



Oracle 技术系列丛书

ORACLE®

AUTHORIZED ORACLE PRESS™—EXCLUSIVELY FROM OSBORNE

Oracle9i JDBC 程序设计

Oracle9i JDBC Programming

(美) Jason Price 著 马朝晖 等译



OFFICIAL • AUTHORIZED

Oracle Press

ONLY FROM OSBORNE



机械工业出版社
China Machine Press



Education

Oracle 技术系列丛书

Oracle9 / JDBC 程序设计

(美) Jason Price 著

马朝晖 等译



机械工业出版社
China Machine Press

本书介绍了如何使用 Java 数据库连接性 (Java DataBase Connectivity, JDBC) 开发访问 Oracle 8i 和 Oracle 9i 数据库程序。涉及了标准的 JDBC 和高性能 Oracle 扩展, Oracle9iAS Containers for J2EE (OC4J)、JDeveloper、性能调整, 以及如何开发 J2EE 组件和 Java 存储过程等。本书含有丰富的示例代码, 对 Java 程序员开发数据库应用程序极有帮助。

Jason Price: Oracle9i JDBC Programming (ISBN 0-07-222254-9).

Copyright © 2002 by The McGraw-Hill Companies, Inc.

Original language published by The McGraw-Hill Companies, Inc. All rights reserved. No part of this publication may be reproduced or distributed in any means, or stored in a database or retrieval system, without the prior written permission of the publisher.

Simplified Chinese translation edition jointly published by McGraw-Hill Education (Asia) Co. and China Machine Press.

本书中文简体字版由美国麦格劳-希尔教育出版公司授权机械工业出版社出版, 未经出版者预先书面许可, 不得以任何方式复制或抄袭本书的任何部分。

本书封面贴有 McGraw-Hill 公司防伪标签, 无标签者不得销售。

版权所有, 侵权必究。

本书版权登记号: 图字: 01-2002-3604

图书在版编目 (CIP) 数据

Oracle9i JDBC 程序设计/ (美) 普莱斯 (Price, J.) 著; 马朝晖等译. - 北京: 机械工业出版社, 2002.10

(Oracle 技术系列丛书)

书名原文: Oracle9i JDBC Programming

ISBN 7-111-11035-8

I . O… II . ①普… ②马… III . 关系数据库 - 数据库管理系统, Oracle9i - 程序设计
IV . TP311.138

中国版本图书馆 CIP 数据核字 (2002) 第 077089 号

机械工业出版社 (北京市西城区百万庄大街 22 号 邮政编码 100037)

责任编辑: 张金梅

北京忠信诚胶印厂印刷·新华书店北京发行所发行

2003 年 1 月第 1 版第 1 次印刷

787mm × 1092mm 1/16 · 28.25 印张

印数: 0 001-4 000 册

定价: 45.00 元

凡购本书, 如有倒页、脱页、缺页, 由本社发行部调换

前 言

Java 已经在企业开发方面取得了相当的地位。可以在三层企业系统的中间层和后端运行 Java，也可以在浏览器中运行 Java。Oracle 数据库是应用最广泛的数据库之一，而且 Oracle9i Application Server (Oracle9iAS) 正在占据越来越多的应用服务器市场份额。Java 数据库连接性 (Java Database Connectivity, JDBC) API 可以用 Java 程序访问数据库，而 JDBC 是大多数主要数据库厂商 (包括 Oracle 公司) 支持的一种标准。

本书将涉及以下内容：

- 如何使用标准的 JDBC 以及 Oracle 公司提供的扩展 (用于使用 Oracle 数据库特有的功能)。
- 如何使用 Oracle9iAS Containers for J2EE (OC4J)。OC4J 能够运行 Enterprise JavaBeans (EJB)、servlet 和 JavaServer Pages (JSP) 等组件。它是 Oracle9i Application Server (Oracle9iAS) 的核心。
- 如何使用 Oracle JDeveloper。JDeveloper 是用于开发和调试 Java 程序的图形化集成开发环境。
- 如何开发 Java 存储过程，这可以取代使用 Oracle 专有的 PL/SQL 过程语言编写业务逻辑。
- 如何使用关系型数据库以及如何使用 Oracle 数据库的对象-关系特性。

如果你不了解 Oracle 数据库，也可以从本书中学到 Oracle 数据库的基础知识。你将看到大量用于讲解 JDBC 的编程示例，还有一些创建本书中使用的各个数据库模式的脚本。这些程序和数据库脚本已经针对 Oracle8i 数据库 (8.1.6 和 8.1.7) 和 Oracle9i 数据库 (9.0.1) 测试过。还有两个脚本演示了最新的一些 Oracle9i 数据库特性，它们只能针对 Oracle9i 数据库运行。

第一部分 “基本的 JDBC 编程”

第 1 章 “JDBC 简介”。介绍在开发和运行包含 JDBC 语句的 Java 程序之前需要安装和配置的软件。其中包含一个使用 JDBC 执行简单的数据库操作的示例程序，并且将编译和运行这个示例程序。另外，我将介绍 Oracle 的 JDeveloper 工具，用这个图形化的集成开发环境可以开发、编译、运行和调试 Java 程序。

