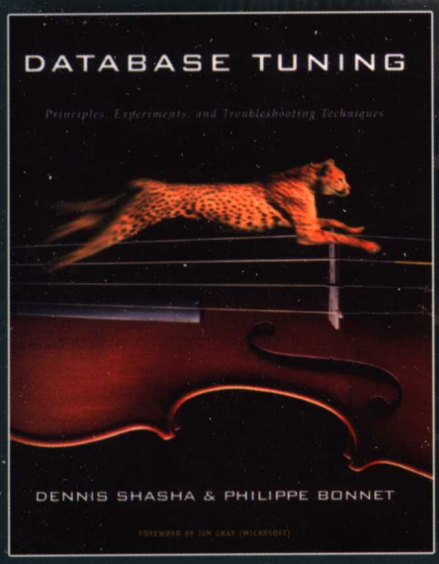
国外计算机科学教材系列

数据库性能调优

——原理与技术

Database Tuning

Principles, Experiments, and Troubleshooting Techniques



[美] Dennis Shasha
Philippe Bonnet 著
孟小峰 李战怀 等译

MORGAN KAUFMANN
電子工業出版社
Publishing House of Electronics Industry
http://www.phei.com.cn

国外计算机科学教材系列

数据库性能调优

——原理与技术

Database Tuning

Principles, Experiments, and Troubleshooting Techniques

[美] Dennis Shasha
Philippe Bonnet 著

孟小峰 李战怀 等译

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内容简介

数据库的性能优化远不是按照厂家所列的有关指南通过短短的几步就可以达到的。要获得最大的优化效果，需要有广泛而深入的基本调优原理的知识，有按系统的方法搜集数据的能力，以及让系统运行更快速的技巧。这既是一门艺术，也是一门科学，而本书将帮助读者获得这方面的技能，从而可以达到对基于各种硬件和操作系统之上的各种数据库系统进行调优的目的。而且，这些技术辅以书中的有关实验结果，可以帮助读者准确评估有关的数据库产品并做出正确的选择。

本书可以作为高年级本科生和研究生的教材，对从事数据库研究和开发的科研人员和工程技术人员也是一本难得的参考书。

版权贸易合同登记号　图字：01-2003-0376

图书在版编目（CIP）数据

数据库性能调优：原理与技术/（美）沙沙（Shasha, D.）等著；孟小峰等译.
-北京：电子工业出版社，2004.5
（国外计算机科学教材系列）
书名原文：Database Tuning: Principles, Experiments, and Troubleshooting Techniques
ISBN 7-5053-9888-1

Ⅰ.数...　Ⅱ.①沙...　②孟...　Ⅲ.数据库系统－教材　Ⅳ.TP311.13

中国版本图书馆 CIP 数据核字（2004）第 039002 号

责任编辑：谭海平　许菊芳
印　　刷：北京智力达印刷有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编：100036
经　　销：各地新华书店
开　　本：787 × 1092　1/16　　印张：17　字数：435 千字
印　　次：2004 年 5 月第 1 次印刷
定　　价：28.00 元

凡购买电子工业出版社的图书，如有缺损问题，请向购买书店调换；若书店售缺，请与本社发行部联系。联系电话：（010）68279077。质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

出 版 说 明

21世纪初的5至10年是我国国民经济和社会发展的重要时期，也是信息产业快速发展的关键时期。在我国加入WTO后的今天，培养一支适应国际化竞争的一流IT人才队伍是我国高等教育的重要任务之一。信息科学和技术方面人才的优劣与多寡，是我国面对国际竞争时成败的关键因素。

当前，正值我国高等教育特别是信息科学领域的教育调整、变革的重大时期，为使我国教育体制与国际化接轨，有条件的高等院校正在为某些信息学科和技术课程使用国外优秀教材和优秀原版教材，以使我国在计算机教学上尽快赶上国际先进水平。

