Oracle 技术丛书

Oracle高效设计

Effective Oracle by Design

Design and Build High-Performance Oracle Applications

(美) Thomas Kyte 著

钟 鸣 郝玉洁 等译

机械工业出版社
China Machine Press

Oracle 技术丛书

Oracle高效设计

Effective Oracle by Design

Design and Build High-Performance Oracle Applications

(美) Thomas Kyte 著

钟 鸣 郝玉洁 等译

机械工业出版社
China Machine Press

本书对 Oracle 及数据库的知识进行了全面深入的讲解，是一本关于 Oracle 的高级手册。本书从开发应用程序的正确方法角度，讲述 Oracle 的重要概念和特性，包括：性能工具包，体系结构选择，语句处理，从基于成本的优化器中获得尽可能多的信息，故障排除等内容。还包括如何编写好的 Oracle 应用程序所涉及的关键问题：高效的管理，高效的设计模式，高效的 SQL，高效的 PL/SQL 程序设计。附录给出了设置和很多常用的脚本。本书内容翔实，实例丰富，语言流畅且浅显易懂，适合作为从事 Oracle 开发人员的参考手册。

Effective Oracle by Design : Design and Build High-Performance Oracle Applications(ISBN 0-07-223065-7).

本书版权登记号：图字：01-2004-3283

图书在版编目 (CIP) 数据

Oracle 高效设计/(美)凯特(Kyte,T.)著；钟鸣等译. －北京：机械工业出版社，2006.1
(Oracle 技术丛书)
书名原文：Effective Oracle by Design:Design and Build High-Performance Oracle Applications
ISBN 7-111-17811-4

Ⅰ.O…　Ⅱ.①凯…　②钟…　Ⅲ.关系数据库－数据库管理系统，Oracle－程序设计
Ⅳ.TP311.138

中国版本图书馆 CIP 数据核字(2005)第 132259 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：隋　曦　　索津莉
北京诚信伟业印刷有限公司印刷 · 新华书店北京发行所发行

2006 年 1 月第 1 版第 1 次印刷
787mm×1020mm 1/16 · 31.75 印张
印数：0 001-4 000 册
定价：68.00 元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换
本社购书热线：(010)68326294

译 者 序

本书是一本关于 Oracle 的高级手册，书中对 Oracle 的许多重要概念和特性进行了独到的讨论。它面向的是那些已经知道如何输入 SQL 语句、如何使用 SQL * Plus 的 Oracle 使用人员和开发人员。书中讲授了如何编写“好的”SQL，以及如何编写“好的”Oracle 应用程序。

本书共 10 章及 1 个附录，各章及附录的内容如下。

第 1 章　构建应用程序的正确方法：讨论关于 Oracle 设计的最好方法的某些“软”问题。这些问题与系统设计、开发、测试、部署和维护有关，但它们并不是 Oracle 所特有的，只不过是从 Oracle 的角度提出而已。这一章是本书所有章节中涉及技术细节最少的一章，但可能是最重要的一章。

这一章中还对 Oracle 的各种文档进行了排序和简介，给希望进一步钻研 Oracle 的各类人员（如编程人员、DBA 等）提供了需要阅读的书籍清单。

第 2 章　性能工具包：作者介绍了自己使用的工具并对为何使用这些工具做了一个概述。另外，还阐述了如何使用这些工具。

第 3 章　体系结构选择：介绍宏观层次上的体系结构选择，它可以帮助用户获得 Oracle 能够达到的可伸缩性和性能。该章不深入探讨 Oracle 服务器进程体系结构及内存结构，不介绍如何建立分区或并行处理，而是将主要精力集中在怎样或何时使用 Oracle 的这些特性上。

该章还讨论了 Oracle 体系结构对应用系统的作用，回答了“如何连接到数据库”、“分区能否使数据库运行得更快”等问题。

第 4 章　高效的管理：介绍作者认为对数据库管理非常有用的一些特性。如 SPFILE、OMF、备份与恢复、字典管理的表空间（DMT）和本地管理的表空间（LMT）、ASSM、让 Oracle 自动管理回退段等。

第 5 章　语句处理：完整地介绍了 Oracle 怎样处理语句，详细讨论了主要语句类型（DDL、DML）的分析、优化和执行，包括查询语句的分析、优化和执行。讨论了绑定变量及其使用。最后，强调要尽可能地避免对语句进行分析。

