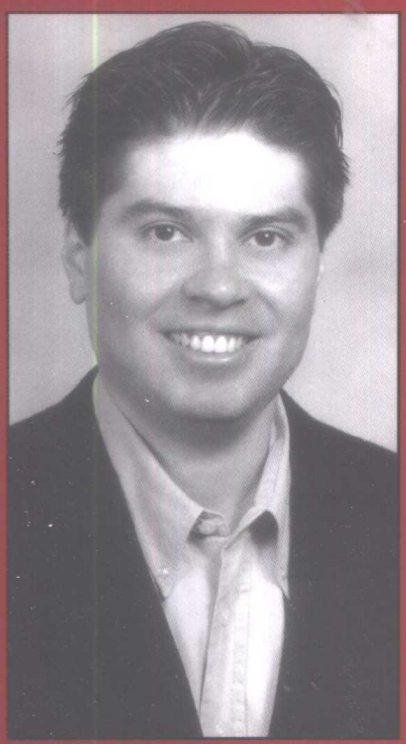
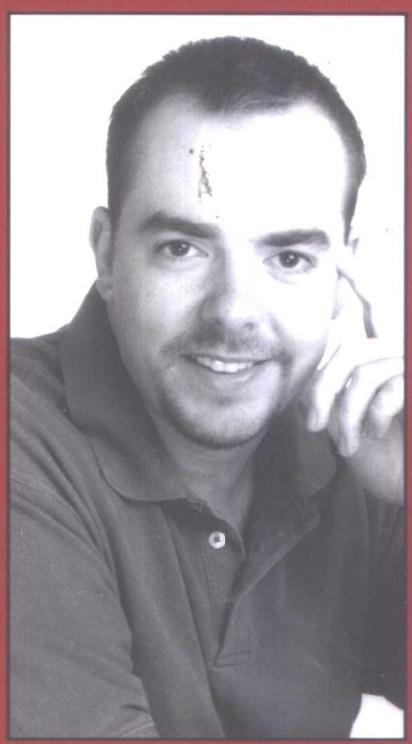
PROGRAMMER TO PROGRAMMER™





BEGINNING Visual Basic .NET Databases

Visual Basic .NET

数据库开发入门经典

Bill Forgey
Denise Gosnell
Matthew Reynolds 著 康博 译

清华大学出版社
http://www.tup.tsinghua.edu.cn

Visual Basic .NET 数据库开发入门经典

Bill Forgey
Denise Gosnell 著
Matthew Reynolds
康 博 译

清华大学出版社

(京) 新登字 158 号
北京市版权局著作权合同登记号：01-2002-3002

内 容 简 介

众所周知，几乎所有的应用程序都必须进行数据访问。通过对本书的学习，您将了解到该如何构建能够有效使用数据库进行数据访问的 Visual Basic .NET 应用程序。

本书主要介绍了数据库设计的基本原理，如何查询数据库、如何在 Windows 应用程序中访问数据库中的数据，以及如何使用 Internet 和 Web 服务来进行远程数据访问。

本书适用于具有一定编程经验的准备开发数据库应用程序的编程人员。

书　　名：Visual Basic .NET 数据库开发入门经典
作　　者：Bill Forgey, Denise Gosnell, Matthew Reynolds 著　康博　译
出 版 者：清华大学出版社(北京清华大学学研大厦，邮编 100084)
http://www.tup.tsinghua.edu.cn
责任编辑：徐燕萍
封面设计：康博
版式设计：康博
印 刷 者：北京市清华园胶印厂
发 行 者：新华书店总店北京发行所
开　　本：787×1092 1/16　**印张**：37.5　**字数**：959 千字
版　　次：2002 年 7 月第 1 版　2002 年 7 月第 1 次印刷
书　　号：ISBN 7-302-05564-5/TP・3285
印　　数：0001～5000
定　　价：65.00 元

出版者的话

随着国际互联网的快速崛起和迅猛发展，计算机之间的互联需求越来越迫切，而目前计算机硬件设备的不兼容性严重束缚了互联网的发展，引发了新一轮的跨平台软件的开发浪潮。软件商纷纷推出新的战略规划和解决方案，Microsoft 提出的.NET 战略就是其中的经典之作。