第 2 章 “数据库与 Oracle”。研究数据库与 Oracle。讲解用于访问数据库的结构化查询语言 (Structured Query Language, SQL)。该章建立一个用于存储在线商店数据的简单数据库。该章还介绍 Oracle 专有的过程式编程语言，PL/SQL。这种语言围绕 SQL 添加了一个编程层，可以使用 PL/SQL 在数据库中定义过程和函数。

第 3 章 “JDBC 编程基础”。介绍用于连接 Oracle 数据库的各种 Oracle JDBC 驱动程序的细节，主要包括：导入 JDBC 包，注册 Oracle JDBC 驱动程序，打开数据库连接，以及执行 SQL 语句在数据库表中获取、添加、修改和删除行。

第二部分 “高级 JDBC 编程”

第 4 章 “高级结果集”。介绍如何在结果集中前后移动以及跳到任意行，还将学习结果集如何感知数据库中的修改。

第 5 章 “PL/SQL 和 JDBC”。介绍如何使用 JDBC 调用 PL/SQL 过程和函数。

第 6 章 “数据库对象”。Java 和 C++ 等面向对象编程语言允许定义类，这些类作为创建对象的“模板”使用。类定义属性和方法，属性用于存储对象的状态，方法用于模拟对象的行为。从 Oracle8i 数据库开始，已经可以在数据库中使用对象，而 Oracle9i 中的对象包含了更多的特性。在这一章中将学习如何创建和使用对象类型。

第 7 章 “集合”。Oracle8i 数据库引入了两个新的称为集合（collection）的数据库类型：varray 和嵌套表（nested table）。varray 与 Java 中的数组类似，可以使用 varray 在数据库中存储一组有序的元素。嵌套表是嵌套在另一个表中的表。该章介绍如何创建和使用集合类型。

第 8 章 “大对象”。在 Oracle8 和所有后续版本中，引入了一种称为大对象（large object, LOB）的新类型。LOB 可以用于存储二进制数据、字符数据和对外部文件的引用。LOB 最多可以存储 4GB 的数据，可以满足当今的多媒体密集型应用程序和网站的需求。该章介绍如何使用 LOB。

第 9 章 “高级事务控制”。当今的数据库可以处理许多个同时访问数据库的用户和程序，这些用户和程序可能在数据库中运行它们自己的事务。数据库软件必须能够满足所有并发事务的需要，并且维护数据库表中存储的行的完整性。可以使用 JDBC 控制事务与数据库中运行的其他事务之间的隔离级别，该章将介绍具体做法。

第三部分 “部署 Java”

第 10 章 “Java 存储过程和触发器”。Oracle9i 和 8i 数据库集成了一个称为 Oracle JVM 的 Java 虚拟机。Oracle JVM 可以在数据库中部署和运行 Java 程序，这些程序按照 Oracle 的官方说法称为 Java 存储过程（Java stored procedure）。Java 存储过程可以取代使用 PL/SQL 编写业务逻辑。触发器使你能够编写在发生某种事件时由数据库自动运行的代码。在这一章中将学习 Java 存储过程和触发器。

第 11 章 “Oracle9iAS Containers for J2EE (OC4J)”。可以使用 Oracle9iAS Containers for J2EE 运行 Java 2 Enterprise Edition (J2EE) 组件。OC4J 可以运行 J2EE 组件，如 Enterprise JavaBeans (EJB)、servlet 和 JavaServer Pages (JSP)。然后，可以在这些组件中使用 JDBC 语句访问数据库。这一章将介绍 OC4J，以及如何使用 OC4J 部署和运行 servlet、JSP 和 EJB。

第四部分 “性能”

第 12 章 “连接缓冲池和缓存”。介绍如何使用数据库连接缓冲池技术改进程序的性能。还将看到如何建立和使用缓存的数据库连接来进一步提高程序的性能。

第 13 章 “性能调整”。介绍标准的 JDBC 性能调整技术以及 Oracle 性能扩展。通常，Ora-

cle 性能扩展提供了比标准 JDBC 更好的性能。

“附录”

附录 A “Oracle 和 Java Type 映射”。包含三个表，其中记载了 Oracle 数据库类型与兼容的 Java 类型之间的映射。

附录 B “Oracle Java 工具参考”。记载了本书中使用的 Oracle Java 工具的语法和命令行选项。

附录 C “JDBC 接口和类精选参考”。提供了一部分 JDBC 接口和类的资料。

附录 D “JNDI 和数据源”。Java 命名和目录接口 (Java Naming and Directory Interface, JNDI) 是一个为 Java 程序提供命名和目录功能的 API。通过 JDBC 使用 JNDI 是非常有用的，因为这样可以注册（即绑定）数据源，然后在程序中查找这些数据源，而不必提供准确的数据库连接信息。这个附录将讲解如何对 JDBC 数据源使用 JNDI。

适合阅读本书的读者

如果你是个 Java 程序员，而且需要编写访问 Oracle 数据库的程序，那么需要阅读本书。如果你是个技术部门的经理，而且需要全面了解 JDBC，那么也会发现本书是极具价值的。本书要求读者具有基本的 Java 知识，但是不要求读者了解 Oracle 数据库，不了解数据库的读者可以在第 2 章中找到理解本书所需的数据库知识。当然，也不要求读者了解 JDBC，因为本书将帮助你成为 JDBC 专家，包括教你如何使用 Oracle JDBC 扩展。

