                if(OrderTime=="")//没有确认订单
                {

                    SqlCommand com2=new SqlCommand("delete from orders where
orderid='"+OrderId+"'",con);
                    //建立适配器
                    SqlDataAdapter sda = new SqlDataAdapter();
                    //建立dataset,用于填充
                    DataSet ds = new DataSet();
                    sda.SelectCommand = com2;
                    //传递查询结果
                    sda.Fill(ds,"orders");
                    this.DataGrid1.DataSource=ds.Tables["orders"];
                    this.DataGrid1.DataKeyField="orderid";
                    this.DataGrid1.DataBind();
                    this.DataGrid1.EditItemIndex=-1;//返回上一级页面
                    this.DataGrid1.DataSource=Class.ExeSelect("select * from users join orders on
users.userid=orders.userid join plane on  orders.planeid=plane.planeid join planeseattype on
plane.planeid=planeseattype.planeid  where username='"+u.username+"' and
orders.planest=planeseattype.planest ");
                    this.DataGrid1.DataKeyField="orderid";
                    this.DataGrid1.DataBind();
                }
                else{
                    SqlCommand com3=new SqlCommand("update planeseattype set
planenumber=planenumber+'"+OrderNumber+"' from planeseattype join plane on
planeseattype.planeid=plane.planeid join orders on plane.planeid=orders.planeid  where
planeseattype.planest=orders.planest and orders.planeid=planeseattype.planeid and
orderid='"+OrderId+"'",con);
                    com3.ExecuteNonQuery();
                    SqlCommand com4=new SqlCommand("delete from orders where
orderid='"+OrderId+"'",con);
                    //建立适配器
                    SqlDataAdapter sda = new SqlDataAdapter();
                    //建立dataset,用于填充
                    DataSet ds = new DataSet();
                    sda.SelectCommand = com4;
                    //传递查询结果
                    sda.Fill(ds,"orders");
                    this.DataGrid1.DataSource=ds.Tables["orders"];
                    this.DataGrid1.DataKeyField="orderid";
```

```
                this.DataGrid1.DataBind();
                this.DataGrid1.EditItemIndex=-1;//返回上一级页面
                this.DataGrid1.DataSource=Class.ExeSelect("select * from users join orders on
users.userid=orders.userid join plane on  orders.planeid=plane.planeid join planeseattype on
plane.planeid=planeseattype.planeid  where username='"+u.username+"' and
orders.planest=planeseattype.planest ");
                this.DataGrid1.DataKeyField="orderid";
                this.DataGrid1.DataBind();
            }
            con.Close();
        }

            if(e.CommandName=="SubmitOrder")//确认订单
            {

                    DateTime dt1 = DateTime.Now;
                    string OrderId=this.DataGrid1.DataKeys[e.Item.ItemIndex].ToString();
                    UserJudge user=(UserJudge)Session["user"];
                    SqlConnection con=Class.connection();//连接数据库
                    con.Open();
                    SqlCommand com=new SqlCommand("select ordernumber from orders join
users on orders.userid=users.userid join plane on  orders.planeid=plane.planeid join planeseattype on
plane.planeid=planeseattype.planeid  where username='"+user.username+"' and
orders.planest=planeseattype.planest and orderid='"+OrderId+"'",con);
                    string OrderNumber=com.ExecuteScalar().ToString();
                    SqlCommand com1=new SqlCommand("update planeseattype set
planenumber=planenumber-'"+OrderNumber+"' from planeseattype join plane on
planeseattype.planeid=plane.planeid join orders on plane.planeid=orders.planeid  where
planeseattype.planest=orders.planest and orders.planeid=planeseattype.planeid and
orderid='"+OrderId+"'",con);
                    com1.ExecuteNonQuery();
                    string OrderState=Convert.ToString("已确认订单");
                    SqlCommand com2=new SqlCommand("update orders set
ordertime='"+dt1+"', orderstate='"+OrderState+"' where orderid='"+OrderId+"'",con);
                    com2.ExecuteNonQuery();
                this.DataGrid1.DataSource=Class.ExeSelect("select * from users join orders on
users.userid=orders.userid join plane on  orders.planeid=plane.planeid join planeseattype on
plane.planeid=planeseattype.planeid  where username='"+user.username+"' and
orders.planest=planeseattype.planest ");
                this.DataGrid1.DataKeyField="orderid";
                this.DataGrid1.DataBind();
                    con.Close();
```

```
        }
    }
```

5.航班管理界面

指定的航班表的删除按钮事件的代码

```
private void DataGrid2_DeleteCommand_1(object source,
System.Web.UI.WebControls.DataGridCommandEventArgs e)
    {
        //删除指定航班

