

实时性 实用性 可靠性

Oracle XSQL

Combining SQL, Oracle Text, XSLT,
and Java to Publish Dynamic Web Content

Oracle XSQL 技术

全面展示如何利用 XSQL 结合
SQL、Oracle Text、XSLT 和 Java
的强大功能发布动态 Web 内容

(美) Michael D. Thomas 著
徐 焱 孙 扬 译



清华大学出版社

Oracle XSQL 技术

(美) Michael D.Thomas 著

徐焱 孙扬 译



清华大学出版社

北 京

内 容 简 介

XSQL 是目前最流行的一种可扩展的结构化查询语言。本书从一个全新的角度全面展示了如何利用 XSQL 结合 SQL、Oracle Text、Oracle RDBMS、Java、XML 和 XSLT 等技术的强大功能发布动态 Web 内容。主要内容包 括：讲解如何使用 XSQL、其他相关技术和 Oracle 数据库数据构建 XML Web 应用程序和 Web 服务；提供有关如何使用定制的行为处理程序、串行器和 XSQL API 增强 XSQL 的各种实用技巧；显示如何使用 XSLT 编写基于数据库数据的脚本；解释如何使用串行器随意生成图像和 PDF 文档等。

本书适合对 Oracle、SQL、XML 和 Java 有所了解并希望通过 XSQL 权衡利用多种 Web 技术进行 Web 动态内容开发的程序员和开发人员。

Michael D. Thomas

Oracle XSQL: Combining SQL, Oracle Text, XSLT, and Java to Publish Dynamic Web Content

EISBN: 0-471-27120-9

Copyright© 2003 by John Wiley & Sons, Inc.

Authorized translation from the English language edition published by John Wiley & Sons, Inc.

All rights reserved.

Chinese simplified language edition published by Tsinghua University Press.

本书中文简体字版由 John Wiley & Sons, Inc. 出版公司授权清华大学出版社出版。未经出版者书面许可，不得以任何方式复制或抄袭本书内容。

版权所有，翻印必究。

本书封面贴有清华大学出版社激光防伪标签，无标签者不得销售。

北京市版权局著作权合同登记号 图字：01-2003-6394

图书在版编目(CIP)数据

Oracle XSQL 技术/(美)托马斯(Thomas, M.D.)著；徐焱，孙扬译。—北京：清华大学出版社，2004

书名原文：Oracle XSQL: Combining SQL, Oracle Text, XSLT, and Java to Publish Dynamic Web Content

ISBN 7-302-08427-0

I.O… II.①托…②徐…③孙… III.关系数据库—数据库管理系统，Oracle IV.TP311.138

中国版本图书馆 CIP 数据核字(2004)第 028020 号

出 版 者：清华大学出版社 地 址：北京清华大学学研大厦、
http://www.tup.com.cn 邮 编：100084
社 总 机：010-62770175 客 户 服 务：010-62776969

组稿编辑：曹 康

文稿编辑：李 阳

封面设计：康 博

版式设计：康 博

印 刷 者：北京市清华园胶印厂

装 订 者：三河市化甲屯小学装订二厂

发 行 者：新华书店总店北京发行所

开 本：185×260 印张：28.75 字数：736 千字

版 次：2004 年 5 月第 1 版 2004 年 5 月第 1 次印刷

书 号：ISBN 7-302-08427-0/TP·6060

印 数：1~4000

定 价：58.00 元

本书如存在文字不清、漏印以及缺页、倒页、脱页等印装质量问题，请与清华大学出版社出版部联系调换。联系电话：(010)62770175-3103 或(010)62795704