电子工业出版社秉承多年来引进国外优秀图书的经验，翻译出版了“国外计算机科学教材系列”丛书，这套教材覆盖学科范围广、领域宽、层次多，既有本科专业课程教材，也有研究生课程教材，以适应不同院系、不同专业、不同层次的师生对教材的需求，广大师生可自由选择和自由组合使用。这些教材涉及的学科方向包括网络与通信、操作系统、计算机组织与结构、算法与数据结构、数据库与信息处理、编程语言、图形图像与多媒体、软件工程等。同时，我们也适当引进了一些优秀英文原版教材，本着翻译版本和英文原版并重的原则，对重点图书既提供英文原版又提供相应的翻译版本。

在图书选题上，我们大都选择国外著名出版公司出版的高校教材，如Pearson Education培生教育出版集团、麦格劳－希尔教育出版集团、麻省理工学院出版社、剑桥大学出版社等。撰写教材的许多作者都是蜚声世界的教授、学者，如道格拉斯·科默（Douglas E. Comer）、威廉·斯托林斯（William Stallings）、哈维·戴特尔（Harvey M. Deitel）、尤利斯·布莱克（Uyless Black）等。

为确保教材的选题质量和翻译质量，我们约请了清华大学、北京大学、北京航空航天大学、复旦大学、上海交通大学、南京大学、浙江大学、哈尔滨工业大学、华中科技大学、西安交通大学、国防科学技术大学、解放军理工大学等著名高校的教授和骨干教师参与了本系列教材的选题、翻译和审校工作。他们中既有讲授同类教材的骨干教师、博士，也有积累了几十年教学经验的老教授和博士生导师。

在该系列教材的选题、翻译和编辑加工过程中，为提高教材质量，我们做了大量细致的工作，包括对所选教材进行全面论证；选择编辑时力求达到专业对口；对排版、印制质量进行严格把关。对于英文教材中出现的错误，我们通过与作者联络和网上下载勘误表等方式，逐一进行了修订。

此外，我们还将与国外著名出版公司合作，提供一些教材的教学支持资料，希望能为授课老师提供帮助。今后，我们将继续加强与各高校教师的密切联系，为广大师生引进更多的国外优秀教材和参考书，为我国计算机科学教学体系与国际教学体系的接轨做出努力。

电子工业出版社

教材出版委员会

译 者 序

数据库技术发展到现在已有近40年的历史，现代数据管理技术与其上应用的结合越来越紧密。相应地，数据库系统的维护成为信息系统管理的瓶颈。作为一个高级的系统软件，数据库管理系统涉及到查询处理、并发控制、故障恢复等关键技术，同时伴随产生了重要的方法学——数据库设计（包括逻辑设计和物理设计）以及新的应用技术如数据仓库等。如何让这样的“复杂”系统高效地运转，为用户提供满意的服务，不是一件简单的事情，应当说是一门学问，这就是本书所探讨的数据库调优技术。

数据库的性能优化远不是按照厂家所列的有关指南通过短短的几步就可以达到的。要想获得最大的优化效果，既需要具有广泛而深入的数据库原理和系统实现知识，又要有扎实的应用设计能力，同时要充分熟悉操作系统和有关的软硬件环境。因此这既是一门艺术，也是一门科学，而本书将帮助读者获得这方面的技能，从而可以达到对基于各种硬件和操作系统之上的各种数据库系统进行调优的目的。而且，这些技术辅以书中的有关实验，这些实验均建立在当前最流行的三种产品（Oracle，IBM 的 DB2 和 Microsoft 的 SQL Server）之上。有关实验结果既会“令系统实现人员大吃一惊”（Jim Gray），也可以帮助读者准确评估有关的数据库产品并做出正确的选择。

