

Oracle Administration and Management

Oracle 数据库 管理与维护技术手册

(修订版)

Michael Ault

著

江漫 张斌 扬帆

等译



清华大学出版社
<http://www.tup.com.cn>



Oracle 数据库管理与维护技术手册

(修订版)

Michael Ault 著

江漫 张斌 扬帆 等译

清华大学出版社

北京市版权局著作权合同登记号：图字：01-2002-2999

内 容 简 介

Oracle 是最先进的数据库系统，执数据库市场牛耳。本书全面讲述了 Oracle 管理和操作的内容，是一本管理 Oracle 数据库的技术、技巧和工具的完全参考手册。本书包含了 15 章的内容，涉及 Oracle 的安装、配置、各种对象管理、安全性管理、系统监控、性能优化、备份和恢复，以及分布式数据库等内容。

本书适合于 Oracle 数据库管理员参考使用。

Michael Ault: Oracle Administration and Management

EISBN: 0-471-21886-3

Copyright©2002 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. 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

图书在版编目(CIP)数据

Oracle 数据库管理与维护技术手册/(美)阿尔特 著;江漫 等译.

—北京:清华大学出版社,2002

书名原文: Oracle Administration and Management

ISBN 7-302-06200-5

I. O… II. ①阿… ②江… III. 关系数据库—数 IV. TP311.138-62

中国版本图书馆 CIP 数据核字(2002)第 106238 号

出 版 者: 清华大学出版社(北京清华大学学研大厦,邮编 100084)

<http://www.tup.com.cn>

责任编辑: 郭东青

封面设计: 康博

版式设计: 康博

印 刷 者: 北京昌平环球印刷厂

发 行 者: 新华书店总店北京发行所

开 本: 787×1092 1/16 **印张:** 53.75 **字数:** 1375 千字

版 次: 2003 年 1 月第 1 版 2003 年 1 月第 1 次印刷

书 号: ISBN 7-302-06200-5/TP·3707

印 数: 0001~3000

定 价: 116.00 元

译 者 的 话

当今社会，是一个信息时代的社会。信息技术的飞速发展大大推动了社会的进步，已经和正在改变着人类的生活、工作和学习等的方式。数据库技术和网络技术是信息技术中最重要的两大支柱。自从 20 世纪 70 年代以来，数据库技术的发展使得信息技术的应用从传统的计算方式转变到了现代化的数据管理方式。在当前热门的信息系统开发领域中，例如管理信息系统、企业资源规划、供应链管理系统、客户关系管理系统及电子商务等，都可以看到数据库技术应用的影子，都离不开数据库技术的坚实支持。

Oracle 系统是一个起步最早、最成功的数据库系统。在当前激烈的数据库市场上，Oracle 系统牢牢居于霸主地位。从某种意义上说，Oracle 系统代表了数据库的发展现状和趋势。许多人把数据库与 Oracle 系统等同起来，学习数据库技术就需要学习 Oracle 系统，学习 Oracle 系统实际上就是深入学习数据库技术的过程。

本书是一本管理和维护 Oracle 数据库的技术、技巧和工具的完全参考手册。对于数据库管理员来说，这是一本不可多得的 Oracle 参考书。本书具有三个特点，即结构合理、内容完整、示例丰富可靠。本书在结构上按照由浅入深的方式进行讲述，因此无论是初级数据库管理员，还是高级数据库管理员都可以从中找到自己需要的内容。本书在内容上，不仅反映了 Oracle9i 的最新功能，而且还涉及了 Oracle8、8i 以及 7 甚至更前的版本内容。读者可以通过阅读这些版本的发展和比较，对相应的内容具有更深刻的认识。本书的作者是一位经验丰富的 Oracle 数据库管理员，他把自己多年的 Oracle 数据库管理和维护经验和工作成果以示例的形式展现在书中。读者可以通过这些示例，更好地理解 Oracle 数据库管理和维护的内容，并且减少自己犯错误的机率。

本书共包含了 15 章，具体讲解了 Oracle 的安装、系统管理、管理表空间、管理关系表、管理对象表、管理索引、管理其他对象、管理表簇、管理用户和安全性、监控数据库对象、监控用户、性能优化、内部优化、管理分布式数据库及备份和恢复这些内容。

本书适合 Oracle 数据库管理员参考使用。也可以作为使用 Oracle 的各种应用程序的开发人员的参考手册。

译 者

2002 年 9 月

前言

已经选择了 Oracle，那么现在该怎么办

Oracle 公司的 Oracle8 和 Oracle8i 产品都具有使数据库现代化的趋势，而这一趋势在 Oracle9i 中更加明显。有些人觉得 Oracle9i 实际上只是 Oracle 的 8.2 版；但 9i 中的众多新特性、新结构以及新功能，都能充分证明它完全是 Oracle 公司推出的新产品。

Oracle 的强大功能来源于它能给用户快速和准确的数据检索能力。这也是关系型系统的主要功能。随着 Oracle 在面向对象方面的扩展，它现在允许在关系型引擎功能的基础上进行实际的建模。面向对象和关系型相结合，直接导致了关系对象型数据库的产生。关系对象型数据库提供了一种逻辑的，通常是简明的数据表示方式。表格状的关系格式(扩展到包括变长数组、嵌套表和对象视图，以及允许存储包含数据的方法)，用一种与关系型范式类似的新方式来表示数据。另外，Oracle8i 将 Java 作为存储代码存储在数据库中的功能，提供了不可估量的能力和灵活性。对 XML(eXtensible Markup Language，可扩展标记语言)的更强大的兼容性，使得 Oracle 走在了先进技术的最前沿。这些特性，再加上关系型的基础，使用户可以查询信息，以便轻松地得到他们想要的数据库，并且是准确地得到他们想要的数据库。这些都是如何实现的呢？

这是通过关系型数据库的表格形式，以及在简单、明确的方法中加入了面向对象的扩展功能来实现的。相关的数据和方法以及这些数据对象之间关系的逻辑集合形成了数据库。Oracle 通过它的 SQL、SQL*Plus、Java、XML 和开发工具，以及其他工具，允许开发人员、用户和管理员查看它们的数据，这在过去是不可能的。

本书的目的是给负责维护和管理 Oracle 数据库的管理员提供一套工具，从而使他们的工作更加简便。通过示例和实际的方案，数据库管理和维护人员将可以深入地洞察 Oracle 数据库管理系统的工作。本书还将给出一些报表的例子，以及产生这些报表的 SQL、SQL*Plus 和 PL/SQL 代码。同时还会给出这些报表的解释。