第 6 章　从基于成本的优化程序获得最大输出：讨论了 Oracle 中两个不同的优化程序，说明为什么 CBO 目前值得重点考虑，特别是在 Oracle9i Release 2 之后的版本不再支持 RBO 的情况下更是如此。

该章讨论了两个对 CBO 影响最大的重要参数，它们分别是：OPTIMIZER _ INDEX _ COST _ ADJ 和 OPTIMIZER _ INDEX _ CACHING。该章中还介绍了影响 CBO 的许多其他参数，并举了许多例子。最后，介绍了 10 053 跟踪事件，这是一个能帮助我们理解优化程序所做事情的工具。

第 7 章　高效的模式设计：应用系统将依赖其物理实现而生存或死亡。选择错误的数据结构会使性能达不到要求，灵活性受到限制。选择正确的数据结构，将会获得良好的性能。该章将考察设计模式时需要考虑的一些东西，然后集中精力讨论表类型、某些有用的索引类型以及压缩等内容。

第 8 章　高效的 SQL：讨论提高查询效率的各种 SQL 技术，并指出，要想开发高效的 SQL，必须对一些较深层次的 SQL 技术有所了解，如访问路径、连接、模式等。另外，为了开发高效的 SQL，深刻理解要解决的问题是至关重要的。

第 9 章　高效的 PL/SQL 程序设计：讨论为什么应该考虑在应用程序中使用 PL/SQL。然后介绍一些能提供高效、高性能 PL/SQL 代码的重要编程技术。

第 10 章　故障排除：介绍一些故障排除的经验和原则。

附录　设置和一些脚本：给出本书许多例子中都用到的 BIG _ TABLE 的建立脚本，以及作者经常使用的一些脚本（实用程序）。

参加本书翻译的主要成员有：钟鸣、郝玉洁。全书由刘晓霞同志审校。同时担任部分翻译及校对工作的还有常征、石永平、王君、张文、魏允韬、田晓涛、耿娜、何江华、梅刚、孙登峰、樊伟、李安娜、李晓军、苏秀玲、赵彦萍、马永良、张启斌等。

由于译者水平有限，书中难免有错误或不当之处，敬请读者批评指正。

译　者

2005 年 11 月 24 日

前　言

本书读者群为Oracle的开发团队，即对整个系统性能有100%控制权的团体。这个团队包括数据建模人员、程序开发人员和DBA。一个流行的神话是DBA负责应用系统性能的方方面面，并且能完全控制系统的性能。可以用赛车来说明这种老观念的谬误。DBA好比赛车场修理处人员，他负责换轮胎，保证给车加满油并让它正常工作。如果你给这位赛车场修理处人员（DBA）一辆Lincoln Navigator（一种极为巨大的卡车），并告诉他你打算用这车去参加方程式大赛，结果会怎样呢？DBA可以保证这辆卡车跑得尽可能快，但他不能让这辆卡车在急转弯时还能保持每小时100多公里的速度。除非把车扔了重新设计制造，否则一旦车辆设计制造出来，DBA能做的事就很少了（这里的车就是应用程序）。

本书主要面向对如何用Oracle设计和建造具有可伸缩性的系统还没有把握的读者群。本书不是一本初学者指南。它是针对那些已经知道如何输入SQL语句、如何使用SQL*Plus的Oracle开发人员的。我们讲授如何编写“好的”SQL，但不讲授SQL基础知识。讲授如何编写“好的”Oracle应用程序，而不讲授如何编写应用程序。

再举一个例子来说明本书将会提供什么信息。假如开发人员是医生，应用程序是病人。有多种医生：

- 急诊室（ER）医生　这些医生对患者进行“筛选”，区分绝症病人与可以救治的病人，在路上进行急救保持病人尽可能长时间存活。他们将会收到吸烟、具有不良饮食习惯又不锻炼的心脏病人，并且要稳定这些病人的病情。
- 手术室（OR）医生　病人在经过筛选且ER医生做过临时性处理后，由OR医生接收。OR医生经过艰苦的手术不仅要挽救病人的生命，而且还要尽可能使他恢复正常。他们给心脏病人做搭桥手术，尽量清理病人的动脉。
- 物理治疗专家（PT）　在OR医生完成手术后，病人就开始了漫长且痛苦的康复过程（不必说花费昂贵），这时就是PT的事了。
- 预防医生　这些医生尽量使病人不打扰上述三种医生。他们劝病人停止抽烟、吃有益于健康的食品、加强锻炼，并且编制一个有很多步骤的程序帮助他们保持良好状态。如果他们工作得当，除非意外事故（比如车祸），否则病人将永远不会再打扰ER、OR或者PT医生。

