

Apress®

SQL Server 2008
Query Performance Tuning Distilled

SQL Server 2008 查询性能优化

[美] Grant Fritchey Sájal Dam 著
姚军 译

 人民邮电出版社
POSTS & TELECOM PRESS

SQL Server 2008 查询性能优化

[美] Grant Fritchey Sajal Dam 著
姚军 译

人民邮电出版社
北京

图书在版编目 (C I P) 数据

SQL Server 2008查询性能优化 / (美) 弗里奇 (Fritchey, G.), (美) 达姆 (Dam, S.) 著 ; 姚军译. — 北京 : 人民邮电出版社, 2010. 8
ISBN 978-7-115-23029-4

I. ①S… II. ①弗… ②达… ③姚… III. ①关系数据库—数据库管理系统, SQL Server 2008 IV. ①TP311.138

中国版本图书馆CIP数据核字(2010)第086918号

版 权 声 明

Original English language edition, entitled SQL Server 2008 Query Performance Tuning Distilled by Grant Fritchey, Sajal Dam, published by Apress 2855 Telegraph Avenue, #600, Berkeley, CA 94705 USA. Copyright © 2009 by Apress L.P. Simplified Chinese-language edition copyright © 2010 by POSTS & TELECOMMUNICATIONS PRESS.
All rights reserved.

本书中文简体字版由 Apress L.P.授权人民邮电出版社独家出版。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,侵权必究。

SQL Server 2008 查询性能优化

- ◆ 著 [美] Grant Fritchey Sajal Dam
译 姚 军
责任编辑 刘映欣
- ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街14号
邮编 100061 电子函件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
北京艺辉印刷有限公司印刷
- ◆ 开本: 800×1000 1/16
印张: 28.75
字数: 627千字 2010年8月第1版
印数: 1-3000册 2010年8月北京第1次印刷

著作权合同登记号 图字: 01-2009-2889号

ISBN 978-7-115-23029-4

定价: 69.00元

读者服务热线: (010)67132705 印装质量热线: (010)67129223
反盗版热线: (010)67171154

内 容 提 要

本书通过大量实例，详细介绍了 SQL Server 数据库系统优化的各种方法和技巧。内容涵盖了数据库应用系统中各种性能瓶颈的表现形式及其发生的根源和解决方法，从硬件瓶颈到查询、索引设计以及数据库管理等，贯穿了数据库系统知识的各个方面。最后以一个实际的工作负载将所有技巧联系起来，并且提供了“宝典”式的最佳实践列表。

本书适合于关心数据库应用系统性能的开发人员和数据库管理人员阅读。通过阅读本书，不仅可以学习到数据库性能管理的许多知识和技巧，还有助于养成良好的编程习惯，为实现高性能的数据库应用系统打下基础。

关于作者

Grant Fritchey 为 FM Global（一家行业领先的工程和保险公司）工作，担任首席 DBA。他使用各种语言（如 VB、C#和 Java 等）开发了许多大规模的应用程序，从版本 6.0 开始使用 SQL Server。他曾经为 3 家失败的 .com 公司担任财务和咨询工作，还是 *Dissecting SQL Server Execution Plans* 一书的作者。

Sajal Dam 拥有位于印度班加罗尔的印度理工学院的计算机科学技术硕士学位，并且使用微软技术超过 16 年。他已经在设计数据库应用和管理软件开发方面拥有了很广泛的背景。Sajal 还在从前端网页到后端数据库的基于微软技术的应用程序上，具备了故障定位和性能优化的大量经验。他有许多为《财富》500 强公司设计可伸缩的数据库解决方案和最大化数据库环境性能的经验。

关于技术审校人

Joseph Sack 为微软公司工作，担任主要领域工程团队的专有支持工程师。从 1997 年起，他就一直为金融服务、IT 咨询、制造业、零售业和房地产业的客户开发并且支持 SQL Server 环境。他是 *SQL Server 2008 Transact-SQL Recipes*、*SQL Server 2005 T-SQL Recipes* 和 *SQL Server 2000 Fast Answers for DBAs and Developers* 等书籍的作者。