本书作者 Dennis Shasha 博士最初写的“Database Tuning：A Principled Approach”（1992，Prentice Hall）是一本专业参考书，也是本书的前身。另一作者 Philippe Bonnet 博士是一位年轻的数据库学者，曾开发过数据库系统 Predator。2002 年 Philippe Bonnet 博士结合本书的内容分别在香港的 VLDB 2002 和郑州的 NDBC 2002 上做了辅导报告，得到与会者的好评。也正是在那个时候，译者认识到本书的价值，并打算翻译成中文。当时与 Philippe Bonnet 博士交换意见后，得到他的极大支持。随后译者在翻译过程中尝试将本书的内容用于教学，在有关讲义和实验方面得到 Philippe Bonnet 博士的大力支持。在此表示感谢。

全书共分 10 章，5 个附录，术语表和索引。本书的翻译、统稿和审校由孟小峰和李战怀共同组织完成。具体翻译分工如下：冯月利、董兵兵（第 1 章，第 5 章）；蒋瑜、于峻涛、韩昆、张小谷、胡志智（第 2 章，附录 B）；陈妍、胡东东、陆世潮（第 3 章）；易蕾、张宁、安靖（第 4 章，附录 D）；赵益宇、秦国（第 6 章，附录 C）；蔡曦、郭研妍、许彦、周芝芝（第 7 章，第 8 章）；李娜、王韵婷、姚佳丽（第 9 章，第 10 章，附录 A）。以上人员完成初稿的翻译。之后，进行了重译，具体分工如下：王明钟、闫剑锋（第 1 章，第 7 章）；张阳、王彦龙（第 2 章）；闫剑锋、张阳（第 3 章，第 4 章）；王勇、张阳（第 5 章，第 6 章）；徐娟、张龙波、陈世亮、孙芳（第 8 章，第 9 章，第 10 章）；董冬梅、闫剑锋（附录 A，附录 B，附录 C，附录 D）；王明钟、闫剑锋（附录 E）；张阳、闫剑锋（术语表）；蒋芸（前言，作者简介，封底等）。全书最后由孟小峰和李战怀负责统一定稿。

本书涉及面广，内容丰富，术语量大。本书译词主要遵从教科书中的习惯用法，并参考《英汉计算机词典》（修订本，1998）等。由于译者水平有限，译文中不当之处在所难免。诚恳读者批评指正并不吝赐教。如果你有任何建议或意见，欢迎发 E-mail 给 xfmeng@public.bta.net.cn，lizhh@nwpu.edu.cn。

译者

2003 年 10 月

序

Jim Gray，微软研究院
摩根卡夫曼数据管理系统丛书编辑

每个数据库产品都有关于性能调优方面的指南，紧接着市场上就会涌现出讨论其中细节的书。如果你仔细阅读了所有这些书，然后做了一些实验，并进行了思考，那么你就能写出这本书。本书总结了数据库产品的设计、实现、管理和使用中的基本性能原则。本书的很多案例都采用了当前最流行的三种产品（IBM 的 DB2、Oracle 公司的 Oracle 和 Microsoft 的 SQL Server）来进行精细的实验，从而用实例说明了数据库调优设计的可行性。本书对新手和有经验的用户都适用，其内容大多来源于作者与华尔街的顾客在事务处理、数据仓库和数据分析应用方面合作的经历，这种案例研究使得书中许多实例都是切实可行的。

对初学者而言，本书对所有系统的 SQL 层的逻辑数据库设计的性能问题给出了明智的建议。至少对我而言，物理数据库的设计是尤其有趣的，因为本书给出了在IBM、Oracle和 Microsoft 系统之间进行设计选择的方法。这些系统有很多内在的不同，而此书中的实例甚至会令系统实现人员大吃一惊。它对上述各种系统中的不同设计的相关性能进行了量化。毫无疑问，没有在任何情况下都是最优的系统，每一个系统都有强项，但也不乏弱点。

这本书可以作为一份指南（相关网站为 www.mkp.com/dbtune/），或者作为特定问题的参考。无论如何，本书都是简单和易懂的。对应用开发人员而言，关于事务设计、事务划分和时间序列数据以及关于调优的章节将有着巨大的吸引力。