在经历这场浪潮的洗涤和考验过程中，全球的软件开发人员都迫切需要了解新的软件技术和开发思路。为了满足国内 IT 从业人员的需要，清华大学出版社从 Wrox 出版公司引进了若干套编程系列丛书，“入门经典”系列是其中不可或缺的入门之作。作为世界著名的编程技术图书的出版公司，Wrox 推出的这套“入门经典”系列丛书主要面向编程的初学者、需要了解.NET 策略的程序员，以及需要迅速掌握多门编程工具的程序员。该丛书依旧秉承了 Wrox 公司“由程序员为程序员而著（Programmer to Programmer）”的创作理念，每本书均由世界顶级的编程高手执笔。他们站在资深程序员的高度，循序渐进地为初学者讲述了.NET 的理念和构架、编程基本思想、编程语言基础、程序的控制和调试、Windows 应用程序的开发、对象编程技术、数据库访问技术、Web 程序开发和.NET 构架应用等最新的软件开发知识，同时辅以大量操作性强的程序为示例，为读者提供了清晰的编程思路和宝贵的编程经验。

为了保证该系列丛书的质量，清华大学出版社迅速组织了一批位于 IT 开发领域前沿的专家学者进行翻译，经过编辑人员的进一步加工整理后，现陆续奉献给广大读者。

读者可以从 www.wrox.com 网站下载所需的源代码和获取相关的技术支持。同时，也欢迎广大读者参与 p2p.wrox.com 网站上的在线讨论，与世界各地的编程人员交流读书感受和编程体验。

前　言

所有软件都是建立在数据处理这一基础之上。无论是嵌入到 VCR(录像机)中定时录音的程序，还是空中交通管制软件，程序总是以某种形式处理着数据。

现在我们知道，高级的应用软件都是将其数据存储在“数据库”中的，这是一个由数据库管理系统(即 DBMS)监控的中央数据仓库。DBMS 要做两件事：首先是处理数据的存储工作。再者是提供一种机制，用于检索、添加、删除和修改数据。DBMS 将尽力用最有效的可行方法来完成这项任务。

在过去的几年中，DBMS 市场已经凭借其自身的优势成长为一个成熟完善的产业。现在它不仅能提供用于大型企业环境的产品，例如 Oracle 9i 或 Microsoft SQL Server 2000 等，也能为桌面应用提供产品，如 Microsoft Access。在某些应用实例中，您甚至可以发现软件包中包含有它们自己的 DBMS 软件，可用来管理其专有数据库。

作为一名程序员，您会发现应用程序经常需要访问由 DBMS 管理的数据。事实上，您也可以发现，使用 DBMS 是存储和处理应用程序数据的最简易方式。然而，面对各种数据库产品，我们怎样才能编写出一个通用的应用程序，能兼容用户所选的任一个数据库呢？

解决这个问题的诀窍在于构建一种能够处理某种数据访问层的应用程序。我们可编写与数据访问层交互的程序，而不是编写专门需要某个特定 DBMS 的程序。而数据访问层的职责是负责切换到 DBMS 自身所使用的“本地”调用。Microsoft 把这种层称之为“通用数据访问”或 UDA。Microsoft 最新的 UDA 工具是 ADO.NET，这是一组综合的对象，它们协同工作共同构成了数据访问层。

本书完整地讲解了如何利用 ADO.NET 的强大功能构建 Visual Basic .NET 应用程序。我们将介绍如何以各种不同的方式使用这项技术，这些方式有：使用 Windows Forms 的桌面应用程序、使用 ASP.NET 的 Web 应用程序，以及 Web 服务等。

本书读者

本书针对那些具备一定 Visual Basic .NET 经验，准备设计数据库应用程序的程序员而编写。如果您具备一定的 Access 经验将会更好，但这不是一定要具备的。

请注意，本书不是一本 Visual Basic .NET 的入门教材，如果您对 Visual Basic .NET 完全陌生的话，请参阅清华大学出版的《Visual Basic .NET 入门经典》一书。

如果您是一位有经验的 VB6 程序员，想迅速地升级到 VB.NET，那么最好参阅清华大学出版社的《Visual Basic .NET 高级编程》一书。

本书内容

Visual Basic .NET 是与非常复杂而又灵活的数据访问技术紧密联系在一起的，所以可以归入本书讨论主题范围的内容非常之多，为了避免涉及太多的内容，我们将重点介绍如下主题：

