混合编程之案例详解

Visual C++＋SQL Server 数据库应用实例完全解析

启明工作室　编著

人民邮电出版社

图书在版编目（**CIP**）数据

Visual C++ ＋SQL Server 数据库应用实例完全解析 / 启明工作室编著.
—北京：人民邮电出版社，2006.4

ISBN 7-115-14633-0

Ⅰ. V… Ⅱ. 启… Ⅲ. ①C 语言—程序设计②关系数据库—数据库管理系统，SQL Server
Ⅳ. ①TP312②TP311.138

中国版本图书馆 CIP 数据核字（2006）第 024976 号

内 容 提 要

本书首先详尽介绍了使用 Visual C++和 SQL Server 开发数据库应用程序应具备的各项技术，然后详细介绍了 7 个流行的系统的开发实例。在本书光盘中还为每个实例制作了 PPT 文件，详细介绍和演示实例所需的实用技术和开发过程，方便读者学习使用。

本书案例贴近实际，内容精炼，适合大中专院校的学生和软件开发人员参考使用。

混合编程之案例详解

Visual C++＋SQL Server 数据库应用实例完全解析

◆ 编　　著　启明工作室
责任编辑　刘　浩

◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街 14 号
邮编　100061　　电子函件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京顺义振华印刷厂印刷
新华书店总店北京发行所经销

◆ 开本：787×1092　1/16
印张：21.5
字数：527 千字　　2006 年 4 月第 1 版
印数：1 – 6 000 册　　2006 年 4 月北京第 1 次印刷

ISBN 7-115-14633-0/TP・5316

定价：36.00 元（附光盘）

读者服务热线：（010）67132692　印装质量热线：（010）67129223

光 盘 说 明

在随书附赠的光盘中，提供了本书实例程序的全部代码。为了便于读者使用光盘中的实例，作者在光盘中按照章名创建每一章节实例的目录。例如，第 2 章中出现的实例将被存储在“X:\第 2 章”目录下（X: 是光盘驱动器号）。在每个实例的目录下，包括一个实例代码目录和数据库备份文件。在相关章节介绍到一个实例时，会明确地说明此实例的存储目录。

在随书附赠的光盘中，还为每个实例制作了一个 PPT 文件，详细介绍和演示实例所需的实用技术和开发过程。第 2 章实例的 PPT 文件名为“第 2 章.PPT”，以此类推。

光盘中不包括 SQL Server 的安装程序。

前　言

数据库应用系统开发是当前最流行的实用技术之一，大多数大型应用系统都需要后台数据库的支持。本书详尽介绍了使用 Visual C++和 SQL Server 开发数据库应用程序所必备的管理及开发技术。在实际应用中，许多大中型网络环境下的应用程序都使用 SQL Server 作为后台数据库，而 Visual C++是非常流行的可视化开发语言，具有功能强大、通用性强和易于扩充等特点。因此 Visual C++和 SQL Server 数据库是开发应用系统时使用广泛的组合。

本书介绍了开发数据库应用系统必须掌握的技术，包括 SQL Server 数据库的管理、SQL 语句和 ADO 数据库访问技术。这些技术会在本书的 7 个实例中得到应用。通过技术与实例相结合的讲解方式，读者可以很快掌握数据库应用系统开发的过程和方法，并将所学技术应用到实际工作中。书中选择的 7 个实例都具有很强实用价值，是当前应用比较广泛的数据库应用系统。

与其他实例类图书相比，本书更注重系统开发过程的完整性。笔者在介绍数据库应用系统实例的过程中，并不是简单讲解实例功能和代码，而是把每个实例作为一个项目来实现。书中的每个实例都按照开发项目的步骤进行介绍，包括系统需求分析、系统体系结构设计、开发工具的选择、开发实例所需的实用技术、数据库结构设计、工程框架设计以及各功能模块的设计等。按照此步骤阅读本书实例，读者可以体验一个项目从无到有、逐步实现的过程。而在实际系统开发过程中，认真完成每个步骤，是项目能够顺利完成的保障。

本书在介绍实例的过程中，力图把每个实例作为一个课题进行研究。本书从了解此课题所需的基本技术着手，分析选择开发工具的方法和系统的实现过程，引导读者完成课题的研究工作。本书还为读者推荐了进一步研究的方向，例如实现数据库开发的三层体系结构和使用更加安全可靠的 PKI 认证体系等，从而使课题的研究更加完整。

在本书附赠的光盘中，除了提供各实例的源代码外，还为每个实例制作了 PPT 文件，详细介绍和演示实例所需的实用技术和开发过程，更加方便读者学习和理解。

本书以程序开发的实用技术为中心，注重提高读者的系统设计和开发能力。读者只要具备基本的 Visual C++编程知识，即可轻松掌握书中内容。

参加本书编写的还有：李晓黎、张晓辉、李树有、刘熙芳、张志祥、刘熙凤、李晓喆、张巍等。由于作者水平有限，书中难免有不足之处，敬请广大读者批评指正（Email：book_better@sina.com）。