Oracle 提供了一个庞大的数据库管理工具集。但是遗憾的是，初级的 DBA(数据库管理员)一般都不具备充分使用这些工具的必备知识。本书的意图就是想改变这一点。在随后的章节中，讲到了数据库的管理和维护的所有步骤，从最初的安装到日常维护一直到最重要的备份、恢复和故障恢复过程，这些可以用来区分成功的 DBA 和普通的 DBA。另外，还将讲到 OEM(Oracle Enterprise Manager)。OEM 是一个用 Java 编写的 GUI 工具，它允许对 Oracle 数据库进行简易的管理和维护；然而，有些时候它是适用的，而有些时候则不行，这在随后的章节中会讲到。

在本书以前的版本中，讲到了 Open VMS、UNIX 和 Windows。在这一版本中，将减少 VMS 的内容以便增加 UNIX 中的一个新的开放源版本——Linux 内容。

Oracle9i 概述

Oracle 9i 版本是一个关系对象型数据库管理系统(Object-Relational Database Management



System, ORDBMS)。Oracle9i(实际上是 9.0.1 发布版本)对 Oracle8i 中的一些新的特性进行了扩展,并作了大量的改动,还添加了一些 Oracle 工具。像曾经提到过的那样,传统的 RDBMS 表中存储的数据叫做关系。这些关系是对数据的二维描述,在关系术语中,行叫做元组,表示记录;列叫做属性,是记录中包含的各种信息。Oracle9i 对 Oracle8i 中提供的面向对象的扩展程序和 RDBMS 添加了新的特性。在关系对象型数据库中,列可以代表一个单独的数值(在标准的关系数据库中),一个变长数组(固定数量的附加记录),或者一个可以存储可变数据数量的副表的 REF。这需要在二维关系视图中添加第三维。而且,在一个关系对象型数据库中,被称作方法的程序可以和表相关联。方法已经超出了以前的触发器的概念范围,这将在前言的后面部分进行解释。在 Oracle8i 的后期版本中,Java——一门新的面向对象的语言,可以用来在一个 Oracle8i 数据库中创建存储对象。Oracle9i 扩展了 Oracle 中 Java 的功能,添加了全新的维以便不时对数据库进行优化,从而支持可变大小的数据库块、多个缓冲池和其他众多的新特性。

Oracle 提供了丰富的工具集以便对数据库进行维护 and 设计。最主要的 Oracle 工具如下:

RDBMS Kernel 数据库内核程序,是 Oracle 的核心。

SQL 结构化查询语言,是关系型语言。

SQL*Plus SQL 语言和数据库的 Oracle 接口。

PL/SQL 过程式 SQL 语言,允许对 SQL 语句进行过程化的处理。

SQL*Loader 可以将 ASCII 码文件中的数据导入到 Oracle 表中。

EXPORT/IMPORT 允许删去数据和结构信息,并且把这些数据和结构信息从 Oracle 数据库插入到归档文件中。

INTERMEDIA 允许访问存储在 Oracle 中的文本型数据。

Developer2000 生成基于 Designer2000 的输入内容的表单、报表和 3GL 代码。

Designer2000 允许生成 ERD(Entity Relationship Diagram, 实体关系图)、FHD(Function Hierarchy Diagram, 函数功能层次图)、Matrix(矩阵)和数据流程图(Data Flow Diagram, DFD)。

Oracle9iAS 基于 Apache Web 服务器技术的一个应用程序,是 Oracle 的扩展程序。

SVRMGR Oracle 替代 SQLDBA 的工具。在 9i 版本中,被 SQL*Plus 新增的功能代替。

OEM Oracle Enterprise Manager,提供在整个企业内部管理多个实例的功能。

RMAN Recovery Manager,为 Oracle 提供自动备份和恢复选项。

Oracle precompilers 为最主要的 3GL 语言提供接口。

JDBC Oracle 对 Java Database Connection 接口的实现。

SQLJ Oracle 的 Java 预编译器,可以在 Java 内自由使用 SQL 语言。

Java 在 8.1 以后的版本中和所有的 9i 版本中,Java 已经移进了 Oracle 内核。

还有很多很多工具,这里没有提到!

要完全理解一个 Oracle 数据库的结构,关键是有使用这些工具的能力。作为一个数据库管理和维护人员,必须对其中的一些工具相当熟悉。

Oracle 不仅仅是一个可以轻松访问数据的程序的集合,还可以将其比作是覆盖了它所驻留的计算机上的操作系统的一个操作系统。Oracle 有它自己的文件结构、缓冲区结构、全局区域以及自身的优化能力,这些功能超出了操作系统提供的功能范围。Oracle 可以控制自己的进程、自己的记录以及保持一致性,并整理自身系统。

就像您所用的系统那样(Windows 除外,它使用的是多线程),Oracle 包括可执行程序、5~9

个独立的进程、一个全局存储区、数据文件以及维护文件。它可以小到几兆，也可以大到全部由太拉字节(2 的 40 次方)组成。图0-1 和 0-2 对典型的 Oracle8i 和 Oracle9i 环境分别作了展示，在阅读前言的下一节的时候可以参照这些框图。