目前还需要各种医生，因为确实会发生事故。但是，最重要的医生或许是最后面的这种预防医生，他们尽力让病人不需要其他三种医生。

本人相信（实际如此），大多数人和书最关注前三种医生。他们极相信“英雄”开发人员——ER或OR医生。或许正因为这一点，良好的设计和实现一般是吃力不讨好的事情。ER和OR把病人从死亡的边缘拉了回来，获得了所有的声誉（做一些奇迹般的事情，挽救系统也是这样）。他们在最后一刻被召集起来，努力挽救病人的生命，并且得到丰厚的报酬；另一方面，物理治疗专家就不那么幸运了，他们接手ER/OR医生处理过的系统，负责维持其运转。

我完全可以站在ER的立场说话，因为事实上我就是那些被召集起来并使系统好起来的“英雄”中的一员。我可以撰写这样一本书，确实，也有人曾经告诉我，应该写这样一本书，但我没有写。

因为我们所缺少的是包括预防医生培训在内的综合方法。这方面的书籍有一些，我喜欢的有Guy Harrison的《*Oracle SQL High Performance Tuning*，2nd *Edition*》（Prentice Hall，2001）和Jonathan Lewis的《*Practical Oracle 8i Building Efficient Databases*》（Addison Wesley，2001）。这些书籍，包括我自己的《*Expert One on One Oracle*》（Wrox Press，2001），致力于消除对英雄的需求。请注意，消防人员在救火时是英雄，但我们都希望永远不要和他们打交道。

本书作为一个指南，提供了进行性能调整的所有结构和方法，讨论的内容包括：

- 开始设计前的调整。
- 针对特定性能目标进行设计，并不断测试。
- 反复试验（保证每个功能项按要求工作，并了解软件的性能，许多性能问题与不了解数据库如何工作有直接的关系）。
- 实际的事故。研究 ER/OR 医生的角色，但这不是本书的最终目标之一。因为你所学到的作为一个预防医生的知识应该有助于限制那些更光彩的医生角色。

大多数书籍的重点完全放在如何当英雄上，本书则不然，本书的重点在于如何成为一个可靠的系统生产者。它着重建造一个完好的系统而不是修补破碎的系统。我们相信，时间会证明建造完好系统的人才是真正的英雄。

目　录

第 1 章　构建应用程序的正确方法

本章将讨论关于Oracle设计的最好方法的某些“软”（softer）问题。这些问题与系统设计、开发、测试、部署和维护有关。它们并不是Oracle所特有的，只不过是从Oracle的角度提出而已。这些问题实际上是每个系统（不管是不是软件系统）都有的。这一章是本书所有章节中涉及技术细节最少的一章，但可能是最重要的一章。很多一犯再犯的错误实际上是规程性而非技术性的。下面试举几个：

- 在开发人员与数据库管理员（DBA）之间垒起一道高墙。本人将分析这种做法的可能结果，提出有助于建立更具建设性的关系的忠告。
- 认为测试环境太昂贵的决策。事实正相反，在长期运行中，没有测试环境代价更高。
- 未充分利用数据库提供的工具。这可能是由于不懂，或许是基于保持独立于数据库的需求，或者可能是出于恐惧或守旧；这通常会导致应用程序开发时间比所需的长，并且不能达到开发人员的希望。

本章将着重讨论这些问题以及相关的内容。

1.1 团队协作

团队协作确实与技术或软件无关。它完全是人员之间的交流。我们在软件开发中遇到的许多问题主要与策略而不是技术有关。我曾经多次看到开发受阻于策略及规程性问题而非技术挑战问题，有多少次我都说不清了。

