中文在线出品

Web 数据库开发技术

需要全本请访问 www.ertongbook.com

第 1 章　数据库基础

1.1　表和数据库

1.1.1　表、记录和字段

表是按某一公共结构存储的一组相似数据。类似于日常生活中的表格，表按行、列方式将相关信息排列成逻辑组。表中的每一行称为记录，每一列称为字段。例如，表1.1是一个客户表，该表包含了客户的姓名、地址和电子邮件等内容。每项内容是一个字段；包含每个客户的姓名、地址和电子邮件等内容的一组信息是一条记录。建立表时，表中的字段必须指定一种数据类型。字段中存储的数据必须与字段所指定的数据类型一致。

表 1.1　客户表

姓　名	地　　址	城市	省份	电　　话	电子邮件
刘小东	福州大学计算机系	福州	福建	0591-2326666	lxd@263.net
张大洋	北京大学管理学院	北京	北京	010-62716666	zdy@sohu.com
谢小莉	浙江大学数学系	杭州	浙江	0571-7042222	xxl@263.net
王小峰	厦门市湖滨东路2号	厦门	福建	0592-5067777	wxf@sohu.c
连星明	上地产业基地四号	北京	北京	010-62982222	lxm@263.net

1.1.2　数据库

数据库将数据表示为多个表的集合，通过建立表与表之间的关系来定义数据库的结构。例如，下面是一个图书订单管理数据库，共包含 Customers、Orders、OrderDetails 和 Books 4个表。Customers 表存放的是客户信息，包括客户号、客户名、电子邮件地址、密码、电话、地址和邮编等，见表1.2。CustID 字段是 Customers 表的主关键字，即每个客户都有惟一的客户号。主关键字是指用来惟一标识表中记录的一个或一组字段。主关键字不能有重复值，如不能使用 CustName 字段作为主关键字，因为很容易出现两个客户同名的情况。

表 1.2　Customers 表

字段名	字段类型	字段长度
CustID	字符型	6
CustName	字符型	16
Password	字符型	20
Email	字符型	20
City	字符型	20
Province	字符型	20
Phone	字符型	20
Address	字符型	50
Zip	字符型	6

Orders 表存放的是订单信息，包括订单号和订购日期等，见表1.3。OrderID字段是Orders表的主关键字，而CustID字段是外部关键字。外部关键字是指用来连接另一个表并在另一个表中为主关键字的字段，显然CustID是这样的字段。

表 1.3 Orders 表

字段名	字段类型	字段长度
OrderID	字符型	6
CustID	字符型	6
OrderDate	日期时间型	
ShippedDate	日期时间型	

OrderDetails表存放的是订单中包含的每本书的订购信息，包括书号、订购数量和折扣等，见表1.4。OrdersDetails和Orders表之间通过OrderID字段建立关联。

表 1.4 OrderDetails 表

字段名	字段类型	字段长度
OrderID	字符型	6
BookID	字符型	6
Quantity	整型	
Discount	实型	

Books表存放的是每本书的详细信息，包括书号、书名、作者、出版社、出版日期、内容简介、分类、子分类、库存量和定价，见表1.5。Books和OrderDetails表之间通过BookID字段建立关联。

表 1.5 Books 表

字段名	字段类型	字段长度
BookID	字符型	6
BookName	字符型	50
Writer	字符型	16
Press	字符型	30
PublishDate	日期时间型	
Statements	备注型	
Category	字符型	16
Subcategory	字符型	16
Quantity	整型	
Price	货币型	

1.1.3 索引

在关系数据库中，通常使用索引来提高数据的检索速度。表中的数据往往是动态增减的，记录在表中按输入的物理顺序存放。当为主关键字或其他字段建立索引时，数据库管理系统将索引字段的内容以特定的顺序记录在一个索引文件上。检索数据时，数据库管理系统先从索引文件上找到信息的位置，再从表中读取数据。这种方法如同图书馆的索引卡片，可以大

大提高检索速度。对小的表来讲，索引也许并不十分必要，但对于大的表来讲，数据如果不以易于访问的逻辑顺序来组织，则很难加以管理。

每个索引都有一个索引表达式来确定索引的顺序，索引表达式既可是一个字段，也可是多个字段的组合。可以为一个表生成多个索引，每个索引均代表一种处理数据的顺序。

1.1.4 关系

关系数据库的最大好处就是能够避免数据的不必要重复，即可以将包含重复数据的表拆分成若干个没有重复数据的简单表，并通过建立表与表之间的关系来检索相关表中的记录。在表与表之间的关系中，习惯上称主表为父表，而通过关系连接的其他表为子表。根据数据复杂程度的不同，表与表之间可能会有4种关系：①一对一关系。指父表中的记录最多只与子表中的一条记录相匹配，反之亦然。②一对多关系。指父表中的记录与子表中的多条记录有关。例如，对于客户表和订单表来讲，每个订单只与一个客户有关，而每个客户可以有多个订单，因此客户表和订单表是一对多的关系。③多对一关系。与一对多是互补的，即父表中的多条记录与子表中的一条记录相关。④多对多关系。指父表中的多条记录与子表中的多条记录有关。