如何获取本书中的示例

本书的所有代码、脚本和其他文件都可以从 Oracle 出版社的网站下载 (www.OraclePressBooks.com)。这些内容都包含在一个 ZIP 文件中。下载这个文件之后，使用 WinZip 打开它，并且从 Actions 菜单选择 Extract 选项。这将创建一个名为 JDBC 的目录，其中包含以下三个子目录：

- **programs** 包含程序。
- **sample_files** 包含第 8 章中使用的示例文件。
- **sql** 包含用于创建数据库模式的 SQL * Plus 脚本。

目 录

前言

第一部分 基本的 JDBC 编程

第 1 章 JDBC 简介	1
1.1 软件需求	2
1.2 配置	3
1.2.1 设置环境变量 ORACLE_HOME	4
1.2.2 设置环境变量 JAVA_HOME	5
1.2.3 设置环境变量 PATH	5
1.2.4 设置环境变量 CLASSPATH	6
1.2.5 在 UNIX 或 Linux 上设置环境 变量 LD_LIBRARY_PATH	6
1.3 你的第一个 JDBC 程序	7
1.3.1 示例程序: FirstExample.java	7
1.3.2 编译并且运行 FirstExample.java	11
1.4 Oracle JDeveloper	12
1.4.1 创建新的工作区和工程	12
1.4.2 将 FirstExample.java 添加进工程	15
1.4.3 将 Oracle JDBC 库添加进工程并且 编译和运行 FirstExample.java	15
1.4.4 调试 FirstExample.java	17
第 2 章 数据库与 Oracle	19
2.1 关系型数据库的概念	20
2.2 结构化查询语言	21
2.2.1 SQL * Plus	21
2.2.2 运行 store_user.sql 脚本	22
2.2.3 数据定义语言 (DDL) 语句	22
2.2.4 数据操纵语言 (DML) 语句	27
2.3 Oracle PL/SQL	46
2.3.1 块结构	47
2.3.2 变量和类型	48
2.3.3 条件逻辑	48
2.3.4 循环	49

2.3.5 游标	51
2.3.6 过程	54
2.3.7 函数	56
2.3.8 包	58
第 3 章 JDBC 编程基础	61
3.1 Oracle JDBC 驱动程序	62
3.1.1 Thin 驱动程序	62
3.1.2 OCI 驱动程序	62
3.1.3 服务器端内部驱动程序	63
3.1.4 服务器端 Thin 驱动程序	63
3.2 导入 JDBC 包	63
3.3 注册 Oracle JDBC 驱动程序	63
3.4 打开数据库连接	64
3.4.1 使用 DriverManager 类的 getConnection () 方法连接 数据库	64
3.4.2 使用 Oracle 数据源连接数据库	66
3.5 创建 JDBC Statement 对象	68
3.6 从数据库获取行	69
3.6.1 步骤 1: 创建和填充 ResultSet 对象	69
3.6.2 步骤 2: 从 ResultSet 对象读列值	70
3.6.3 步骤 3: 关闭 ResultSet 对象	72
3.7 向数据库中添加行	72
3.8 修改数据库中的行	73
3.9 从数据库中删除行	74
3.10 处理数值	74
3.11 处理数据库 null 值	75
3.12 控制数据库事务	77
3.13 执行数据定义语言语句	77
3.14 处理异常	78
3.15 关闭 JDBC 对象	79
3.16 示例程序: BasicExample1.java	80

3.17 准备好的 SQL 语句	86	6.3.4 从 products 表中删除行	143
3.18 Oracle JDBC 扩展	89	6.4 在 object_products 表上执行 DML	143
3.18.1 oracle.sql 包	89	6.4.1 在 object_products 表中插入行	143
3.18.2 oracle.jdbc 包	93	6.4.2 从 object_products 表中选择行	143
3.18.3 示例程序: BasicExample3.java	96	6.4.3 更新 object_products 表中的行	144
第二部分 高级 JDBC 编程		6.4.4 从 object_products 表中删除行	144
第 4 章 高级结果集	101	6.5 在 object_customers 表上执行 DML	144
4.1 可滚动结果集	102	6.5.1 在 object_customers 表中插入行	144
4.1.1 在可滚动结果集中移动	104	6.5.2 从 object_customers 表中选择行	145
4.1.2 在可滚动结果集中判断位置	105	6.6 在 purchases 表上执行 DML	145
4.1.3 示例程序: AdvResultSet		6.6.1 在 purchases 表中插入行	146
Example1.java	106	6.6.2 从 purchases 表选择行	146
4.2 可更新结果集	109	6.6.3 更新 purchases 表中的行	147
4.2.1 更新行	109	6.7 Oracle9i 数据库类型继承	147
4.2.2 删除行	111	6.8 使用弱类型的 Java 对象访问	
4.2.3 使用可更新结果集更新或		数据库对象	148
删除行时发生的冲突	111	6.8.1 使用 STRUCT 插入数据库对象	149
4.2.4 插入行	111	6.8.2 将数据库对象选择进 STRUCT	151
4.2.5 示例程序: AdvResultSet-		6.8.3 使用 STRUCT 更新数据库对象	153
Example2.java	112	6.8.4 删除对象	155
4.3 结果集可以“看见”哪些数据库修改	115	6.8.5 示例程序: ObjectExample1.java	155
4.3.1 refreshRow() 方法	116	6.8.6 弱类型对象引用	160
4.3.2 修改何时可见	117	6.8.7 示例程序: ObjectExample2.java	163
4.3.3 可滚动敏感结果集	117	6.9 强类型的接口和自定义类	169
4.3.4 示例程序: AdvResultSet-		6.10 使用 JPublisher 产生自定义类	169
Example4.java	118	6.10.1 从命令行运行 JPublisher	170
4.4 元数据	120	6.10.2 从 JDeveloper 运行 JPublisher	174
第 5 章 PL/SQL 和 JDBC	125	6.11 使用实现了 SQLData 接口的自定义类	175
5.1 调用 PL/SQL 过程	126	6.11.1 SQLData 实现的类型映射	175
5.2 调用 PL/SQL 函数	128	6.11.2 使用自定义 Java 对象插入	
5.3 示例程序: PLSQLExample1.java	130	数据库对象	176
5.4 使用 PL/SQL 包和 REF CURSOR	133	6.11.3 将数据库对象选择进自定义	
第 6 章 数据库对象	137	Java 对象	177
6.1 创建对象类型	138	6.11.4 使用自定义的 Java 对象更新	
6.2 使用对象类型定义列对象和对象表	140	数据库对象	177
6.3 在 products 表上执行 DML	141	6.11.5 示例程序: ObjectExample3.java	178
6.3.1 在 products 表中插入行	141	6.12 使用实现了 ORADData 接口的自定义类	183
6.3.2 从 products 表中选择行	142	6.12.1 使用 JPublisher 产生实现了	
6.3.3 更新 products 表中的行	142	ORADData 接口的自定义类	183
		6.12.2 使用自定义 Java 对象插入、	

