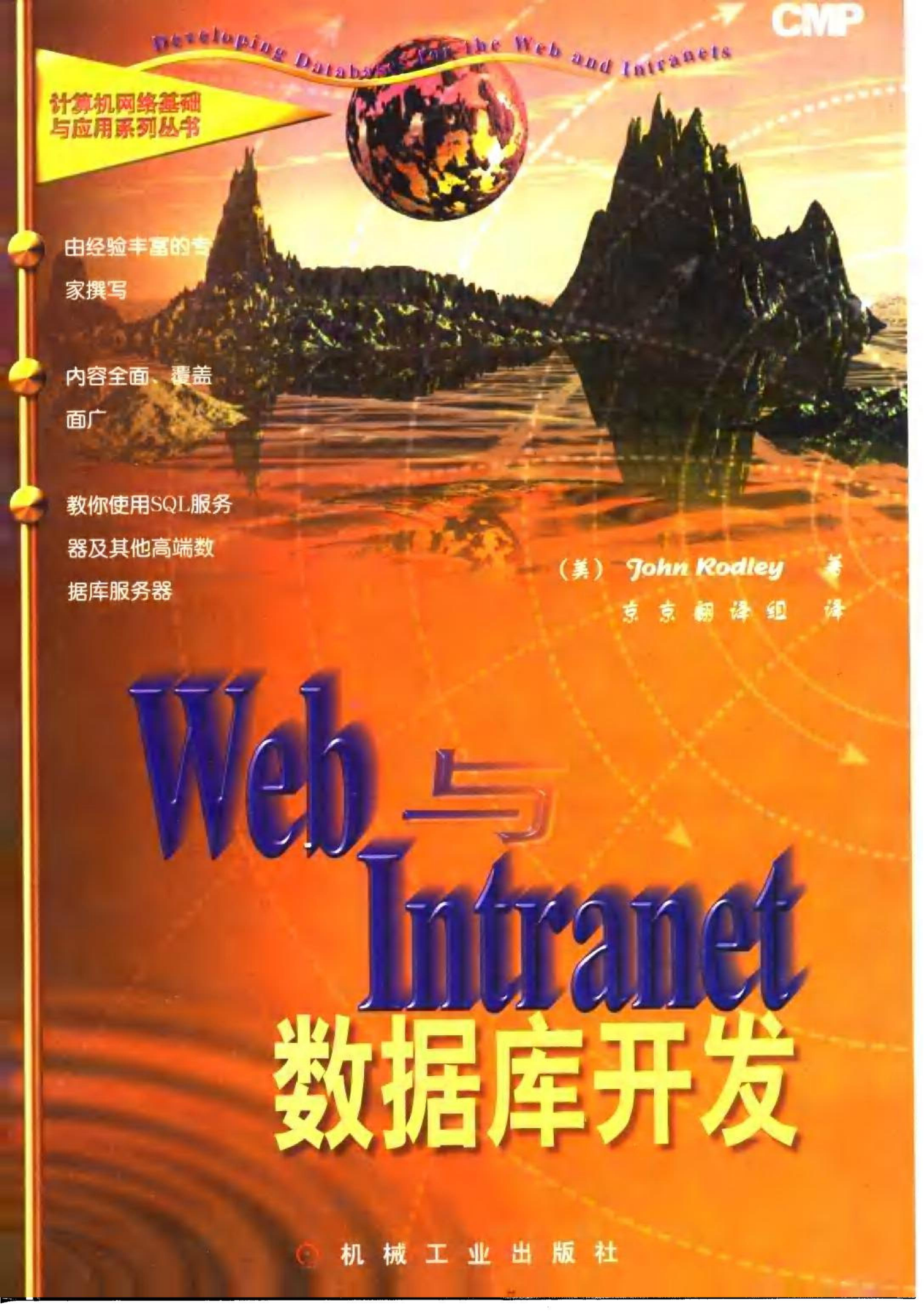
CMP
Developing Database for the Web and Intranets
计算机网络基础
与应用系列丛书
由经验丰富的专
家撰写
内容全面、覆盖
面广
教你使用SQL服务
器及其他高端数
据库服务器
(美) John Rodley 著
京京翻译组 译
Web与Intranet
数据库开发
机械工业出版社

计算机网络基础与应用系列丛书

Web 与 Intranet 数据库开发

（美）John Rodley 著
京京翻译组 译

机械工业出版社

本书向读者揭示了 Web 数据库创建最核心的技术。内容包括：用三种不同的途径来创建高性能的数据库应用程序；Web 服务器运用的内部技术；怎样使用 ODBC、微软 SQL 服务器以及其他高端数据库服务器；对任意数量的并发用户，寻求 Web 数据库性能优化的最佳方式；如何创建快速、灵活的“购物手推车”系统；阻挡对 Web 安全系统的渗透的“反黑客”技术以及如何为 Web 数据库创建 Java 程序片等，本书是网络应用程序开发人员尤其是 Web 数据库开发人员不可多得的参考书。

John Rodley: Developing Databases for the Web and Intranets.

Authorized translation from the English Language edition Published by Coriolis Group Books.

Copyright 1997 by The Coriolis Group Books.

All rights reserved.

本书版权登记号：图字：01-97-0557

图书在版编目（CIP）数据

Web 与 Intranet 数据库开发/（美）罗德利（Rodley, J.）著；京京翻译组译．—北京：机械工业出版社，1997.8

（计算机网络基础与应用系列丛书）

书名原文：Developing Databases for the Web and Intranets

ISBN 7-111-05872-0

Ⅰ.W… Ⅱ.①罗… ②京… Ⅲ.全球网络：互联网络-数据库管理系统-软件开发 Ⅳ.TP311.13

中国版本图书馆 CIP 数据核字（97）第 16754 号

出版人：马九荣（北京市百万庄南街 1 号　邮政编码 100037）

责任编辑：何伟新

北京大地印刷厂印刷　·新华书店北京发行所发行

1997 年 8 月第 1 版第 1 次印刷

787mm×1092mm　1/16 · 20.25 印张　505 千字

印数：6000 册

定价：34.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

引　言

从本世纪80年代开始，随着局域网应用的爆炸性增长，许多PC程序员都争先恐后地拥入了客户机/服务器编程的世界。在另一方面，如将局域网的增长比作“爆炸”，那么Web的发展无疑就是“核爆炸”。从前，成千上万的程序员竭尽所能变出形形色色的魔术，编写出有特色的、独立的Windows程序，他们现在却突然置身于Web无所不在的空间里。在这里，成百万台客户机和服务器争相吸引公众的注意力。这是一个令人激动的场所，也是一个老式魔术再也玩不起花样的地方。