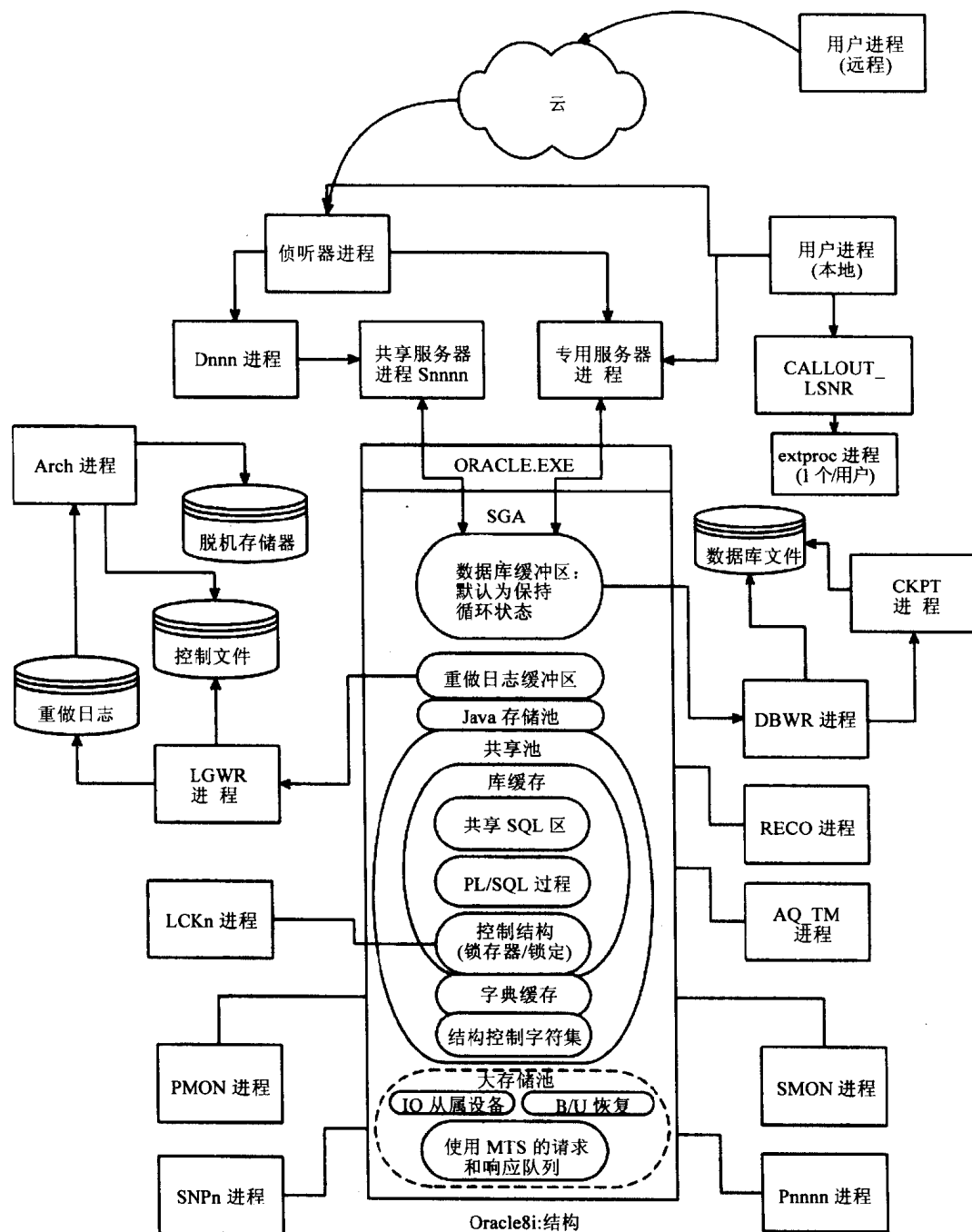


图0-1 Oracle8i 的结构和文件

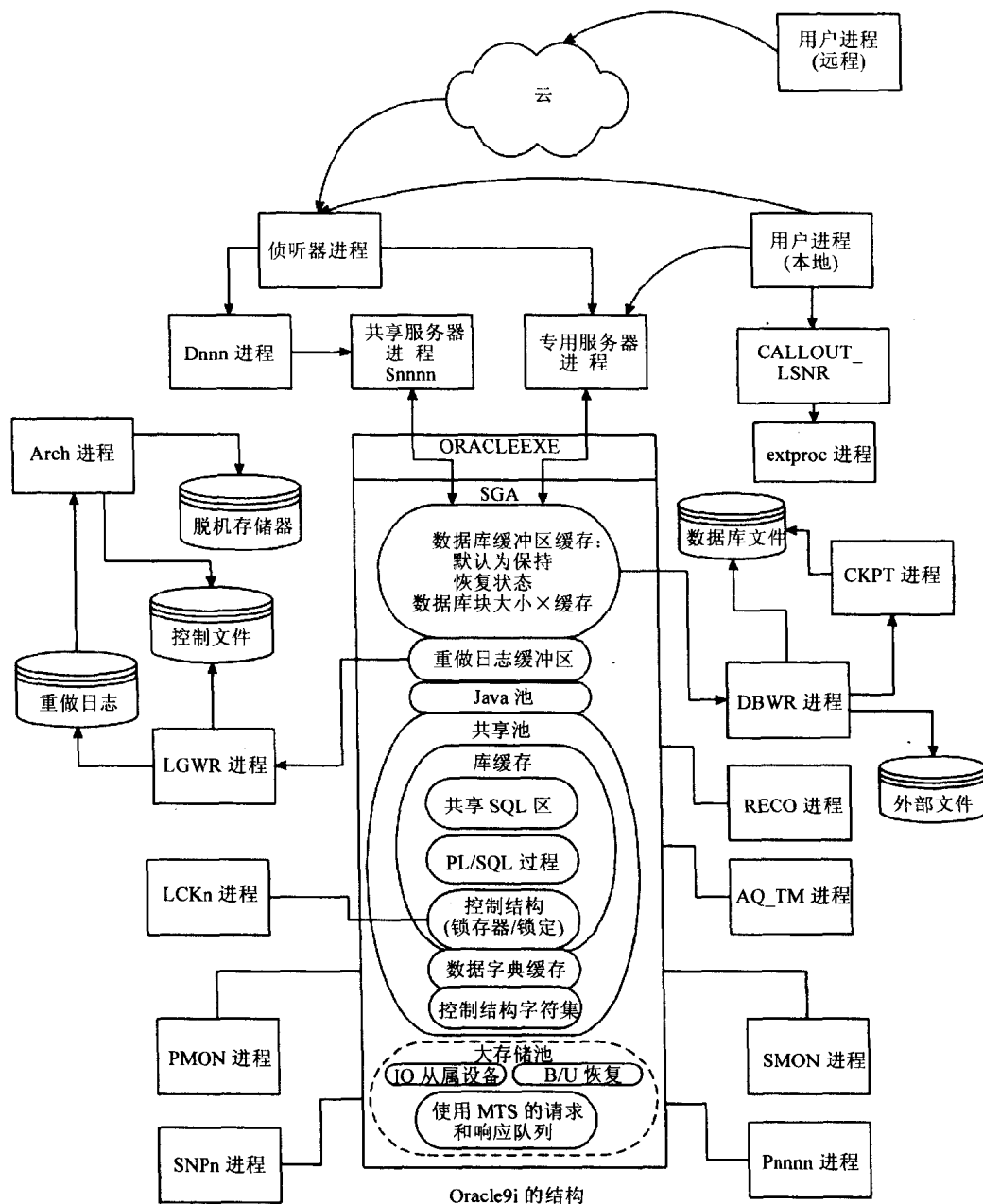


图0-2 Oracle9i 的结构和文件

让我们看看工作在 UNIX、Linux 或者 Windows 环境中的一个典型的 Oracle 系统。在 NT(使用线程)、Linux 或者 UNIX 中,最少可能有十几个独立的进程是属于 Oracle8i 的;对 Oracle9i,则更多。其中的五个是 Oracle 的基本进程(PMON、SMON、DBWR、LGWR 和 CKPT),它们在系统启动 Oracle 时就会自动启动。另外的进程可能会在数据库使用归档(ARC)、使用 TCP/IP 协议或者在并行或者分布式模式运行时启动。Oracle 的作业队列、快照进程、高级队列选项以及标记进程都添加到了进程的统计中。这些进程描述如下:

DBWR(数据库写入器) 从 SGA 中的缓冲区到数据库文件处理数据传输。可能有多个数据库写入器进程,以及数据库写入器从属进程,这些取决于 Oracle 的发布版本级别。

LGWR(日志写入器) 从重做日志缓冲区中传输数据到数据库的重做日志的数据库文件中。可以配置多个写入器，从而使用多个可以写日志的位置。

SMON(系统监控器) 在实例启动的时候执行实例恢复操作，并且负责清除临时段。在并行的环境中，SMON 可以恢复失效的节点。

PMON(进程监控器) 恢复已经失败并已在缓存中清除的用户进程。PMON 可以从一个失败的进程中恢复资源。

ARCH(归档进程) 只有在归档日志起作用的时候才进行激活。ARCH 写入到归档日志数据文件中的重做日志数据文件。有可能会有多个归档进程，这取决于在初始参数中定制的存档日志位置的数量。

RECO(分布式) 解析失败。

Transaction Recoverer 恢复分布式事务处理。

LCKn(锁进程) 用于在 Oracle 并行服务器(Oracle parallel server, OPS)中或者实际的应用程序簇的环境中锁定的中间实例。

Dnnn(调度) 允许多个进程共享一个确定数量的 Oracle 服务器。将对请求下一个可用服务器的进程进行排队并确定其路径。该进程用在安装 Oracle 的多线程服务安装中。

Snnn(服务器) 促使数据库所需的所有调用来解决用户的重复请求。它将给调用它的 Dnnn 进程返回结果。这是一个多线程的服务程序进程，仅仅在数据库使用到 MTS 时才会进行配置。

LISTENER TCP/IP server 如果将要运行 TCP/IP，那么该进程作为一个侦听器进程，将会同时运行(每个节点可能有多个)。其他的通信协议也可以拥有侦听器进程。

CKPxx 检查点进程，开始于对登录 Oracle 的检查点操作进行优化。在 Oracle8 以前的版本中，该进程的启动是可选的；在 Oracle8 及其以后的版本中，它会自动启动。

Snpxx 快照进程和作业队列。在 8i 中，可以配置的数量最多为 32 个。

EXTPROC 标记队列，每个会话将会有有一个进程执行标记。

QMNn Oracle 队列的监控进程(8i 中新增加的功能)。它们对 Oracle 高级队列(Oracle advanced queueing, Oracle AQ)中使用到的队列进行监控。最多可以有 10 个 QMN 进程。这些进程取代了前一版本中的 AQ_TXX 单进程。

图 0-3 显示了在 SuSE7.2 系统中运行一个典型配置的 Oracle9i 实例时，执行“ls-eflgrep oracle”命令的结果。注意，普通的配置包括一个最小数量的多线程服务器配置(d000-d002 和 *s000 进程)。如果允许使用 Oracle Universal Installer 对数据库进行安装，那么可以通过默认方式这样做。如果不需要 MTS，那么就必须从初始化文件中删除配置参数，以便中止对进程的配置。

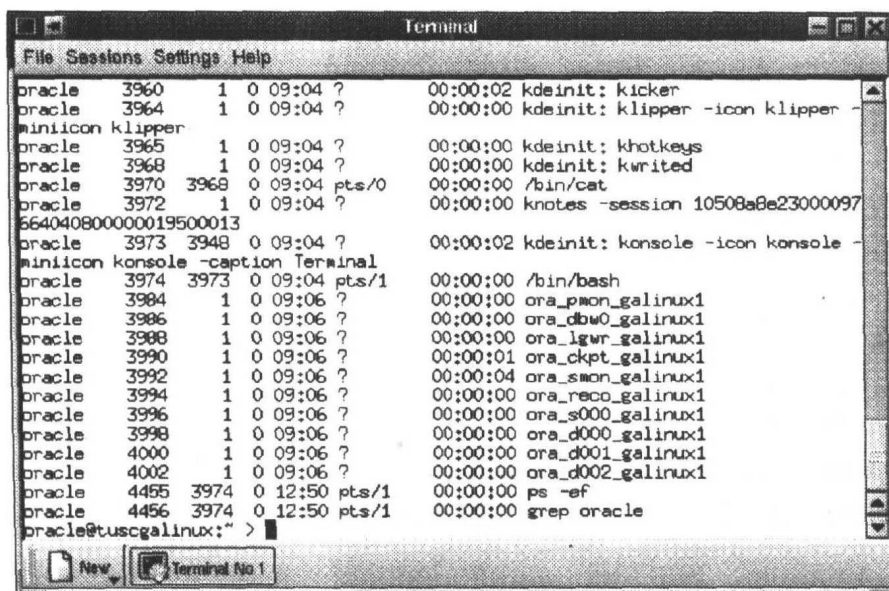


图0-3 Linux 操作系统中有关 Oracle9i 的进程清单

在 Windows 或者 W2K 操作系统中想要得到一个类似图 0-3 中的列表可能会有一些问题。但是，如果从微软的站点下载了 Resource Kit 4.0，它将会提供几个监控多线程进程的工具，那么就使得监控变得简单了。通过这种方式监控多线程进程的工具中有一个就是 Pviewer，它允许用户从主进程列表中挑选出一个进程，同时还将告知用户所有的相关线程和它们的内存使用情况。图 0-4 列出了一个来自 Windows NT 4.0 Server 操作系统上的 Oracle8i, 8.1.7 中的实例的示例。