目 录

第 1 章	Oracle XSQL 简介	1
1.1	XSQL 的功能	1
1.1.1	Web 应用程序开发的问题	2
1.1.2	作为关键技术的 XSQL	4
1.2	XSQL 和其他 Oracle 技术	9
1.2.1	Oracle Text	10
1.2.2	XML 支持	10
1.2.3	Oracle JDeveloper	10
1.3	XML 介绍	11
1.3.1	XML 的角色	11
1.3.2	格式规范的文档与有效文档	12
1.3.3	文档结构	13
1.3.4	处理指令	14
1.3.5	属性	14
1.3.6	语法本质	15
1.3.7	命名空间	17
1.3.8	模式	17
1.4	小结	18
第 2 章	开始学习 XSQL	19
2.1	基本安装	19
2.1.1	安装剖析	20
2.1.2	从头开始安装 XSQL	21
2.1.3	配置 Java	23
2.2	在现有环境下进行安装	24
2.2.1	Web 和应用程序服务器集成	24
2.2.2	配置数据库	26
2.2.3	配置数据库连接	26
2.3	小结	26
第 3 章	认识 XSQL	27
3.1	加载 XSQL 示例	27
3.2	创建一个示例用户和表	28
3.2.1	设置连接定义	28

3.2.2	第 1 个 XSQL 页面	29
3.3	第 1 个 XSLT 样式表	30
3.3.1	XSQLConfig.xml	31
3.3.2	XSQLConfig 元素	31
3.3.3	servlet 元素	31
3.3.4	processor 元素	32
3.3.5	http 元素	34
3.3.6	connectiondefs 元素	34
3.3.7	actiondefs 元素	34
3.3.8	serializerdefs 元素	34
3.4	安全问题	35
3.4.1	已知问题	35
3.4.2	考虑 XSQL 安全	38
3.5	小结	40
第 4 章	XSQL 体系结构	41
4.1	简介	41
4.2	Java 部分	42
4.2.1	JVM	42
4.2.2	驱动器	43
4.2.3	servlet 引擎	43
4.2.4	JSP 运行时	44
4.3	XSQL 外观	45
4.3.1	XSQL 命令行工具	45
4.3.2	XSQLRequest 类	46
4.3.3	XSQL servlet	46
4.4	XSQL 页面处理程序	47
4.5	Oracle XML 模块	49
4.5.1	XML 分析器	49
4.5.2	XML SQL	50
4.5.3	XSLT 处理程序	50
4.6	核心 XSQL 组件	50
4.6.1	XSQL 页面	51
4.6.2	行为	51
4.6.3	行为处理程序	51
4.6.4	串行器	52
4.7	小结	53

第 5 章 编写 XSQL 页面	54
5.1 查询数据库	54
5.1.1 多个 xsql:query 行为	56
5.1.2 xsql:query 细节	58
5.1.3 xsql:no-rows-query	59
5.1.4 规范模式	60
5.1.5 格式化日期	62
5.2 其他内置行为	63
5.2.1 xsql:dml	64
5.2.2 xsql:ref-cursor-function	65
5.2.3 xsql:include-owa	65
5.2.4 xsql:include-request-params	66
5.2.5 xsql:include-param	66
5.2.6 xsql:include-xml	66
5.2.7 xsql:set-page-param	67
5.2.8 xsql:set-session-param	67
5.2.9 xsql:set-cookie	68
5.2.10 xsql:set-stylesheet-param	69
5.2.11 xsql:action	69
5.2.12 xsql:include-xsql	69
5.2.13 xsql:insert-request	70
5.2.14 xsql:update-request	71
5.2.15 xsql:delete-request	71
5.2.16 xsql:insert-param	72
5.3 链接到 XSLT 样式表	72
5.3.1 处理指令细节	73
5.3.2 基于客户端类型选择样式表	74
5.4 小结	74
第 6 章 XSQL 参数	75
6.1 引用参数	75
6.2 参数的 XML 表示	76
6.3 参数类型	79
6.3.1 请求参数	79
6.3.2 页面私有参数	80
6.3.3 会话参数	81
6.3.4 cookie	82
6.4 设置默认参数	84

