第 3 章

动态网站与数据库技术

数据库是动态网站的基础，动态网站是数据库技术与网页技术结合的产物。动态网页上需要显示的内容一般来自于后台数据库，当用户浏览到动态网页时服务器会实时地从数据库中把用户需要的信息读出来放在网页中呈现给用户，如果数据库中的数据发生了变化，如被添加、修改或删除之后，用户浏览到的网页内容也会随之改变。

那么一个关键的问题就是：网页程序是怎么和数据库打交道的？答案是使用 SQL 语言。SQL 是专门用于数据库访问的语言，把 SQL 语句嵌入到网页程序就可以实现动态网站。不仅如此，对数据库进行运行、管理和维护工作也是通过 SQL 语句来完成的。所以，熟练掌握 SQL 语言是学习数据库的重点，是开发和维护网站工作必备的技能。

本章主要介绍在网站开发中如何使用数据库，主要包括网站程序与数据库的连接方式、基本 SQL 语句、存储过程和事务在开发中的应用。

3.1 数据库操作语言——SQL

SQL(Structured Query Language)是一种功能强大的结构化查询语言，是各种关系数据库操作的标准语言。尽管市场上的数据库产品很多，但只要访问的是关系数据库，那么其操作语言就是标准的、统一的。

利用 SQL 语言提供的命令可以实现对数据库中所有数据的存取和访问操作。SQL 命令中命令动词和保留字字母不区分大小写，一条 SQL 语句可以以一行或多行的形式输入，最终以分号结束。

1. 功能分类

SQL 作为关系数据库的操作语言，功能非常强大，主要包括以下三大类，如表 3-1 所示。

表 3-1 SQL 语言分类

SQL 分类	英 文 全 称	功 能	包括主要命令动词
数据定义 DDL	Data Definition Language	定义数据库中各种对象	CREATE、DROP、ALTER
数据操纵 DML	Data Manipulation Language	实现数据的查询、插入、删除以及修改等操作	SELECT、INSERT、UPDATE、DELETE
数据控制 DCL	Data Control Language	用于定义数据库用户的权限	GRANT、REVOKE

2. 特点

SQL 主要有以下特点。

(1) SQL 是非过程化语言。使用 SQL 时只需告诉数据库需要干什么，不必描述怎么干。因此，对于用户来说，不需要关心 SQL 语句具体的执行细节，只需要理解其逻辑含义，具体细节和过程由数据库管理系统来完成。

(2) SQL 语言简单易用。SQL 语言完成核心功能的语句只用了 9 个动词，而且语法结构很接近于自然语言。

(3) SQL 是一种面向集合的操作语言。它的操作对象和操作结果都是元组的集合，对数据的处理都是成组而不是一条一条处理的。采用这种操作方式，大大加快了数据的处理速度。

(4) SQL 语言统一。SQL 命令具有交互式和嵌入式两种执行方式。交互式 SQL 是指用户在数据库管理系统产品的工具环境(如 Oracle 提供的 SQL * Plus 工具、MySQL 提供的 mysql 工具)中直接输入 SQL 命令对数据库进行操作；嵌入式 SQL 是指将 SQL 语句嵌入到高级语言程序(如 Java、VB、Pro * C/C++、C # 等)中来实现对数据库的操作。两种方式中 SQL 语言格式、用法基本相同，这种统一的语言风格给程序员设计程序提供了很大的方便。

3.2　数据库连接技术

开发动态网站中的网页与数据库打交道之前都需要和数据库建立连接，然后才能执行具体的数据存取操作，并在网页中处理这些数据。动态网页开发中常用的数据库连接技术主要包括 ODBC、ADO、JDBC 和数据库连接池。

3.2.1　数据库连接方式

1. ODBC 技术

ODBC(Open Database Connectivity)即开放数据库连接技术，是微软和一些数据库厂商联合制定的、实现应用程序和关系数据库之间通信的接口标准。它本质上是一组数据库访问的应用程序编程接口(API)，由一组函数调用组成，核心是 SQL 语句。