选择和更新数据库对象	187	8.2.7 使用 get 方法读取 CLOB 和 BLOB 列	245
6.12.3 示例程序: ObjectExample4.java ...	188	8.2.8 使用流读取 CLOB 和 BLOB 列	247
6.12.4 强类型的对象引用	193	8.2.9 使用 BFILE 指针读取外部文件 ...	249
第 7 章 集合	199	8.2.10 示例文件: LobExample2.java ...	251
7.1 varray	200	8.3 LONG 和 LONG RAW 列	257
7.1.1 创建 varray 类型	200	8.3.1 写 LONG 和 LONG RAW 列	258
7.1.2 使用 varray 类型定义表中的列	201	8.3.2 示例程序: LongExample1.java ...	260
7.1.3 用元素填充 varray	201	8.3.3 读取 LONG 和 LONG RAW 列	262
7.1.4 选择 varray 元素	201	第 9 章 高级事务控制	267
7.1.5 修改 varray 元素	202	9.1 ACID 事务属性	268
7.2 嵌套表	202	9.2 事务隔离	268
7.2.1 创建嵌套表类型	202	9.3 使用 JDBC 的实际例子	269
7.2.2 使用嵌套表类型定义表中的列	202	9.4 示例程序: AdvTransExample1.java ...	272
7.2.3 用元素填充嵌套表	203	9.5 分布式事务	276
7.2.4 选择嵌套表元素	203		
7.2.5 修改嵌套表元素	203	第三部分 部署 Java	
7.3 Oracle9i 多级集合类型	204	第 10 章 Java 存储过程和触发器	281
7.4 使用弱类型对象访问集合	206	10.1 Oracle JVM 体系结构	282
7.4.1 使用 ARRAY 插入集合	206	10.2 Java 存储程序的特性	283
7.4.2 将集合选择进 ARRAY	208	10.2.1 使用默认的数据库连接	284
7.4.3 使用 ARRAY 更新集合	209	10.2.2 输出	284
7.4.4 示例程序: Collection- Example1.java	212	10.3 实际例子	286
7.5 使用强类型对象访问集合	216	10.4 从 PL/SQL 过程调用 Java 存储程序	300
7.5.1 使用 JPublisher 产生自定义类	217	10.5 从 Java 调用 Java 存储程序	301
7.5.2 使用自定义 Java 对象插入集合 ...	220	10.6 使用 JDeveloper 装载和公布 Java 存储程序	303
7.5.3 将集合选择进自定义 Java 对象 ...	221	10.7 数据库触发器	311
7.5.4 使用自定义 Java 对象更新集合 ...	222	第 11 章 Oracle9iAS Containers for J2EE (OC4J)	319
7.5.5 示例程序: Collection- Example3.java	224	11.1 OC4J 概述	320
第 8 章 大对象	229	11.2 安装 OC4J	321
8.1 示例文件	230	11.3 启动、重新启动和关闭 OC4J	321
8.2 大对象 (LOB)	231	11.3.1 启动 OC4J	321
8.2.1 示例表	231	11.3.2 重新启动 OC4J	321
8.2.2 put、get 和流方法	232	11.3.3 关闭 OC4J	322
8.2.3 使用 put 方法写 CLOB 和 BLOB 列 ...	233	11.4 定义数据源	322
8.2.4 使用流写 CLOB 和 BLOB 列	237	11.5 servlet	323
8.2.5 使用 BFILE 列存储外部文件 的指针	240	11.5.1 部署 servlet	326
8.2.6 示例程序: LobExample1.java	241	11.5.2 调用 servlet	327