6.5	使用绑定变量	85
6.6	解决参数名冲突问题	85
6.7	小结	88
第 7 章	用 XSQL 进行数据库修改	89
7.1	输入 HTML 窗体	89
7.1.1	窗体参数和规范模式	89
7.1.2	使用 xsql:insert-request	92
7.1.3	使用 xsql:update-request	93
7.1.4	使用 xsql:delete-request	94
7.2	用 xsql:dml 修改	95
7.3	处理 XML 文档	96
7.3.1	处理提交的 XML	96
7.3.2	处理 XML 参数	99
7.3.3	作为 XML 插入 XML	100
7.4	小结	101
第 8 章	Oracle SQL	102
8.1	SQL 语法	102
8.1.1	区分大小写	102
8.1.2	词法约定	102
8.1.3	数据类型	103
8.1.4	运算符	106
8.1.5	SQL 表达式	106
8.2	描述表	107
8.3	SELECT 查询	107
8.3.1	目标子句	108
8.3.2	元素子句	110
8.3.3	WHERE 子句	115
8.3.4	ORDER BY 子句	122
8.3.5	进行集合操作	123
8.3.6	假想的双重表	124
8.4	管理表	125
8.4.1	创建表	125
8.4.2	更改表	128
8.4.3	删除表	129
8.5	添加和修改数据	130
8.5.1	事务	130
8.5.2	序列	131

8.5.3	INSERT 语句	132
8.5.4	UPDATE 语句	133
8.5.5	DELETE 和 TRUNCATE 语句	133
8.6	视图	134
8.6.1	创建和更改视图	134
8.6.2	使用视图	135
8.7	约束	136
8.7.1	约束类型	136
8.7.2	使用约束	137
8.8	格式化日期	139
8.8.1	XSQL 日期与 Oracle 日期的比较	139
8.8.2	日期格式元素	140
8.9	SQL 函数	142
8.9.1	使用聚集函数	142
8.9.2	一般意义上的数值函数	149
8.9.3	对数和三角函数	153
8.9.4	字符函数	154
8.9.5	日期函数	163
8.9.6	转换函数	170
8.9.7	混杂函数	173
8.10	小结	173
第 9 章	PL/SQL	174
9.1	初识 PL/SQL	174
9.2	结构	176
9.3	声明变量	177
9.3.1	标量变量声明	177
9.3.2	记录声明	178
9.3.3	光标	180
9.3.4	数组结构	180
9.4	PL/SQL 块和执行块	181
9.5	PL/SQL 中的 SQL 语句	182
9.6	控制结构	183
9.6.1	条件控制结构	184
9.6.2	循环控制结构	185
9.6.3	顺序控制结构	187
9.7	光标	187
9.8	包	190

9.9	过程和函数	191
9.10	异常	193
9.11	触发器	195
9.12	引用光标	195
9.13	PL/SQL 和 XSQL	196
9.14	小结	197
第 10 章	使用 Oracle Text	198
10.1	简单的关键字搜索	198
10.2	词根、模糊、通配符和探测法搜索	199
10.3	在 XML 文档内搜索	200
10.4	其他特性	202
10.5	小结	202
第 11 章	检索 XML	203
11.1	使用 XSQL 检索 XML 十分困难	203
11.2	使用 xsql:include-owa 创建 XML	204
11.2.1	从 PL/SQL 生成 XML	206
11.2.2	XMLType	208
11.3	小结	211
第 12 章	XSLT	212
12.1	开始学习 XSLT	212
12.1.1	XSLT 的作用	212
12.1.2	推模型和拉模型	213
12.2	XHTML 和样式表的创建	215
12.2.1	定义 XHTML	215
12.2.2	简单的 XHTML 转换	218
12.2.3	迁移 HTML 的技巧和窍门	221
12.3	小结	223
第 13 章	深入 XSLT	224
13.1	根元素	224
13.1.1	xsl:stylesheet 语法	224
13.1.2	示例	226
13.2	控制输出	226
13.2.1	xsl:output 语法	226
13.2.2	示例	227
13.3	模板	228
13.3.1	xsl:template 语法	228