前　言

本书目标

数据库调优可以使数据库应用运行得更快。“更快”通常指的是更高的吞吐量或者说是更短的应用响应时间。

为了使系统运行得更快，数据库调优人员必须调整应用程序构建的方式、数据库系统的数据结构和参数、操作系统的配置甚至硬件。因此，最好的数据库调优人员能够解决牵涉到广泛的应用和计算机系统知识的问题。

本书的目的是给读者补充以上这些知识，并有以下 3 个目标：

1. 帮助对数据库管理系统、操作系统和硬件之上的应用进行优化。
2. 提供对数据库管理系统进行选择的标准，包括一组用于在特定情况下对系统某些方面进行测试的实验数据和脚本。
3. 讲解解决优化难题所遵循的基本原理。

实现第一个目标的最好办法，是在阅读本书的同时，参考特定系统的优化指南，这两者在许多方面互相补充：

- 本书提供的调优概念可以在不同系统、不同版本间进行移植，而优化指南将会把这些概念与特定的系统和版本特性相结合。
- 本书包含了专业数据库优化师（包括我们自己）的经验和智慧，并且提供了可用的实验案例脚本。因此，本书提供了更多的建议和策略，而这些在数据库系统本身的优化指南中是找不到的。
- 本书比大多数优化指南涉及的范围更广，例如应用程序与数据库服务器的工作分配，事务的设计以及硬件的购买等。

给老师的提示：本书主要是为应用专家编写的，但是也可以在高年级的大学数据库课程中使用。事实上，我们以及我们的一些同事已经在使用这本书的前一版“Database Tuning: A Principled Approach”（1992，Prentice Hall）来授课。

如果学生已经掌握了数据库系统的外部视图的基础、查询语言、面向对象的概念和概念设计，可以有如下的选择：

- 对那些不久将会设计数据库管理系统的学生，最好讲授查询处理、并发控制和恢复的知识，这是传统方法。
- 对那些主要是使用和管理数据库管理系统的学生，最好教授给他们一些调优的基本原则。

其实这两种方法可以很好地相互补充，例如，在传统的方法中会讲授 B-树的实现。而讲述调优时，可以说明针对不同类型的查询，B-树和哈希结构各自的优缺点。再举一个例子，在传统方

法中会讲授并发控制的加锁算法，而在调优方法中，可以讲授在不牺牲串行化的前提下，对事务进行分割以实现高并发性的方法。

我们仅假设读者已经学习过关系查询语言 SQL、数据结构的高年级本科课程，如果可能，还有操作系统的本科课程，因此在写作时尽量不涉及其他的知识。本书还深入地讨论了并发控制、恢复以及查询处理的原理。

如果你正在使用本书作为教材，我们可以提供演讲笔记，请发 E-mail 到我们的信箱 shasha@cs.nyu.edu 或者 bonnet@diku.dk。

能学到的知识

大多数企业中的工作人员发现，购买数据库管理系统通常比从头开发要好。以前的抱怨——“商业系统的性能不能很好地满足应用”——已经让位于新的认识，即数据库管理系统提供的便利（特别是标准的接口、工具、事务功能以及数据结构的独立性）与它的价格是相称的。关系数据库系统通常不能满足数据密集型应用（如搜索引擎、电路设计、金融时间序列分析和数据挖掘）中的表述能力和性能要求。采用关系数据库系统的应用通常有如下特征：大数据量、并发应用的频繁更新以及对容错的需求。

异常处理的应用没有在这里提及，因为它们表现为偶发的大量更新且不考虑容错性。而且，SQL 数据模型仅在事后才关注它们。

关系系统适用于很多常见的应用：关系系统能很好地捕捉聚集，因此它们正从部分销售商那里获得数据仓库的市场。而且，它们已经扩展了事务模型以支持电子商务应用。

关系系统是数据库中如此重要的部分，故本书着重讨论了其中几种主要的产品：DB2，SQL Server，Oracle，Sybase 等。另一方面，本书也探讨了对顺序查找（Fame，KDB）或者主存（TimesTen）进行优化的系统的特征。