“我们与他们”、数据库开发团队人员与给他们提供技术支持的DBA人员之间的关系经常被描述成这样，很发人深省。在这种氛围下，DBA常常感到有必要保护数据库，以免它受到开发人员的损坏。另一方面，开发人员也感到他们需要以某种方式对抗DBA，以实现特定的系统功能。有时，为了尽量让这两伙人配合工作，我感到与其说自己是一个现场数据库专家还不如说自己是个媒婆。

必须注意，协作是双方的事情。开发人员常常感到DBA为了证实开发人员需要某种权限或可以使用某种功能而给开发人员增加了许多负担。事实上，DBA这样做是有充足理由的。授予数据库权限应该经过深思熟虑。DBA要求开发人员说明为什么需要某种权限（比方说CREATE VIEW权限）是完全合理的。稍后我们还要对此作深入的讨论。将某些数据库功能（如数据库视图、存储过程或触发器）全权授予他人是不合理的。

下面是一个例子（来自本人的AskTom网站）：

“存储过程——它们有害吗？

使用存储过程有什么缺点？有开销吗？本人是一个Oracle DBA初学者。有一位SQL程序设计员要求我授予他CREATE PROCEDURE系统权限……”

在我的回答中，解释了存储过程的好处，说明存储过程能够怎样帮助DBA调整应用系统（不需要深入研究应用系统），并强调存储过程是一种非常好的安全机制、一种非常好的提高性能的工具，而且也是一种非常好的程序设计工具。我建议该DBA允许使用存储过程，但只允许开发人员将不良的SQL放置在自己的客户机应用程序中。此外，我还指出，如果SQL位于存储过程中，则DBA至少应该能检查它，并帮助调整和管理它。

反响很强烈，这证明在开发人员阵营与DBA阵营之间存在一条鸿沟。DBA们的看法是，“保证数据库安全是我们的工作。”开发人员的看法是，“编程是我们的工作，允许我们做那件事是他们的工作。”一位开发人员表达的观点是，检查进入数据库的每行代码是DBA的工作。一位DBA反驳这句话，说，事实上这是开发人员的工作。如此等等，争论还在继续。

事实上，要DBA审查进入数据库的每个代码行是不可能的，因为DBA一般不是开发人员，不需要了解源代码。

显然，DBA和开发人员常常互相感到自己具有完全不同的目的，但事实并非如此。DBA不仅仅保护数据库，也不仅仅服务于开发人员。

抱有“保护数据库免遭开发人员损坏”想法的DBA就像过于溺爱孩子的父母一样，一般不会有好的结果。再说，开发人员编程的目的也不是为了对抗DBA。开发人员有自己的工作要做，而且要尽力做好。双方

的共同目标是建立一个满足用户需求、性能良好、具有所需伸缩性、可维护的功能完善的数据库系统，并且成本尽可能低。除非 DBA 和开发人员互相配合，向着这个共同目标前进，否则取得成功的机会将非常小。如果两类人员之间的观点存在鸿沟，许多目标就不能实现。

DBA 与开发人员的作用

通常，DBA 对数据库及其如何工作的知识比开发人员更丰富，而开发人员的软件开发知识要比 DBA 更丰富一些。这是由他们的工作性质决定的。

DBA 一般负责掌握数据库的体系结构，知道如何修补数据库，了解数据库如何工作。为了完成特定工作需要对数据库体系结构有清晰的了解，否则就不能成功地进行备份和恢复。如果 DBA 不理解数据库控制文件、数据文件及重做日志之间的体系结构关系，在备份和恢复中就会经常犯错误。DBA 的使命是与数据库连在一起的。

开发人员一般是程序设计人员或分析人员，他们仅把数据库当作一种工具，一种达到目的的手段。在许多情形下，他们把大量的精力花在“非数据库”的工作上，如界面设计等。

DBA 与开发人员需要某种程度沟通。如果两类人员能够配合工作，则开发人员会逐渐更多地了解数据库，而 DBA 也能更好地促进开发过程。

我在这里列出了两个应该做什么和不应该做什么的清单，一个针对 DBA，另一个针对开发人员。这有助于澄清一些非常有害的观点。

1. DBA 应该做什么和不应该做什么

DBA 不应该认为自己的工作就是保护数据库免遭开发人员的损坏。数据库是开发人员的工具，DBA 能够也应该帮助开发人员更好地利用数据库，而不是企图保护数据库免遭开发人员的损坏。