        SqlConnection con=Class.connection();
        con.Open();
        SqlCommand com=new SqlCommand("select planeid from plane where
planename='"+this.PlaneNameDropDownList.SelectedValue+"'",con);
        string planeId=com.ExecuteScalar().ToString();
        SqlCommand com1=new SqlCommand("delete from orders where
orders.planeid='"+planeId+"'",con);
        com1.ExecuteNonQuery();
        SqlCommand com2=new SqlCommand("delete from  planeseattype  from plane join
planeseattype  on plane.planeid=planeseattype.planeid   where
planename='"+this.PlaneNameDropDownList.SelectedValue+"'",con);
        com2.ExecuteNonQuery();
        this.DataGrid1.DataSource=Class.ExeSelect("select * from plane join planeseattype on
plane.planeid=planeseattype.planeid");
        this.DataGrid1.DataKeyField="planeid";
        this.DataGrid1.DataBind();
        SqlCommand com3=new SqlCommand("select * from plane",con);//重新绑定航班号
        SqlDataReader sdr=com3.ExecuteReader();
        this.PlaneNameDropDownList.DataSource=sdr;
        this.PlaneNameDropDownList.DataTextField="planename";
        this.PlaneNameDropDownList.DataValueField="planename";
        this.PlaneNameDropDownList.DataBind();
        sdr.Close();
        con.Close();
    }

    private void DataGrid2_CancelCommand_1(object source,
System.Web.UI.WebControls.DataGridCommandEventArgs e)
    {
```

```
            this.DataGrid2.EditItemIndex=-1;//取消更改信息
            SqlConnection con=Class.connection();
            con.Open();
            this.DataGrid2.DataSource=Class.ExeSelect("select * from plane join planeseattype on
plane.planeid=planeseattype.planeid where
planename='"+this.PlaneNameDropDownList.SelectedValue+"' ");
            this.DataGrid2.DataKeyField="planeid";
            this.DataGrid2.DataBind();
            con.Close();

        }
```

指定的航班表的编辑按钮事件的代码

```
private void DataGrid2_EditCommand_1(object source,
System.Web.UI.WebControls.DataGridCommandEventArgs e)
        {

            this.DataGrid2.EditItemIndex=e.Item.ItemIndex;//编辑指定航班
            SqlConnection con=Class.connection();
            con.Open();
            this.DataGrid2.DataSource=Class.ExeSelect("select * from plane join planeseattype on
plane.planeid=planeseattype.planeid where
planename='"+this.PlaneNameDropDownList.SelectedValue+"' ");
            this.DataGrid2.DataKeyField="planeid";
            this.DataGrid2.DataBind();
            con.Close();

        }
```

指定的航班表的更新按钮事件的代码

```
private void DataGrid2_UpdateCommand_1(object source,
System.Web.UI.WebControls.DataGridCommandEventArgs e)
        {
            //更新航班信息
            string PlaneId=this.DataGrid2.DataKeys[e.Item.ItemIndex].ToString();
            string PlaneName=((TextBox)(e.Item.Cells[0].Controls[0])).Text;
            string PlaneCompany=((TextBox)(e.Item.Cells[1].Controls[0])).Text;
            string PlaneType=((TextBox)(e.Item.Cells[2].Controls[0])).Text;
            string PlaneStartPlace=((TextBox)(e.Item.Cells[3].Controls[0])).Text;
            string PlaneEndPlace=((TextBox)(e.Item.Cells[4].Controls[0])).Text;
            string PlaneStartTime=((TextBox)(e.Item.Cells[5].Controls[0])).Text;
            string PlaneEndTime=((TextBox)(e.Item.Cells[6].Controls[0])).Text;
            string PlaneSeatType=((TextBox)(e.Item.Cells[7].Controls[0])).Text;
            int PlaneNumber=Convert.ToInt32(((TextBox)(e.Item.Cells[8].Controls[0])).Text);
            decimal  PlanePrice=decimal.Parse(((TextBox)(e.Item.Cells[9].Controls[0])).Text);
```

```
                DateTime starttime=Convert.ToDateTime(PlaneStartTime);
                DateTime endtime=Convert.ToDateTime(PlaneEndTime);
                SqlConnection con=Class.connection();//连接数据库
                con.Open();
                SqlCommand com=new SqlCommand("update plane set planename='"+PlaneName+"',
planecompany='"+PlaneCompany+"',planestartplace='"+PlaneStartPlace+"',planeendplace='"+PlaneEnd
Place+"',planestarttime='"+starttime+"',planeendtime='"+endtime+"' where
planeid='"+PlaneId+"'",con);
                com.ExecuteNonQuery();
                SqlCommand com1=new SqlCommand("update planeseattype set
planetype='"+PlaneType+"'where planeid='"+PlaneId+"'",con);
                com1.ExecuteNonQuery();
                SqlCommand com2=new SqlCommand("update planeseattype set
planenumber='"+PlaneNumber+"', planeprice='"+PlanePrice+"'  from planeseattype join plane on
planeseattype.planeid=plane.planeid where planeseattype.planeid='"+PlaneId+"'and
planest='"+PlaneSeatType+"'",con);
                com2.ExecuteNonQuery();
                this.DataGrid2.EditItemIndex=-1;//返回上一级页面
                this.DataGrid2.DataSource=Class.ExeSelect("select * from plane join planeseattype on
plane.planeid=planeseattype.planeid where plane. planeid='"+PlaneId+"' ");
                this.DataGrid2.DataBind();
                SqlCommand com3=new SqlCommand("select * from plane",con);//重新绑定航班号
                SqlDataReader sdr=com3.ExecuteReader();
                this.PlaneNameDropDownList.DataSource=sdr;
                this.PlaneNameDropDownList.DataTextField="planename";
                this.PlaneNameDropDownList.DataValueField="planename";
                this.PlaneNameDropDownList.DataBind();
                sdr.Close();
                con.Close();
        }
```

指定的航班表的取消按钮事件的代码