编者

2006 年 4 月

目　录

第 1 章　数据库编程实用技术 …… 1

1.1　SQL Server 数据库管理…… 1
1.2　常用 SQL 语句介绍…… 9
1.3　ADO 数据访问技术…… 14
1.4　Visual C++数据库访问控件 …… 21
1.5　数据库应用系统工程规划…… 33

第 2 章　人事工资管理系统…… 40

2.1　概述 …… 40
2.2　系统分析与总体设计…… 40
2.3　设计工程框架 …… 46
2.4　部门管理模块设计…… 56
2.5　员工信息管理模块设计…… 64
2.6　员工照片管理模块设计…… 71
2.7　员工工资管理模块设计…… 77
2.8　用户管理模块设计…… 95

第 3 章　固定资产管理系统…… 99

3.1　概述 …… 99
3.2　系统分析与总体设计…… 99
3.3　设计工程框架 …… 104
3.4　基本信息管理模块设计……113
3.5　固定资产管理模块设计……116
3.6　用户管理模块设计…… 138

第 4 章　考勤管理系统…… 139

4.1　概述 …… 139
4.2　系统分析与总体设计…… 139
4.3　设计工程框架 …… 144
4.4　基本信息管理模块设计…… 152
4.5　考勤管理模块设计…… 160
4.6　统计查询模块设计…… 173
4.7　用户管理模块设计……176

第 5 章　汽车维修管理系统 ……177

5.1　概述……177
5.2　系统分析与总体设计 ……177
5.3　设计工程框架……183
5.4　基本信息管理模块设计 ……192
5.5　业务信息管理模块设计 ……201
5.6　业务统计模块设计 ……211
5.7　用户管理模块设计 ……214

第 6 章　学校收费管理系统 ……215

6.1　概述……215
6.2　系统分析与总体设计 ……215
6.3　设计工程框架……221
6.4　基本信息管理模块设计 ……230
6.5　学生收费管理模块设计 ……244
6.6　查询统计模块设计 ……252
6.7　用户管理模块设计 ……255

第 7 章　电脑报价管理系统 ……256

7.1　概述……256
7.2　系统分析与总体设计 ……256
7.3　设计工程框架……262
7.4　基本信息管理模块设计 ……271
7.5　配件管理模块设计 ……277
7.6　报价管理模块设计 ……282
7.7　统计查询模块设计 ……291
7.8　用户管理模块设计 ……293

第 8 章　医院收费管理系统 ……294

8.1　概述……294
8.2　系统分析与总体设计 ……294

8.3 设计工程框架 …… 300
8.4 基本信息管理模块设计…… 308
8.5 信息管理模块设计…… 312
8.6 收费管理模块设计……320
8.7 数据统计管理模块设计……332
8.8 用户管理模块设计……336

第 1 章　数据库编程实用技术

本章将介绍后面实例中使用到的 Visual C++数据库编程实用技术。使用 Visual C++开发数据库应用程序需要掌握如下基础技术：

- 基本的 Visual C++编程方法；
- 基本的后台数据库管理方法，例如创建数据库、创建表、创建视图以及备份和还原数据库等；
- 常用 SQL 语句的使用；
- 在 Visual C++使用 ADO 数据库访问技术。

本书假定读者掌握了基本的 Visual C++编程方法，因此只对上述后 3 项技术进行介绍。

1.1　SQL Server 数据库管理

本书实例采用 SQL Server 2000 作为后台数据库，这一小节将介绍管理 SQL Server 数据库的基本方法，从而为读者理解实例的实现过程奠定基础。

1.1.1　企业管理器

企业管理器是 SQL Server 2000 数据库系统中最重要的管理工具，可以在企业管理器中对 SQL Server 数据库进行各种操作。

依次单击“开始”\“程序”\“Microsoft SQL Server”\“企业管理器”，运行 SQL Server 2000 企业管理器。企业管理器的主界面如图 1.1 所示。

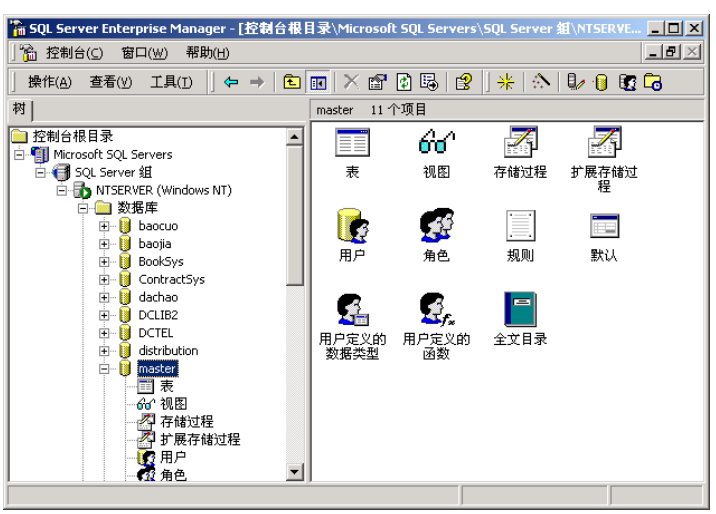


图1.1　SQL Server企业管理器

企业管理器的窗口分为两个部分，左侧窗口使用树状结构描述了 SQL Server 2000 的逻辑结构，右侧窗口显示左侧窗口中选择对象的具体内容。SQL Server 组可以包含多个 SQL Server 服务器，右键单击“SQL Server 组”，并选择菜单中的“新建 SQL Server 注册”，就可以将 SQL Server 服务器添加到树中，从而实现对网络中所有 SQL Server 数据库的管理。