没有正确的推断就不要限制某些特性或功能。通常，有的推断是出于恐惧、不了解或消极的经验。下面是一些常见的推断：

- 不允许使用视图。原因是 DBA 曾经有过由于视图而导致性能低下的经历，因此认为视图有害。如果你经常这样推断，在遇到性能不良的查询时最终会连 SQL 也不允许用。
- 不允许使用存储过程。这确实让我糊涂了，因为 DBA 应该非常希望所有东西都在存储过程中。如果这样，他们就能确切知道何种模块依赖于何种对象。假如问题是由 SQL 引起的，就能很方便地调整性能不好的模块（只需从数据字典中读出代码并处理它们即可）。假如某天 DBA 需要调整一个 Java 2 企业版（J2EE）的应用程序，因为他不是开发人员，不会用 J2EE 编程，就不知道从何处下手。如果应用程序的数据库部分位于数据库中，调整数据库访问就很容易了。
- 不允许使用版本 6 之后的特性。这在老 DBA 中很常见，他们排斥新东西，排斥 PL/SQL、触发器、应用环境、细粒度的审计、细粒度的访问控制等。他们希望数据库只允许读取（如果可能的话，甚至不允许连接）。我怀疑他们根本不想负任何责任。如果所有开发人员都只使用简单的数据操纵语言（DDL），DBA 的工作就是微不足道的了。这对公司是一种很大的损失，因为它为数据库付出了大量的金钱而不能充分利用数据库。
- 不允许采用 N 版本的新特性。深层次的想法是让所有人用某个旧特性处理问题，并说，我们将在三四年内一直应用这个特性。问题是在这三四年内，你可能会损失很多东西，因为一个新特性可能会减少成小时、成天计的工作量（从而节省金钱）。本地管理的表空间（Oracle 8i 版本 1 中引入）就是一个很典型的例子。这个特性有很多好处，但许多 DBA 不允许使用。理由从来都是非技术性的：他们害怕（恐惧，不自信，怀疑）。

“我从资料上知道本地管理表空间的优点，而且主动问过比我资格老的 DBA，是否对于新的数据仓库可将表空间改为本地管理的。他说本地管理的表空间还有性能问题……”

他说的不错，是有性能问题，但全都是对性能有正面影响的“问题”。DBA 过去使用的表空间（字典管理

的表空间）对性能具有负面的影响。见闻不广才会拒绝新特性。

不过，在感到某个特性或功能多么独一无二、多么酷以至于迫不及待要使用它时，也不要变成“特性痴迷者”。例如，数据库中的可扩展标记语言（XML）功能，有 XML 并不表示任何东西都应该以 XML 来存储。它只不过是一项特性，可以根据需要取舍。

DBA 应该做的事：

- 将开发人员视为可向其传授数据库知识的人员。经过一段时间，你会发现开发人员实际上也想把事情办好。如果你教给他们正确的方法，教他们如何找出适合自己的正确方法，他们就会采用这些方法。人人都希望把事办好，但需要指导。你应该传播自己的知识。
- 对新的特性进行评价和测试，不要错过它们。不要由于某些新特性不完善而一概否定所有新特性。任何事物都有其适用和不适用的地方，新特性也不例外。或许新特性的所谓“问题”是由于你使用不恰当而引起的。铁锤用来钉钉子是好工具，但用来钻孔就不行了。
- 用具体的事例来支持你的规定和手续。不要总是说，“我曾经听人说它很慢”，或者“我听说有缺陷”。不要相信道听途说。这与相信某些性能调整神话的性质完全相同，比方说相信“如果你获得 99.9% 的高速缓存命中率，你的工作就不会有问题”，或者相信“一个表应该在一个区中”。本书将用实际的例子来支持所提出的观点。应该少信点别人的话。