11.6	JavaServer Pages (JSP)	327
11.7	Enterprise JavaBeans (EJB)	329
11.7.1	会话 bean 示例	330
11.7.2	在独立的 Java 程序中使用 bean	340
11.7.3	在 servlet 中使用 bean	343

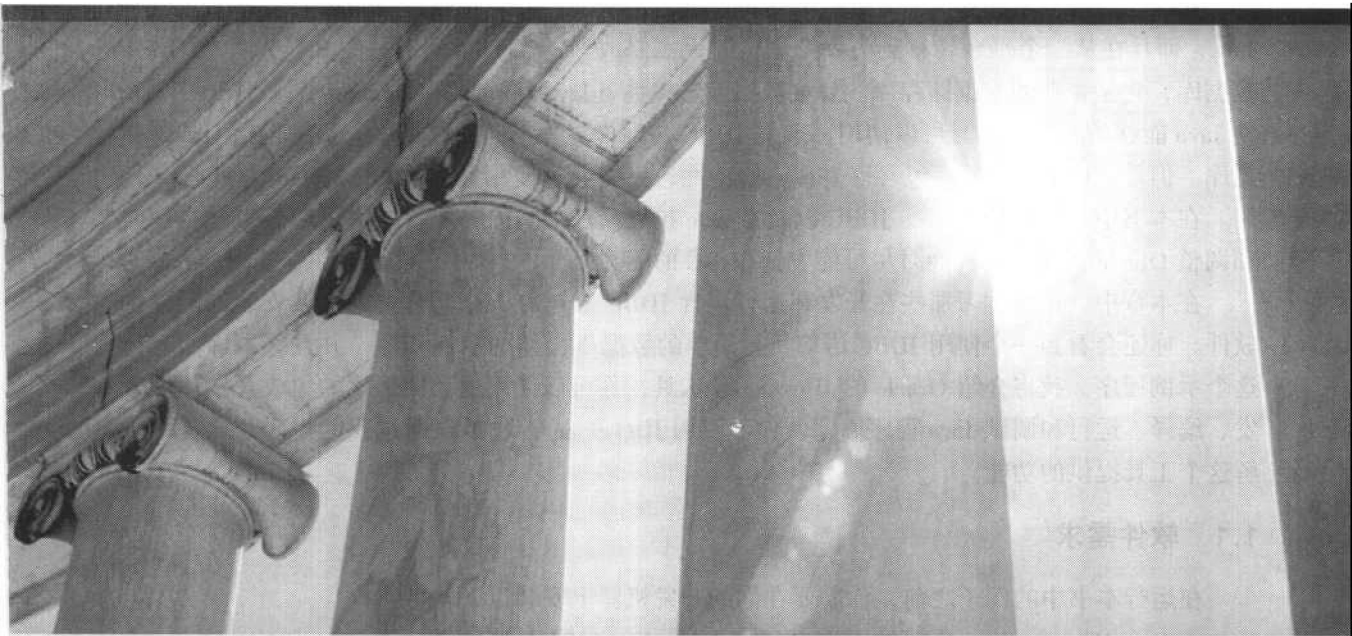
第四部分 性能

第 12 章	连接缓冲池和缓存	347
12.1	连接缓冲池和缓存包	348
12.2	使用 Thin 驱动程序实现连接缓冲池	348
12.3	连接缓存	353
12.3.1	创建连接缓存	354
12.3.2	请求、使用和关闭连接实例	354
12.3.3	关闭连接缓存	356
12.3.4	示例程序: ConnectionCache Example1.java	356
12.3.5	控制 PooledConnection 对象的 数量	359
12.3.6	示例程序: ConnectionCache Example2.java	360
12.4	使用 OCI 驱动程序实现连接缓冲池	363
12.4.1	创建 OCI 连接缓冲池	363
12.4.2	请求、使用和关闭连接实例	365
12.4.3	关闭 OCI 连接缓冲池	365
12.4.4	示例程序: ConnectionPool- Example2.java	365
第 13 章	性能调整	369
13.1	perf_test 表	370
13.2	禁止自动提交模式	370

13.3	批处理	373
13.3.1	标准更新批处理	373
13.3.2	Oracle 更新批处理	379
13.4	行预获取	384
13.4.1	标准行预获取	384
13.4.2	Oracle 行预获取	388
13.5	定义结果集列的类型	393
13.5.1	defineColumnType () 方法	393
13.5.2	定义列的长度	393
13.5.3	结构化的对象、对象引用和 数组列	394
13.5.4	示例程序: DefineColumnType Example.java	394
13.6	语句缓存	396
13.6.1	启用语句缓存和设置语句 缓存尺寸	396
13.6.2	语句缓存的类型	397
13.6.3	使用隐式语句缓存	397
13.6.4	使用显式语句缓存	402
13.7	调整 SQL 语句	407
13.7.1	使用 WHERE 子句限制行数	407
13.7.2	添加额外的索引	407
13.8	结束语	407

附 录

附录 A	Oracle 和 Java Type 映射	409
附录 B	Oracle Java 工具参考	413
附录 C	JDBC 接口和类精选参考	421
附录 D	JNDI 和数据源	433