一个 SQL Server 服务器下可以创建多个数据库，展开“数据库”节点，将显示指定服务器中的所有 SQL Server 数据库。数据库由表、视图、用户等数据库对象组成，在企业管理器中，可以添加、修改、删除和查看这些数据库对象。

1.1.2 创建数据库

在创建数据库之前，必须先确定数据库的名称、所有者（创建数据库的用户）、大小以及用于存储该数据库的文件和文件组。除了数据库名称外，其他参数都可以由系统默认设置。

打开 SQL Server 企业管理器，可以按照下面的方法来创建数据库。

（1）展开服务器组，然后展开服务器实例。

（2）右击“数据库”，然后单击“新建数据库”命令，打开“数据库属性”窗口，如图 1.2 所示。

（3）键入新数据库的名称，假定数据库名为 UserMan。默认情况下，系统自动地使用指定的数据库名作为前缀创建主数据库和事务日志文件，例如 UserMan_Data.mdf 和 UserMan_Log.ldf。

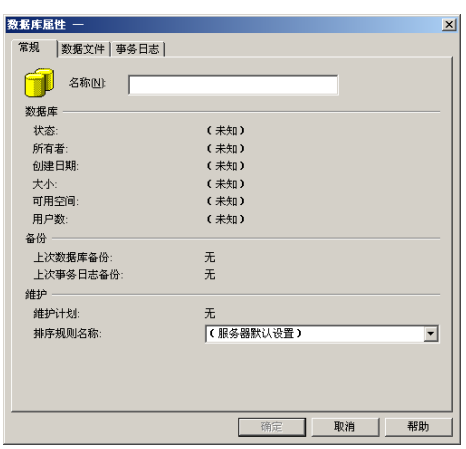


图 1.2 数据库属性

（4）如果需要设置数据文件和事务日志文件，可以分别单击“数据文件”和“事务日志”选项卡。建议读者采用默认设置。

（5）单击“确定”按钮，在企业管理器中可以看到新建的数据库。

1.1.3 创建表

表是包含数据库中所有数据的数据库对象。在设计数据库结构时，最主要的工作就是决定

数据库由哪些表组成，然后定义每个表的结构。

表的结构由若干个列（字段）组成。在定义表结构时，需要指定列名、数据类型、长度和其他列属性。

SQL Server 2000 的常用数据类型如表 1.1 所示。

表 1.1　　SQL Server 2000 常用数据类型说明

数据类型	描述
Image	可以用来存储超过 8 KB 的可变长度的二进制数据
Char	固定长度的字符数据类型。Char[(*n*)]表示长度为 *n* 个字节的固定长度且非 Unicode 的字符数据。*n* 必须是一个介于 1~8 000 之间的数值。存储空间为 *n* 个字节
Varchar	是非固定长度的字符数据类型。Varchar[(*n*)]表示长度为 *n* 个字节的可变长度且非 Unicode 的字符数据。*n* 必须是一个介于 1~8 000 的数值。存储空间为输入数据的字节的实际长度，而不是 *n* 个字节。所输入的数据字符长度可以为零
Datetime	1753 年 1 月 1 日到 9999 年 12 月 31 日的日期和时间数据，精确度为 0.03s（等于 3.33ms 或 0.00333s）
Int	-2^{31}（-2 147 483 648）～$2^{31}-1$（2 147 483 647）的所有整型数据。存储空间为 4B
Smallint	-2^{15}（-32 768）～$2^{15}-1$（32 767）的整型数据。存储空间为 2B
Tinyint	0～255 的整型数据。存储空间为 1B
Decimal 和 Numeric	带定点精度和小数位数的 Numeric 数据类型。Decimal[(*p*[, *s*])] 和 Numeric[(*p*[, *s*])] 表示定点精度和小数位数。使用最大精度时，有效值从 $-10^{38}+1$～$10^{38}-1$。*p* 表示精度，指定小数点左边和右边可以存储的十进制数字的最大个数。精度必须是从 1 到最大精度之间的值。最大精度为 38。*s* 表示小数位数，指定小数点右边可以存储的十进制数字的最大个数。小数位数必须是 0～*p* 之间的值。默认小数位数是 0，因而 $0 \le s \le p$。最大存储空间基于精度而变化
Float [(*n*)]	-1.79E + 308～1.79E + 308 之间的浮点数字数据。*n* 为用于存储科学记数法 Float 数尾数的位数，同时指示其精度和存储空间。*n* 必须为从 1～53 之间的值
Real	-3.40E + 38～3.40E + 38 之间的浮点数字数据。存储空间为 4B
Bit	bit 数据类型只能包括 0 或 1。可以用 bit 数据类型代表 TRUE 或 FALSE、YES 或 NO。例如，询问客户是否为初次访问的问题可存储在 bit 列中，类似于其他数据类型中的布尔型

在定义 char、varchar、decimal、numeric 和 float 数据类型时，需要指定数据的长度。

下面在数据库 UserMan 中创建一个表 Users，用于保存用户信息，结构如表 1.2 所示。

表 1.2　　表 Users 的结构

字段名	数据类型	长度	具体说明
UserId	int		用户编号
UserName	Char	10	用户名
UserPwd	Char	10	密码

在 SQL Server 2000 企业管理器中，展开服务器组，然后展开服务器实例。展开要添加表的数据库，右击“表”目录。选择菜单中“新建表”命令，打开表设计器，如图 1.3 所示。