致 谢

写一本书真的很难。但幸运的是，写这些东西时我能有这么好的支持团队。感谢来自 Apress 的人们：Jonathan Gennick，感谢你给我的机会和支持；Richard Dal Porto，感谢你的组织；感谢 Douglas Pundick 完美的编辑工作，很抱歉为你添了这么多麻烦；同样感谢 Kim Wimpsett 和 Kelly Winqvist，对我的技术编辑，我唯一能说的是，“我配不上你们”；我衷心地感谢 Joseph Sack 对本书所做的杰出的工作。你们完成了令人惊奇的工作，如果本书中尚有遗留的错误那都是我的责任，而不是你们的。

我还必须感谢所有 SQL Server 中心论坛的人们。Steve Jones 和 Andy Warren 在这里创建了一个社区。Steve、Gail、Jeff、Jack、Barry、Lynn 和所有其他没有特别提到的人们，感谢你们多年来的帮助。我和一群了不起的人们在一家出色的公司中工作，在这里我有了学习和成长的机会。感谢我的经理 Jack Dewar 和他的老板 Bill Hawkins。还要感谢 DBA 团队 Peter、Ted、Basem、AJ、Dave M.、Dave H.、Chris K.、Chris P.、Yuting、Ray、Raj 和 Det，你们教给我很多。

感谢 Red Gate 的人们制作的了不起的工具，包括 SQL Prompt，我用它编写和格式化了本书的 T-SQL 代码。

最后，在人们开始演奏音乐，将我赶下舞台之前，我希望感谢我的家人——我的妻子和孩子。这是你们让我写的第二本书，感谢你们对我的宽容。

Grant Fritchey

前 言

性能常常是开发系统时最后想到的事情之一。很不幸，这意味着它将在系统投产之后成为最大的问题。不能简单地依靠一个电话，告诉你运行于服务器 Z 的数据库 Y 上的过程 X 运行很慢，而必须有一个机制来查找这些信息。也不能用“慢”来说明工作的状态。慢是和什么相比？上一周？还是上一个月？和开发时相比？一旦你认定某些部件运行缓慢，必须要指出原因。它是否需要一个索引？它是否有一个未用的索引？是由于 CPU、磁盘、内存、用户数，还是数据量？已经认定了慢的部件和原因后，还必须做一些事情。如何优化？重写查询？修改 Where 子句？当开始性能调整时，这些问题会没完没了。

本书将提供回答这些问题所需要的工具。将会说明，如何建立一个收集服务器上运行的 SQL Server 实例及其中的数据库性能数据的机制。还将介绍收集单个 T-SQL 调用数据的巧妙方法。过程中，将讨论索引结构、选择和维护，编写 T-SQL 代码的最佳方法，以及许多其他的主题。编写本书的目标之一，是使用类似于在真实世界中所能看到的查询类型的实例来说明这些主题。所介绍的工具和方法大部分都可以用于 SQL Server 标准版，但是有些只可用于 SQL Server 企业版——到时会有相关提示。

本书将有条理地介绍进行这些工作和回答这些问题的工具，从而消除当前在性能优化中常见的猜测工作。性能问题不是什么可怕的事情，使用正确的工具，可以从容而可靠地解决性能问题，并赢得同伴和客户的尊敬以及直接帮助他们成功。

本书的读者

本书适合于任何负责系统性能的读者。这无疑包括数据库管理员，因为他们负责建立系统，创建基础架构，并且随时对其进行监控。开发人员也被包括在内，因为除了他们，还有谁需要生成结构良好和高性能的 T-SQL 代码？数据库开发人员比任何人都更是目标读者，因为数据库开发正是我的工作。