13.3.2	xsl:apply-templates 语法	230
13.3.3	xsl:call-template 语法	231
13.3.4	示例	232
13.4	取值选择	235
13.4.1	xsl:value-of 语法	235
13.4.2	示例	236
13.5	迭代	238
13.5.1	xsl:for-each 语法	238
13.5.2	示例	239
13.6	条件逻辑	241
13.6.1	boolean 表达式	241
13.6.2	xsl:if 语法	241
13.6.3	xsl:choose、xsl:when 和 xsl:otherwise 语法	242
13.6.4	示例	243
13.7	处理 XML 实体和文本	246
13.7.1	xsl:element、xsl:attribute 和 xsl:attribute-set 语法	247
13.7.2	xsl:attribute 语法	248
13.7.3	xsl:attribute-set 语法	249
13.7.4	元素创建示例	249
13.7.5	xsl:text 语法	251
13.7.6	xsl:comment 语法	252
13.7.7	xsl:copy 语法	253
13.7.8	xsl:copy-of 语法	254
13.7.9	xsl:namespace-alias 语法	255
13.7.10	xsl:processing-instruction 语法	256
13.8	对元素编号	257
13.8.1	xsl:number 语法	257
13.8.2	xsl:decimal-format 语法	258
13.9	变量和参数	259
13.9.1	xsl:variable 语法	259
13.9.2	变量示例	260
13.9.3	xsl:param 语法	263
13.9.4	xsl:with-param 语法	264
13.9.5	参数示例	265
13.10	重用样式表	267
13.10.1	xsl:include 和 xsl:import 语法	267
13.10.2	xsl:apply-imports 语法	267
13.11	排序	268

13.11.1	xsl:sort 语法	268
13.11.2	示例	268
13.12	空白处理	269
13.13	混杂元素	270
13.13.1	xsl:key 语法	270
13.13.2	xsl:message 语法	270
13.13.3	xsl:fallback 语法	270
13.14	XPath	271
13.14.1	表达式和数据类型	271
13.14.2	位置路径	272
13.14.3	轴线	273
13.15	XPath 和 XSLT 函数	277
13.15.1	XSLT 特定函数	278
13.15.2	节点设置函数	280
13.15.3	字符串函数	282
13.15.4	布尔函数	285
13.15.5	数值函数	286
13.16	小结	287
第 14 章	构建 XSQL Web 应用程序	288
14.1	应用程序体系结构	288
14.1.1	简单的 XSQL 体系结构	289
14.1.2	开发用户体系结构的过程	290
14.1.3	扩展选项	294
14.1.4	使用 JavaScript 和 CSS 扩展接口	296
14.1.5	数据库层次上的扩展	297
14.2	示例应用程序	298
14.2.1	需求	298
14.2.2	应用程序接口设计	298
14.2.3	数据库需求	302
14.2.4	数据库设计	304
14.2.5	数据库实现	306
14.2.6	编写 SQL	307
14.3	集成 XSQL 和 XSLT	311
14.3.1	制作 XSQL 页面	312
14.3.2	编写样式表	316
14.4	传递参数	329
14.4.1	使用 SQL 传递	329

14.4.2	设置样式表参数	330
14.4.3	使用其他 XSQL 参数	331
14.5	无状态分页	334
14.5.1	分页的难点	334
14.5.2	纯粹的 XSQL 无状态分页	335
14.6	XSQL 数据编辑器	340
14.6.1	编辑器体系结构	340
14.6.2	XSQL 开发	341
14.6.3	开发 XSQL 接口	342
14.7	JavaScript 与 XSQL 开发	346
14.7.1	JavaScript 与 XSLT 集成	346
14.7.2	JavaScript 与 XSQL 数据	347
14.8	错误处理	349
14.8.1	XSQL 错误	349
14.8.2	XSQL 错误模板	350
14.8.3	处理错误的方式	351
14.9	小结	353
第 15 章	命令行工具	354
15.1	使用命令行工具	354
15.2	文本示例	355
15.3	编写脚本	356
15.4	创建静态 Web 页面	358
15.5	小结	359
第 16 章	使用 XSQL 的 Web 服务	360
16.1	体系结构	360
16.2	简单的 Web 服务客户	364
16.3	XML 到 XML 的转换	369
16.4	XML 处理行为	374
16.5	小结	375
第 17 章	Web 浏览功能外的 XSQL	376
17.1	示例程序	376
17.2	XSQL 类指南	379
17.3	在 servlet 内使用 XSQL	380
17.4	XML 分析: DOM 和 SAX 的比较	381
17.5	Oracle DOM API	381
17.6	小结	382