第一部分 基本的 JDBC 编程

第 1 章 JDBC 简介

Java 已经确立了它作为企业计算领域的领先者的地位。你可以在所有主要计算平台上使用 Java, 而且在从一个平台转移到另一个平台时无需修改你的 Java 代码。另一个领先者是 Oracle 数据库, 它是企业的数据储存库。Java 数据库连接性 (Java Database Connectivity, JDBC) API 使 Java 能够访问数据库。可以使用标准的 JDBC 访问许多种不同的数据库, 其中包括 Oracle 数据库。但是, Oracle 数据库包含许多独特的特性, 只能通过使用标准 JDBC 的 Oracle 扩展来使用。在本书中, 你将学习标准 JDBC 以及 Oracle 扩展。Oracle 扩展可以在使用 Oracle 数据库时和调整 Oracle 扩展的高性能时尽可能发挥 JDBC 的能力。

在本章中, 你将学习那些在开发和运行包含 JDBC 语句的 Java 程序之前需要安装和配置的软件。你还会看到一个使用 JDBC 语句执行简单的数据库操作的示例程序, 并且将编译和运行这个示例程序。我将介绍 Oracle 的 JDeveloper 工具, 你可以使用这个图形化的集成开发环境开发、编译、运行和调试 Java 程序。即使你不使用 JDeveloper, 也可以通过我的说明和屏幕图了解这个工具提供的功能。

1.1 软件需求

在运行本书中的例子之前, 你需要在你的开发机器上安装以下软件:

- Sun Microsystems 公司的 Java Development Kit (JDK) 版本。
- Oracle Client 软件, 它带有 Oracle JDBC 驱动程序。

你可以在 Sun 的 Java 网站 <http://java.sun.com> 下载 JDK 并且查看完整的安装说明。这个网站还包含 Java 和 JDBC 的所有参考资料。你应该花时间查找和研究这些资料, 包括 JDBC API 文档。

你可以从 Oracle Technology Network (OTN) 网站 <http://otn.oracle.com> 下载 Oracle 软件的试用版 (包括 Oracle9i 数据库和 JDeveloper)。Oracle Client 软件也随数据库软件一起发布, 但是可以在运行 Oracle Universal Installer 时单独安装它。你还可以从 OTN 网站获得 Oracle 产品的文档。

另外, 你需要能够访问运行着以下软件的机器:

- Oracle9i 数据库 (版本 9.0.1 或更高) 或 Oracle8i 数据库 (版本 8.1.6 或更高)。如果你希望使用 Oracle9i 中新增的数据库对象和集合增强功能 (将在第 6 章和第 7 章中讨论), 那么需要访问 Oracle9i 数据库。如果你只能访问 Oracle8i 数据库, 那么仍然能够使用对象和集合, 但是不能使用 Oracle9i 中新增的增强功能。
- Oracle Net, 用于连接数据库。
- Oracle9iAS Containers for J2EE, 即 OC4J (版本 9.0.2 或更高)。在编写本书时, 可以从 OTN 单独下载 OC4J, 但是当你阅读本书时, 可能需要获得整个 Oracle9i Application Server 才能得到 OC4J。我将在第 11 章中说明安装和使用 OC4J 的细节。

Oracle 数据库、Oracle Net 和 OC4J 不必安装在你进行开发的本地机器上, 它们可以在网络上任何地方的机器上运行。通常, 你的公司会有一个人担任数据库管理员 (database administrator, DBA), 他负责建立和管理数据库。如果你不太了解数据库, 那么应该请教 DBA。在

下一章中，你将学到数据库的基本知识。

你使用的 JDK 和 Oracle JDBC 驱动程序的版本很重要：它们必须彼此兼容，而且与你使用的数据库兼容。你应该查看文档和 readme 文件，了解你使用的数据库支持 JDK 和 Oracle JDBC 驱动程序的哪些版本。OTN 上有可兼容软件版本的信息。

提示 无论你使用什么数据库版本，你通常应该使用 JDK 和 Oracle JDBC 驱动程序的最新版本，但是要检查文档以确保它们是兼容的。

在本书中，我使用以下的软件和版本：

- Java 2 Standard Edition JDK (1.3.1_01)。
- Oracle Client 软件和 Oracle JDBC 驱动程序 (9.0.1)。
- Oracle9i (9.0.1) 和 Oracle8i (8.1.6 和 8.1.7) 数据库。
- Oracle Net (9.0.1)。
- Oracle9iAS Containers for J2EE (9.0.2)。
- Oracle9i JDeveloper (beta 版)。

你实际上不需要 JDeveloper，但它是非常有用的 Java 程序开发和调试工具。在本章后面你将学习 JDeveloper。

Oracle Client 软件

在你的机器上安装 Oracle Client 软件的目录称为 ORACLE_HOME 目录。这个目录包含一些子目录，其中之一是 BIN 目录，它包含 Oracle 可执行程序。另一个目录是 jdbc 目录，它包含以下内容：

- 文本文件 Readme.txt。你应该阅读这个文件。它包含 Oracle JDBC 驱动程序的重要信息，如它支持的新特性和安装信息。
- doc 目录，其中有一个 ZIP 文件，它包含 Oracle JDBC API 参考文档。一定要解压这个 ZIP 文件的内容，并且研究这些参考资料。随着逐渐了解 Oracle JDBC，你会发现这些参考资料是无价的。
- demo 目录，其中有一个 ZIP 文件，它包含 Oracle 提供的示例 Java 程序。
- lib 目录，其中包含许多 ZIP 文件和 Java 归档 (Java Archive, JAR) 文件。我将在下一节中讨论这些文件。