在本书中，将有许多相同的原理（结合实例研究代码）被运用到不同的产品中。因此，在一种系统中使用的例子也会在其他系统中出现。

在讨论完对所有应用以及大多数数据库系统都适用的原理后，我们接着讨论针对页面支持、数据仓库、异构系统和金融时间序列的调优策略。本书没有直接讨论诸如 SAP 的企业资源计划系统，但是由于它们是基于关系引擎实现的，故讨论的调优理论也同样适用。

怎样阅读本书

本书使用非形式化的风格，而且尽量对每一项建议都给予解释。偶尔，在定性的推理陷入僵局后，本书会提供大量的由经验而得的方法（有实验支持的），以帮助做出明智的决定。我们鼓励重做实验以得到自己的结论。

本书的网站 www.mkp.com/dbtune 包含了能在各种数据库系统中运行的 SQL 脚本和实验程序以及生成数据的脚本。这个网站还包括每个实验设置以及得到结果的详细信息。

部分实验的结果不是在最新的 DBMS 上得到的，这在一本书中是无法避免的，但是不会影响我们实现自己的目标：不是提供任何针对数据库管理系统所有版本的确切结果，而是指明对性能影响重大的参数。我们鼓励读者使用我们提供的脚本，针对自己感兴趣的版本做实验。

致谢

许多人参与编写了这本书。首先，要感谢优秀的章节作者：Joe Celko（数据仓库）和 Alberto Lerner（性能监视）。其次，要感谢几位提供了中肯建议的技术专家：Christophe Odin（Kelkoo），Hany Saleeb（数据挖掘），以及 Ron Yorita（数据连接）。

本书来源于美国纽约大学和丹麦哥本哈根大学的教学经验以及在贝尔实验室，NCR，Telcordia，瑞士联邦银行，Morgan Stanley Dean Witter，Millenium Pharmaceuticals，Interactive Imagination 等部门的调优经验。我还想向 Martin Appel，Steve Apter，Evan Bauer，Latha Colby，George Collier，Marc Donner，Erin Zoe Ferguson，Anders Fogtmann，Narain Gehani，Paulo Guerra，Christoffer Hall-Fredericksen（帮助实现了用于实验的工具），Rachel Hauser，Henry Huang，Martin Jensen，Kenneth Keller，Michael Lee，Martin Leopold，Iona Lerner，Francois Llirbat，Vivian Lok，Rajesh Mani，Bill McKenna，Omar Mehmood，Wayne Miraglia，Ioana Manolescu，Linda Ness，Christophe Odin，Jesper Olsen，Fabio Porto，Tim Shetler，Eric Simon，Judy Smith，Gary Sorrentino，Patrick Valduriez，Eduardo Wagner，Arthur Whitney 和 Yan Yuan 表达我的感激之情。

还要感谢我们在 Oracle 的同事，特别是 Ken Jacobs, Vineet Buch 和 Richard Sarwal，他们给予了法律和技术上的建议。

还要感谢美国国家科学基金、高级研究项目局、NASA 以及美国和其他国家的基金组织——不是因为它们对编写本书的直接支持，而是它们为本领域的很多研究提供了支持。很多商业技术（层次数据库和位图）以及更多的理论都来源于研究原型。

在 Morgan Kaufmann，同事们共渡了一段美好时光。坚强的资深编辑 Diane Cerra 成功克服了一次错误的开端，并在很多未知的方面显示了她的能力，她还从 Lothlarien Homet 选择了一个合格的继任者。开发和产品编辑 Belinda Breyer 和 Cheri Palmer 在参与本书的编辑时准时、胜任并坚持不懈。这个组的其他成员包括技术编辑 Robert Fiske、校对 Jennifer McClain、封面设计 Yvo Riezebos 和索引员 Ty Koontz，在此一并向他们表示感谢。他们还帮助发现了优秀的审校人员。还要特别感谢 Krishnamurthy Vidyasankar，Gottfried Vossen 和 Karen Watterson，他们的建议极大地改善了本书的质量。