1.1.5 创建表和数据库

由于本书以后各章中的许多范例均要用到图书订单管理数据库，因此本节谈谈如何使用Microsoft SQL Server来创建该数据库。Microsoft SQL Server是美国微软公司推出的、使用相当广泛的数据库管理系统，它的当前版本是Microsoft SQL Server 7.0，其中包括Microsoft SQL Server企业版和Microsoft SQL Server桌面系统。Microsoft SQL Server企业版运行于Windows NT Server 4.0和Service Pack 4 及以上版本，而Microsoft SQL Server桌面系统运行于Windows 95/98和Windows NT Workstation等。Microsoft SQL Server包含一套图形化工具，如Service Manager（用于启动和关闭数据库服务）、Enterprise Manager（用于创建和修改数据库及备份数据库等）和Performance Monitor（用于分析执行效率，排除效率瓶颈）等。

1．创建数据库

创建数据库的方法如下：

（1）在Windows 2000中安装Microsoft SQL Server 7.0企业版。

（2）选择“开始/程序/Microsoft SQL Server 7.0/Enterprise Manager”命令，启动Enterprise Manager，见图1.1。

（3）在控制台树中右击Databases节点，然后选择快捷菜单的“New Database”命令，打开“Database Properties”对话框。

（4）输入数据库的名称，如“ebooks”，见图1.2。

（5）单击“确定”按钮。

注意：在Microsoft SQL Server中，每个数据库都有两个相关的文件：一个是扩展名为.mdf的数据库文件，另一个是扩展名为.ldf的事务日志文件．可以分别在“General”和“Transaction Log”选项卡中设置这两个文件的存放位置和占用的空间．