1.2 配置

一旦你下载并且安装了所需的软件，下一步就是配置你的机器以便开发和运行包含 JDBC 的 Java 程序。你必须在你的机器上设置四个环境变量：

- ORACLE_HOME。
- JAVA_HOME (如果你使用 Java 2 Enterprise Edition，那么还要设置 J2EE_HOME)。
- PATH。
- CLASSPATH (如果你使用 Java 2 Enterprise Edition，那么还要设置 J2EE_CLASS-

PATH)。

如果你正在使用 UNIX 或 Linux，那么还需要设置环境变量 LD_LIBRARY_PATH。在下面几节中，你将学习如何设置这些环境变量。

1.2.1 设置环境变量 ORACLE_HOME

ORACLE_HOME 目录就是安装 Oracle 软件的目录。你需要将你的机器上的环境变量 ORACLE_HOME 设置为这个目录。

1. 在 Windows 2000 上设置环境变量

要想在 Windows 2000 上设置环境变量，首先需要打开 Environment Variables 对话框。为此，需要执行以下步骤：

- 1) 打开 Control Panel。
- 2) 双击 System。这将显示 System Properties 对话框。
- 3) 选择 System Properties 对话框的 Advanced 选项卡。
- 4) 点击 Environment Variables 按钮。这将显示 Environment Variables 对话框。

检查当前是否定义了系统变量 ORACLE_HOME。如果定义了，检查它是否被设置为你安装 Oracle 软件的目录。如果没有定义 ORACLE_HOME，就点击 System Variables 区域中的 New 按钮来创建它。图 1-1 是具有 ORACLE_HOME 设置的 Environment Variables 对话框（在我的机器上，ORACLE_HOME 被设置为 F:\oracle\Oracle90）。

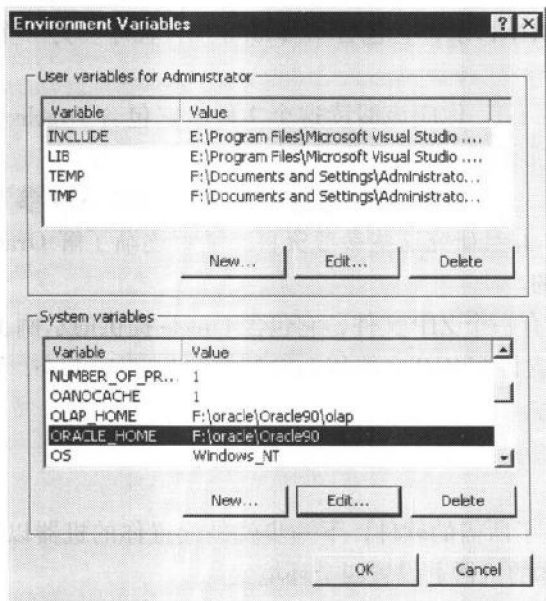


图 1-1 具有 ORACLE_HOME 设置的 Environment Variables 对话框

2. 在 UNIX 或 Linux 上设置环境变量

要想在 UNIX 或 Linux 上设置环境变量，需要在一个特殊文件中添加几行，需要修改的文

件取决于你使用哪种 shell。如果你使用 Bourne 或 Korn shell，那么需要使用以下语法在 .profile 文件中添加两行：

```
ORACLE_HOME=install_directory
export ORACLE_HOME
```

你应该将 install_directory 替换为安装 Oracle 软件的目录。如果在 Linux 上使用 Bash shell，那么使用相同的语法，但是要在 .bash_profile 文件中添加代码行。例如：

```
ORACLE_HOME=/usr/local/oracle
export ORACLE_HOME
```

如果你使用 C shell，就需要使用以下语法在 .login 文件中添加环境变量：

```
setenv ORACLE_HOME install_directory
```

例如：

```
setenv ORACLE_HOME /usr/local/oracle
```

1.2.2 设置环境变量 JAVA_HOME

环境变量 JAVA_HOME 指定你安装 JDK 的目录。例如，如果你将 JDK 安装在 F: \jdk1.3.1_01 目录中，就需要使用前一节中说明的步骤将 JAVA_HOME 变量设置为这个目录。

注意 如果你使用 Java 2 Enterprise Edition，那么还需要将 J2EE_HOME 设置为安装 J2EE 的目录。详细的安装说明参见 Sun 的文档。

1.2.3 设置环境变量 PATH

环境变量 PATH 包含一个目录列表。当你使用操作系统的命令行输入一个命令时，机器会搜索 PATH 中的目录，寻找在你的命令中指定的可执行文件。你的机器中可能已经有了对某些目录的 PATH。你需要向现有的 PATH 中添加以下两个目录：

- 安装 JDK 的目录中的 bin 子目录。
- 安装 Oracle 的目录中的 BIN 子目录。

例如，如果你在 Windows 2000 上的 F: \jdk1.3.1_01 目录中安装 JDK，在 F: \oracle \Oracle90 中安装 Oracle 软件，那么就在你的 PATH 中添加 F: \jdk1.3.1_01 \ bin; F: \oracle \ Oracle90 \ BIN。注意分隔这两个目录的分号。使用我在前面说明的步骤在 Windows 2000 上设置 PATH。