首先输入列名。列名必须惟一，而且要符合 SQL Server 的标识符规范。

选择一列，在表设计器的上半部分可以编辑列名、数据类型、长度和允许空等属性。如果选中“允许空”复选框，则该列数值可以为空，否则在输入数据时，必须输入该列数据。在表设计器的下半部分，可以编辑列的其他属性，包括描述、默认值、精度、小数位数和标识等。

输入完成后，单击“保存”图标，将弹出“输入表名”对话框，如图 1.4 所示。

输入表名后单击“确定”按钮，可以在企业管理器中看到新建的表。右键单击表，在弹出

菜单中选择“设计表”命令，可以打开表设计器对其进行修改。

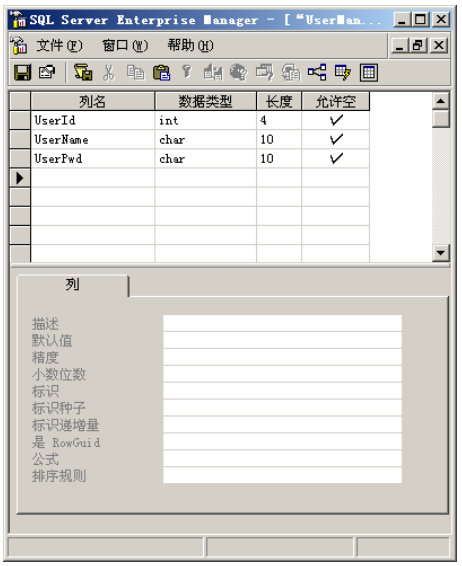


图 1.3　表设计器



图 1.4　输入表名

右键单击表，在弹出菜单中选择“打开表”/“返回所有行”，可以表格的形式查看表中的数据，如图 1.5 所示。

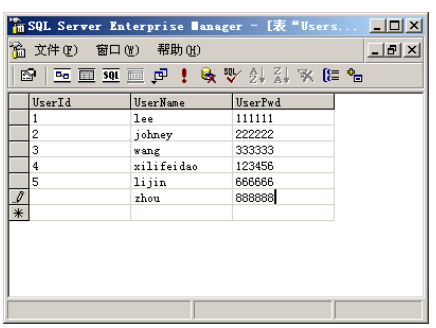


图 1.5　查询表中数据

在表中，有的列可以被设置作为行的惟一标识，也就是说在表的所有行中，此列的数据是惟一的。通常情况下，可以把编号列设置为惟一标识列。为了描述这种惟一标识列，数据库中引入了键（key）的概念。在 SQL Server 中，有两种类型的键。

- 主键（PRIMARY KEY）　表中的一组列，对表中的每一行这些列都有惟一的值。定义主键可以强制在指定列中不允许输入空值，如果要插入行的主键值已经存在，则此行不允许被插入。表只能有一个主键。
- 外键（FOREIGN KEY）　是用于建立和加强两个表数据之间的链接的一列或多列。有时称这两个参与表为外键表和主表。外键可以约束表中的行，因此也称为外键约束。外键确保添加到外键表中的任何行在主表中有相应的行。

在表设计器中单击要定义为主键的数据列，如果要选择多个列，可同时按下 CTRL 键。右击该列，然后选择“设置主键”命令。此列的前面出现了主键符号，同时“允许空”的标记被去掉。

通常使用编号列作为主键。列 UserId 并没有实际意义，只是用来标识表中的每条记录。因此，没有必要由用户设置列 UserId 的值。可以定义该列的值由系统自动分配。在表设计器中选择列 UserId，在下面的属性栏中设置“标识”为“是”、“标识种子”为 1、“标识递增量”为 1，表示该列的值由系统自动生成，每次递增量为 1，如图 1.6 所示。

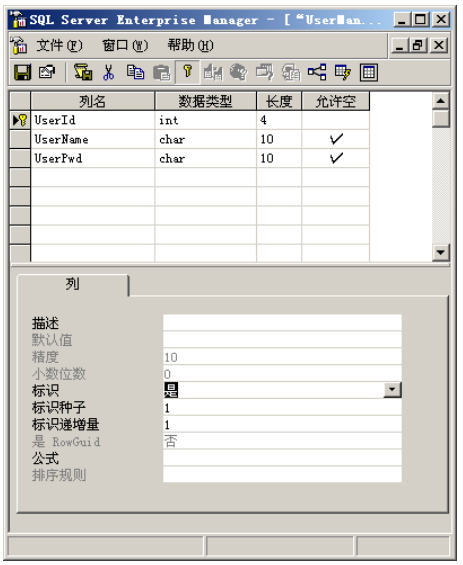


图 1.6　设置标识列

1.1.4　创建视图

视图是虚拟的表，它根据用户指定的条件从表中查询数据，并以表格的形式返回结果。视图中的数据可以来自于一个或多个表。

例如，可以从用户表 Users 中提取 UserName 和 UserPwd 创建一个视图，方法如下。

（1）在企业管理器中，展开指定实例的数据库，右键单击“视图”项，在弹出菜单中选择“新建视图”命令，打开“视图设计器”窗口。此窗口由 4 部分窗格组成，由上到下分别为关系图窗格、网格窗格、SQL 窗格和结果窗格，如图 1.7 所示。