一个基于 ODBC 的应用程序对数据库的操作不依赖任何 DBMS，不直接与 DBMS 打交道，所有的数据库操作由对应的 DBMS 的 ODBC 驱动程序完成。也就是说，不论是 Access、MySQL、SQL Server 还是 Oracle 数据库，均可用 ODBC API 进行访问，用户直接将 SQL 语句送给 ODBC 即可。由此可见，ODBC 能以统一的方式处理所有的关系数据库。

应用程序要访问一个数据库，首先必须用 ODBC 管理器(在控制面板中)注册一个数据源，管理器根据数据源提供的数据库位置、数据库类型及 ODBC 驱动程序等信息，建立起 ODBC 与具体数据库的联系，这样只要应用程序将数据源名提供给 ODBC，ODBC 就能建立起与之相应的数据库的连接。

2. ADO 技术

ADO(ActiveX Data Object)是微软公司基于 COM 的数据库应用程序级的编程接

口。通过 ADO 连接数据库，可以灵活操作数据库中的数据。

使用 ADO 访问关系数据库有两种途径：一种是通过 ODBC 驱动程序；另一种是通过数据库专用的 OLE DB Provider，后者有更高的效率。

3. JDBC 技术

JDBC(Java Data Base Connectivity)是 Java 应用程序访问关系数据库的接口。JDBC 是一种用于执行 SQL 语句的 Java API，可以为多种关系数据库提供统一的访问接口。

在 JSP 中一般使用 JDBC 访问数据库。

4. 数据库连接池技术

在传统方式下，应用程序每次访问数据库时都要进行数据库连接的建立操作，对于一些大型网站如银行网站的网络环境下的数据库应用，由于用户众多，系统开销会很大，无法满足用户的需求，这时通常使用数据库连接池技术。

数据库连接池技术的核心思想是：连接复用，通过建立一个数据库连接池以及一套连接使用、分配、管理策略，使得该连接池中的连接可以得到高效、安全地复用，避免了数据库连接频繁建立、关闭的开销，这项技术能明显提高对数据库操作的性能。

3.2.2 网页访问数据库程序举例

不同的网站开发语言在连接数据库的方法上虽然不尽相同，但是其基本原理是相同的。在网站开发中连接数据库时还需要根据使用的网站开发语言了解其数据库连接方法，一般都是固定模式，不需要去理解，只要套用模式即可。下面是 JSP 以 JDBC 方式分别对 Oracle、SQL Server、MySQL 这 3 种数据库进行连接、操作的程序例子。

1. JSP 通过 JDBC 连接访问 Oracle 数据库

下面给出了 JSP 参考代码，程序名为 p_Oracle.jsp，此程序已在 Windows 平台 Tomcat 5.5+jdk1.5.0_02+Oracle 9i 环境运行通过。

```
<%@ page contentType="text/html;charset=gb2312"%>
<%@ page import="java.sql.*"%>
<html>
<body>
<table width="500" border="1">
<%
//加载 Oracle JDBC 驱动程序
Class.forName("oracle.jdbc.driver.OracleDriver");
//数据库地址
String url=" jdbc:Oracle:thin:@127.0.0.1:1521:stud ";
String user="tbuser";                          //数据库用户名
String password="tbuser";                      //数据库用户口令

//连接数据库
Connection conn= DriverManager.getConnection(url,user,password);
//向数据库发送 SQL 语句
Statement stmt=conn.createStatement(
ResultSet.TYPE_SCROLL_SENSITIVE,ResultSet.CONCUR_UPDATABLE);
```

```
String sql="SELECT * FROM zhutib ";
ResultSet rs=stmt.executeQuery(sql);
//处理 SQL 执行结果
while(rs.next()) {%>
<tr><td><%=rs.getString("ztxh")%> </td>
<td> <%=rs.getString("ztbt")%></td>
<%}%>
</table>
<%out.print("太好了!数据库操作成功了!");%>
<%//关闭所创建的各个对象
rs.close();
stmt.close();
conn.close();
%>
</body>
</html>
```