```
private void DataGrid2_CancelCommand_1(object source,
System.Web.UI.WebControls.DataGridCommandEventArgs e)
        {
                this.DataGrid2.EditItemIndex=-1;//取消更改信息
                SqlConnection con=Class.connection();
                con.Open();
                this.DataGrid2.DataSource=Class.ExeSelect("select * from plane join planeseattype on
plane.planeid=planeseattype.planeid where
planename='"+this.PlaneNameDropDownList.SelectedValue+"' ");
                this.DataGrid2.DataKeyField="planeid";
                this.DataGrid2.DataBind();
                con.Close();
```

```
}
```

第六章　总结

6.1 本系统的特色

本系统将查询航班和预定机票放在一个界面上，这样就大大提高了网站的运行速度，节省了不少资源，使用户查询和预定航班更快捷。同时系统可以为未注册的用户提供查询航班，查看实时航班信息和热门航班的服务，为他们的出行提供了一个很好的参考。

6.2 该系统还需要进一步完善的地方

1.本系统没有实现在线支付功能

2.本系统的界面的艺术感还不是很强

3.本系统没有实现为用户提供订票优惠的功能

结束语

毕业设计是我在大学期间的最后一项作业，同时也是对我在四年大学期间所学的理论知识一次很好的实践机会。我将这四年学到的计算机理论知识运用到这次毕业设计中，提高了自己的实际动手能力，并且使我的理论知识得到了巩固和加强，为以后的工作和学习打下一个良好的基础。俗话说“实践是检验真理的唯一标准”，以前我也看过一些关于 ASP.NET 的书籍，但是很少在计算机上实际操作一下，因此在实际编码过程中遇到了很多的问题。每当我遇到困难的时候，我便会去网上搜索相关资料，去图书馆借阅相关书籍，在计算机上反复进行试验，找寻解决问题的办法，有时为了解决一个问题我会忘记睡觉，一直工作到深夜。如果实在是自己解决不了的问题，我会向我的指导师李艳平老师求助，最终在自己不屑的努力和李艳平老师的大力帮助下，完成了这次毕业设计，并在实际测试中达到了预期的目标。毕业设计是我在大学期间完成的规模最大的一个项目，我深深地体会到了流程图的重要性，设计一个好的流程图，会大大缩减编码的时间，并且不会产生逻辑错误，造成不可弥补的损失。因为这次毕业设计的时间很紧，任务很重，而且自己掌握的理论知识和实际技术也不是很多，所以我编写的民航订票管理系统还存在着一些不足的地方，有待进一步的提高和完善。

最后我要衷心的感谢在毕业设计中帮助过我的李艳平老师和我身边的同学们，如果没有你们耐心和细致的帮助，我不可能如此顺利的完成毕业设计，谢谢大家！

参考文献

[1]沈阳，李勇敢编著．ASP.NET 程序设计教程[M]．北京：电子工业出版社， 2006：23-56．

[2]杨天奇，王文，何朋，李会锋编著．ASP.NET 网络编程[M]．北京：机械工业出版社，2007：67-102．

[3] 翁健红编著．基于 C#的 ASP.NET 程序设计[M]．北京：机械工业出版社，2007：35-58．

[4] 武新华，秦连清等编著．ASP.NET+SQL 数据库案例精粹[M].西安：西安电子科技大学出版社,2007：78-92．

[5]李兰友，杨晓光编著．ASP.NET 实用程序设计[M].北京：清华大学出版社，2006：125-178．

[6]萨师煊，王珊编著． 数据库系统概论第三版[M]．北京：高等教育出版社，2005：84-128．

[7]杜亮编著.亲密接触 ASP.NET[M].北京：清华大学出版社，2002：156-217．

[8] 张文仲编著．ASP.NET 网络开发技术[M].北京：人民邮电出版社，2006：53-96．

[9] Alex Homer,Dave Sussman 编著．ASP.NET2.0 技术详解[M].北京：人民邮电出版社，2007：232-289．

[10]Kari1 Waston,Christian Nage1 等编著． C#入门经典（第三版）[M]．北京： 清华大学出版社，2006：310-416．

[11]Chris UIIman,John Kauffman 等编著． ASP.NET1.1 入门经典(Visua1 C#.NET2003 编程篇)[M]．北京： 清华大学出版社，2004：123-312．