（2）右击关系图网格，选择“添加表”命令，打开“添加表”对话框，如图 1.8 所示。

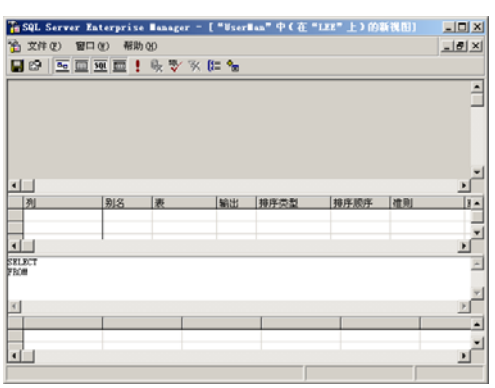


图 1.7　视图设计器

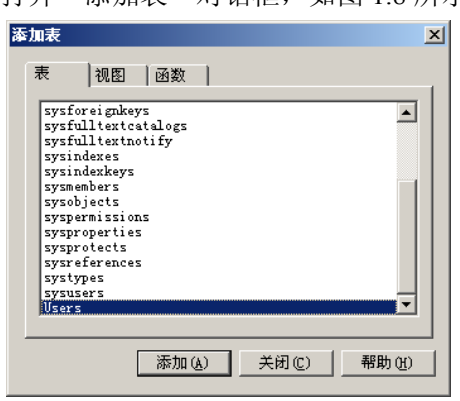


图 1.8　“添加表”对话框

（3）选择表 Users，单击“添加”按钮，将表 Users 添加到关系图网格中。

（4）在关系图网格中，选择表 Users 中的 UserName 和 UserPwd 字段。

（5）设置完成后，单击工具栏中的“运行”按钮，运行视图。结果出现在“结果窗格”中，如图 1.9 所示。

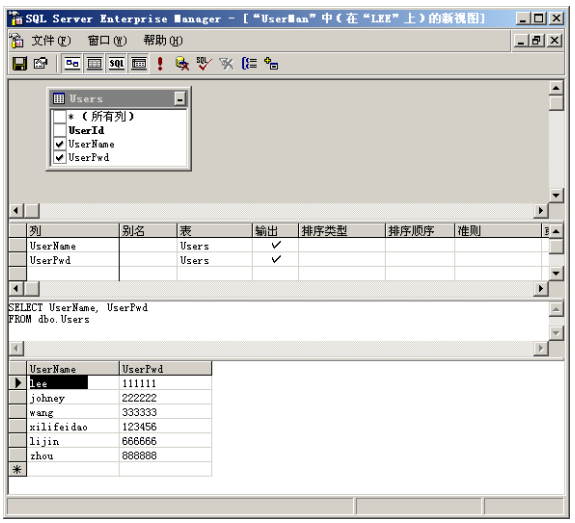


图 1.9　创建新视图

（6）单击“保存”图标，在“另存为”对话框中输入视图名 UserView。

关闭“视图设计器”，视图 EmployeeView1 已经出现在企业管理器中。

1.1.5　备份和还原数据库

为了保证用户数据的安全性，防止数据库中的数据意外丢失，SQL Server 2000 提供了备份和还原数据库组件。定时地备份数据库是保护数据库的最佳选择。本书后面实例的后台数据库都以备份文件的形式保存在附赠光盘的对应目录下，读者在学习具体实例时，请参照本节介绍的方法和相关章节的说明还原数据库。

可以在企业管理器中，按照如下方法对数据库进行备份。

（1）在企业管理器中，展开要备份的数据库所在的服务器组，然后展开服务器。展开“数据库”文件夹，右击要备份的数据库。

（2）在弹出菜单中选择“所有任务”/“备份数据库”，打开“SQL Server 备份”窗口，如图 1.10 所示。

（3）在“名称”框内，输入备份集名称。读者也可以选择在“描述”框中输入对备份集的描述。

（4）在“备份”选项下单击“数据库 / 完全”，此选项支持数据库的完整备份，而其他选项只能备份数据库的一部分。

（5）在“目的”选项下，单击“磁带”或“磁盘”。如果计算机上没有安装磁带设备，则此选项不可用，只能选择“磁盘”。

（6）单击“添加”按钮，打开“选择备份目的”对话框。选择“文件名”选项，默认的文件为 C:\Program Files\Microsoft SQL Server\MSSQL\BACKUP\。注意，这只是一个目录，读者还需要自己输入一个备份文件名。如果要备份到指定的目录，请单击“…”按钮，如图 1.11