不仅老式的魔术不再管用，用户到应用程序之间的途径也充满了艰辛。事实上，如将坐在浏览器前的一名互联网用户连至公司服务器，其间难免会穿过一大帮相识或不相识的人，而这些人相互之间难免有喜欢对方的，也有不喜欢对方的，黑客由此而产生。即便那些有经验的程序员，他们第一次在Web空间里历险的时候，经常也会在沉陷得很深的时候，才发现自己忘记了一些真正重要的东西。

如果这一小段演说打动了您的某根心弦，那么这本书便是为您准备的。在这本书里，我们将经历一次有意义的Web编程探险，通过三种不同的途径将商业应用程序放到World Wide Web里。首先，在第1章到第4章里，我们将采取最困难的方法，设置一个简单的数据库，并用C、CGI和一个独立的多用户数据库编写一些查询屏幕以及少许数据输入功能。在这一部分里，示范的Newfoundland Dog数据库将使用当今的技术，将数据库事务处理限制在服务器内，并通过HTTP服务器传递所有通信数据。根据这种思路，我们将从头构建一个SQL Server数据库；用C语言构建一个完整的CGI数据库访问程序；并草拟HTML文档，用它把各部分的信息链接到一块儿。在第4章，我们最终将得到一个完整的Web数据库应用程序（它的实际效果可访问http：//www.channell.com/users/ajrodley获知）。

在第5章，我们将完成一些更有趣的工作，使用像Java这样的工具为一种典型的商业应用程序——无所不在的股票报价系统——提供活动界面。在当今的WebMaster之间，活动Web内容无疑是一个很热的话题。在这里，我将采取一个关键的步骤：将活动内容与有用的数据链接起来。

第6章的工作仍很有趣，我们准备构建目前最热门应用程序之一：一个国际互联网商店。本章将涉及Web应用程序开发时比较头痛的一些问题，比如怎样使用来自JavaSoft公司的Jeeves Web服务器进行身份鉴定以及状态维护。

在第7和第8章里，我们将回顾前面创建的三个应用程序，并对一些特殊的问题进行更深入的讨论。第7章讨论的是安全问题，第8章讨论的数据库问题——不能运作时怎么办。

Web应用程序的开发就像乘坐快速的旋转木马。没有方便的时间或地点可供上去，旋转的地板会使我们受到伤害。尽管有些方案今天看起来管用，但到了明天也许就会过时，甚至会显得相当愚蠢。事物的发展本来就是这样的，推陈出新，才会有历史的进步。当您读到本书时，如果发现其中的一部分内容已略嫌陈旧，那么我一点都不会感到震惊。但是，为了安全跳上这个不断旋转的木马，也不是全无技巧可言的。至少，在起跳以前，可以绕着它加速

跑，与它同步以后，再一跃而上，我希望，这本书能帮助您朝正确的方向快速起步。

本书是为程序员编写的。它覆盖了大面积的问题，使用了多种不同的工具，因此，其中肯定充斥着各种代码示例以及对数据库和编程问题的深层次讨论。对大部分基本技术来说，我只是简单地进行了一些介绍，多数情况下，都是很快地切入正题。因此，为了充分使用这本书，您应该较熟悉C程序、SQL和基本的关系型数据库知识。至于是不是还应熟悉Java，我却有些犹豫，因为目前只有很少一部分人掌握它，但事实上，Java是相当有用的一种工具。另外，如果您恰好是一名能干的C/C++程序员，那么用Java编写的东西看起来将非常熟悉，而且几乎能毫无阻碍地理解它——至少语法结构是这样的。但是，不管怎样，在本书的下半部分之前，您不会看到有关Java的任何迹象，到真正接触到Java的时候，我们已经建好了一个能正常使用的Web数据库。

本书使用了许多HTML代码，但就我个人的意见来说，为阅读这本书，深层次的HTML知识并非特别必要的。如果您有足够的C知识，那么使用一种HTML编辑器（我用的是SoftQuad公司的HotMetal），花上半小时的时间，就能很轻松地掌握HTML的运用。

工具和移植

下面列出的是本书用到的一些工具，所有示例都要以这些工具为基础，注意，只有使用一模一样的工具，本书的示范代码才能保证在不作任何修改的前提下编译与运行。但是，由于历史和其他方面的原因，您或许不得不使用其他工具与平台，根据我的经验，对这里使用的例子来说，交换工具并不是特别困难。但是，大家都知道，任何事情都是易说难做的，所以，我在下面除列出工具与平台以外，还指出了它们可能带来什么样的移植问题。如果您不能确定这本书是否适用于您目前的系统配置，那么请从下表里寻求解答。

浏览器：Netscape Navigator

Web应用程序开发的前提是：无论用户使用什么浏览器，都能将有用的信息提供给他们。所以，使用什么浏览器不能成为不买这本书的理由。

Web服务器：Jeeves、IIS以及Netscape Commerce Server

这个问题不难解决。之所以同时列出这三种Web服务器，是因为它们是目前最流行的，而且我必须使用它们。在其他服务器里，没有提供一些高级的特性，比如“安全插座层”（SSL）等。但是，我用过的大多数服务器（O'Reilly，Apache……）都支持基本的CGI功能，这对Newf数据库的运行是足够的了。

数据库服务器：Microsoft SQL Server

同样，这个问题也不重要。尽管我提供的大多数配置信息都专门针对SQL Server，但是，对所有多用户、关系型数据库管理系统来说，这些配置后面隐藏的基本思路都是一致的。我试着对每个问题都用足够的深度解释，揭示出它们背后的基本概念，以便您能方便地应用于其他平台。但是，这样或许要求您手头放上一本使用手册，以便随时参考，当然，如果您患有“手册恐惧症”，恐怕就不易将本书的代码移植到其他数据库平台。