2．开发人员应该做什么和不应该做什么

对于开发人员：

- 不要企图绕开 DBA；应该与他们配合工作。如果你提出具体的修改意见并有充分的理由，他们没有不听的。如果你想干什么就干什么，不和他们商量，他们当然要排斥你了。
- 不要假想 DBA 要和你作对。很多时候，一定的规定和手续是有道理的。必须照章办事，有不同意见可以提出来，但不要乱放炮。
- 请 DBA 告诉你为什么。如果你建议使用本地管理的表空间，但老资格的 DBA 说不能使用，因为有“性能问题”，那么可以问问他是什么样的性能问题。并解释说你想知道为什么本地管理的表空间会有这些不好的性能问题（顺便说一下，没有不好的性能问题）。
- 知道你自己在说什么。否则，别人会对你的话打折扣。采用恰当的语气说话：从某个合理的假设开始。构造一个试验来证明（或反驳）。保证你的试验足以证明你的观点，并且具有可重复性。让别人重复你的试验。准备好结果并让别人来评论它。这样，你在说明自己的方法是正确的时候，就会立于不败之地了。

简而言之，我认为，DBA 与开发人员需要作为一个整体配合工作。他们应该坐到一起，互相交流，在规章制度的基础上互相支持。他们不应该认为自己是独立的群体，并且互相推诿。“他们与我们”的想法对建造高速、可靠、可用的系统极为有害。开发人员和 DBA 与其相互隔离，还不如团结协作、取长补短更有好处。

1.2 阅读资料文档

Oracle 数据库带有 100 多本手册（Oracle9i 版本 2 有 108 本），共有 46 000 多页。要搞清楚从何处开始阅读似乎很棘手。事实上，许多人被这么多文档吓住了，只好不去理会这些资料，就像它们不存在一样。其实，只要有人告诉你从何处开始阅读也很简单，这就是本节要做的工作。我们先介绍关键的文档，然后建议哪些是必读的。

“我从你的网站上学到不少东西。你经常提出的观点是阅读《*Concepts Guide*》（概念指南），这对我帮助很大。所以，几个星期前我在笔记本电脑中拷贝了一份 PDF 文档，并读了一遍。我很高兴自己这样做（不久还要再读一遍）。许多以前不理解的东西现在都清楚了，现在，我已经成为许多开发人员和 DBA 的顾问，我知道他们多半没读过这份资料。”

这是我最希望得到的奖品。常常有人向我提以“别给我指出看什么文档资料”为开头的问题。这种要求

我几乎总是置之不理，因为他们的问题已经在文档资料中给出了很好的回答。我常常以“好的，关于这个问题，《*Concepts Guide*》在某节是这样说的……”开始回答。我不止一次说过，如果你从头到尾阅读一下《*Concepts Guide*》，只要记住其中10%的内容，就会掌握大多数人掌握的90%以上的Oracle知识。而且，如果以后再遇到问题，你就可以说，“我记得有这事。我们来看看《*Concepts Guide*》中怎么说。”

1.2.1 指南的指南

虽然有100多份文档，但我在本节中的建议还是非常简单。这里汇集的指南是应该从头到尾阅读的（精读哪些决定于你是开发人员还是DBA）。其余文档可根据需要和兴趣作为参考资料。

1．概念指南

对于每个Oracle版本，《*Concepts Guide*》（概念指南）都是必读的文献。它包含以下一些内容：

- **什么是Oracle** 对数据库、内存结构、分布式数据库、并发控制、数据一致性、安全、管理和其他内容进行介绍。
- **数据库结构** 对数据如何存储的深入介绍。引入段、区、块、表空间等概念。
- **Oracle实例** Oracle实例是什么：其启动与关闭过程，应用程序如何与Oracle实例打交道，内存与进程体系结构怎样，如何管理资源。
- **数据** 模式对象（如表、视图、索引等）、每种对象类型可用的选项、所有数据类型（系统固有的类型与用户定义的类型）的概览。
- **数据访问** SQL、PL/SQL以及Java与数据库的交互，模式对象、事务处理和触发器之间的关系及如何管理和维护。
- **并行操作** 何时进行并行操作（包括并行查询、并行DML、管理的并行操作等）以及如何操作。
- **数据保护** 这或许是最重要的内容了，它包括诸如在数据库内如何并发和一致地工作，何时以及如何强制实施数据完整性，其中如何保证安全，为保护数据可使用什么方法等问题。数据保护还要说明怎样使用权限和角色，如何进行数据审计。

《*Concepts Guide*》的另一好处是可以作为其他文档的“索引”，而且它是完全免费的。通常，各内容以“See Also”结尾，链接到其他文档的相应位置。《*Concepts Guide*》是一个较高层次的概述，是其他文档的元数据文档。对它的学习，会将你引向其他相关的Oracle文档。你可以根据自己的兴趣和时间进行深入研究。