所示。

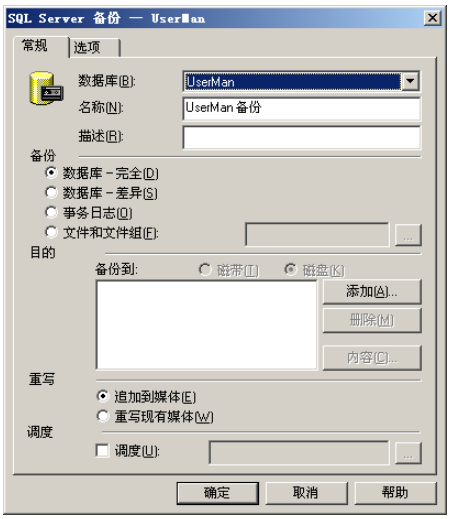


图 1.10　SQL Server 备份对话框

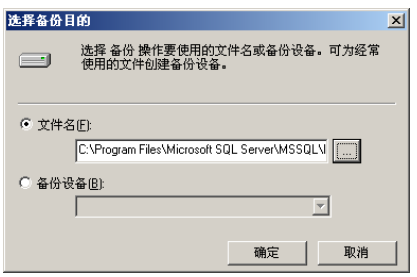


图 1.11　选择备份目的

（7）单击“确定”按钮，返回“SQL Server 备份”对话框。

（8）在“重写”选项下，读者可以选择：

- “追加到媒体”，将备份追加到备份设备上任何现有的备份中；
- “重写现有媒体”，将重写备份设备中任何现有的备份。

（9）如果需要设置定期执行备份操作，请选择“调度”复选框，然后单击后面的“…”按钮，打开“编辑调度”对话框，如图 1.12 所示。

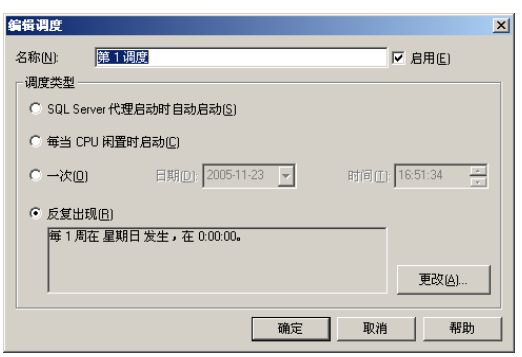


图 1.12　编辑调度

（10）在“编辑调度”对话框中设置完成后，请单击“确定”按钮，返回“SQL Server 备份”对话框。

（11）如果一切都设置完成，则在“SQL Server 备份”对话框中，单击“确定”按钮，开始备份。

如果备份远程计算机上的数据库，则备份文件将在远程计算机上生成。数据库 UserMan 的备份文件保存在附赠光盘的“第 1 章\UserMan”。

还原 SQL Server 数据库的过程如下。

（1）在企业管理器中，展开要还原的数据库所在的服务器组，然后展开服务器。右击“数据库”，在弹出菜单中选择“所有任务”/“还原数据库”，打开“还原数据库”窗口，如图 1.13 所示。在“还原为数据库”文本框中输入要还原的数据库名称。

（2）在“还原”选项中选择“从设备”，如图 1.14 所示。

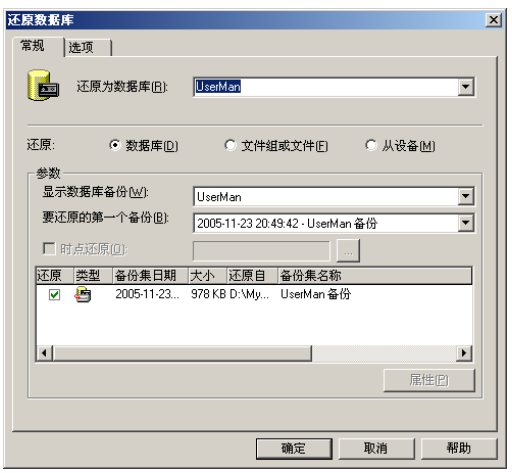


图 1.13　“还原数据库”对话框

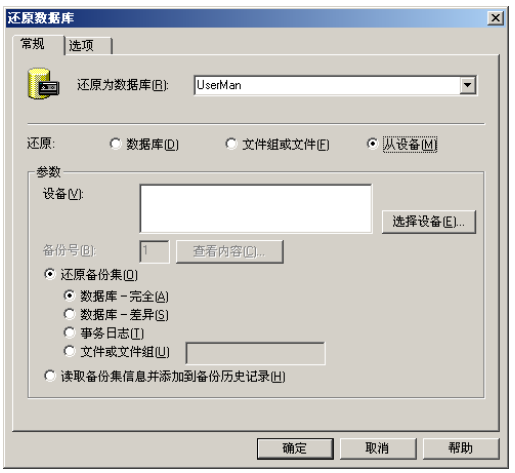


图 1.14　选择“从设备”选项

（3）单击“选择设备”按钮，打开“选择还原设备”对话框，如图 1.15 所示。

（4）在“选择还原设备”对话框中，同样可以看到“磁盘”和“磁带”两种设备。选择“磁盘”，然后单击“添加”按钮，打开“选择还原目的”对话框，如图 1.16 所示。选择“文件名”选项，并输入目的文件。也可以单击“…”按钮，在弹出的窗口中选择目的文件。