- 基本的数据库设计原理
- SQL Server 桌面引擎
- 使用 T-SQL 查询数据库
- 使用 Visual Studio .NET 的 Server Explorer 运行查询、视图和存储过程等
- ADO.NET 和数据集对象
- 将数据读入数据集中，将其捆绑在用户界面的一个控件上，改变数据集中的数据，并将这种改变保存在底层的数据库中
- XML 在 ADO.NET 中的角色
- 使用 Web Forms 和 Web 服务的 Internet 数据库应用程序

本书使用环境

使用本书要求计算机运行下列软件：

- Windows 2000、XP 或者 NT 4 Server
- 由 Windows 2000 和 Windows XP 提供的 IIS 5
- Internet Explorer
- Access XP 或 2000
- Visual Studio .NET 专业版(或 Visual Studio 的更高版本。例如，.NET 的企业版将运行得更好。然而，编写本书时，这些版本还无法得到，所以本书是采用专业版编写的)。
- SQL Server 2000 桌面引擎，它附带在 Visual Studio .NET 中。

本书成书于 Visual Studio .NET 正式版发布之前。如果本书中的用法和处理 Visual Studio .NET 最终版本所必需的用法之间，如果存在实质性的变化，我们将在 Wrox 在线勘误表服务中提供免费升级。

用户支持

我们一贯重视读者的意见，并想知道每位读者对本书的看法，包括读者喜欢和不喜欢的内容，以及读者希望我们下一次能够完善的地方。您可以发送电子邮件(地址为 feedback@wrox.com)向我们反馈意见。请确保反馈信息提到本书的书名。

范例代码

当您访问 Wrox 公司站点(地址为 http://www.wrox.com/)时，通过 Search 工具或书名列表，可以方便地定位您所需要的书目。然后，单击 Code 列中的 Download 超链接，或者单击本书的具体页面中的 Download Code 超链接，就可以下载相应的范例代码。

从我们的站点中下载的文件都是使用 WinZip 压缩过的文档。将附件保存到本地磁盘上的文件夹中后，需要使用解压程序(例如 WinZip 或 PKUnzip)来解压文件。在解压文件时，通常将代码解压到每一章所在的文件夹中。在解压的过程中，应确保解压程序(WinZip、PKUnzip 等等)解压时被设置为使用原有文件夹名。

勘误表

我们已经尽最大努力确保本书中的文本和代码没有错误，但是错误肯定还是在所难免。如

果您发现本书存在错误，例如拼写错误或不正确的代码片段，请给我们反馈信息，我们将不胜感激。勘误表的发送可以节约其他读者学习本书的时间，而且能够帮助我们提供更高质量的信息。我们将检查您的反馈信息，如果正确，将被粘贴到本书的勘误页面上，或者在本书的后续版本中使用。

要在我们的站点上找到勘误表，请访问 http://www.wrox.com/，并通过 Advanced Search 或者书名列表轻松定位本书页面。然后，单击 Book Errata 超连接即可。

E-mail 支持

如果您希望直接向了解本书的专家咨询本书中的问题，可以发送电子邮件到 support@wrox.com，要求在邮件的主题中带上本书的书名和 ISBN(国际标准书号)的后 4 位数字。一个典型的电子邮件应包括下面的内容：

- 在主题中必须有本书的书名、ISBN 的后 4 位数字和问题的页数。
- 信息的主题应包括读者的名字、联系信息和问题。

我们将不返回您的无用邮件，因为我们仅仅需要有用的详细资料，以便可节约您和我们的时间。当您发送一个电子邮件信息时，它将得到下面这些支持：

- 用户支持：首先，您的信息将被送到我们的用户支持人员手中，由他们先进行阅读。归档一些被频繁提到的问题，并立即回答有关本书或者 Web 站点的任何常见问题。
- 编辑支持：接着，一些有深度的问题将被送到对本书负责的技术编辑手中，他们在程序设计语言或者特定的产品上有着丰富的经验，能够回答相关主题的详细技术问题。
- 作者支持：最后，如果编辑不能回答您的问题(这种情况很少发生)，他们将请求本书的作者。我们将尽量保护作者免受干扰，以便不影响其写作。然而，我们也非常高兴转寄给他们一些特殊的问题。所有 Wrox 公司的作者都为他们的书提供技术支持。作为回应，他们将发送电子邮件给用户和编辑，从而使所有的读者受益。