此为试读，需要完整PDF请访问：www.ertongbook.com

嵌入的 SQL 预处理器和库：MS ESQL/C

将 SQL Server 选定为数据库平台以后，再选择微软 ESQL/C 就是顺理成章的一件事情。但是，涉及代码移植的时候，这应是您最不用担心的一个问题。事实上，这里使用的嵌入 SQL 代码与我为 Sybase 及 IBM DB2/2 编写的嵌入式 SQL 代码没有任何不同之处。

C 编译器：Microsoft Visual C++

C 编译器的问题也不用费太大的心思。在本书里，我没有使用任何 Microsoft 基础类或任何 GUI 库，涉及移植问题时，这些东西会为我们带来很大的麻烦。我们在这儿使用的全是纯命令行应用程序，因此，任何一个有经验的程序员都能将它们移植到当前的任何一种操作系统里（甚至 Unix），同时不用耗费大量的时间与精力。唯一可能带来麻烦的地方是 newfq.exe 主函数里使用的异常控制代码，但是，一旦您真正理解了它的工作原理，对它进行修改也应是小事一桩。

操作系统：Windows NT 3.51 & 4.0

为保证程序完全可靠，在本书的大多数项目里，我都宁可使用 Unix，事实上，编制每一个项目的时候，我都回想起了自 70 年代末期以来编写 Unix 程序的愉快经历。Unix 多任务总要比其他竞争对手领先一筹，当今流行的大多数出色的编程思想都是从 Unix 移植过来的。不幸的是，通常都有一个（或两个、或三个）重要的原因限制我们不能使用 Unix，在多数情况下是费用问题。在 Unix 的世界里，任何东西都变得非常昂贵。所以我们选择了 Windows NT，将其作为唯一一种性价比很高的方式展示我们必须展示的那些编程思想。另外，根据我在 Unix 世界的经验，本书编写的代码应该很容易移植到其中，因为我随时都注意了不用受限于操作系统的特殊调用与工具。

Java

我们在股票报价器和购物手推车应用程序里使用的 Java 代码是完全能够移植的。如果您的操作系统提供了一个 Java 解释器（目前大多数都提供了这样的一个解释器），那么 Java 代码应该不加修改（甚至毋需重编译）即可运行。所有代码均用 Symantec 的 Cafe 生成，并符合 JDK 的 1.02 规范。

JDBC

有些 Java 例子依赖于某个安装好的 JDBC 驱动程序和类库。我在这儿使用的是 WebLogic 的 jdbcKona/MSSQLServer，其他方案无疑也是可行的。目前，JDBC 是尚处在发育期的一种标准。因此，我们能看到采用不同方式实现的兼容标准。

本书的技术支持

针对我的另一些书籍的作者，我提供了充分的技术支持。如果您对我写作的东西存在任何疑问，请放心地给我写信。如果您对某些内容不甚理解，或者不能应用于实际工作，请不要埋在心里——请向我致电。根据我的经验，通过几天的电子邮件交换，大多数问题都可以

迎刃而解。与我联系时，最好的方式就是向下述地址发电子邮件。

john.rodley@channel1.com

也可以通过出版商的 Web 站点与我联系。

http://www.coriolis.com

作为本书读者的特权，可免费下载我所有书籍示范代码的最新版本，地址如下：

http://www.rodley.com

或

http://www.channel1.com/users/ajrodley

除此以外，这些站点本身即运行了示范代码的最新版本。所以，举个例子来说，您完全可以登录到 rodley.com，然后检索 Newfoundland Dog Database。

在阅读本书的过程中，建议大家花点儿时间登录到支持站点。我们的读者经常都能带来重要的观点与意见，无论它们是否正确，都为大家提供了论证问题，并获得真理的绝好机会。因此，有空请来坐坐，接受或反驳其他读者的观点。通过这一过程，您往往能学到本书范围以外其他有价值的东西，同时，也能对我们的工作有所帮助。

目　录

引　言

第一部分

第 1 章　预备知识 …… 1
1.1　为什么要出版 …… 1
1.2　硬件 …… 2
1.2.1　停电 …… 2
1.2.2　硬盘故障 …… 2
1.2.3　软件错误 …… 3
1.3　网络连接 …… 3
1.4　HTTP 服务器 …… 5
1.4.1　安全防护 …… 7
1.4.2　CGI 支持 …… 7
1.5　数据库引擎 …… 8
1.5.1　并发数据库连接 …… 8
1.5.2　客户机/服务器结构的优势何在 …… 11
1.5.3　DBMS 级别的安全防护 …… 11
1.5.4　对象数据库 …… 12
1.6　还会遇到其他什么危险 …… 12
1.7　编程接口 …… 13
1.7.1　C、C++、Java 和脚本语言 …… 13
1.7.2　固有调用和 ODBC …… 13
1.7.3　JDBC …… 15
1.7.4　专门的一揽子方案 …… 16
1.8　总结 …… 16
1.9　Web 资源 …… 17
1.9.1　JDBC …… 17
1.9.2　ODBC 和固有调用 …… 17
1.9.3　对象数据库 …… 17
1.9.4　专门的一揽子方案 …… 17
1.9.5　数据库引擎 …… 17
1.9.6　Web 服务器 …… 17
1.9.7　编程接口 …… 18
第 2 章　提供一个被动视窗的只读数据库 …… 19
2.1　Newfoundland Dog Database …… 19
2.1.1　访问配置文件 …… 20
2.1.2　用 SQL Server 构建数据库 …… 23
2.2　Web 数据库的界面为何如此简陋 …… 28
2.3　HTML 表单 …… 29
2.3.1　表格 …… 32
2.3.2　预格式化文本 …… 34
2.3.3　元素命名 …… 36
2.4　用通用网关接口构建查询 …… 37
2.4.1　将表单信息输入 CGI 程序 …… 37
2.4.2　CGI 规范的其他元素 …… 50
2.5　总结 …… 52
2.6　Web 资源 …… 52
第 3 章　动态创建 HTML …… 53
3.1　处理致命错误 …… 53
3.2　数据库查询 …… 61
3.3　动态创建 HTML …… 85
3.4　HTML 模板处理器 …… 99
3.5　总结 …… 120
3.6　Web 资源 …… 120