2．新特性指南

如果不知道有什么东西，就不可能会使用它。《*New Features Guide*》（新特性指南）说明相对于前几个版本，本版本有什么新特性。对于每个特性，该指南给出简明扼要的介绍，并指出在何处可以找到更详细的说明。它还包含一份说明每种Oracle版本（个人版、标准版、企业版）包含哪些特性的清单。这份清单可以减少许多麻烦。如果你正在设计一个运行在Oracle标准版上的系统，通过它可以知道你能够使用哪些新特性。在Oracle 8i以及之前的版本中，本指南名为Getting to Know。

此外，现在大多数文档的开始处都含有一节“What's New In”。新特性指南给出了管理、开发、性能、可伸缩性、可用性等方面的新特性。个别文档还给出了更详细的“What's New”清单。例如，《*Administrators Guide*》（管理员指南）给出“What's New in Administration（管理中的新功能）”一节，而《*Application Developers Guide*》（应用开发人员指南）给出“What's New in Application Development（应用开发中的新功能）”一节。

3．应用开发人员指南

实际上，对于各种Oracle新特性，有相当多的以《*Application Developers Guide*》（应用开发人员指南）开始的指南，如《*Advanced Queuing*》（消息软件）、《*LOBs*》（大对象）、《*Object Relational Features*》以及《*Workspace Management*》等。我建议所有开发人员都应该读一下*Fundamentals*指南。这本全面的指南介绍如下内容：

- 开发人员可访问的各种编程环境
- 设计数据库模式
- 利用约束维护数据完整性
- 如何对数据进行索引——应该考虑什么、相应引擎如何处理 SQL 语句、如何使用动态 SQL、如何使用 PL/SQL、如何实现安全性等。

《*Concepts Guide*》只是说明存在这些特性，而《*Application Developers Guide*》说明如何使用它们。

4．PL/SQL 用户指南与参考手册

PL/SQL 是开发人员开发基于 Oracle 的应用程序时的一个最重要的语言。理解它能做什么，不能做什么非常重要。《*PL/SQL Users Guide and Reference*》（PL/SQL 用户指南与参考手册）涵盖了 PL/SQL、错误处理、语法、程序包/过程以及许多与 PL/SQL 相关的基础知识。

5．性能调整指南与参考手册

我每天都要使用并且最喜欢的一本指南是《*Performance Tuning Guide and Reference*》（性能调整指南与参考手册）。这本指南在 Oracle 8i 以及更早的版本中称为《*Designing and Tuning for Performance*》（性能设计与调整）。此指南的前半部分主要针对开发人员，不过也与 DBA 有关。它描述了优化程序具体是如何工作的，如何正确收集统计数据，不同的物理结构如何工作，使用它们的恰当时机（例如，使用索引编排表的恰当时机）。最重要的是，它完整地描述了开发人员需要使用的基本工具，如：Explain Plan、SQL _ TRACE、TKPROF、Autotrace 等，甚至还描述了 Statspack。

本书后半部分全是关于 DBA 工作的，介绍诸如建立高性能数据库、内存配置、理解操作系统交互和资源使用、配置共享服务器与专用服务器、如何收集统计数据、何时及如何使用性能视图、如何使用其他性能工具等内容。这是一本在部署和调整数据库之前必读的指南。即使你对 Oracle 有极为丰富的经验，也会从本指南中学到许多新东西。

6．备份与恢复概念

备份与恢复是 DBA 不容忽略的一件大事。即使你使用某个工具自动备份，即使你自认为已经完全掌握备份知识，也应该仔细阅读《*Backup and Recovery Concepts Guide*》（备份与恢复概念指南）。对备份与恢复工作有深入透彻的理解是 DBA 需要具备的一项基本素质。

仅阅读 RMAN 指南是不够的。这样说是因为许多人向我提出一个问题，他们仅阅读了 RMAN 指南，不理解为什么复原了上周的控制文件后却不能完全恢复数据库。他们对各种文件如何互相配合工作不了解，不知道恢复到给定状态需要什么。除非你想让公司花费更多的金钱，否则就应该读一下备份与恢复概念指南。如果不理解其中的某些内容，应该再次阅读直到弄懂为止。然后做一下试验，看你的理解是否正确。如果不正确理解，恢复时出错是必然的事情。