目　录

第1章 基本原理

1.1 原理的作用

调优需要有广泛的常识，这使得它既简单又复杂。

称调优简单，是因为调优者不必纠缠于复杂的公式和法则。许多学术和业界的研究者都在尝试将调优和查询处理建立在数学基础之上。但是，比较复杂的研究由于其依据的假设不现实，往往对实际工作没有帮助；反之，比较简单的研究却有助于对问题的理解，这在下面的章节中会提到。

称调优复杂，是因为如果要完全理解常识所依赖的原理，还需要对应用、数据库软件、操作系统和硬件的物理性能有广泛而深刻的理解。大多数有关调优的书仅提供了由经验得到的方法，却没有提及它们的局限性。

例如，有的书会说：在事务响应时间很关键的时候，不要使用聚集函数（如AVG）。因为这些函数必须扫描大量的数据，可能导致阻塞其他查询。通常来讲，这个规则是正确的，但在对通过索引得到的少量元组求平均值时，此规则便不成立。这说明调优者必须熟知此方法的依据，即访问大部分共享数据的长事务会延迟并发在线事务。优秀的调优者能够正确地运用这个规则，把它当做原理（在高并发度环境下不要扫描大量的数据）的实例，而不是错误地把规则本身当做原理。

本书认为理解了原理就可以更好地运用从经验中学到的方法。而且，理解了原理还有助于发现解决特定问题的新方法。所以，本书从最基本的原理开始讨论。

1.2 五个基本原理

以下五个基本原理渗透着对性能的考虑：

1. 全局考虑，局部修整。
2. 划分打破瓶颈。
3. 启动成本高，运行成本低。
4. 服务器和客户端之间合理的任务分配。
5. 性能价格比。

下面将详细描述每个原理及其相关应用的例子，有些例子可能会用到后面章节和术语表中的术语。

1.2.1 全局考虑，局部修整

只有正确地找出问题，并将对数据库的干涉减到最少，才能有效地进行调优。这就要求进行一些试验以获得正确的结论。以下两个例子描述了常见的错误：

- 全局调优常常先检查硬件设备的统计数据，然后得出处理器的利用率、输入输出活动、页

面调度等数据。如果某项值（如磁盘活动量）过高，初级调优者就会购买新硬件（买更多的磁盘）。但是在许多情况下，这样做并不妥当。例如经常执行的查询不使用索引而直接对表进行扫描，又或者日志文件和常被访问的数据共用同一磁盘，这些都会导致很多的磁盘活动。这时，在不同磁盘间创建索引或者移动数据文件，都可能比购买新硬件更能降低成本且更有效。

数据库管理员没有增加数据库缓冲区的大小而引起许多不必要的磁盘访问，也是导致磁盘活动过多的常见原因。

- 调优者常常会测量某个查询所用的时间，如果比较长就尽量缩短。但如果该查询很少执行，则对它进行调优就有点得不偿失了。例如将仅占运行时间 1%的查询的速度提升两倍，最多可使系统加速 0.5%（该例子同时也说明如果查询是很关键的，就必须尽量提高其效率）。所以，应该首先将问题定位到一个查询上并对它进行优化，但必须保证这是个关键查询。

即使是对特定问题进行处理，也必须考虑全局。开发者可能会要求采用他们指定的查询并“找到使执行尽可能快的索引”。通常，理解应用的目的有助于找到更简单的解决方法，从而把工作做得更好，这意味着要和设计者一起详细讨论所有的事情，就像临床医生一样，应尽力解决隐藏在表面现象之后的问题。