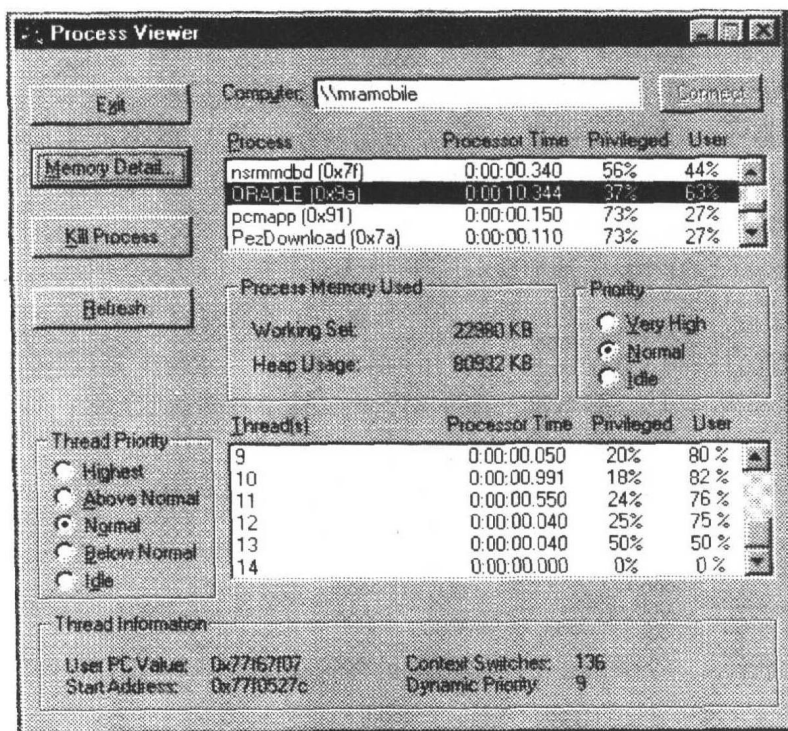


图0-4 PViewer 软件输出 NT 4.0、SP6a 和 Oracle9i 进程的示例窗口

当然，如果需要查看实际的进程名称，那么就必须进入 Oracle 并查看 V\$SESSION 表。程序清单 0-1 中展示了一个 NT 4.0 操作系统中的 V\$SESSION 表输出的一个报表。

程序清单 0-1: Oracle 进程名称的报表输出示例

Date: 02/07/02		Page: 1	
Time: 06:41 PM		DBAUTIL	
		Oracle Threads Report	
		aultdb1 database	
	Decimal Hex		
Process	Owner	Thread Thread	PID
-----	-----	-----	-----
User	Internal		1
PMON	Internal	822 336	2
DBW0	Internal	849 351	3
LGWR	Internal	850 352	4
CKPT	Internal	851 353	5
SMON	Internal	853 355	6
RECO	Internal	854 356	7
User	Internal	855 357	8
User	Internal	857 359	9
User	Internal	866 362	10
User	Internal	867 363	11
User	DBAUTIL	884 374	12
12 rows selected.			

在允许多用户的系统中，每个用户进程可能会产生若干个子进程，这取决于进程的活动类型。根据 Oracle 的配置，一个单独的并行查询语句可以为单用户启动数十个查询从属进程！

全局存储区，又叫做系统全局区(System Global Area, SGA)，是一个仅仅为 Oracle 保留的 CPU 存储区。它包含用来加速整个事务的缓冲区，并且帮助保持系统的完整性和一致性。没有数据是直接磁盘上进行更改的，所有的数据更改操作都是通过 SGA 来进行操作的。SGA 的大小和配置是由一个叫做 INIT.ORA 的文件或者初始化文件来定义的，该文件包含了 SGA 中每种类型的缓冲区或者共享池区的大小等信息。

同时还存在组成典型实例的共享程序。这些共享程序可能少到只有一个(比如, Oracle 内核)，也可能非常复杂，如整个工具集。

像前面说过的那样，Oracle 可以看作是一个存在于现有操作系统上的操作系统。它有独立的结构和构造。有关 Oracle 的安装，最重要的当然是数据库了。为了访问某个 Oracle 数据库，必须至少拥有一个分配给该数据库的 Oracle 实例。实例包含子进程、全局区域以及分配给它的相关结构。可以有多个实例和一个数据库相连接。数据库和实例的许多特征在创建的时候就已经决定了。单个的 Windows、Linux 以及 UNIX 平台可以同时几个实例进行操作；在 Windows、Linux、以及 UNIX 操作系统中，它们可以和自己的数据库相连接，或者共享同一个数据库。在一个共享数据库的环境中，所谓的远程应用程序簇，或者在 Oracle9i 中称作 RAC，需要额外的许可和进程。

Oracle 文档的重量超过 60 磅，不管用什么方式进行阅读，都不是一件容易的事。要完全包



含这些文档，至少需要一张光盘。

关系型术语简述

毫无疑问，有些人想知道我在这本书中究竟将讲述一些什么内容，像关系、元组和属性等术语到底是什么意思？在任何领域，90%都是在学习术语——该领域的特殊语言。在 Oracle 中，术语就是和关系型数据库相关的那些概念。这些术语很多都要归功于 E.F.Codd 博士，他明确地表示了数据的这些规则(叫做“范式”)，以及关系数据库设计的关系代数基础。

您可能已经接触过像范式、元组、主键和外键等话题。本书的目的不是给出一个关系理论的完整教程。但是，我的确想阐明这个“关系语言”的含义，因此那些没有关系术语基础的人将会发现，该书对他们和对那些已经对关系术语很熟悉的人同样适用。

我已经提到过表、元组和属性，还接触到了关系。现在让我们更加清楚地看看关系，特别是它们在关系数据库中的应用。想想您所工作的公司或者是您正想求职的公司。公司有雇员，或者，叫做公司雇用的员工。这就是一个关系。关系就是包含在实体中的信息之间的逻辑。在这个案例中，信息来自这些实体：A(公司)和 B(员工)。

一个员工可以有多份工作吗？当然可以有。一个公司可以有多个员工吗？是的。那么现在重新说明这个关系：一个公司可以雇用一個或者多个员工。一个员工可以被一个或者多个公司雇用。

这叫做多对多关系。当然，也存在其他类型的关系。在一个公司内，一个员工在一段时间内通常在一个部门工作，而一个部门可以有很多员工。这叫做一对多关系。一般来说，绝大多数多对多关系都可以分解成一对多关系，一对多关系是关系数据库中最主要的关系类型。一个关系存在于两个实体之间。在我们的例子中，“员工”实体将会映射到一个表中。显示一个关系型数据库的逻辑结构的图叫做实体关系图(ERD)。图0-5 显示了一个简单的实体关系图。