第 18 章 自定义行为处理程序	383
18.1 开始	383
18.1.1 行为处理程序简介	383
18.1.2 把 XML 添加到数据报	385
18.1.3 比较行为处理程序和 servlet 开发	387
18.2 行为处理程序 API	388
18.2.1 XSQLActionHandler 接口	388
18.2.2 XSQLActionHandlerImpl 基类	390
18.2.3 XSQLPageRequest	392
18.2.4 访问 servlet 功能	393
18.3 数据库交互	395
18.3.1 使用内置的行为处理程序	395
18.3.2 JDBC 连接	401
18.3.3 使用 XSU 类	402
18.3.4 添加 XMLType 对象	403
18.4 参数与输入	404
18.4.1 访问 XSQL 数据	405
18.4.2 替换参数值	407
18.4.3 设置页面参数	409
18.4.4 内部行为处理程序通信	412
18.5 小结	414
第 19 章 串行器	415
19.1 串行器简介	415
19.1.1 串行器的作用	415
19.1.2 使用串行器	416
19.2 使用 Apache FOP 创建 PDF	417
19.2.1 FOP 体系结构	417
19.2.2 使用 XSQL FOP 串行器	418
19.3 创建自定义的串行器	422
19.3.1 文本串行器	423
19.3.2 二进制串行器	426
19.3.3 串行化 BLOB	430
19.3.4 使用 Java AWT 创建 JPEG	433
19.4 小结	441
附录 A 资源	442
附录 B 相关标准	444

第1章 Oracle XSQL简介

欢迎进入可扩展结构化查询语言(XSQL)开发这个令人激动的领域!为什么该领域如此让人激动呢?因为它效率和高且容易使用。XSQL不是一种令用户叫绝、眼花的技术,也不是人们一直谈论不休的最新X标准,问题“但它现在能为我们做些什么呢?”的焦点是没有一种技术能完全实现它的功能。Web服务器不会保存所有的公司库存数据,数据库本身也不会以一种吸引人的、可用的方式将数据呈现给用户。这样就诞生了XSQL。XSQL使您可以很容易地在目前业界中应用的大多数健壮、成熟、可用的技术之间找到平衡点,这些技术包括:标准查询语言(SQL)、超文本标记语言(HTML)、超文本传输协议(HTTP)、可扩展标记语言(XML)、Java和Oracle关系数据库管理系统(RDBMS)。

每种技术在其领域内都被论证为最佳技术。用于查询关系数据时,SQL绝对占据主导地位。HTML和HTTP是Internet领域中的双胞胎。它们都有缺陷,但它们同时也在发展。Java在企业应用程序中取得了空前的成功,并将继续保持这种势头。XML是一种以平台和应用程序独立的方式结构化数据的标准。不久前,Oracle的RDBMS在其领域内成为领先技术,并占有了市场。

在本书中,XSQL使您可以把这些强大的语言结合到一起。您将学习如何使用XSQL在Web上实时呈现您的数据库数据;只用XSQL和(可扩展样式表语言转换)就可以开发出完整的应用程序;会看到如何使用XSQL随意创建图形,并表示基于动态数据的PDF(可移植文档格式)文档。这其中最简单的情况甚至不需要进行任何Java编码。您只需对与数据库进行复杂交互的部分使用Java编码即可。