1.2.2 划分打破瓶颈

大多数情况下，系统运行缓慢不是由于所有部件都饱和引起的，而是由于系统中的某个部分限制了整体的性能，这些部分称为瓶颈。

可以把瓶颈想像成高速公路上的交通阻塞，常常是因为大量车辆必须从一个窄道通过，或者因为一条路上的车流要和另一条路上的车流汇合造成的。无论哪种情况，瓶颈就是公路交通网中车辆数与车道数的比率最高的那段路。解决的方法是先找到瓶颈所在，然后采用下面两个策略之一：

1. 让司机在车道数较少的路段开得快些。
2. 多修几个车道以减少每条道上的车辆数，或者鼓励司机避开车流高峰。

第一个策略是局部调整（例如采用增加索引或者重写查询的方法，以便更好地利用现有索引），应该首先考虑。第二个策略意味着进行划分。

数据库系统中的划分将负载分散到更多的资源上执行，或者在时间上并行分配，从而减少某个部件上的负载量。很多情况下划分的确能解决瓶颈问题，下面是一些例子，具体的技术细节会在后文中详细讨论。

- 某银行有 N 个支行，大多数客户通过本地的支行访问自己的账户。如果中心系统过载，最自然的解决办法就是将第 i 个支行的客户账户数据放入子系统 i 中。这就是物理资源的空间划分形式。
- 锁争用通常只涉及少数资源，如数据文件经常争用空闲列表（未用的数据库缓冲页列表）。解决的办法是将这些资源分成块以减少在每个锁上的并发争用。对空闲列表资源而言，这意味着要创建多个空闲列表，每个都包含指向部分空闲页的指针。需要空闲页的线程可随机锁住并访问其中一个空闲列表。这就是临界资源的逻辑划分形式。
- 当一些长事务和许多短在线事务需要同时访问相同的数据时，系统会因为锁争用和资源争

用而导致性能降低。死锁会强制长事务中断，长事务也可能阻塞短事务执行。而且，长事务可能占用大量的缓冲池，此时即使不存在锁争用的情况也会延缓短事务的执行。可行的解决办法是在几乎没有在线事务活动时才执行长事务，并且在负载过重的情况下将它们串行化，这样就不会彼此干扰（时间划分）。另一个解决办法是允许只读长事务使用其他存储设备上的过时数据（空间划分）。

从数学角度讲，划分就是将集合划分为多个两两互不相交的部分。以上 3 个例子（后面许多例子也是如此）分别说明了空间、逻辑资源和时间上的划分。遗憾的是，划分并不一定能提高性能。例如，通过支行来划分数据会增加额外的跨行事务通信开销。

所以，像大多调优一样，划分必须谨慎使用。本节的主旨很简单，那就是：发现瓶颈时，首先尽量加速它所在的部件；如果没有效果，再进行划分。

1.2.3　启动成本高，运行成本低

许多人造物都会耗费大量的资源用来启动，例如汽车（它的点火系统、应急刹车）、电灯泡（它的寿命主要取决于被打开的次数）和数据库系统。

- 在磁盘上开始一个读操作是很耗时的，但是一旦开始，磁盘就能高速地传输数据。因此从一个磁道连续读 64 KB 的数据，很可能比从该磁道两次读取 512 字节数据所花费的时间还短。因此，经常被扫描的表应当连续存放在磁盘上。当某个重要的查询从包含成百上千列的表中只查取少数几列时，垂直划分是个好策略。
- 在分布式系统中，通过网络发送一条消息后的反馈时间比在一条消息里发送很多字节所增加的时间开销高很多。实际上，发送 1 KB 的包，时间开销只比发送 1 字节的包大一点。这意味着，发送大块数据比发送小块数据好。
- 即使对简单查询而言，解析、进行语义分析、选择存取路径的系统成本也是惊人的（在大多数系统中超过 10 000 条指令）。这表明，应该事先编译经常执行的查询。
- 在使用标准编程语言（诸如 C++, Java, Perl, COBOL 或者 PL/1）调用数据库系统时，某些系统（如大多数关系系统）中，打开连接并进行调用会导致很大的开销。所以，程序最好是执行简单的 SELECT 调用，然后循环引用其结果，而不是循环地多次调用数据库（每次都使用 SELECT）。另外，缓存数据库连接也是个不错的选择。