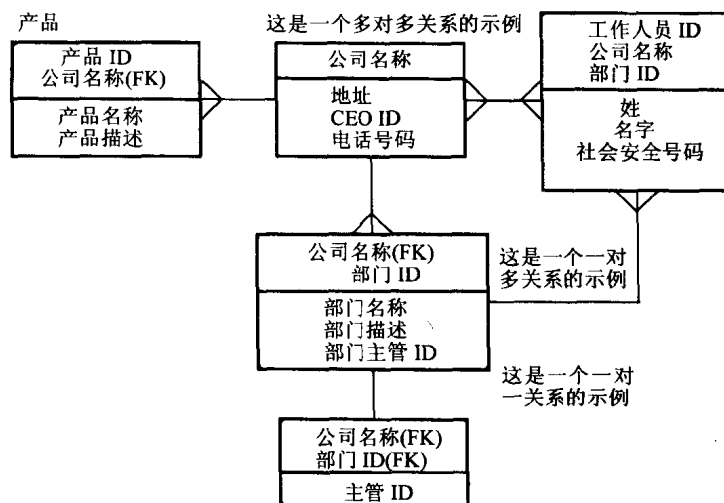


图0-5 一个 ERD 图示例

关系数据库的另外一个方面是它的函数。没有函数，一个数据库将毫无用处。函数用在比较高的层次上，比如：“提供一种方式对雇员的工作经历进行跟踪和报告”。函数可以拆散，

或者只要您愿意，也可以进行分解，直到分离到最原始的原子状态。分解一个函数的极端例子是直到操作中只包含独立的属性，例如添加、删除、更新以及查询。

举例来说，假如我们想从表中检索一条记录(或者元组)，更新其中的一列(或者属性)，并将该数据行返回到表中。在这个例子中，可以考虑使用一个函数，update of attribute x。换个角度，该函数可以分解成独立的查询、修改和对列的更新操作。在绝大多数情形下，不需要分得过于详细。数据库要执行的函数可以构成函数(功能)层次图。图0-6 展示了一个简单的函数(功能)层次图(Function Hierarchy Diagram, FHD)。

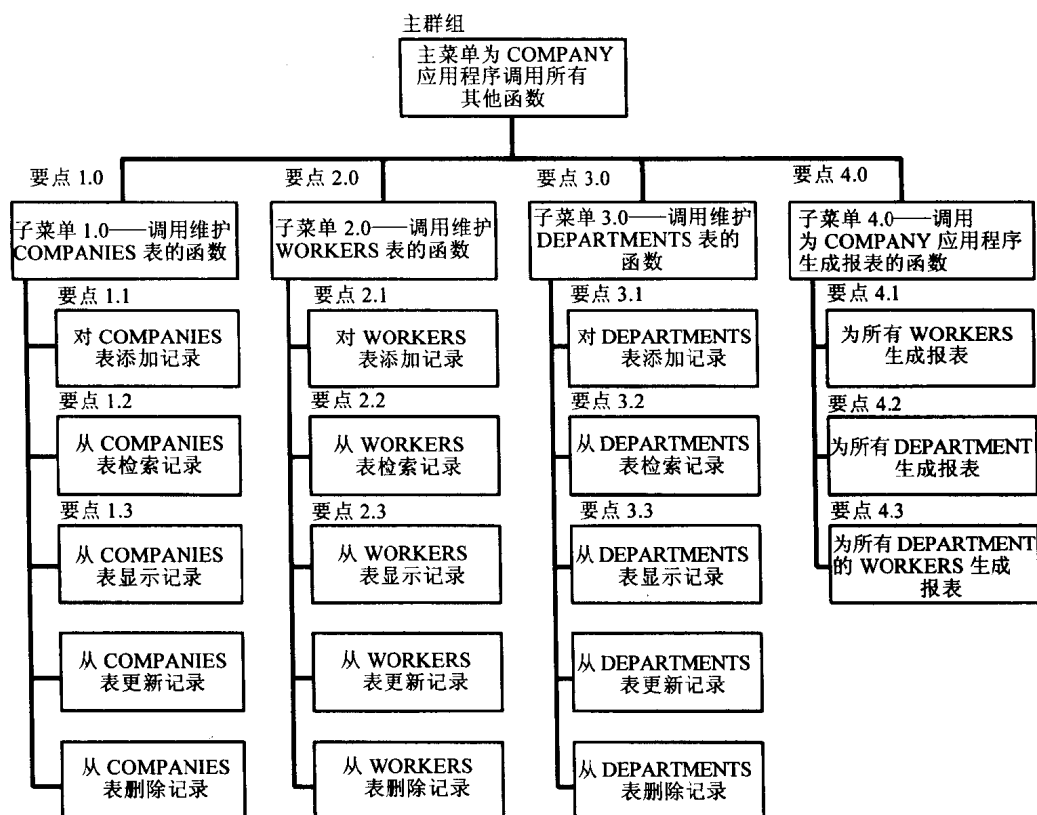


图0-6 一个简单的FHD示例

关系型数据库最后一个组成部分是模块。一个模块执行一个或者几个函数。一个模块可以映射到表单、报表、菜单或者一个过程中。例如，一个单独的提供表单的模块可以处理大量的原子函数，例如，对一个表甚至一组表以及数据记录进行添加、更新检索以及删除操作等。

总结一下，一个关系数据库由包含属性的实体构成。这些属性和实体可以映射到一个表中。实体向表每添加一行，它就映射一次。这些数据行叫做元组。每个实体通过关系和一个或多个其他实体相联系。关系双向有效，在程度上必须恰当，比如说，一对多关系或者多对多关系。关系还必须具有可选择性，比如可能存在和绝对存在。

函数用来告诉系统对实体和关系执行什么样的操作。实体和关系映射到函数。模块实现了这些函数且把这些函数映射到表单、报表、菜单或者一个过程中。

所有这些取决于Codd博士提出的规则。整个规则集是很复杂的，并且对我们绝大多数人来说理解起来比较困难。幸运的是，可以用它们来生成规范化的规则。这些规范化的规则简单、明了，如下所述：



- 预报器(Precursor) 每个实体都是由属性和/或者关系组合来惟一地标识。
- 第一范式(First Normal Form, 1NF) 删除实体中重复的属性或属性组。
- 第二范式(Second Normal Form, 2NF) 删除那些只与部分惟一标识符相关的属性。
- 第三范式(Third Normal Form, 3NF) 删除那些依赖于非惟一标识符属性的属性。