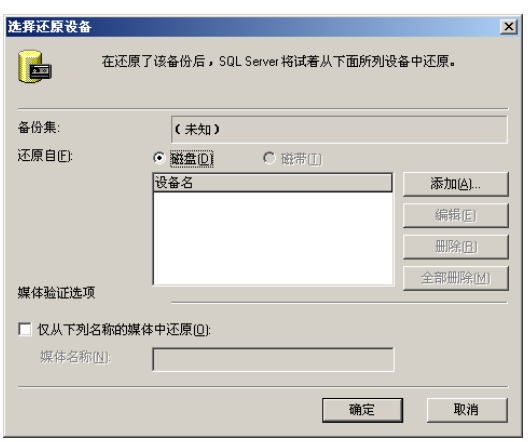


图 1.15　选择还原设备

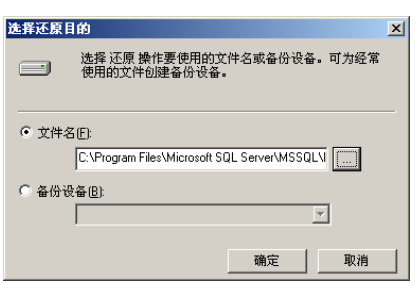


图 1.16　选择还原目的

（5）在“选择还原目的”对话框中单击“确定”按钮，返回“选择还原设置”对话框；

（6）在“选择还原设置”对话框中，可以看到“设备名”列表中出现了选择的设备。单击“确定”按钮，返回“还原数据库”对话框。

（7）在“还原数据库”对话框中，同样可以看到“设备”列表中出现的选择设备。单击“确

定”按钮，开始还原数据库。还原操作完成后，将出现一个对话框，提示还原成功。

读者可以按照上面的步骤将附赠光盘中的“第 1 章\UserMan”还原到数据库中。

1.2　常用 SQL 语句介绍

SQL 是结构化查询语言的缩写，是访问数据库的标准语言。标准化的数据库系统都对 SQL 语言提供支持，这给数据库应用程序开发人员带来了很大的方便，无论后台数据库是 SQL Server、Oracle，还是 IBM DB2，甚至是小型数据库 Access，都可以使用标准的 SQL 语句对它进行操作。

可见，要开发数据库应用程序，必须首先掌握 SQL 语言。本节将介绍几个常用的 SQL 语句，为读者阅读后面的实例奠定基础。

在阅读本节之前请将附赠光盘中的“第 1 章\UserMan”还原到数据库中。

1.2.1　执行 SQL 语句的环境

在 SQL Server 中，可以使用查询分析器执行 SQL 语句。依次点击“开始”\“程序”\“Microsoft SQL Server”\“查询分析器”，可以运行 SQL Server 2000 查询分析器。首先需要输入登录信息，如图 1.17 所示。sa 是 SQL Server 的系统管理员用户，在安装 SQL Server 时可以设置 sa 用户的密码（本书附赠的数据库密码均为 sa）。输入密码后，单击“确定”按钮，进入查询分析器，如图 1.18 所示。

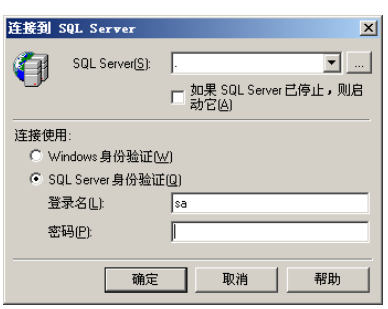


图 1.17　连接到 SQL Server

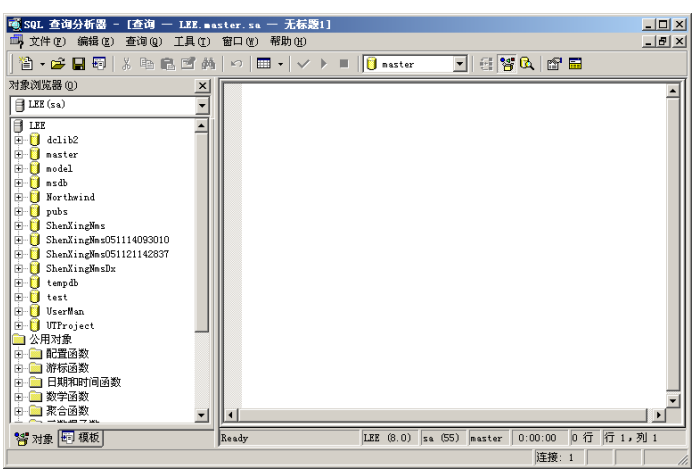


图 1.18　查询分析器

在右侧的窗格中可以输入要执行的 SQL 语句，单击工具栏中的▶按钮执行 SQL 语句。在工具栏中有一个列表框，可以通过它选择 SQL 语句作用于的数据库。

1.2.2　SELECT 语句

SELECT 语句是最常用的 SQL 语句，它的功能是从数据库中查询数据，并以表格的形式返回查询结果。下面是一个最简单的 SELECT 语句：

```
SELECT * FROM Users
```

它的功能是返回表 Users 中的所有记录。在查询分析器中执行此语句，如图 1.19 所示。

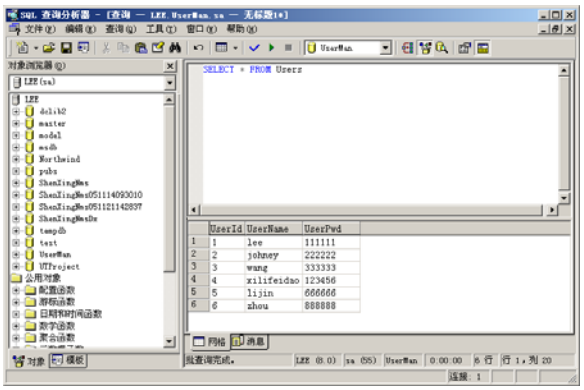


图 1.19　执行简单的 SELECT 语句

在 SELECT 语句中，*表示表中的所有字段，FROM 关键字后面的是要查询的表。下面结合实例介绍 SELECT 语句的各种使用方法。

1．设置列标题

默认情况下，返回结果的标题为表的列名。对于不熟悉数据库结构的用户来说，这些列名很不容易理解。可以使用 AS 子句自定义显示标题。例如：

```
SELECT UserName AS 用户名, UserPwd As 密码 FROM Users
```