2. JSP 通过 JDBC 连接访问 SQL Server 数据库

下面给出了 JSP 参考代码，程序名为 p_SQLServer.jsp，此程序已在 Windows 平台 Tomcat 5.5+jdk1.5.0_02+SQL Server 2005 环境运行通过。

```
<%@ page contentType="text/html;charset=gb2312"%>
<%@ page import="java.sql.*"%>
<html>
<body>
<table width="500" border="1">
<%
//加载 SQL Server JDBC 驱动程序
Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
//数据库地址
String url="jdbc:sqlserver://localhost:1433;" +
           "databaseName=STUD;integratedSecurity=false;";
String user="newuser";                          //数据库用户名
String password="newuser123";                   //数据库用户口令

//连接数据库
Connection conn= DriverManager.getConnection(url,user,password);
//向数据库发送 SQL 语句
Statement stmt=conn.createStatement(
ResultSet.TYPE_SCROLL_SENSITIVE,ResultSet.CONCUR_UPDATABLE);
String sql="SELECT * FROM zhutib ";
ResultSet rs=stmt.executeQuery(sql);
//处理 SQL 执行结果
while(rs.next()) {%>
<tr><td><%=rs.getString("ztxh")%> </td>
<td> <%=rs.getString("ztbt")%></td>
<%}%>
</table>
<%out.print("太好了!数据库操作成功了!");%>
```

```
<%//关闭所创建的各个对象
rs.close();
stmt.close();
conn.close();
%>
</body>
</html>
```

3. JSP 通过 JDBC 连接访问 MySQL 数据库

下面给出了 JSP 参考代码，程序名为 p_MySQL.jsp，此程序已在 Windows 平台 Tomcat 5.5+jdk1.5.0_02+MySQL 6.0 环境运行通过。

```
<%@ page contentType="text/html;charset=gb2312"%>
<%@ page import="java.sql.*"%>
<html>
<body>
<table width="500" border="1">
<%
//加载 MySQL JDBC 驱动程序
Class.forName("com.mysql.jdbc.Driver");
//数据库地址
String url="jdbc:mysql://localhost/TB?useUnicode=true&characterEncoding=gb2312";
String user="root";                    //数据库用户名
String password="root";                //数据库用户口令

//连接数据库
Connection conn= DriverManager.getConnection(url,user,password);
//向数据库发送 SQL 语句
Statement stmt=conn.createStatement(
ResultSet.TYPE_SCROLL_SENSITIVE,ResultSet.CONCUR_UPDATABLE);
String sql="SELECT * FROM zhutib ";
ResultSet rs=stmt.executeQuery(sql);
//处理 SQL 执行结果
while(rs.next()){%>
<tr><td><%=rs.getString("ztxh")%> </td>
<td> <%=rs.getString("ztbt")%></td>
<%}%>
</table>
<%out.print("太好了!数据库操作成功了!");%>
<%//关闭所创建的各个对象
rs.close();
stmt.close();
conn.close();
%>
</body>
</html>
```

对以上 3 个 JSP 程序比较后发现，访问数据库都经历连接的建立、利用 SQL 操作数据、释放连接 3 个环节，而只有加载的 JDBC 驱动程序和数据库连接地址串格式两处不同。

3.3　动态网站中的数据处理

在动态网站中都要对用户数据进行必要的操作，即对数据库进行相应的操作。在网站中对数据库通常有以下几种操作。

(1) 建立存储数据的二维表(创建表)。

(2) 添加数据。

(3) 更改数据。

(4) 删除数据。

(5) 从数据库中查询出所关心的数据。

这些操作都需要通过 SQL 语句完成。

本章如无特别说明(除了 3.8 节之外)，所有语句和举例将都采用 Oracle 语法，并且所有实例均已在 Oracle 9.2.0.1.0 环境中运行通过。

3.3.1　定义网站数据库表

网站数据主要以数据库表对象的形式组织并存储。这需要两步才能完成：第一步定义一个数据库表，也就是定义表结构，相当于在数据库中建一张只有表结构的空表，以后再填数据；第二步向该表添加数据。