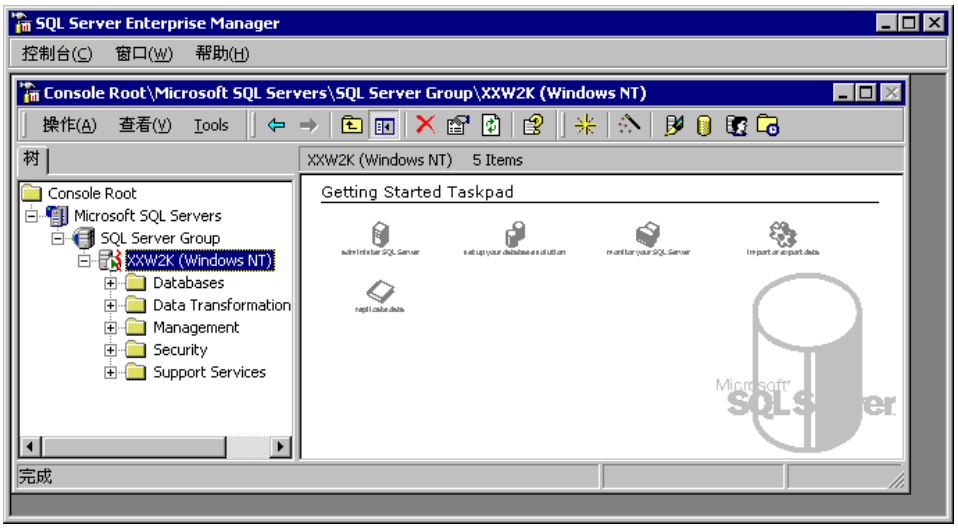


图 1.1 Enterprise Manager

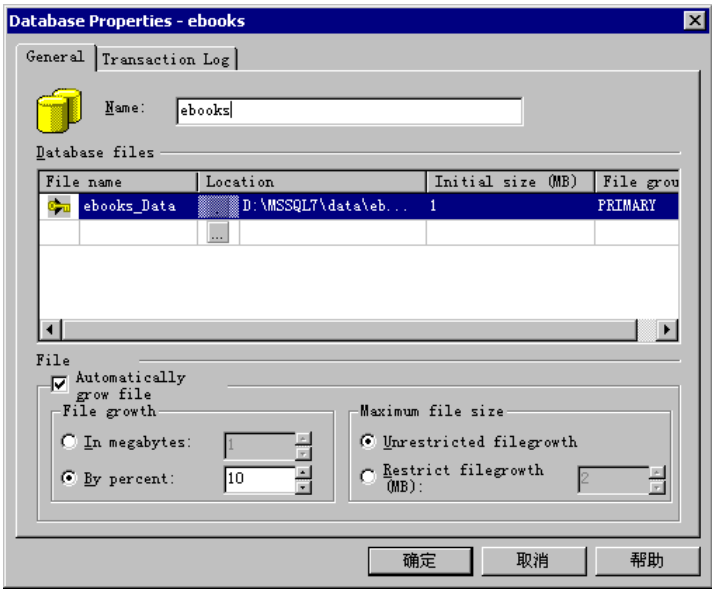


图 1.2 输入数据库的名称

2．创建新表

创建新的数据库后，就可以往数据库中添加新表，方法如下：

（1）**在**Enterprise Manager**中展开**Databases**节点，再展开**ebooks**节点。**

（2）**右击**Tables**节点，然后选择快捷菜单的“**New Table**”命令。**

（3）**在“选择名称”对话框中输入表名，如“**Books**”，见图**1.3。

需要全本请访问 www.ertongbook.com

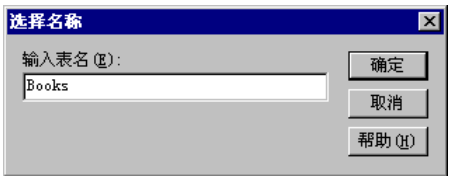


图 1.3 输入表名

（4）单击“确定”按钮，然后在表设计器窗口中设计表的结构，包括设计表中每个字段的名称、类型和长度等，见图1.4。若要设置主关键字，请选中相应的字段，如BookID字段，然后单击工具栏的“Set primary key”按钮。当字段是主关键字时，该字段不允许出现空值。

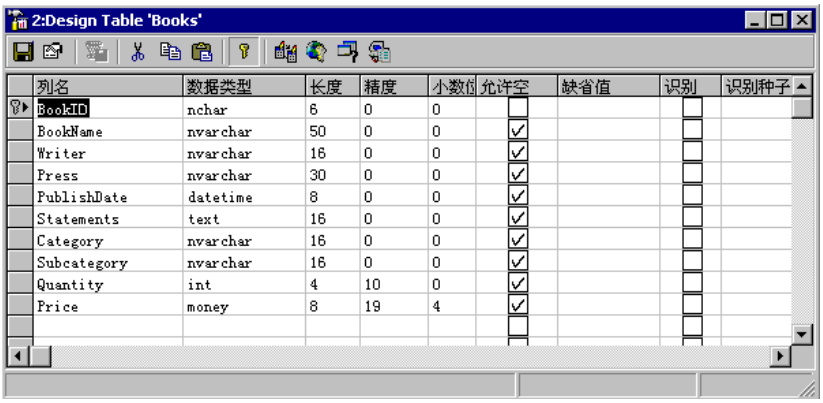


图 1.4 设计表的结构

（5）单击标题栏的“关闭”按钮。

（6）按同样的方法创建Customers、Orders和OrderDetails表。

创建表后，如果要往表中添加记录，请单击Tables节点，在右边窗格中右击相应的表（如Books表），然后从选择快捷菜单指向“Open Table”子菜单，再选择“Return all rows”命令，接着输入每条记录的各个字段的值，见图1.5。

3．为表创建索引

为表创建索引的方法如下：

（1）在Enterprise Manager中右击要创建索引的表（如Customers表），然后选择快捷菜单的“Design Table”命令，打开表设计器。

（2）单击工具栏的“Table and Index Properties”按钮，打开“属性”对话框，然后切换到“索引/键”选项卡。

（3）单击“新键”按钮，然后输入索引名并确定用于索引的字段，见图1.6。

（4）单击“关闭”按钮。

注意：在图1.6中，若选中“创建惟一索引”复选框，则表示该索引不允许重复值.