Wrox 公司的支持过程仅仅对那些与我们出版的书目内容直接相关的问题提供支持，对于超出常用书目支持的问题，您可以从 http://p2p.wrox.com/ forum 中的公共列表中获得支持信息。

p2p.wrox.com 站点

为了便于作者和其他人讨论，特将讨论内容加入到 P2P 站点的邮件列表中，而且我们惟一的系统将在邮件列表、论坛、新闻组以及所有其他服务(一对一的邮件支持系统除外)上提供 programmer to programmer™(由程序员为程序员而作)联系。如果您向 P2P 发送一个问题，它一定会被登录邮件列表的 Wrox 公司作者和其他相关专家检查到。无论您是在阅读本书，还是在开发自己的应用程序，都可以在 p2p.wrox.com 站点中找到许多对自己有所帮助的邮件列表。

按照下面的步骤可以预定一个邮件列表：

(1) 登录 http://p2p.wrox.com/站点。

(2) 从左边的主菜单栏选择一个适当的类别。

(3) 按照说明订阅并填写自己的邮件地址和密码。

(4) 回复您收到的确认邮件。

(5) 使用预定管理程序加入更多的邮件列表并设置自己的邮件首选项。

本系统提供最好支持的原因

您可以选择连接整个邮件列表，也可以只接收每周的邮件摘要。如果您没有时间和工具来接收邮件列表，可以直接查找我们的在线文档。独特的 Lyris 系统可以将一些没用的垃圾邮件删除，并保护您的电子邮件地址不被侵扰。出现加入和离开列表、以及任何有关列表的其他常见问题时，请发送邮件到 listsupport@p2p.wrox.com。

目　录

第1章　关系型数据库设计

本章将介绍一些设计和实现数据库的背景知识。绝大多数的应用程序，无论是用 Visual Basic .NET 还是用其他编程语言开发的，都包含有一定存储容量的数据库，所以，深入理解优秀数据库的设计原理是至关重要的。在对数据库的整体情况进行简短介绍之后，本章将开始设计和实现一个具体类型的数据库——关系型数据库。不必担心能否这么快就理解所有的数据库术语，到本章结束时，你将清楚地理解下列问题：

- 数据库的概念
- 关系型数据库和平面文件数据库间的比较
- 关系型数据库的优点
- 如何分析业务需求以确定数据库中应包含的信息
- 基于特定的业务需求，如何确定数据库中需要包含的适当元素
- 如何定义键码和关系
- 数据规范化的目标以及可以带来哪些好处
- 如何定义索引
- 综合这些知识创建物理数据库

最后，在设计关系型数据库时，我们再回顾一下本章的重要内容。

1.1　数据库的概念

从本质上讲，数据库是一种用有组织的方式存储数据的电子化手段。数据可以是企业或个体需要记录的任何东西，而在计算机出现之前，它们只能被记录在纸张上。数据一旦被存储进来后，数据库中的数据就可以被程序作为信息进行检索、处理并显示给浏览者。用来存储数据的实际数据库结构可以有多种不同的形式。在检索和修改信息时，每一种结构都有其特定的优势。在下一节，我们将讨论用平面文件结构和关系型数据库结构存储数据时的区别，然后还要讨论每一种方法的优劣所在。

1.1.1　平面文件与关系型数据库

平面文件是数据库的最基本形式——所有的信息都存储在单一的文件中。平面文件为用户所需存储的每个信息项提供一个字段。虽然平面文件很容易创建，但效率不是太高。由于包含了大量的重复信息，这种文件确实也浪费了大量的存储空间，在多个文件都包含关联信息的复杂系统中更是如此。这将加大信息维护和检索的难度。以前你使用过的电子数据表就是平面文件数据库的一个最常见的例子。为了进一步说明该如何组织平面文件中的数据，及为什么这种结构可能会引发问题，我们来假设一个例子。

假设你使用电子数据表记录客户的订单，如表 1-1 所示：

表 1-1

订单编号	订货日期	订货说明	数量	单 位	价格	客户名字	客户地址
1000	1-Aug-01	Tofu	1	40 - 100 g pkgs	23.25	Jane Doe	123 Somewhere St., Anytown, IN 46060 USA
1000	1-Aug-01	Jack's New England Clam Chowder	1	12 - 12 oz cans	9.65	Jane Doe	123 Somewhere St., Anytown, IN 46060 USA
1000	1-Aug-01	Grandma's Boysenberry Spread	3	12 - 8 oz jars	25	Jane Doe	123 Somewhere St., Anytown, IN 46060 USA
1001	2-Aug-01	Uncle Bob's Organic Dried Pears	1	12 - 1 lb pkgs	30	John Smith	345 Anywhere St., Somewhere, IN 46001 USA
1001	2-Aug-01	Tofu	1	40 - 100 g pkgs	23.25	John Smith	345 Anywhere St., Somewhere, IN 46001 USA