表建立之后如果不甚满意，或者随着时间的变化可能有的列变得过时没用，而有的属性需要增加进来，那么就需要对表的结构定义做些修改操作。修改结构包括增加新列、删除旧列以及改变一些列的宽度等。

本节就针对表(结构)的定义和修改操作展开介绍，这部分属于 SQL 的 DDL 部分。

1. 创建表

创建新表通过 CREATE TABLE 语句完成，基本格式为

```
CREATE TABLE [用户.] <表名>
    ( <列名> <数据类型> [,...] );
```

说明：

(1) 在该命令格式中，“[ ]”表示可选，“< >”表示必选，“,...”表明可以重复前面的内容，在本语法中表明可以定义多个列。如不特别申明，本书中提到的语句格式一律遵循此约定。

(2) [用户.]指定新表属于哪个用户，省略则表示属于当前用户。

(3) <表名>指用户所要创建的表的名称，它可以由一个或多个列(属性、字段)组成，同时要为列指定合适的数据类型。

数据类型是用来表现数据特征的，它决定了数据在计算机中的存储格式、存储长度以及数据精度和小数位数等属性。表中的每一列都必须确定数据类型，数据类型一旦确定，也就确定了该列数据的取值、范围及存储格式。数据库中最常用的数据类型包括字符型、数值型、日期型，如表 3-2 所示。不同的 DBMS 数据类型的标识符不尽相同。

表 3-2 常用数据类型

大 类	数据类型	说 明	举 例
字符型(需要定界符,一般用单引号)	CHAR(n)	用于定义固定长度的字符数据,其中 n 表示字符串的长度	学生性别列,值有 2 个:“男”、“女”,因此性别可以定义为 CHAR(2);邮政编码,如“050021”,通常定义为 CHAR(6)
	VARCHAR(n)	用于定义可变长度的字符数据,可变长度是指在定义之初,系统并不为变量分配长度为 n 的空间,而是在使用中按需分配,最大长度为 n	通信地址,如“石家庄体育南大街”,应用字符型,而且地址长短不等,故应定义为 VARCHAR 型,长度取所有地址中最长的,如 VARCHAR(100)
数值型	INT	用于定义整型数据,不需指定长度	年龄、成绩、人数等都是整型数,如年龄 18 岁:18,总职工 680 人:680。数值型数据经常做数学计算
	NUMBER(n,m)	用于定义整型或带小数位数的数据,n 表示总位数,m 表示小数点后位数	工资、电费、平均成绩等,如工资 930.35 元:930.35
日期型(需要定界符,一般用单引号)	DATE	用于定义日期类型的数据,不需指定长度。DATE 是 Oracle 日期类型,默认格式形如“20/2 月/1985”,SQL Server、MySQL 中日期则为 DATETIME,默认格式形如“1985/2/20”	工作日期、出生日期等,如某人生日为 1985 年 2 月 20 日:“20/2 月/1985”(Oracle)、“1985/2/20”(SQL Server、MySQL)

【例 3-1】 建立“学生选课”数据库中的 student 表。SQL 语句为

```
CREATE TABLE student
    (sno CHAR(11),
    sname VARCHAR(10),
    ssex CHAR(2),
    sage INT,
    sdept CHAR(2),
    sdate DATE );
```

执行结果如图 3-1 所示。

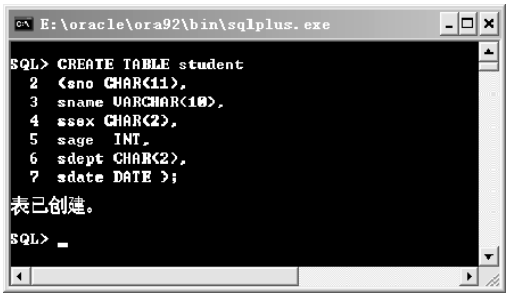

图 3-1 创建表 student

【例 3-2】　建立“学生选课”数据库中的 sc 表。SQL 语句为

```
CREATE TABLE sc
    (sno CHAR(11),
    cno CHAR(3),
    grade INT );
```