本章是XSQL的总体介绍,也是对一些基础技术的介绍。所涉及的第1个主题是XSQL的功能简介,其中包含一些简单代码示例,这些示例阐述了XSQL功能的强大性。接着讨论XSQL和其他Oracle技术的关系。您不必专门为Oracle才使用XSQL,其实XSQL和Oracle可以结合得很好。本章结尾部分介绍XML的深层开发。XSQL和XSLT均派生于XML,因此在深入学习前需要充分理解XML。

1.1 XSQL的功能

在学习任何新技术前,全面了解该技术可以解决哪些问题是值得的。前面,指出XSQL可以把各种技术结合到一起。在本节中,您会从更高层次上看到XSQL如何实现这种结合。然而最重要的是明白为什么在开始时各种技术的集成是一个问题。

首先要理解Web应用程序开发的关键问题是什么。Web和企业应用程序的结合就像任何结合一样存在问题。可以看到,XSQL极大简化了这种关系的关键部分:数据库访问。这就引起了下一个讨论:数据库如何满足现代Web应用程序开发?通过本书会明白,Oracle数据库不仅可以存储关系型信息,也可以存储XML信息。XML可依次对Web应用程序开发的基础性问

题提供许多解决方案，因为 XSQL 基本上是基于 XML 的，因此理解 XML 如何提供这些方案很重要。伴随着 XSQL 的技术几乎总是 XSLT。XSLT 使您几乎可以把 XML 转换为任何东西。在这个技术繁杂的世界里，从一种数据格式向另一种数据格式进行转换的问题时常出现。XSLT 解决了这个问题，本节将对其具体内容进行介绍。本节最后将说明 XSQL 如何在所有这些技术间架起桥梁。

1.1.1 Web 应用程序开发的问题

本章介绍的都是观点，历史是提出优秀观点的最好见证。因此要开始讨论 Web 应用程序开发的当前状态，首先要考虑 Web 本身的历史。开始时——回溯到 1990 年——只有 HTML、URL (统一资源定位器)和 HTTP。该系统的真正诱人之处是超链接。Berners-Lee 认为可以把文档链接到一起——并以此方式——跨越网络！他在 1990 年的论文“信息管理，一个提议！”中清楚地说明了这一点：

图形，然后是对此文档的引用，以及所有与被引用文档的网络地址相关的内容，这样当阅读此文档时就可以通过单击跳到其他文档中。

从此概念中，我们现在所知道的超链接就诞生了。现在，Tim Berners-Lee 虽然不是第 1 个超链接的构想者，但他正确实现了自己的系统并使其足够简单，因此获得了广泛传播。他也得到了其他一些因素的协助。首先，他赶上了合适的时机，当时 Internet 的底层协议，传输控制协议/Internet 协议(TCP/IP)正式形成并已广泛使用。第二，他是在学术环境内发明此系统，这是许多 Internet 标准的共同特性。在学术环境比在商业范围更容易获得协议集的自由共享。在 20 世纪 90 年代早期，对许多软件公司来说，免费发布复杂的软件和设计还是闻所未闻的事情。

然而，Web 增长的关键原因是它开始就是一个很简单的系统。HTTP 开始于一个特别简单的协议，并且直到今天也大致保持这种情况。您发送请求并接收响应，指出如何响应是服务器的责任，对响应作出反应则是客户端的责任。其中只有 3 种可能的方法：GET、POST 和极少用到的 HEAD。这使得开发 Web 服务器很容易实现。客户端只需要理解一些可能的响应即可。同样，HTML 的设计思想也很简单。代替使用功能更强大但相对更复杂的 SGML，Berners-Lee 只选择了其中一个子集。实际上开发服务器和客户端十分容易，以至于大多数 Web 早期开发由开发人员和学生在业余时间或由致力于某一特定项目的专业人员就可以完成。需要注意的是两个机构对 Web 的诞生起了重要作用——欧洲粒子物理研究所(CERN)和位于美国 Urbana-Champaign 的 Illinois 大学的超级计算应用程序国家中心——这两个机构在实验室发明 Web 时并没有进行针对网络计算的研究。