第二部分

第 4 章　运行中的数据库示例 …… 121
4.1　工作原理 …… 121
4.2　这种方法的缺点 …… 123
4.2.1　域检验 …… 123
4.2.2　用 Java 进行域检验 …… 127
4.2.3　响应限制 …… 136
4.2.4　显示限制 …… 142
4.2.5　性能 …… 147
4.3　总结 …… 149
4.4　Web 资源 …… 150
第 5 章　活动视窗的客户机/服务器数据库 …… 151
5.1　样本程序 …… 151
5.2　具体的做法 …… 151
5.2.1　触发器 …… 152

5.2.2 扩展保存进程 …… 153
5.2.3 注册扩展存储进程 …… 156
5.2.4 扩展存储进程与 Java 服务器的通信 …… 157
5.2.5 Java 服务器 …… 163
5.2.6 创建 Java 固有方法 …… 166
5.2.7 为什么要用 Java 服务器 …… 184
5.2.8 Java 程序片用户界面 …… 185
5.2.9 综合 …… 199
5.2.10 编译和链接存储进程和固有方法 DLL …… 200
5.3 存在的一些问题 …… 201
5.4 总 结 …… 201
5.5 Web 资源 …… 202
第 6 章 通过 Web 接收输入 …… 203
6.1 场景管理 …… 203
6.1.1 甜饼 …… 203
6.1.2 甜饼恶魔 …… 205
6.1.3 隐藏域 …… 205
6.2 购物手推车 Web 站点 …… 205
6.3 为我们提供服务的一个 Web 服务器 …… 209
6.3.1 我们的服务程序片 …… 211
6.3.2 Doggie Diamonds 甜饼 …… 220
6.3.3 准备运行的一个小型 JDBC …… 221
6.3.4 同步和开放式连接 …… 225
6.3.5 用户界面的结构 …… 225
6.3.6 为购物手推车添加货物 …… 242
6.3.7 结帐程序 …… 257
6.3.8 还剩下什么工作 …… 271
6.4 总结 …… 272
6.5 Web 资源 …… 272

第三部分

第 7 章 安全防护 …… 273
7.1 拓扑 …… 273
7.2 操作系统和网络 OS 的安全 …… 277
7.3 数据库安全 …… 277
7.4 别人能偷走我的数据吗 …… 279
7.5 Web 服务器安全 …… 283
7.5.1 缓冲区溢出 …… 284
7.5.2 知识就是力量 …… 296
7.5.3 切记不要公开源代码 …… 297
7.6 硬壳和纵深防卫 …… 297
7.7 窃听和安全插座连接 …… 298
7.8 总结 …… 300
7.9 Web 资源 …… 301
第 8 章 操作问题 …… 302
8.1 经常备份 …… 302
8.2 警告和意外 …… 309
8.3 总结 …… 315

第一部分

第1章 预备知识

World Wide Web 最终还是与商业联姻了。在个人主页、动画、声音和其他大量有趣却不赚钱的领域里进行了短暂的冲锋以后，Web 终于体会到了自己真正的价值：用一种及时和令人愉悦的方式向人们提供大量有用的信息。而伴随着大量信息，几乎只意味着一件事情：大量的数据库编程，这就是编著本书的原因。

为数据库换上一张 Web 的脸面并不需要涉及高深的技术，但也没有想象中的那么容易。贯穿本书，我们将开发三个 Web 数据库应用程序：

- Newfoundland Dog Database（一个关于饲养狗的数据库）。
- 一个股票报价系统。
- 一个在线商店。

每个应用程序都为大家展示了 Web 数据库问题的不同的侧面，而且我们要用一系列工具解决它们。

在我们接触数据库编程问题的本质以前，首先要建立一个合适的环境。这个环境里涉及到以后准备使用的硬件，连入网络空间的方式，以及在网络计算机里需要运行的软件服务。准备好这个工作环境以后，还有一系列相当宽松的要求，它们描述了 Web 数据库服务器的样子。

如果您已经拥有一台连入网络的计算机，那么可以跳过有关硬件与网络连接的简介部分；如果您已对基本的数据库编程有着很好的理解，那么可以跳过本章的后面一半内容；另外，如果您实在没有什么耐心，那么完全可以跳至第 2 章，直接开始编写代码。

1.1 为什么要出版

我们中的许多人都准备在 Web 里放置一个属于自己的数据库。地球上的每个信息收集者都准备聘请一系列人员，在人类知识库里占据一小块地盘，让世界上的所有人都能使用自己的信息。在这种情况下，如果再对设置 Web 数据库的必要提出疑问，那么通常都会遭到嘲笑，人们会说：啊，这个问题太愚蠢了！毕竟，人们根深蒂固的概念是：Net 是一样好东西，而 Web 则是更好的东西。但至于为什么，许多人可就说不上来了。因此，就我自己的看法来说，如果问一问在 Web 里使用数据库到底有什么好处，也不能算是多此一举的事情。从本质上说，在 Web 里使用数据库有三方面引人注目的优点：

- 潜在的用户是与国际互联网连接的数百万乃至上千万的人。这是推动 Web 数据库开发的直接利益因素。用户们都实实在在地守在他们的浏览器前面，一旦您的数据库能通过一个网络 URL 访问，那么需要做的全部事情就只有引导用户查看与使用它们。当然，这是我们比较理想的一种期望。这类似于我们常常乐观地说“瞧，中国有上十亿人口呢”。在本世纪里，美国成千上万的商人们都怀着这个乐观的态度看待中国的“巨大”市

场。他们花费了许多年的时间与上百亿的资金培养中国的消费者，尽管目前尚未达到预期的目标，但是，和中国毕竟会成为一个大市场一样，Web 数据库的巨大潜力终究有一天也会被完全发掘出来。例如，我们的 Newfoundland Dog Database 已将信息的触角延伸到了数十个国家的成千上万名用户，其中包括一些听起来似乎不太可能的国家，比如芬兰和爱沙尼亚等，这个试验性质的数据库应用程序获得了令人相当满意的效果。