【例 3-3】　建立“贴吧”数据库中的 zhutib 表。SQL 语句为

```
CREATE TABLE zhutib
    ( ztxh INT,
    ztbt VARCHAR(50),
    ztzw VARCHAR(1000),
    ftr VARCHAR(20),
    ftrq DATE );
```

2. 修改表

表被创建之后，还会经常做一些修改。比如“学生选课”数据库中 student 表在创建之后，经常需要打印班级名单和班级成绩单，因此存储每个学生的班级信息很有必要；相反，由于该数据库注重的是成绩及相关数据的管理，而“年龄”信息与成绩联系不大，因此“年龄”列可以去掉。

SQL 语言中通过 ALTER TABLE 命令为已存在表增加新列、删除不需要的列以及更改某列的数据宽度或数据类型。但需要注意：数据宽度一般应越来越大，数据类型改变后应与原来相容。ALTER TABLE 命令的一般格式为

```
ALTER TABLE <表名>
    ADD <新列名><数据类型>
    | DROP COLUMN <列名>
    | MODIFY <列名><数据类型>;
```

说明：

(1) <表名>指定需要修改的表。

(2) 在该命令格式中，“|”表示前后内容可以任选其一。此处表示可以从 ADD、DROP COLUMN、MODIFY 三个子句中任选其一。

(3) ADD 子句用于增加新列。表在增加一列后，所有的数据记录在新增加的列上值都为空。空值(用 NULL 表示)是数据库中一个特殊值，表示实际值未知或无意义，它与字符空格、数值 0 不是一回事。在一个表中，如果一行中的某列没有值，那么就称它为空值。关于空值的操作较为特殊，后面再予介绍。

(4) DROP COLUMN 子句用于删除指定列。

(5) MODIFY 子句用于修改指定列的定义。

【例 3-4】　向“学生选课”数据库中的 student 表增加一个班级列 sclass。SQL 语句为

```
ALTER TABLE student
ADD sclass VARCHAR(20);
```

执行结果如图 3-2 所示。

图 3-2 增加新列

【例 3-5】 向“贴吧”数据库中的 zhutib 表增加回贴数一列 hts。SQL 语句为

```
ALTER TABLE zhutib
ADD hts INT;
```

【例 3-6】 删除“学生选课”数据库中的 student 表中 sage 一列。SQL 语句为

```
ALTER TABLE student
DROP COLUMN sage;
```

如果同时删除两列或更多列,则需要使用如下语法:

```
ALTER TABLE student
DROP (ssex, sclass);
```

上述命令将 student 表的 ssex 与 sclass 列同时删除,此时注意不允许再使用 COLUMN 保留字,且删除的列应放在括号中。

本例的执行结果如图 3-3 所示。

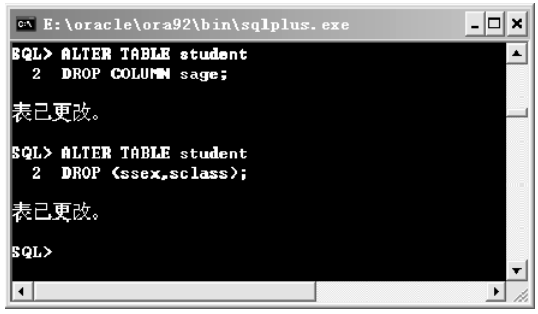

图 3-3 例 3-6 执行结果

【例 3-7】 将“学生选课”数据库中的 student 表中 sname 列的宽度增加到 20 个字符。SQL 语句为

```
ALTER TABLE student
MODIFY sname VARCHAR(20);
```