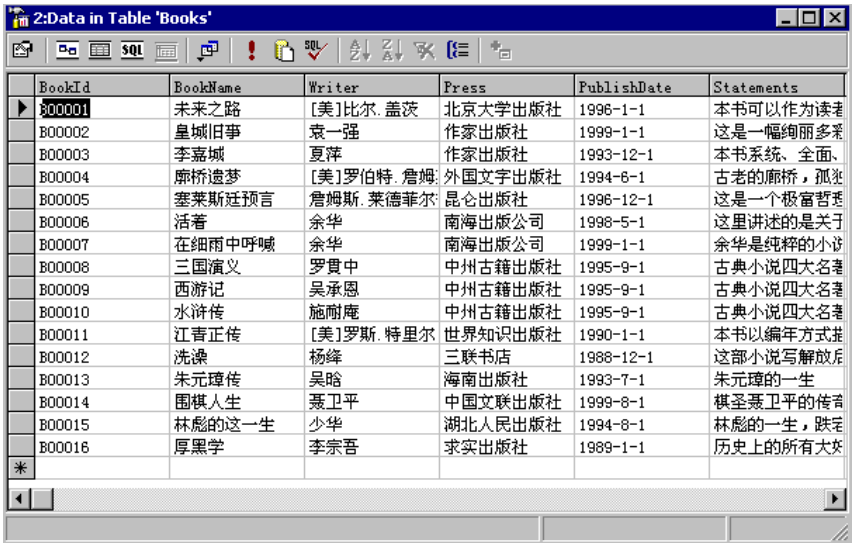


图 1.5　添加记录

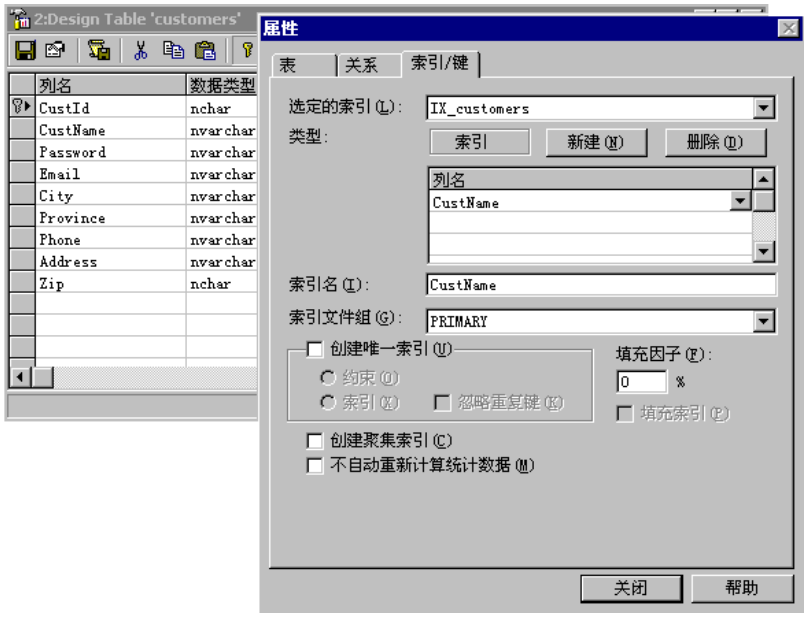


图 1.6　“索引/键”选项卡

1.2　SQL 语言

SQL（Structured Query Language）**语言是由**Ray Boyce**和**Don Chamberlin**于**1974**年提出的。**IBM**公司**San Jose Research Laboratory**在所研制的**System R**系统上实现了**SQL**语言。**SQL**语言的功能包括查询、操纵、定义和控制，是一个综合的、通用的关系数据库语言。自产生之日起，**SQL**语言便成了检验关系数据库的试金石，而**SQL**标准的每一次变更都指导着关系数据库产品的发展方向。

SQL语言本身并不是一个很完整的编程语言，如它不支持流控制等。一般都是将SQL语言与其他编程语言（如Visual Basic，Visual C++，Delphi和PowerBuilder等）及脚本语言（如VBScript和JavaScript等）结合起来使用。

从1982年开始，美国国家标准局（ANSI）即着手SQL的标准化工作。1986年10月，ANSI的数据库委员会X3H2批准SQL作为关系数据库的美国标准，同年公布了标准SQL文本。此后不久，国际标准化组织（ISO）也作了同样的决定。目前，SQL标准有3个版本。基本SQL定义是ANSI X3 135-89，一般叫做SQL-89。SQL-89定义了模式定义、数据操作和事务处理。SQL-89和随后的ANSI X3 168-1989构成了第一代SQL标准。ANSI X3 135-1992描述了一种增强功能的SQL，现在叫SQL-92标准。SQL-92包括模式操作、动态创建、SQL语句动态执行和网络环境支持等增强特性。在完成SQL-92标准后，ANSI和ISO即开始合作开发SQL3标准。SQL3的主要特点在于抽象数据类型的支持，为新一代对象关系数据库提供了标准。

目前，各数据库厂商都纷纷推出各自的支持SQL语言的数据库系统，但完全按SQL标准实现的并不多。由于不同的数据库系统在支持SQL标准的同时均对SQL标准做了许许多多的扩充，因而形成了各种不同的“方言”。例如，Microsoft SQL Server使用的Transact-SQL就是在SQL-92标准基础上扩充的“方言”。在Microsoft SQL Server 7.0中，可以用交互式的图形化工具Query Analyzer来编写并执行SQL语句，见图1.7。