- 标准的、功能强大的用户界面外壳：Web 浏览器。尽管 HTML 存在一系列限制，但它毕竟是一种被广泛接受的标准。只需按标准行事，就能保证信息最终进入那些不由我们控制的用户软硬件配置环境。从前开发商业性数据库应用程序的时候，我们有权指定用户与我们的数据库进行通信的软件与硬件环境，但现在却时过境迁了，用户们可随心所欲地选择自己喜爱的系统平台。但是，我们仍有充分的理由假定用户的软件能理解 HTML。
- 数据库计算机到用户浏览器之间一条随时备用的数据管道。在当今销售的许多分布式应用程序里，仍然依赖于专利性质的软件包，这些软件包能通过电话线相互通信，并能通过某些约定（或令人费解的“协议”）对信息进行中继。费用低廉的国际互联网服务广泛流行以后，像这样的一些技术便应扔进垃圾堆了。用户随时都可依赖自己与网络空间（或称 Net，专指国际互联网，下同）的连接，这样一来，数据库供应商只需将数据库放到 Net 即可完事，不用担心协议的问题。在许多公司里，大多数员工都已成为网络社会的一员。

但是，为了真正能与数百万 Web 用户沟通，我们必须知道如何将自己的数据库计算机与 Net 相连，以及怎样与 HTML 兼容浏览器进行沟通。为完成这个目标，第一步便是设置具备 Web 能力的硬件。

1.2 硬件

Web 数据库计算机的许多要求与其他普通数据库服务器计算机的要求都是相同的。总的来说，它要求容量大、速度快以及相当可靠。大量 RAM 和硬盘空间是非常基本的条件。DBMS 是系统开销比较大的一种程序，如今的产品往往都使用了大量线程和进程，它们对内存和 CPU 都提出了苛刻的要求。对硬盘空间的需求最终取决于准备保存的数据量大小。

可靠性是另一个重要的因素。经常丢失数据的数据库是一文不值的，为了将数据丢失的几率降低到最小，必须考虑到下面这些可能的情况。

1.2.1 停电

停电造成的影响是不言而喻的，但是，这个问题很容易解决。具有自动关机功能的不间断电源（UPS）已有多年的发展历史，应该相当成熟。在这儿需注意的一个问题是，关机（切断电源）的同时也应正确中止 DBMS 的运行。

1.2.2 硬盘故障

这并非危言耸听，所有硬盘最终都是会崩溃的，因为任何东西都有一个极限寿命。尽管这听起来令人不好受，但实情确实如此，没有任何一种硬件能永远健康和长寿。Diablo（恐惧）游戏里的“永不磨损”装备只是一种神话。为有效解决这个问题，可考虑使用一个 RAID 磁盘阵列，它应具备数据冗余和热交换功能，以便能在不关机的前提下随时备份重要数据。另外还有一些比较普通的方法，比如定时用磁带备份等等。事实上，为减轻硬盘崩溃造成的损

此为试读，需要完整PDF请访问：www.ertongbook.com

失，对任何数据库服务器来说，最起码的要求就是经常（比如每周）进行磁带备份。

1.2.3 软件错误

这种错误很难发现，也很难解决，但是必须做到对它心中有数。您的 DBMS、操作系统、网络 OS、固件（BIOS 中保存的信息）、显示驱动程序、磁带备份软件——所有这些都存在着或大或小的缺陷（或称 Bug、臭虫）。有时，这些臭虫会原形毕露，导致机器的崩溃。同样，为了对付这类问题，唯一真正有效的方法仍是经常备份。

通过前面的讨论，我们事实上已经简要总结了防止数据丢失的几个步骤，或者说已拟定了一份“计划”。但是，如果不能正确地执行这个计划，那么也无法发挥出它应有的作用。怎样才能知道自己的数据保护计划万无一失呢？很简单，亲自动手测试它。

UPS 是最容易测试的部件。将 UPS 的电源从墙壁上拔掉，然后观察系统是否能正确关闭；再插上电源，观察系统是否能正确恢复。

磁带备份测试也同样简单，尽管它要花费更多的工作。随便关掉系统，然后用最新的备份恢复。如果正确无误，就表明磁带备份没有问题。

这听起来似乎过于简单，有经验的人员会对这种测试不屑一顾。对于您来说，这种测试或许真的有些多余。但我曾接触过许多客户，他们详尽制订了灾难恢复计划，但却从来没有人测试过它们的有效性，没有人花时间去做这些“没有价值”的事情。有的客户勤勤恳恳对他们的系统备份了很多年的时间（从来没有测试过使用的磁带），到最后真正出现事故，需要恢复以前的数据时，却发现这些磁带上根本没有记录下任何东西（一个程序上的错误使然）。不要让这种悲剧在您身上重演！不仅要很好地计划，忠实地执行，而且还要事先和定期进行测试。

我们的数据库计算机不仅需要容量大（无论哪方面）、速度快，而且至少还要使用一个不间断电源、磁盘备份设备以及一个可选的 RAID 磁盘阵列，从而保证数据的完整性，以及获得最大的效能。

1.3 网络连接

与把其他机器连入 Net 相比，将这个大型、快速的数据库服务器连入 Net 在大多数地方都没有什么不同。图 1-1 为大家展示了基本的连接方式，用户直接连入干线。为了与这些用户进行沟通，必须将数据库计算机与干线相连。大多数情况下，这需要向当地的国际互联网服务供应商（ISP）购买连接通道。这种连接通常由下面这些形式构成。

- 一条租用或拨号线路，通过 28.8 KB/S 的 Modem 连接，运行 SLIP 或 PPP 协议，以便建立一个 TCP/IP 连接。
- 一条 ISDN 线路，通过 ISDN Modem 以 64 KB/S 或 128 KB/S 的速率连接，运行 PPP 协议，以便建立 TCP/IP 连接。
- 一条分段式 T1 线路，利用专用硬件建立与干线的 TCP/IP 连接。

我们真正需要的是这样一台机器，它运行 TCP/IP，能通过某种类型的硬件连入干线，TCP/IP 连接的建立最终是由自己的国际互联网服务供应商建立的。ISP 为我们的机器提供了一个全世界独一无二的 IP 地址，拥有这种 IP 地址以后，国际互联网里的其他计算机只需访问这个地址，就可以进入我们的机器。