正因为简单，Web 才能像野火一样广泛传播。它传播如此迅速，以至于一半以上的美国人都访问过 Web。大多数在办公桌上有一台计算机的人都会访问 Web。这种普及性使其成为应用程序开发的理想平台。不必担心在每个人的桌面上安装一个软件应用程序，或需要客户在家里安装软件的问题，也不必担心平台差异性问题的。仅从逻辑观点看，Web 是许多应用程序首选的平台。

但存在一个小问题：Web 最初的设计意图只是为了共享简单的文档！这就引出了一些问题，其中最明显的就是 Web 的不稳定性。HTTP 最初不支持 cookie，而 cookie 使您可以把不同的 HTTP 事务绑定到一起放入用户会话中。只共享静态文档时，则不必把不同的 HTTP 事务绑定到一起。文档总会保持一致，所以用户以前请求过的文档是什么就不重要了。但是当数据为动

态时，它经常会涉及当前请求内容前所请求的内容。购物卡应用程序就是一个很好的例子。当用户准备购买某个商品时，Web 应用程序必须通过几个 HTTP 事务跟踪被选中的商品。

除了 cookie 外，还有一些技术可以解决此问题。XSQL 完全支持 cookie 和 servlet 会话。在本书后面的内容中会介绍这些机制。尤其需要指明的是：支持会话的机制在最初设计后加入到源 HTTP 中的，就像 JavaScript 和连接数据库的概念被加入到 Web 一样。也许最重要的是，HTML 文档用于传送信息还不够充分。HTML 只给出了 Web 浏览器如何显示信息的结构。这是很重要的功能，但也意味着不能按语意解释 HTML。由此诞生了 XML。

Web 原本可以设计的更好，但事后聪明总是没有用的。实际上，Web 是非常优秀的，因为它很容易扩展。虽然其最初是为静态文档而设计的，但添加对图形和动态数据的支持十分容易。Web 服务器并不关心其发送或接受信息的种类。HTTP 只描述以一种简单和轻便的方式转换信息的机制。如果 HTML 包含 JavaScript，那么就更好了——理解使用信息的方式是浏览器的责任。

同样，创建数据库驱动的 Web 页面只是服务器的责任。Web 浏览器不知道与数据库交互的前期内容。严格地说，HTTP 服务器进程也不知道。它只知道把某些 URL 传递到 servlet 和其他与数据库交互并产生动态结果的服务器端模块。

这种发展一直持续到目前 Web 服务的出现。HTTP 十分简单，所以您可以很容易地在自己的代码中嵌入简单的 Web 客户端代码。然后就可以从远程计算机上获得数据并在自己的程序中使用它们。HTTP 不关心发送的内容，因此可以使用 XML 来结构化数据。在其接收的数据方面，HTTP 也是很松散的，因此可以把数据块发回到服务器。这样，最初设计用来简化共享文档的协议就可以被用作强大的分布式应用程序的基础。

开发 Web 应用程序的过程正日益变得成熟。早期的 Web 应用程序开发人员在遇到各种问题时必须自己制定解决方案。其实这样的方案有很多。最好的方案得到了认可并被广泛采纳。Java 和 Java 数据库连接(JDBC)就是很好的例子，XML 和 XSLT 也是如此。

XSQL 架构也是 Web 开发中的另一发展产物。有了 XSQL，产生和数据库交互的动态 Web 页面几乎就像编写 HTML 页面一样简单。而在此之前，您还必须用 Java、Perl 或 C++等语言编写一个模块来管理数据库连接，执行数据库的 SQL 查询，然后才能处理结果。这还只是开始，从中还必须指出对结果的操作。因为结果的数目和类型可能对同一模块发生很大变化，您不得不处理诸如在一个页面上放多少结果和如何对不同类型进行格式化的问题。此模型，常被称为三层模型，如图 1-1 所示。