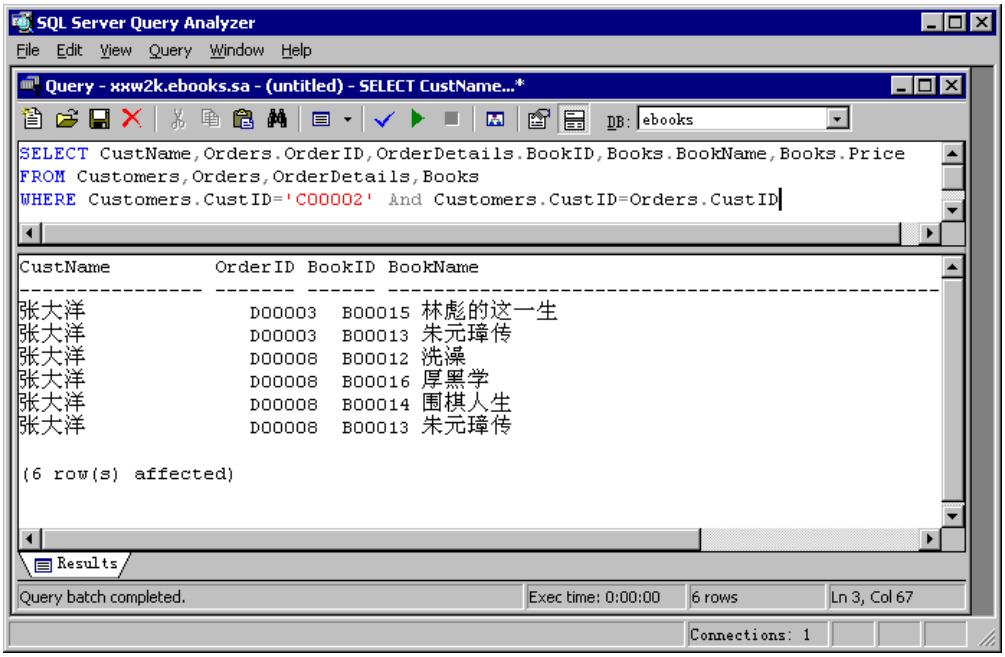


图 1.7 交互式的图形化工具 Query Analyzer

注意：要打开Query Analyzer，请选择“开始/程序/Microsoft SQL Server 7.0/Query Analyzer”命令。

1.2.1 SQL 的构成

SQL由命令、子句和运算符等元素构成，这些元素结合起来组成用于创建、更新和操作

数据库的语句。SQL命令分两大类：DDL（数据定义语言）命令和DML（数据操纵语言）命令。

DDL 命令用于创建和定义新的数据库、字段和索引：

- CREATE 创建新的表、字段和索引。
- DROP 删除数据库中的表和索引。
- ALTER 通过添加字段或改变字段定义来修改表。

DML 命令用于创建查询，以便从数据库中排序、筛选和抽取数据：

- SELECT 在数据库中查找满足特定条件的记录。
- INSERT 在数据库中插入新的记录。
- UPDATE 更新特定的记录和字段。
- DELETE 从数据库中删除记录。

SQL 子句用于定义要选择或操作的数据：

- FROM 指定要操作的表。
- WHERE 指定选择记录时要满足的条件。
- GROUP BY 将选择的记录分组。
- HAVING 指定分组的条件。
- ORDER BY 按特定的顺序排序记录。

SQL运算符分两类：逻辑运算符和比较运算符。逻辑运算符（AND、OR和NOT）用于连接两个表达式，通常在WHERE子句中使用；比较运算符（<、<=、>、>=、=、<>、BETWEEN、LIKE和IN）用于比较两个表达式的值。

SQL还有一些用来计算的函数，即合计函数。例如，用AVG函数计算平均值，用COUNT函数返回记录数，用SUM函数计算总和，用MAX函数计算最大值等。

1.2.2 SELECT 语句

SELECT 语句的功能是从现有的数据库中检索数据，即将满足一定约束条件的一个或多个表中的字段从数据库中挑选出来，并按一定的分组和排序方法显示出来。简单的选择查询只有FROM子句，FROM子句用来指定数据的来源，即指出记录来自哪些表。如果某一字段出现在多个表中，则要用.（小圆点）来指定字段所属的表。例如，以下语句检索Books表的所有记录和字段：

```
SELECT * FROM Books M
```

又如，以下语句检索Books表的所有记录，但结果只含书名、作者和出版社：

```
SELECT BookName , Writer , Press FROM Books
```

SELECT语句的WHERE子句用来指定选择记录时要满足的条件。如果没指定WHERE子句，则查询将返回表中的所有记录。WHERE子句必须在FROM子句的后面。例如，以下语句用于查询客户号等于C00003的客户的订单情况：

```
SELECT CustName , Orders.OrderID , OrderDetails.BookID , Books.BookName , Books.Price
FROM Customers , Orders , OrderDetails , Books
WHERE Customers.CustID='C00003' And Customers.CustID=Orders.CustID
And Orders.OrderID=OrderDetails.OrderID And OrderDetails.BookID=Books.BookID
```

又如，以下语句用于查询1999年上半年的订货情况：

```
SELECT * FROM Orders WHERE OrderDate Between '1999-1-1' And '1999-6-30'
```

又如，以下语句用于查询所有姓刘的客户：

```
SELECT * FROM Customers WHERE CustName Like '刘%'
```

WHERE子句可以包含多个表达式，表达式之间用AND和OR等运算符连接。

SELECT语句的GROUP BY子句用于对记录分组，即将指定字段列表中具有相同值的记录合并成一组。如果含有合计函数，则每组中含有摘要，否则省略摘要。例如，以下语句按订单号分组统计每张订单的总额：

```
SELECT OrderDetails.OrderID,
SUM(OrderDetails.Quantity*OrderDetails.Discount*Books.Price)
FROM OrderDetails, Books
WHERE OrderDetails.BookID=Books.BookID
GROUP BY OrderDetails.OrderID
```

SELECT语句的HAVING子句用于确定带GROUP BY子句的查询中具体显示哪些记录，即用GROUP BY子句完成记录分组后，可以用HAVING子句来显示满足指定条件的分组。例如：

```
SELECT OrderDetails.OrderID,
SUM(OrderDetails.Quantity*OrderDetails.Discount*Books.Price)
FROM OrderDetails, Books
WHERE OrderDetails.BookID=Books.BookID
GROUP BY OrderDetails.OrderID
HAVING Sum(OrderDetails.Quantity*OrderDetails.Discount*Books.Price)>=100
```

SELECT语句的ORDER BY子句用于对记录排序。例如：

```
SELECT * FROM Books ORDER BY BookName
```

默认是升序。若想降序，则要在用作排序的字段后加DESC关键字。例如：

```
SELECT * FROM Books ORDER BY BookName DESC
```