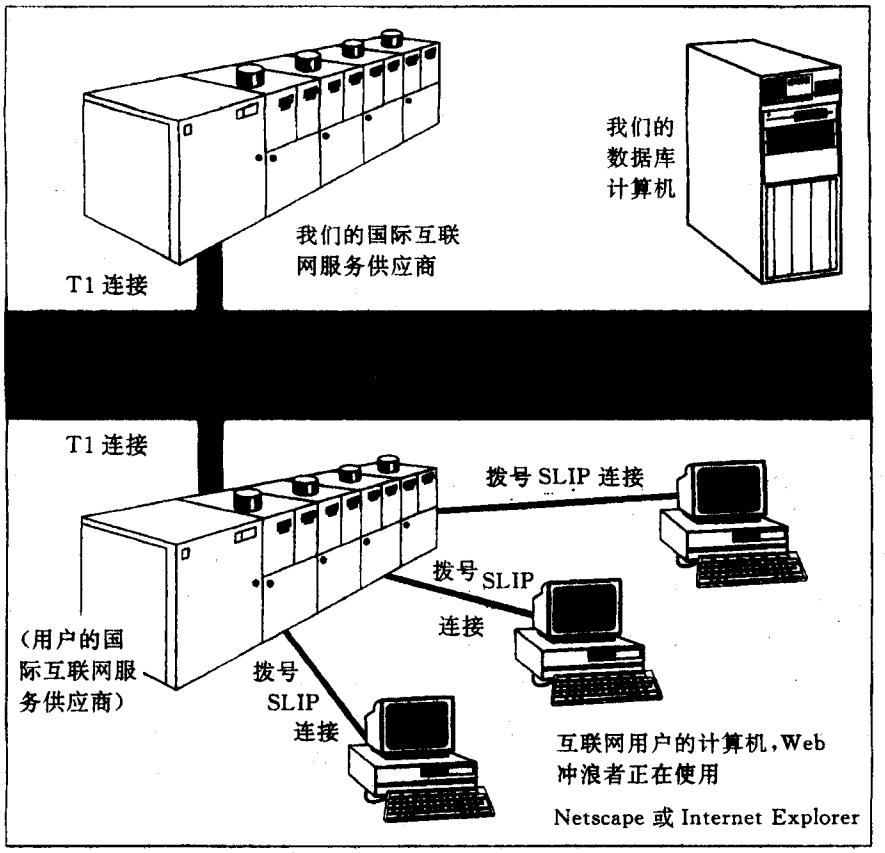

图 1-1　将计算机连入网络空间

IP 地址是进入我们 Web 城堡的大门，这种地址与一般的邮件地址相似。如果将邮件寄至奥迪斯街 50 号，那么我就能收到这份邮件。如果应用程序连至 IP 地址 199.9.1.13，那么就能与我的 ISP 连上。我们通常听说的 URL 采用像 http：//www.channel1.com 这样的形式，但这实际只是表面现象。通过一种名为 DNS（域名服务）的机制，站点名 www.channel1.com 将转换成 IP 地址 199.9.1.13。

通过 IP 地址采用的形式，我们可判断自己需要什么类型的 Net 访问。基本的 PPP 或 SLIP 连接通常为每名呼叫者自动分配一个准随机 IP 地址，因此，如果您选用廉价的拨号 PPP 方式与自己的 ISP 连接，那么每次呼叫 ISP 时都会获得一个不同的 IP 地址。这样一来，别人显然无法通过一个固定的地址找到您。

因此，这为我们的 Web 数据库服务器又多增加了两项要求。

- 与干线之间的一个 TCP/IP 连接。
- 一个固定的 IP 地址。

防火墙

尽管有关防火墙的全方位讨论已超出了本书的范围，但我们仍然不应忽视它的存在，因为防火墙几乎肯定是任何国际互联网数据库服务器设置的一个组成部分。防火墙的宗旨是将单位内部的网络环境与外部的网络通信隔开，在中间架设一道抵御外来入侵的“墙”。

实际的防火墙通常是运行防火墙软件的一台单独的计算机。在国际互联网里送入和取回的信息都要通过这个防火墙服务器。所有防火墙都采取了形形色色的技术，能将危险的通信拒之于自己的内部网络以外。在所有这些技术里，大多数都需要检查每个数据包，分辨它的始发地和目的地，以及它准备做什么，然后在这些信息的基础上，决定是否允许它通行。

由于 Web 服务器的宗旨是服务于各种各样的网络通信，其中当然包含了形形色色危险的信息，它通常安装于运行防火墙软件的那台计算机。这样一来，防火墙便不能对 Web 服务器起到保护作用。

如果数据库计算机是内部网络的一部分，那么必须在内部网和国际互联网之间建起一道防火墙。

1.4 HTTP 服务器

为了将数据库放置到 Web 空间，必须让大量组件协同工作。图 1-2 为大家展示了只提供 HTML 文档服务的一台 Web 服务器的模块图。

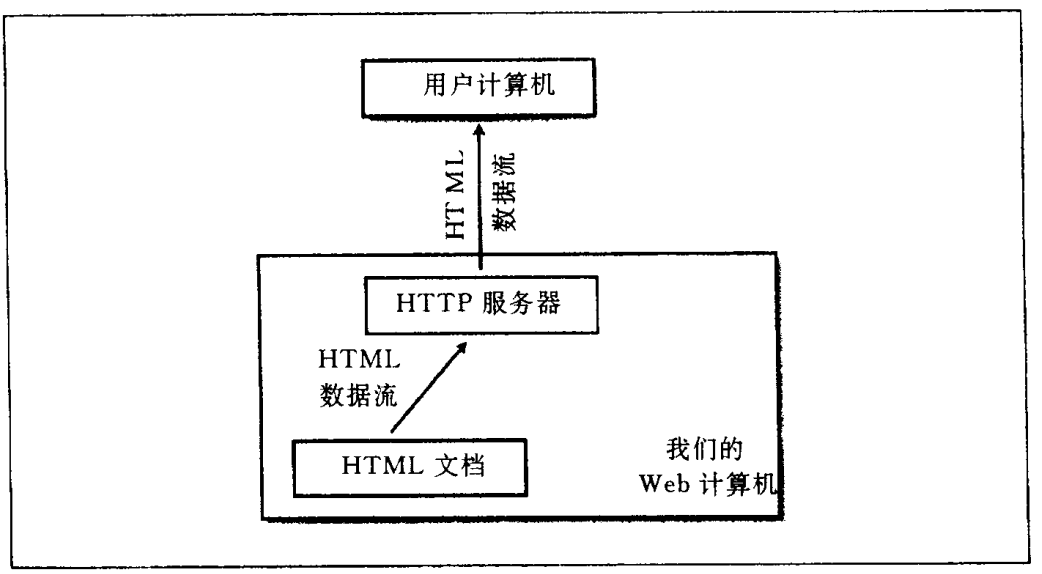

图 1-2 提供 HTML 文档服务的 Web 服务器

但是，实际提供 Web 服务时，问题却要稍微复杂一些。利用一个 CGI URL，用户可在服务器里实际调用并执行一个程序。此时，CGI 程序将实时（动态）创建 HTML 文档，然后由 Web 服务器将那个动态的 HTML 文档传回浏览器。添加了 CGI 以后，模块图就变成了图 1-3 的那个样子。