请注意这个电子数据表是如何容纳订单和客户信息的，例如，Jane Doe 订购了编号为 1000 的 ToFu(食品名称，下同。译者注)，还有 Jack 的 New England Clam Chowder、Grandma 的 Boysenberry Spread。每个数据项都列在电子数据表的一个单独的行中。进一步还可看出，正如上面订单中灰色部分指示的那样，订单编号、订单日期，连同 Jane Doe 的名字和地址被多次列出，因为她们订购了多种产品而导致其信息被存储了好多次。

所以说订单编号、订单日期、客户名字和客户地址字段包含了冗余的信息，换句话说，相同的信息重复出现在了若干个地方。冗余的信息造成了数据库的臃肿，而这正是因为它包含了由相同信息组成的多个数据项。在电子数据表中记录订单信息时，这也将造成额外的工作量，这完全是相同信息被重复记录之过。更为不幸的是，多次输入这些信息将极大地增加出错的机会，例如名字或地址拼写错误。

使用平面文件的另外一个问题就是维护。那将出现什么情况呢？例如，Jane Doe 迁居了，你就必须在电子数据表中修改她的地址，如果使用这种平面文件形式，那么，你就不得不多次修改她的地址——因为每次订货都会留下地址信息。如果她恰好是一个喜欢购物的客户，那就意味着上百处都需要修改。如果她的地址仅被保存在一个地方，那么只需修改一处就可

以了，但是在这个例子中却并非如此。在这个简单的例子中，你已经亲眼目睹了这种平面文件数据库的一些最普遍的问题：数据冗余和额外的维护需求。

在理解了平面文件数据库的概念，并且意识到了这种形式可能会在什么地方引起问题后，我们再来讨论一种克服了这些缺点的数据库类型——关系型数据库。用最简单的术语表示就是：关系型数据库可以被看作是信息项的集合，这些信息项被分解到不同组中，这些组以一种或多种方式相互关联。用数据库的术语表示，这些组常被称为表。这个概念听起来挺复杂的，但事实上不难理解。修改一下先前的示例，看一下如果使用关系型数据库它将会怎么样，很快你就会发现，大多数的概念并非复杂得难以理解。

回想一下在平面文件电子数据表中存储订单和客户信息的情景。每一笔订单都是由多个订单数据项组成的，并且每位客户发出多笔订单。关系型数据库在存储这样的信息时，会将其拆分到 3 个单独的表格中：Customers、Orders 和 OrderItems。如表 1-2 所示：

表　1-2

Customers	Orders	OrderItems
Customer_Id	Order_Id	Item_Id
Customer_First_Name	Customer_Id	Order_Id
Customer_Last_Name	Order_Date	Item_Description
Customer_Address1		Quantity_Ordered
Customer_City		Item_Price
Customer_State		Quantity_Per_Unit
Customer_Zip		
Customer_Country		

Customers 表为每个客户保存单独的记录项。Orders 表为每笔订单保存单独的记录项。最后，OrderItems 表为订单中的每一条产品订购保存单独的记录项。它的含义是，每一笔订单中可以包含一条或多条产品订购记录。这样，客户信息就从订单中分离出来而被单独保存了，订单的每一条产品订购记录也从订单中分离出来而被单独保存。我们可以看到，Orders 表中包含了一个 Customer_ID 字段，这个字段和 Customers 表中的 Customer_ID 字段相关联。我们还可以看到，OrderItems 表中包含了一个 Order_ID 字段，它可以和 Orders 表中的 Order_ID 字段相关联。我们将在本章的“定义表之间的关系”一节中讨论如何使表之间产生关联。目前，我们只需知道它是一种用来消除数据冗余的机制，可以解决在平面文件形式下看到的重复的客户名字和地址等问题。关系型数据库中没有这样的重复数据。例如，想修改 Jane Doe 的地址时，我们只需修改她在 Customers 表中惟一的数据项即可。更妙的是，当 Jane Doe 订购产品时，我们不用重复输入她的地址。如果她曾经在这里订过货，那么她的详细资料就已