ORDER BY子句中还可包含多个字段，这样记录先按第一个字段排序，然后对值相等的记录再按第二个字段排序，依此类推。例如：

```
SELECT * FROM Customers ORDER BY Province, City
```

1.2.3 SELECT…INTO 语句

SELECT…INTO语句用来从查询结果中建立新表。例如：

```
SELECT CustName, City, Province INTO #Customers2
FROM Customers WHERE Province IN ('北京', '天津')
```

1.2.4 DELETE 语句

DELETE语句的功能是删除FROM子句列出的、满足WHERE子句条件的一个或多个表中的记录。例如，可以用下面的语句从Customers表中删除姓“刘”的记录：

```
DELETE FROM Customers
```

```
WHERE CustName Like '刘%'
```

1.2.5 INSERT INTO 语句

INSERT INTO语句用于添加一个或多个记录到表中。例如，以下语句用于往Books表中添加一条新的记录：

```
INSERT INTO Books
(BookID, BookName, Writer, Press, PublishDate, Statements, Category,
SubCategory, Quantity, Price)
VALUES ('B00017', '永乐大帝', '何宝善', '北京燕山出版社', '1997-12-1',
'准确概括了明成祖超人的帝王之才和复杂的思想性格', '文艺', '传记', 166, 24)
```

1.2.6 UPDATE 语句

UPDATE语句按某个条件来更新特定表中的字段值。例如，以下语句用于将OrderDetails表中所有折扣改为0.8：

```
UPDATE OrderDetails SET Discount = 0.8
```

1.3 客户/服务器结构的数据库系统

计算机网络技术的发展，使计算机资源的共享成为可能。在一个企业中，使用网络技术，采用客户/服务器（Client/Server）结构的数据库系统，不仅可以实现对数据库资源的共享，而且可以提高数据库的安全。

传统的客户/服务器结构的数据库系统是两层的，基本思想是：服务器提供数据的存储和管理等，客户端运行相应的应用，通过网络获得服务器的服务，使用服务器上的数据库资源。客户端和服务器通过网络连接成一个互相协作的系统。客户/服务器结构把硬件和软件合理地配置和设计，极大地推动了联机企业信息系统的实现。与主机/终端模式相比，客户/服务器结构可以更好地实现数据服务和应用程序的共享，并且系统容易扩充，更加灵活，简化了企业信息系统的开发。当信息系统的规模扩大或需求改变时，不必重新设计便可在原有基础上进行扩充和调整，从而保护了企业在硬件和软件上的已有投资。客户/服务器结构很快成为企业信息建设的主要模式，对数据库乃至 IT 业的发展产生了深远的影响。

随着Internet/Intranet技术的普及和电子商务技术的发展，客户/服务器结构的数据库系统正由2层向3层或多层发展。在图1.8所示的层结构中，应用程序的任务在逻辑上被划分为3种服务：用户服务、业务服务（中间层服务）和数据服务。用户服务在客户机上，提供了可视化的用户界面来表示信息和收集数据；业务服务在一个或多个业务服务器上，是联系用户服务和数据服务的“桥梁”，它们响应用户发来的请求，实施业务规则，处理和分析业务数据，获得业务结果；数据服务在数据库服务器上，负责数据的组织管理，包括数据的定义、维护、访问和更新以及管理并响应业务服务的数据请求。

需要全本请访问 www.ertongbook.com

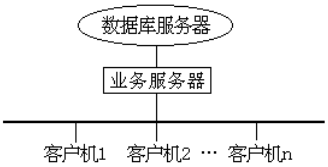


图 1.8 3 层结构

1.4 ODBC 简介

ODBC（Open Database Connectivity）是Windows开放服务体系结构的一个组件，是使用十分广泛的数据库接口。ODBC是以开发者工具包形式发行的一种面向SQL的API（应用程序编程接口），适用于多种DBMS（数据库管理系统）。使用ODBC，可以实现用相同的代码来访问多种不同格式的数据库，简化了数据库的访问，也为程序的跨平台开发和移植提供了极大的方便。

ODBC由应用程序、驱动程序管理器、驱动程序和数据源等部分组成，见图1.9。应用程序执行并调用ODBC接口来提交SQL语句并接收SQL的执行结果；驱动程序管理器用来管理ODBC驱动程序，并将数据源映射成特定的驱动程序动态链接库；驱动程序针对某种特定的DBMS，每个驱动程序是一个动态链接库，实现ODBC函数调用并与数据源交互。

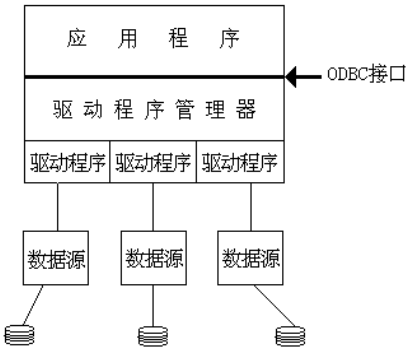


图 1.9 ODBC 的结构

数据源（DSN：Data Source Name）由待访问的数据库及与之相关的操作系统、DBMS和网络平台组成。一个DSN就是对数据库的一个命名连接。DSN分用户、系统和文件3种。用户DSN和系统DSN将信息存储在Windows注册表中。用户DSN只有用户可以看见，且只能用于当前机器中；系统DSN允许所有用户登录特定服务器去访问数据库，任何具有权限的用户均可访问系统DSN。文件DSN将信息存储在后缀为.dsn的文本文件中，如果将此文件放在网络的共享目录中，则可被网络中的任何一台工作站访问到。在Web应用程序中访问数据库时，通常都是建立系统DSN。下面是在Windows 2000中建立一个系统DSN的步骤：