现在让我们进入正题，在系统里再添加一个数据库，如图 1-4 所示，我们已在 Web 服务器计算机里添加了一个数据库。

现在，我们的重点是如何利用 Web 服务器从数据库里取出信息，以及如何将它传至用户的浏览器。为了将数据从数据库里取至用户浏览器，图 1-3 为大家展示了两种明显的方法——创建静态的 HTML 文档，或通过 CGI 程序创建动态的 HTML 文档。

HTTP 服务器（Web 服务器）的职责说明相当简单：将文档发送给任何发出请求的浏览器。如换用更多的技术性术语进行说明，那就是：针对 HTML 文档请求，监听一个特定的端口，然后通过那个端口将请求的文档传送给发出请求的浏览器。下面是具体的步骤。

(1) 服务器站点监听一个特定的端口。
(2) 浏览器连至那个端口。
(3) 浏览器向服务器请求一个文档。
(4) 服务器将那个文档传给浏览器。

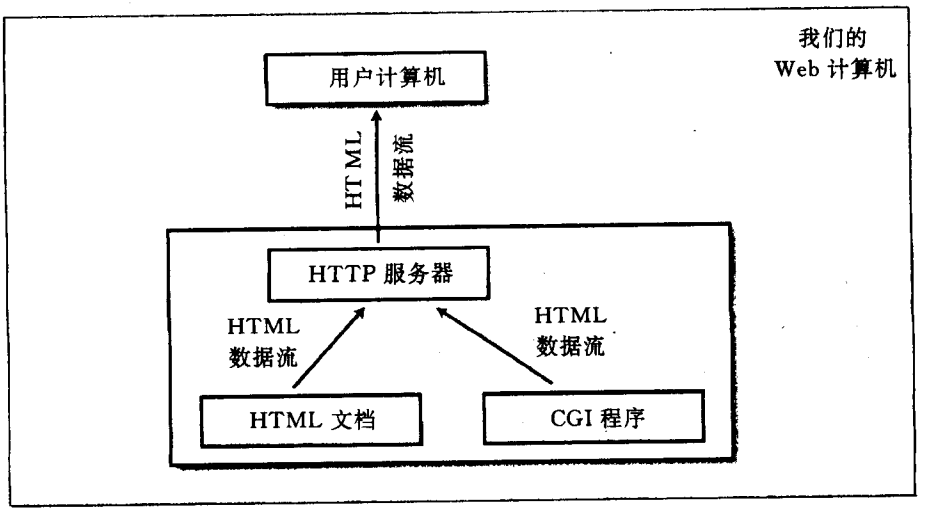

图 1-3 提供 HTML 文档服务及 CGI 程序的 Web 服务器

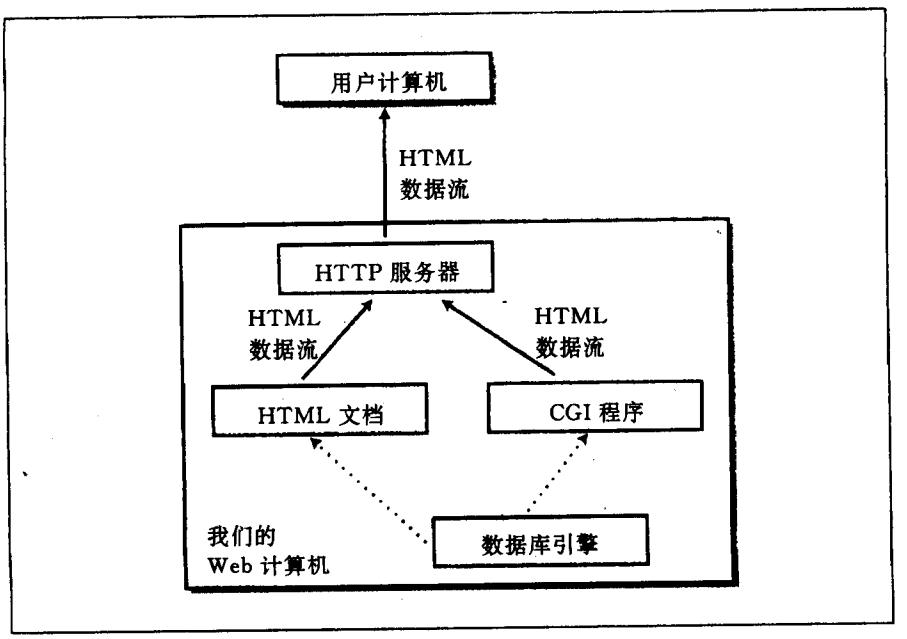

图 1-4 提供了 HTML 文档、CGI 程序以及一个数据库的 Web 服务器

举一个更具体的例子，假设您现在希望观看我过去的主页。此时，可在浏览器的 URL 地址框里输入：http：//www.channel1.com/users/ajrodley/index.html，然后指示浏览器与这

个地址连接。随后，浏览器会取得站点信息（http：//www.channel1.com），然后在其中寻找两方面重要的信息。首先是站点名，即 www.channel1.com，通过您的 DNS 服务器，浏览器会把这个名字转换成 IP 地址，即 199.1.13.9。其次，HTTP 协议会告诉浏览器使用端口 80，因为那是 HTTP 服务器的默认端口。因此，为了建立物理性的网络连接，浏览器必须知道所有这两方面的信息：IP 地址和端口号。