3. 删除表

删除表会将该表的定义以及所有数据一并删除。一般格式为

DROP TABLE <表名>;

【例 3-8】　删除“学生选课”数据库中的 student 表。

```
DROP TABLE student;
```

3.3.2　操纵网站数据

数据库表可以存储网站的数据，但是表创建之后，表中还没有任何的数据，必须向表中添加记录行数据才能实现数据的存储。

网站维护工作中，除了要向网站添加新数据之外，还经常要对网站现有数据进行修改更新、对网站某些数据做删除等，这些属于 SQL 的 DML 部分，是 SQL 的核心功能部分，称为数据操纵。SELECT 语句在 SQL 中使用频率最高、用法也最为复杂，在 3.3.3 小节专门介绍。

1. 向网站添加新数据

SQL 语言使用 INSERT 语句添加新的数据行，一般一次添加一行数据，有时也可以成批添加。

(1) 添加一条记录数据

添加一条记录的 INSERT 语句格式为

```
INSERT INTO <表名> [(列名[,...])] VALUES (值 1[,...]);
```

说明：

① [(列名[,...])]称为列名列表，用于指明要添加值对应的列。如果表中某些列在[(列名[,...])]中没有出现，则新记录在这些列上将取空值；如果所有的列都有对应值添加，则这部分可以省略不写。

② (值 1[,...])称为值列表，与 [(列名[,...])]或表定义的列必须在个数、先后次序、数据类型、宽度要求等方面完全一致；否则系统不是出现错误，就是添加的数据与字段含义不符。

【例 3-9】　将两名新生信息添加到“学生选课”数据库的 student 表中。两名新生信息为：①张一平，男，18 岁，计算机系(CS)，2007 年 9 月 1 日入学，分配学号是 20070501101；②马骏，男，数学系(MA)，19 岁，2007 年 9 月 2 日入学，分配学号是 20070501105。

用如下 SQL 语句向数据库添加“张一平”学生的信息：

```
INSERT INTO student
VALUES ('20070501101','张一平','男',18,'CS','1/9 月/2007');
```

“张一平”各项信息完整，添加命令省略了列名列表，命令简洁，但是注意各项信息与表结构定义顺序(参见例 3-1)要相一致。命令执行结果如图 3-4 所示。

而“马骏”同学信息通过以下语句添加：

```
INSERT INTO student (sno,sname,sage,sdate,ssex,sdept)
VALUES ('20070501105','马骏',19,'2/9 月/2007','男','MA');
```

图 3-4 添加数据时未指定列名列表

虽然该生各项信息也很齐全，但是对表列的定义顺序却不十分清楚，所以只能在添加的命令中提供上列名列表，这样只需要确保值的列表与提供的列名列表一致即可。命令执行结果如图 3-5 所示。

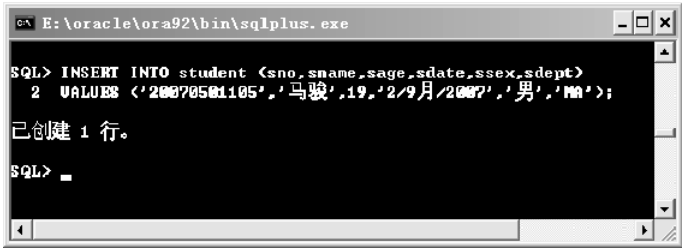

图 3-5 添加数据时指定列名列表

如图 3-6 所示，如果不指定列名列表，值就会根据表列的定义顺序进行对应添加，结果导致了错误。错误发生在第 4 项数据上，要求的年龄是数值型数据，而提供的却是一个日期型数据。

图 3-6 添加数据时值列表与表列未对应

【例 3-10】 新添加的记录仅部分列有值。

```
INSERT INTO student (sno, sname, ssex, sdept)
VALUES ('20070501102','赵敏','女','CS');
```

执行结果如图 3-7 所示。

新添记录在其他列上取空值，意即该学生的年龄、入学时间信息未知，这可以从图 3-8 所示的查询结果得到验证。