简单来说,在第三范式中,一个实体中的属性必须直接并且只和惟一标识符相关联。这个惟一标识符叫做主键。它可以是一个独立的数字,类似社会安全号码,或者是属性的组合,叫做连接或者复合键。比如说姓氏和出生年月日等。一般来说,这些主键可以看作通过将键映射到其相关联的实体中来加强与实体的联系。这时,它们就变成了外键。

对于那些希望更详细地了解有关实体关系图、规范化和其他相关主题的读者,推荐查阅下面这些书籍:

Atre, S. Data Base: Structured Techniques for Design, Performance and Management. Second edition. New York: John Wiley & Sons, 1998。

Barker, Richard. CASE*METHOD: Entity Relationship Modelling. Reading, MA: Addison-Wesley, 1990。

Barker, Richard 和 Cliff Longman. CASE*METHOD: Function and Process Modelling. Reading, MA: Addison-Wesley, 1990。

Barker, Richard. CASE*METHOD: Tasks and Deliverables. Reading, MA: Addison-Wesley, 1990。

Braithwaite, Ken S. Relational Databases: Concepts, Design, and Administration. New York: McGraw-Hill, 1991。

Date, Chris J. Introduction to Database Systems, 6th ed., vol.1 Reading, MA: Addison-Wesley, 1994。

面向对象术语简述

毫无疑问,很多人知道大量的有关面向对象的技术,甚至比我知道得更多。不过,简单复习一下这些基本术语会对我们很有好处。

面向对象领域的核心当然是对象了。那么什么是对象呢?在《Visual Basic 5: Object-Oriented Programming》(Coriolis Press, 1997)一书中, Gene Swartzfager 解释到:“对象是一个包含相关过程和数据集合的软件包。”反过来,过程又叫做方法,数据元素又叫做属性。因此,对象就是包含属性和方法的软件包。对象要完全融入到面向对象领域中的话,必须是相互独立的。这时,将它们作为可重用模块是非常理想的,因为可以对其进行独立的维护,这种维护与其他对象无关,只需实现和增强恰当的人机交互界面和收发消息标准就行了。遗憾的是,很多公司忘记了最后这一点。

对象通过消息进行通信。消息通常包括三个部分:对象的名称、对象的方法(或者成员),以及属性列表。对象还包含在定义对象类的每个成员的类中。单独的对象叫做该对象的实例。类中对象的特定实例与该对象的惟一的差别在于特定实例运行时的属性不一样。

其他与“对象说法”相关的术语有:封装、多态性和继承(我曾经碰到一个程序员为封装多态性而痛苦不已,但是自从使用了继承之后,他的工作就显得很轻松了)。这三个主要的面向对

象技术到底是什么呢？

- 封装是面向对象程序的特性，它意味着彻底的独立。也就是说，它和其他对象无关，其他对象也与它无关。由于封装的对象不通过外部的通信就可以传递属性、返回属性，所以它可以称作数据隐藏或者数据绑定工具。
- 多态性允许多个类或者对象使用同一个方法。重载 Oracle 的 PL/SQL 函数来处理多种数据类型就是多态性应用的例子。在 PL/SQL 中，重载函数只是几个有着相同的名称和参数结构的函数或过程，但是它们传递的参数是不同的数据类型。用户并不知道存在多个函数或者过程：他或者她只需要调用函数或者过程就可以得到返回值。
- 继承的意思显而易见。我们已经接触到了类的概念。类可以分为父类和子类。继承允许子类从他们的父类继承行为。一般来说，父类是一个有遗传函数的一般的类，使用子类来扩展更多的特有行为。Oracle9i 支持有限的继承。

还有一些 Oracle 特有的定义，以便扩展 Oracle 的面向对象功能。这些与 Oracle 如何使用面向对象技术相关。让我们一起看一些 Oracle 特有的定义，来完成对面向对象技术术语的讨论。

- **对象类型(Object type)** 对象类型(在早期发布的文档中，叫做抽象数据类型)是定义 Oracle 对象的数据类型。Oracle 对象是一个 Oracle 对象类型的实例。Oracle 的对象类型既可以是永久性的，比如说表或者视图，也可以是临时性的，比如说 PL/SQL 中的存储结构。CREATE TYPE 命令用来创建 Oracle 对象类型。永久性的对象类型将拥有对象标识符(OID)，而非永久性的对象类型则不会拥有对象标识符。
- **对象表(Object table)** 对象表是一个 Oracle 对象类型表。如果在 Oracle9i 中导入 Oracle8 以前的表，那么它将不会自动转换成对象表。要使关系表变成对象表，必须创建对象的部分视图。只有代表实际对象表的对象才拥有 OID。在早期的 Oracle8 文档中，对象表被称作为盘区(extent)表或者类型表。
- **对象标识符(Object identifier, OID)** 如前所述，只有存储在对象表中的对象才拥有一个对象标识符(OID)。OID 用来确保对象全局惟一，它是由一个 128 字节的十六进制数组成的。仅仅通过其自身，OID 是不能用来对对象实例进行定位的。只有包含定位数据的 REF(将在第 5 章中讨论)，才可以用来对对象实例进行定位。
- **对象(Object)** 一个对象数据类型的单独实例。可能有或者没有相应的 OID。如果对象来自一个对象表，则它将拥有一个 OID；如果对象来自一个 Oracle 调用接口变量或者 PL/SQL 变量，则将没有 OID。
- **嵌套对象(Nested Object)** 如果一个对象的对象类型用来定制对象表中单独的一列，我们就称其为嵌套对象。
- **嵌套表(Nested Table)** CREATE TYPE 命令可以用来创建类型，该类型实际上是一个表。然后，这个表类型就可以用来创建这种类型的表，该表可以作为第二个对象表中的一列。这就产生了嵌套表。如果现在对此弄不明白也不用着急。在后面几章中，您将会看到一些更易懂的例子。
- **数据类型(Datatype)** 在 Oracle9i 中有三种数据类型：内置类型(built-in)、库类型(library)和用户定义的类型(user-defined)。内置的数据类型包括标准的 NUMBER、DATE、VARCHAR2、LONG、LONG RAW、BLOB 等数据类型。用户定义的数据类型是用户或