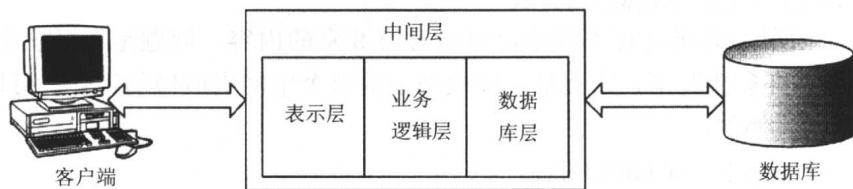


图 1-1 三层模型

前面已经讨论过，用户接口(UI)层只知道如何向用户表示数据。数据库储存数据，通常具有超出任何特定应用程序的各种目的。这把许多工作都留给了中间层。许多体系结构都喜欢引用中间层，使之包含业务逻辑。这种委婉的说法暗示出中间层只是一些绝对简单的集合，规定诸如“填装货舱到容量的 80%”和“对海运提供 10%的折扣”等规则。客户端关注所有的 UI

部分，而数据库则执行管理数据的艰苦工作。

当回过头来检查中间层时，它通常不像是一个画在白板上的素描。您会在中间层内发现许多硬编码的 SQL 和 UI 代码。虽然许多应用程序开发组尽量把表示层和数据库层分隔开，但是很难定义其严格的界限。即使使用 Java Servlet Page(JSP)等脚本语言，代码片段也通常对系统有较深的依赖性。如果 UI 设计者决定要在返回的数据集中有轻微的差别，或者希望它们以不同的方式排序，那么该怎么办呢？您必须找出 SQL 语句并进行修改。这可能会在系统的其他地方反应出来。您可能必须创建一个新类。因此，要进行一个相对简单的 UI 改动，必须对数据库层进行改动。当必须扩展系统时，可能就会发现您的 UI 代码只有很少一部分可重用。

现在暂时假定，一个特定的应用程序实现了各层之间较好的分离。每个人在开始编码前都要阅读所有的设计模式书籍并在白板上花费大量的时间。他们要有足够的时间进行正确的分离，管理者或客户必须理解这一点的重要性。还有一个问题：系统的输出是 HTML。如果您要使数据对 Web 服务器或无线方式等新平台可用，那么该怎么办呢？必须把 UI 层导向一个全新的用户接口类型。因为代码是只使用 HTML 思维方式编写的，可能必须重写所有的接口部件。如果系统设计的很完美，也需要做大量的工作。现在假定系统设计的并不完美。

Web 曾经是最大的非主流应用程序开发平台。它开始于瑞典一个学术机构的内部项目，并逐渐发展为当今最强大的技术之一。它有自身的错综复杂性，但其适应能力是无限的。Web 成功的关键是将其理解作为一种不断演化的技术。开发 Web 应用程序的艺术也在不断进化，一个成功的 Web 应用程序开发人员总是能预见到下一进化结果。下面我们开始介绍 XSQL 如何大大简化开发此平台的过程。

1.1.2 作为关键技术的 XSQL

XSQL 是一种关键技术，而不仅是基础技术。Oracle RDBMS 是基础技术的一个典型示例。许多公司的整个业务活动都建立在 Oracle 数据库之上。Java 和 HTTP 也是基础技术。然而，XSQL 更像是一个拱门的楔石——以一种简单和谐的方式把各种技术结合起来的块。

要明白其中的原因，研究在开发一个数据库驱动的 Web 应用程序时实际所需的东西。忘记所有曾听到过的三层或 n 层概念。实际上，在许多数据库驱动的 Web 页面中，数据库查询都被美化了。对这些 Web 页面，其需求其实很简单：

- 提取数据的 SQL
- 产生 HTML(用于表示信息)的方式

创建 SQL 很简单，如果您还不知道，可参考第 8 章的内容。问题是您必须从 SQL 结果集进入 HTML 标记。许多情况下，这就是三层模型中的整个中间层所做的工作。可以从一些分类的结果集开始介绍，例如：

```
>SELECT ename, job, sal FROM emp
      WHERE deptno=20
      ORDER BY sal;
```

ENAME	JOB	SAL
SMITH	CLERK	800
ADAMS	CLERK	1100