7．恢复管理员参考手册

《*Recovery Manager Reference*》（恢复管理员参考手册）介绍备份数据库的工具。请忘掉那些陈旧的脚本，扔掉 tar、cpio、dd、ocopy。RMAN 就是你想要的书记员。这个工具的功能（块级、现场恢复、备份保持策略、热备份等）很强，值得学习。

8．管理员指南

当然，我知道 DBA 已经是管理员了，但 DBA 阅读《*Administrators Guide*》（管理员指南）还是能知道一些新东西的。从这本资料中你可能第一次发现，数据库中还有一个资源管理器（8i 中的新功能，9i 中进行了增强）。或许，你会学到细粒度的审计（Oracle 9i 中的新功能）。我敢保证，阅读此指南你一定能学到某些新东西。

与数据库管理有关的某些 Oracle 新特性只能在本文档中才能找到。它们是一些“通用程度”不足以放入《*New Features Guide*》（新特性指南）中的特性。本指南列出了所有与数据库管理有关的新特性，这些新特性

不会与别的新特性混淆。

1.2.2 阅读路线

这里给出一个阅读路线，说明必读的资料。提供三类资料，分别是：针对所有人员的资料，针对 DBA 的资料，针对开发人员的资料。

1．开发人员与 DBA 必读

对于这两类人员，需要阅读以下资料：

- 《*Concepts Guide*》
- 《*New Features Guide*》

2．开发人员必读

对于开发人员，还需要阅读以下资料：

- 《*Application Developers Guide*（*Fundamentals*）》
- 《*PL/SQL Users Guide and Reference*》
- 《*Performance Tuning Guide and Reference*》（对于 8i 或更早的版本，称为 Designing and Tuning for Performance Guide）

如前所述，开发人员应该阅读《*Performance Tuning Guide and Reference*》的前半部分，并酌情阅读其他部分。对于 Oracle 9i 及以后的版本，在阅读此文档前，应考虑先阅读《*Performance Method*》（9i 版本 1）或《*Performance Planning*》（9i 版本 2）。《*Performance Tuning Guide and Reference*》描述了必须理解的内容，如可伸缩性、系统体系结构、应用设计原理等。

3．DBA 必读

在阅读了《*Concepts Guide*》和《*New Features Guide*》后，DBA 还应该阅读：

- 《*Backup and Recovery Concepts*》
- 《*Recovery Manager Reference*》
- 《*Backup and Recovery Concepts*》（这不是排版错误，这里重复给出是因为它非常重要。备份和恢复一件不允许出错，但 DBA 又最容易搞错的事情。必须阅读并理解它。）
- 《*Administrators Guide*》
- 《*Performance Tuning Guide and Reference*》（重点在后半部分）

4．推荐阅读

在阅读了上述指南后，可根据需要选择读物。如果你是需要处理 XML 的开发人员，有不少于 3 种 XML 指南可读。如果你对 Java 感兴趣，也有这方面的指南。如果你是 DBA，需要了解怎样建立和配置失败切换，也可以找到相应的手册。

随着对这些资料的阅读和钻研，你会发现它们非常不错。并不是我为 Oracle 工作才这样说，而是因为我的大多数 Oracle 知识都是来自这些资料。我确实阅读了《*Concepts Guide*》。我撰写“新特性研讨”的方法就是钻研《*New Features Guide*》。建议读者访问 http://otn.oracle.com，并点击文献链接，阅读这些文献。

1.3 避免黑盒综合症

没有 Oracle 数据库的基本知识，对它怎样工作不了解，开发人员常常会采用错误的方法，并且容易犯错误。许多时候，人们使用数据库就像它是一个黑盒子一样，把它当成像收音机中的电池那样的可换日用品。由于怀有这种观点，许多人使用数据库，但尽可能地避免依赖数据库，就像数据库是某种不好的东西一样。他们以数据库独立的名义，拒绝利用特定数据库的特性，拒绝扩展其功能。这实际是选择了取消而不是利用公司为之付了金钱的大多数功能。这表示，他们采取了自己编写更多代码的办法，这需要做更多的维护工作，并且比从市场上