者 VAR 建立的特殊的数据类型。库类型是由第三方建立并提供给用户使用的类型(一般是特殊形式的用户定义的数据类型)。

- **大对象(Large Object, LOBS)** LOBS 是 Oracle8 中新增加的特性。有多种类型的 LOB: 字符型的 LOB(CLOB)、二进制的 LOB(BLOB)、国际字符 LOB(NCLOB)和存储在数据库外部的 LOB。对长的二进制文件比如说数字化电影或者音轨一般使用 BFILE 文件格式。
- **外部过程(External Procedure)** Oracle8 的另外一个新的特性就是可以调用没有存储在数据库中的 PL/SQL 过程。这里所说的过程是指 3GL(第三代语言)程序, 只支持 C 和 Java 语言。同样也指对 3GL 外部程序的调用。
- **构造器(Constructor)** 这是一个 Oracle 内核生成的方法, 用于实例化 Oracle 对象类型实例。它与用户所创建的方法的不同之处在于, 它是由系统创建的, 并且当对某个对象类型进行调用或者需要一个对象类型而又没有隐式的或者显式的 SELF 参数时, 可以自动应用(使用一些“对象语言”)。
- **转发类型定义(Forward type definition)** 我倾向于将其叫做类型中的类型(type-in-type)。从本质上来说, 它意味着用户可以定义一个对象类型, 然后在第二、三或者任何数字的后续类型定义中使用该类型。在一个特殊的情形中, 两种类型在它们的定义中相互引用。如果您是一个 C 语言程序员, 这与转发声明等同。第 5 章中将会展示这种类型中的一个类型示例的定义。
- **对象视图(Object view)** 对象视图允许普通的关系表像对象关系数据库中的对象一样被引用。对象视图进程为关系型的行提供合成的 OID。第 7 章将会演示这种技术。
- **统一建模语言(Universal Modeling Language, UML)** 最新的建模语言可以使用面向对象的方法进行设计。
- **Java** 一门面向对象的编程语言。在 Oracle8 和 Oracle9i 中, 可以用来创建内部存储的对象以及 Java 脚本、Java 小应用程序以及应用程序。

数据库管理员的职责

不能将数据库管理员(DBA)和数据管理员(DA)混为一谈。数据管理员负责通过命名规则和数据字典对数据进行管理, 数据库管理员则负责数据库在物理上的实现。这两个管理员的职能有相互重叠和包含的部分。实际上, 本书对两者不加区分, 因为 DBA 的职能是和 DA 的职能紧密联系的。DBA 为 DA 和用户提供支持和技术建议。

DBA 的职能范围通常会不断地变化和扩展。他可能包含数据库的物理设计和实现、执行状态的监控和优化, 甚至对接口程序进行测试和配置, 比如说数据库使用的 X Window 模拟器。

数据库管理员在协同系统管理员对数据库访问进行优化时, 必须要有移动数据文件的自由。DBA 应当和系统管理员协作以确保对可用资源的合理使用。在一个公司里, 没有比数据库管理员和系统管理员之间发生争斗更容易导致失败的事了。

很多公司都认为 DBA 必须对他们的应用程序相当熟悉。在 DBA 职位上呆久了以后, 我发现事实并非如此。可以肯定地说, 如果要设计和创建一个应用程序, 那么就需要对相关知识相当熟悉, 但是如果仅仅是维护和优化, 那么就并不需要了解相关知识。实际上, 我发现, 在很

多实例中(无双关意思),太多的应用程序层次的知识可能会使 DBA 看不到在事务层次上实际发生的事情。在每个数据库中,优化最后都归于这个问题:事务是在以最小的时间、最高效的方式运行吗?在事务层上看一个数据库就像在一个森林中看树一样:您需要知道当前这棵树的类型,但是您不需要了解整个森林。

根据《Oracle9i Database Administrators Guide》(版本 1,代号是:A90117-01,2001 年 6 月,Oracle 公司)一书,可以将一个数据库管理员的主要职责列表如下:

- 安装和升级 Oracle 服务器和应用程序工具。
- 为数据库分配系统存储空间并计划将来所需的存储空间。
- 在应用程序开发人员设计出一个应用程序后,创建主要的数据库存储结构(表空间)。
- 创建应用程序开发人员设计出的应用程序的主要对象。
- 根据应用程序开发人员提供的信息,如有必要,对数据库结构进行修改。
- 对用户进行登记并维护系统安全。
- 保证与 Oracle 的许可协议相适应。
- 控制和监控用户对数据库的访问。
- 计划对数据库的信息进行备份和恢复。
- 维护归档在磁带上的数据。
- 备份和恢复数据库。
- 与 Oracle 公司联系,取得技术支持。

在理想的结构中,数据管理员直接向总监或者其他高级经理汇报工作。这个 DA 是“不受别人支配的”,而不像海军舰船上那些办公室执行官。同样数据库管理员也应当向这一管理层的经理人递交类似报表。DA 和 DBA 需要紧密合作以确保数据库的物理和逻辑结果通过数据字典紧密地连结在一起。数据库管理员同时还兼有数据库管理的职能,这也可能是一个专门的职位。下一个级别是应用程序管理人员,他们控制单个的应用程序。接下来是开发和维护团队的应用程序管理人员,他们也可能是独立董事。这种组织结构图如图0-7所示。如果数据库足够大,特别是在 Oracle9i 中,由于其复杂的安全性要求,可能还需要一名专门的数据库安全管理人员。

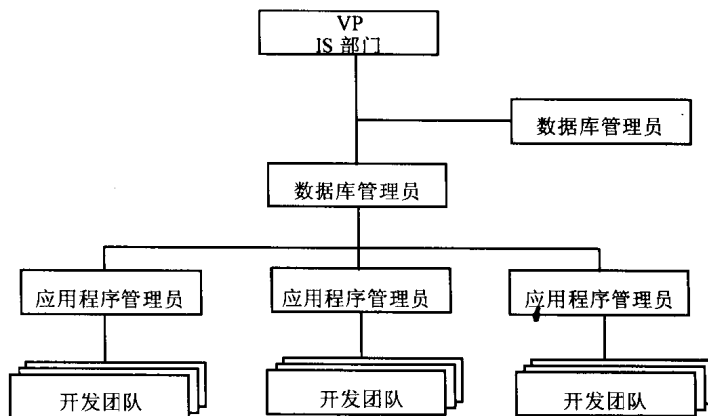


图0-7 信息系统数据管理组织结构图表示例

DA 和 DBA 之间应当多交流,数据库管理者(Database Manager, DBM)和 DBA 必须亲密合作以对数据库系统进行优化。应用程序管理者应当保证其应用程序符合命名规则并同 DBM 和