运行结果如图 1.20 所示。

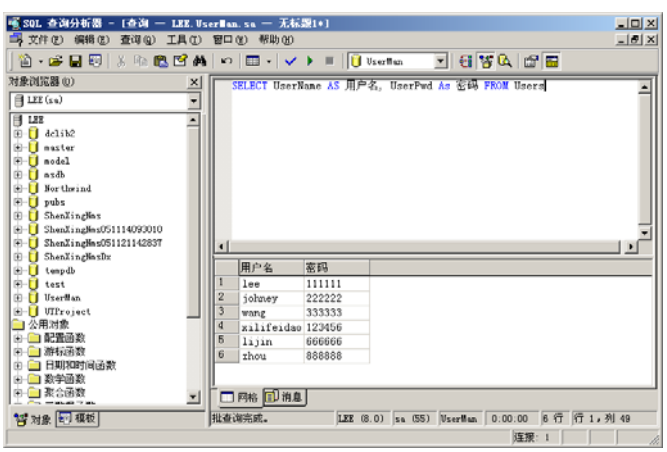


图 1.20　设置列标题

2．设置查询条件

可以使用 WHERE 子句设置查询条件。例如，查询用户名为 lee 的用户信息，可以使用如下代码：

```
SELECT UserName AS 用户名, UserPwd As 密码 FROM Users WHERE UserName = 'lee'
```

在 SQL 语言中，使用单引号定义字符串。运行结果如图 1.21 所示。

可以在 WHERE 子句中使用 LIKE 关键字和通配符，用于模糊查询。例如，要查询所有用户名中包含字母 i 的用户信息，可以使用下面的语句。在 SQL 语言中，使用%表示通配符，代表任意字符串。运行结果如图 1.22 所示。

```
SELECT UserName AS 用户名, UserPwd As 密码 FROM Users　WHERE UserName LIKE '%i%'
```

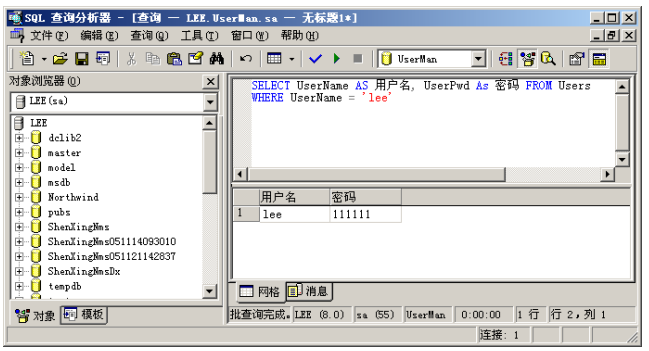


图 1.21　设置查询条件

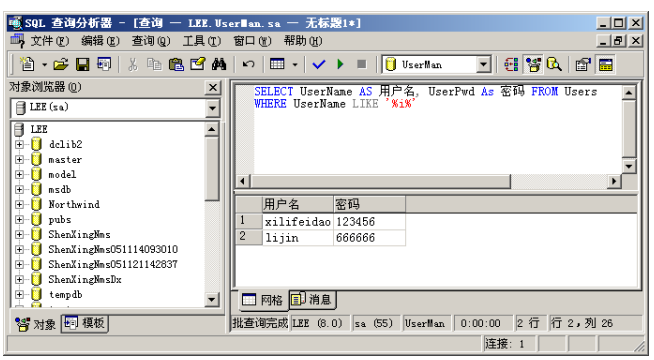


图 1.22　使用通配符进行模糊查询

3．排序查询结果

可以使用 ORDER BY 子句对查询结果进行排序。例如，按用户名的升序排列所有返回结果，可以使用如下代码（运行结果如图 1.23 所示）：

```
SELECT UserName AS 用户名, UserPwd As 密码 FROM Users ORDER BY UserName
```

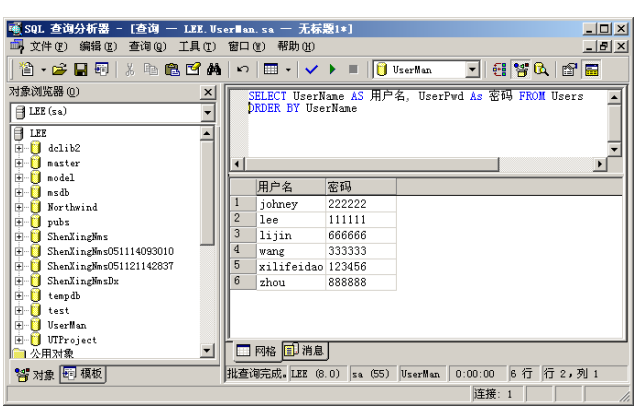


图1.23　排序查询结果