要想在 UNIX 或 Linux 中向现有的 PATH 中添加目录，需要修改对于你的 shell 合适的文件。例如，如果你使用 Linux 的 Bash shell，就将以下代码行添加进 .bash_profile 文件：

```
PATH=$PATH:$JAVA_HOME/bin:$ORACLE_HOME/BIN
export PATH
```

注意分隔这两个目录的冒号。

1.2.4 设置环境变量 CLASSPATH

环境变量 CLASSPATH 包含 Java 类包的位置列表。位置可以是目录名，也可以是包含类的 ZIP 文件或 JAR 文件。ORACLE_HOME \ jdbc \ lib 目录包含许多 ZIP 文件。需要添加进 CLASSPATH 的文件取决于你使用的 JDK 和你需要的特性：

- 如果你使用 JDK 1.2.x (或更高版本)，那么在 CLASSPATH 中添加 classes12.zip。如果你需要 National Language 支持，那么还要在 CLASSPATH 中添加 nls_charset12.zip。
- 如果你使用 JDK 1.1.x，那么在 CLASSPATH 中添加 classes111.zip。如果你需要 National Language 支持，那么还要在 CLASSPATH 中添加 nls_charset11.zip。
- 如果你需要使用 Java Transaction API (JTA)，那么在 CLASSPATH 中添加 jta.zip。jta.zip 文件位于 ORACLE_HOME \ jdbc \ lib 目录。
- 如果你需要使用 Java Naming and Directory Interface (JNDI)，那么在 CLASSPATH 中添加 jndi.zip。jndi.zip 文件位于 ORACLE_HOME \ jdbc \ lib 目录。

在第 6 章中，你将使用 SQLJ 将 SQLJ 文件转换和编译为 Java 类文件。我将在第 6 章讨论 SQLJ，但是你需要提前将 translator.zip 和 runtime.zip 添加进 CLASSPATH。这两个文件位于 ORACLE_HOME \ sqlj \ lib 目录。

注意 你还需要将当前目录添加进 CLASSPATH。办法是在 CLASSPATH 中添加一个点号 (.)。这样，当你运行你的程序时，Java 就可以找到当前目录中的类。

对于 Windows 2000，典型的 CLASSPATH 如下：

```
.;F:\oracle\Oracle90\jdbc\lib\classes12.zip;F:\oracle\
Oracle90\jdbc\lib\nls_charset12.zip;F:\oracle\Oracle90\sqlj\
lib\translator.zip;F:\oracle\Oracle90\sqlj\lib\runtime.zip;
```

如果你使用 Windows 2000，就使用前面说明的步骤设置 CLASSPATH。如果你使用 Linux，那么应该将以下代码行添加进 .bash_profile 文件：

```
CLASSPATH=$CLASSPATH:..$ORACLE_HOME/jdbc/lib/classes12.zip:
$ORACLE_HOME/jdbc/lib/nls_charset12.zip:$ORACLE_HOME/sqlj/
lib/translator.zip:$ORACLE_HOME/sqlj/lib/runtime.zip
export CLASSPATH
```

注意 如果你使用 Java 2 Enterprise Edition，那么还需要设置 J2EE_CLASSPATH。详细的安装说明参见 Sun 的文档。

1.2.5 在 UNIX 或 Linux 上设置环境变量 LD_LIBRARY_PATH

如果使用 UNIX 或 Linux，那么还需要将环境变量 LD_LIBRARY_PATH 设置为 \$ORACLE_HOME/lib。这个目录包含 JDBC OCI 驱动程序使用的共享库。

1.3 你的第一个 JDBC 程序

在本节中，你将看到一个简单的程序。它连接一个数据库，显示用于连接的用户名，并且显示从数据库获取的当前日期和时间。这个程序的源代码文件是 FirstExample.java。

注意 如果你下载本书的 ZIP 文件并且进行解压，那么就会创建一个 programs 目录，其中包含本书中的所有程序，当然也包含 FirstExample.java。这个 ZIP 文件包含本书中使用的所有程序、脚本和其他文件。对下载和解压这个 ZIP 文件的说明参见本书的前言。

1.3.1 示例程序：FirstExample.java

我将列出 FirstExample.java 程序的代码，然后简短地说明主要语句的作用。目前不必太关心这个程序的细节，你将在第 3 章中学习这些细节。FirstExample.java 程序的源代码如下：

```
/*
   FirstExample.java connects to the database as the "scott"
   user; the program then retrieves this username along with
   the current date and time from the database
*/

// import the standard JDBC package
import java.sql.*;

public class FirstExample {

    public static void main (String args []) {

        // declare Connection and Statement objects
        Connection myConnection = null;
        Statement myStatement = null;

        try {

            // register the Oracle JDBC drivers
            DriverManager.registerDriver(
                new oracle.jdbc.OracleDriver()
            );

            // create a Connection object, and connect to the database
            // as scott using the Oracle JDBC Thin driver
            myConnection = DriverManager.getConnection(
                "jdbc:oracle:thin:@localhost:1521:ORCL",
                "scott",
                "tiger"
            );

            // create a Statement object
```