（1）选择“开始/程序/管理工具/数据源（ODBC）”命令，打开ODBC数据源管理器，然后切换到“系统DSN”选项卡，见图1.10。

（2）单击“添加”按钮，弹出“创建数据源”对话框，见图1.11。



图 1.10 “系统 DSN”选项卡

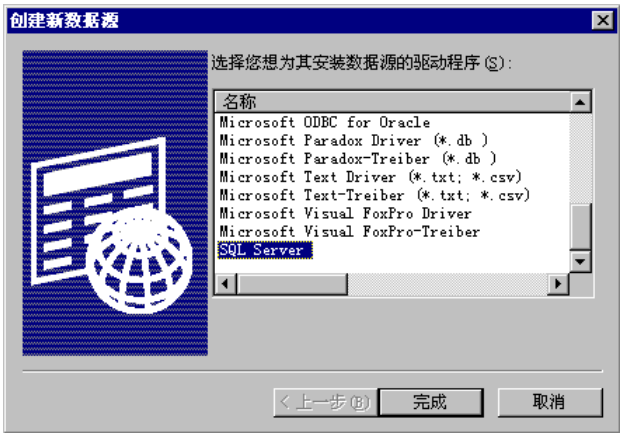


图 1.11 “创建数据源”对话框

（3）选择SQL Server，然后单击“完成”按钮。

（4）输入系统DSN的名称并指定SQL Server所在的服务器，见图1.12。



图 1.12 输入 DSN 的名称并指定 SQL Server 所在的服务器

（5）单击“下一步”按钮，选择SQL Server验证登录ID的方式，见图1.13。这里选中“使用用户输入登录ID和密码的SQL Server验证”单选钮，然后在“登录ID”和“密码”文本框中输入对ebooks数据库有存取权限的SQL Server账号和密码。

（6）单击“下一步”按钮，选中“改变默认的数据库为”复选框，然后选中ebooks数据库，见图1.14。

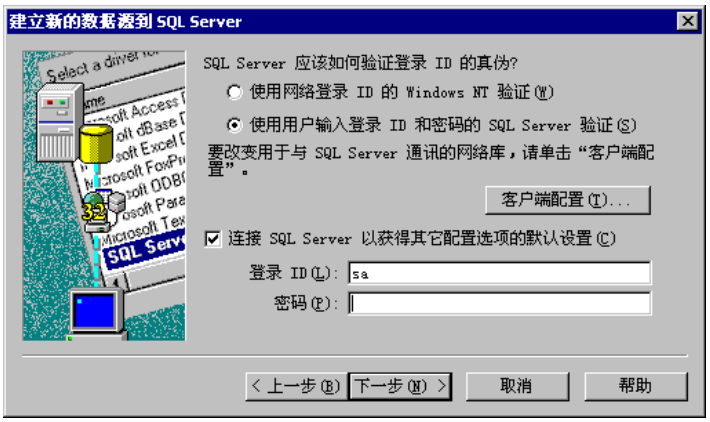


图 1.13 选择 SQL Server 验证登录 ID 的方式



图 1.14 将数据源对应的默认数据库改为 ebooks

（7）单击“下一步”按钮，到下一画面时单击“完成”按钮。

（8）在图1.15中，单击“测试数据源”按钮。当弹出对话框提示测试成功时，表示DSN设置正确。

（9）单击“确定”按钮，完成系统DSN的建立。



图 1.15 “ODBC Microsoft SQL Server 安装”对话框

1.5 OLE DB 简介

OLE DB是一种数据访问的技术标准，目的是提供统一的数据访问接口。这里的数据既可是DBMS数据源，也可是非DBMS数据源。DBMS数据源包括主机数据库（如IBM DB2）、服务器数据库（如Oracle和Microsft SQL Server）及桌面数据库（如Microsoft Access）；非DBMS数据源包括存放在Windows和Unix文件系统中的信息、电子邮件、电子表格、Web上的文本或图形及目录服务等。OLE DB使得数据的消费者（应用程序）可以用相同的方法来访问各种数据，而不用考虑数据的具体存储地点、格式和类型。OLE DB将传统的数据库系统划分为多个逻辑组件，见表1.6，组件之间相对独立又相互通信。