通过与 IP 地址 199.1.13.9、端口 80 连接，浏览器现在打开了与 HTTP 服务器的一个双向信道，这台服务器位于我的 ISP 那里。随后，浏览器会将 URL 剩余的部分（/users/ajrodley/index.html）传递给服务器，以此作为一个文档请求。请求的 index.html 文档通过网络抵达浏览器以后，浏览器会对其中包含的 HTML 代码进行解释，并在屏幕上显示出其中的内容。

如果有兴趣了解具体的 HTTP 服务器是什么样子，可参阅帕特里克·罗顿（Patrick Naughton）编著的《Java Handbook》一书。在这本书里，罗顿展示了一个相当基本的 HTTP 服务器，它使用的 Java 代码不超过 300 行。对各种 Web 服务器来说，区别主要在于安全防护、非标准的数据类型、CGI 支持、进程控制以及数据库集成等方面。

1.4.1 安全防护

安全防护这个问题值得用一整章的篇幅大写特写，但我们在这儿只是对它稍加探究。Web 服务器可提供两种类型的安全防护措施——授权和链接加密。

“授权”要求用户通过口令访问 Web 页，每一天，我们都需要用相同或不同的“用户名/口令”登录进入特定的系统。对有些服务器来说，比如 Netscape，可用授权屏幕保护单独的页不受非法侵入，甚至能保护整个目录。

“链接加密”是指对浏览器和 Web 服务器之间传递的数据进行加密。这样一来，即便有人暗中截获了在国际互联网里中途传送的资料，也能保证他们无法理解其中的含义。网景（Netscape）公司早已公布了一种标准，如今许多浏览器和服务器都应提供了对它的支持，这一标准即是众所周知的 SSL，即“安全插座层”。如果浏览器和服务器通过 SSL 连接到一起，那么中间传输的所有数据都会得到加密，对大多数国际互联网事务处理来说，这一级别的链接安全防护应该足够了。

1.4.2 CGI 支持

HTTP 服务器提供的另一项特性是支持一种名为“通用网关接口”（CGI）的标准。CGI 标准定义了在与 Web 服务器进行交互的一个 HTML 文档里，如何调用可执行程序。大多数 Web 服务器都支持“CGI 目录”（或称 cgi-bin）的概念，这种目录里保存的所有文件都应该是 CGI 程序。例如，在我的 Web 服务器里，假设已将/cgiDir 配置成 CGI 目录使用。如果一个浏览器请求获得文档 http：//www.myserver.com/cgiDir/AProgram.exe，那么此时并不是读取名为 AProgram.exe 的文件，然后将它的所有字节都传给浏览器，而是由 Web 服务器执行那个程序，然后将程序的输出结果反馈回浏览器。

这是一种相当有用的技术。利用 CGI，我们几乎可以完全自由地创建动态 HTML。事实上，在后面的 Newfoundland Dog Database 示范应用程序里，当根据用户查询创建一个页时，我们便采用了这种技术。

我们的 Web 服务器必须支持 CGI，支持 Web 页级别的安全防护，并且还有可能需要支持链接加密。

当然，和我们在这儿的只言片语相比，CGI 本身要复杂得多。在后续的章节里，我们还要

对 CGI 进行更加深入的讨论。

1.5 数据库引擎

数据库引擎是 Web 数据库站点最重要的组成部分之一。它决定了实际保存数据时采用的机制，并允许我们对数据进行查询与更新。

在附录 A 里，我们讨论了当今市场出售的一些 DBMS 软件包的优缺点。在后续的章节里，大多数时候使用的都是 Microsoft SQL Server。但是，我们会尽量避免使用这种产品专用的一些功能，这样做的目的很简单，使您应能方便地对代码进行修改，从而移植到自己使用的数据库引擎里。

针对 Web 评估一个数据库引擎的时候，首先应考虑的是多用户问题，也就是说，您的数据库必须能够应付多名并发访问的用户。针对这个问题，存在多种解决方案，我们在后面会详细讨论。但是，从总体上说，数据库引擎需要支持一定数量的并发访问。

为进一步简化问题，我们准备为 DBMS 添加另两个限制条件。

- 必须是关系型数据库。
- 必须支持 SQL/92 语言规范。

这些限制并不会为我们带来特别的麻烦，当今大多数流行的数据库软件包都符合这些标准。附录 A 列出了其中的一部分软件包。

1.5.1 并发数据库连接

针对准备选用的数据库引擎，我们已将并发连接能力指定成一种必须满足的条件。这个条件并不是可有可无的。由于数据库的并发访问能力决定了有多少人能同时使用它，所以大多数 RDBMS（关系型数据库管理系统）开发商都根据数据库允许的最大并发连接数量为自己的产品定价。对于每个并发连接，1000 美元的价格是相当普遍的，因此，为决定自己的购买策略，事先应估算出最终可能有多少个并发连接。另外，由于每个并发连接的价格较高，所以我们也有理由将并发连接的数量减小到最低程度。

不幸的是，并发连接的数量很难精确地计算，我们需考虑到下面这些问题。

- 平均和最长的事务处理时间。
- 用户愿意等待事务处理结果的时间。
- 试图同时执行事务处理的用户平均和最大数量。

如果您试着计算上述任何一个参数，很快就会意识到解决这个问题的难度有多大。

1. 平均和最长的事务处理时间

相比之下，这是最容易获得的一个参数，因为我们可以实地进行测试。首先可设置一个测试用的数据库，安排一些乐意参加的典型的事务处理用户，从而构建一个测试小组，测算每次事务处理用去的时间。根据这些结果，计算出平均和最大时间。

这种测试实际还存在许多问题。首先是事务处理的混合，我们仅仅依靠假定进行这种混合。对有些应用程序来说，用户会创建他们自己的查询，这样就使我们的假定变得毫无用处。另外，我们不可能模拟机器在重负载通信情况下的运作。在测试时，一次事务处理可能花 5s 的时间就能完成，但在某些重载环境下，则有可能花 50s 的时间才能完成。我们能作出的最好的假定就是：测算出来的数字在大多数情况下都是正确的。

2. 用户愿意等待事务处理结果的时间

此为试读，需要完整PDF请访问：www.ertongbook.com