以上 4 个例子列举了 4 种不同的初始成本：获取第一个要读的字节；发送消息的第一个字节；执行前预备一个查询；有效地利用调用。这些例子所要阐明的道理是一样的：用尽可能少的启动次数来获取最好的性能。

1.2.4　服务器和客户端之间合理的任务分配

要从数据密集型的系统获得最好的性能，不仅需要对系统的数据库管理部分进行调优，还需要考虑如何在数据库系统（服务器）和应用程序（客户）之间分配工作。特定的任务应该被分配到哪里呢？这取决于以下 3 个主要因素：

1. **客户与服务器的相关计算资源**：如果服务器过载，其他部分都正常工作，部分任务就应该迁移到客户端去。例如，一些应用程序会将计算密集型的工作分配给客户端。
2. **相关信息的获取**：当数据库发生变化时（如表插入）系统应该有些响应（如写到屏幕上

去）。设计良好的系统应该在内部使用触发器，而不是在应用程序中进行轮询。这是因为轮询法会定期地查看表是否有变化，而触发器仅在发生变化时被激发，因此只需要很小的开销。

3. **数据库任务和屏幕是否进行交互**：如果存在交互，那么访问屏幕的那部分应该放在事务之外执行，原因是屏幕交互会占用大量时间（至少几秒）。如果事务 *T* 持有时间片，那么 *T* 会阻止其他事务访问它所持有的数据，所以事务应被分成 3 步：
   （a）一个用来检索数据的短事务。
   （b）一个交互会话，它发生在客户端，并且在事务环境之外（不持锁）。
   （c）用来设置交互会话期间变化的另一个短事务。

下一章和附录 B 里还会用到类似的例子，讨论在不牺牲隔离性的前提下对事务进行细分的方法。

1.2.5 性能价格比的权衡

为提高应用程序的速度，需要提供一定的内存、磁盘或计算资源，下面将介绍这一点。

- 增加 RAM 可使系统增加缓冲空间，减少磁盘访问的次数，从而增加系统的速度。当然，RAM 并不是免费的。
- 增加索引会使关键查询执行起来快很多，但是这需要更多的磁盘空间和更多的内存，还需要更多的处理器时间和磁盘访问，用来完成不使用索引的插入和更新操作。
- 用临时的划分将长查询和在线更新分离后，会发现分配给这些长查询的时间太少了。这时，可以创建专用的归档数据库供长查询使用。从性能角度看，这是个很好的解决办法，但需要购买并维护一个新的计算机系统。

对以上问题，一位关系数据库供应商的顾问简单地概括说："想要速度，那么你愿意出多少钱呢？"

1.3 基本原理和知识

数据库调优是一门知识密集型的学科。调优者必须综合考虑缓冲池大小、数据结构、锁争用和应用程序需求间的复杂关联，并在此基础上做出判断。因此后继章节在详细讨论各个话题的同时，也强调了它们应用于特定环境时应考虑的调优因素。以下是每章的简要描述。

- 第 2 章讨论底层的系统功能，它们是数据库系统的基础。
  对数据库系统性能提高至关重要的并发控制和恢复原则，以及对这些子部件进行调优的指南。
  对调优和监控而言都很重要的操作系统结构方面。
  最有可能提高系统性能的硬件调整。
- 第 3 章讨论了索引的选择。
  跟索引选择最有关联的查询类型。
  很多数据库系统提供的数据结构（B-树，哈希和堆结构），一些应用程序可能会用到的数据结构。
  稀疏索引和稠密索引。
  聚簇（主）索引和非聚簇（二级）索引。