表 1.6 OLE DB 逻辑组件

逻辑组件	说 明
消费者（Consumers）	消费者是使用OLE DB对存储在数据提供者中的数据进行控制的应用程序。除了典型的数据库应用程序外，还包括需要访问各种数据源的开发工具或语言等
提供者（Providers）	提供者是暴露OLE DB的软组件。提供者大致分两类，即数据提供者（Data Providers）和服务提供者（Service Providers）。数据提供者是提供数据存储的软组件，小到普通的文本文件，大到主机上的复杂数据库，或者电子邮件存储，都是数据提供者的例子；服务提供者位于数据提供者之上，它是从过去的DBMS中分离出来且能独立运行的功能组件，如查询处理器和游标引擎等，这些组件使得数据提供者提供的数据能以表格状的形式向外表示（不管真实的物理数据是如何组织和存储的），并实现数据的查询和修改功能。服务提供者不拥有数据，但通过OLE DB来封装某些服务
业务组件（Business Component）	业务组件是利用数据服务提供者专门完成某种特定业务信息处理的，可重用的功能组件。

OLE DB标准的具体实现是一组API函数，这些API函数符合COM（组件对象模型）。使用OLE DB API，可以编写能访问符合OLE DB标准的任何数据源的应用程序，也可以编写针对某些特定数据存储的查询处理器和游标引擎。

第 2 章　Web 简介

2.1　概述

Web（World Wide Web或WWW）的中文名称是万维网，它不是传统意义上的物理网络，而是在超文本基础上形成的庞大信息网。Web的最大特点是拥有友好的图形界面、简单的操作方法及图文并茂的显示方式。

Web是由许许多多Web站点构成的。每个Web站点是一组资源的集合，其上的资源一般存放在Internet/Intranet的某台服务器上，但也可能分布在多台服务器上。Web站点提供信息的基础是HTML（Hypertext Markup Language）页面（中文称网页）。每个Web站点是一组精心设计的网页，这组网页并不是杂乱无章的，而是围绕着某些主题，有机地连在一起，形成一个整体。组成Web站点的一组网页中一定有一个网页是其他网页的根，这就是主页。用户在客户机上通过浏览器访问Web站点时，一般都是先看到站点的主页。主页是Web站点的门脸，是向访问者介绍该Web站点并帮助他们浏览站点的网页。如果将Web看成一个巨大的图书馆，那么Web站点就像一本本书，主页就像每本书的封面和目录，网页就好比书中特定的页。

Web基于客户/服务器结构，客户是浏览器，而服务器就是Web服务器。浏览器和Web服务器通过HTTP（Hypertext Transfer Protocol）交换信息。Web上的大多数交互均可看成请求和响应，即浏览器向Web服务器发出请求（一般是要求显示一个用户想看的网页），而Web服务器则向浏览器返回响应（通常是一个HTML页面或图像等）。

2.2　HTML 文档

Web站点提供信息的基础是网页，而网页是由HTML文档生成的。每个HTML文档由许多元素构成，这些元素告诉浏览器如何显示HTML文档。每个元素都用HTML代码或标记定义，每个标记总是用尖括号“<”和“>”括起来。大多数HTML标记都双边标记，前边的称头标，后边的称尾标，尾标比头标多一个反斜杆（/），头标与尾标之间的内容就是该标记的作用范围。有些HTML标记是单边标记，即只有头标，而没有尾标。例如，下面是一个简单的HTML文档：

```
<HTML>
<HEAD><TITLE>电脑培训</TITLE></HEAD>
<BODY>
<IMG SRC="e:\myinfo\电脑.jpg">
<H1>欢迎访问东方电脑培训中心</H1>
<HR>
<A HREF="e:\myinfo\introduce.htm">中心简介</A>
<A HREF="e:\myinfo\contents.htm">培训内容</A>
```

```
<A HREF="e:\myinfo\inform.htm">报名方法</A>
<HR>
</BODY></HTML>
```

各种标记的具体含义如下：

- <HTML>…</HTML> **表示当前文档是 HTML 文档。**
- <HEAD>…</HEAD> **表示网页的页头。**
- <TITLE>…</TITLE> **表示网页的标题。**
- <BODY>…</BODY> **表示网页的实际内容。**
- <H1>…</H1> **表示一级标题。**
- <A>…</A> **表示超级链接（也称热点或链接）。超级链接可以是文字或图形，通常带下划线或以变色等方式显示。在网页中单击超级链接，就可以非常方便地从一个地方跳转到另一个地方。**
- <HR> **表示一条水平线。**
- <IMG> **表示一幅图。**

HTML文档是纯文本文档，可以用任何文本编辑器来编写，如可用Windows“记事本”程序来编写。编写后，选择“文件/另存为”命令即可保存。保存时文件名必须以.htm或.html为扩展名。保存后，就可以用浏览器来打开HTML文档。打开时，浏览器并不是显示其中的HTML代码，而是显示经解释后生成的网页。例如，图2.1就是以上代码经浏览器解释后生成的网页。



图 2.1 在浏览器中显示 HTML 文档

注意：虽然通过编写HTML文档来生成简单的网页并不太费事，但当HTML页面较复杂时，编写工作量就会大大增加．因此，出现了类似Macromedia Dreamweaver和Microsoft FrontPage这样的网页设计器．有了网页设计器，编写HTML页面就像编制Microsoft Word文档那样简单而又容易．

需要全本请访问 www.ertongbook.com