任何有能力编写 T-SQL、设计表、实现索引或者在 SQL Server 系统上操纵服务器设置的人都在一定程度上需要这些信息。

本书的结构

本书的目的是尽量使用真实的查询。为此，需要一个“真实”的数据库。我本可以创建一个

并强迫大家去下载，但是这里我选择使用一个 Microsoft 创建的样板数据库 AdventureWorks2008 来代替——这可以通过 CodePlex 得到 (<http://www.codeplex.com/MSFTDBProdSamples>)。建议保持一份便于恢复的拷贝，并且在阅读本书的两个主题之后复位样板数据库。Microsoft 随时会更新这些数据库，所以可能看到和本书中所列出的不同的数据组或不同的查询表现。在一定程度上，本书基于前面章节所介绍的知识来构建。但是，许多章节介绍该主题中的独特信息，所以可以挑选特定的章节阅读，从第 1 章到第 16 章顺序阅读可以获得最大的好处。

- 第 1 章“SQL 查询性能调整”介绍性能调整的迭代过程。你将得到发布性能基线、识别瓶颈、解决问题以及量化改进的第一印象。
- 第 2 章“系统性能分析”从监控运行 SQL Server 的 Windows 系统开始，说明作为数据收集机制的性能监视器和用户自定义函数。
- 第 3 章“SQL 查询性能分析”定义了查看“隐藏的秘密”以及了解系统上运行何种查询的最佳方法，它提供了 Profiler 和跟踪工具的细节。本章中第一次介绍了多种用于监控查询的动态管理视图和功能。
- 第 4 章“索引分析”说明了索引和索引架构，它定义了聚簇索引和非聚簇索引之间的差别，说明了在不同类型的查询中哪种索引工作得最好，还介绍了基本的索引维护。
- 第 5 章“数据库引擎调整顾问”介绍了微软的工具——数据库引擎调整顾问。本章详细介绍了数据库引擎调整顾问的使用方法；介绍了各种调用工具的机制并说明在真实负载下的使用方法。
- 第 6 章“书签查找分析”针对典型的性能问题——关键字查找，这也被称为书签查找。本章研究各种书签查找操作的解决方案。
- 第 7 章“统计分析”介绍了统计的概念。优化器使用统计做出关于查询执行的计划。本章的主题是维护统计，理解它们的存储方式，学习其工作方式，并学习其对查询的影响。
- 第 8 章“碎片分析”说明如何随时索引碎片。将给出识别索引碎片的方法，索引出现碎片时查询发生了什么问题，以及消除索引碎片的机制。
- 第 9 章“执行计划缓冲分析”介绍 SQL Server 用于存储执行计划的机制。计划重用是 SQL Server 中的一个重要概念。将给出识别计划是否被重用的方法，以及用于查看缓冲的各种机制。本章还介绍了新的动态管理视图，允许比以前更多的缓冲访问。
- 第 10 章“存储过程重编译”显示 SQL Server 如何及何时重新编译存储在缓冲中的计划。将给出计划重新编译可能伤害或者帮助系统性能的方法，以及强制或者避免重新编译的机制。
- 第 11 章“查询设计分析”展现编写系统中执行良好的查询的方法。本章研究常见的错误并提供解决方案，以及给出多种避免常见瓶颈的最佳实践。
- 第 12 章“阻塞分析”讲述发现服务器上有多个对话争夺资源的最佳方法。将给出如何监控阻塞以及避免阻塞的会话的方法和技术。

- 第 13 章“死锁分析”说明系统上发生死锁的情况，并给出识别涉及死锁的会话的方法。本章还介绍避免死锁或发生死锁后修复代码的最佳实践。
- 第 14 章“游标开销分析”解析在面向数据集的 T-SQL 代码中出现的游标的固有开销。但是，当游标不可避免时，需要理解它们的工作方法、它们所做的工作，以及直接消灭它们不是一个选择时调整它们的最佳方法。
- 第 15 章“数据库负载优化”示范了取得前面所有章节中出现的信息并将它们用在真实数据库负载中的方法。将识别性能最差的过程并通过多种调整方法使其达到更优的性能。
- 第 16 章“SQL Server 优化检查列表”总结了前面各章，得到一组检查列表和最佳实践。本章的目标是快速回顾所有已经学习到的知识。

下载代码

可以从 Apress 网站 (<http://www.apress.com>) 的 Source Code (源代码) 部分下载本书使用的代码实例。大部分代码是保存在 .sql 文件中的普通 T-SQL，可以在任何 SQL Server T-SQL 编辑工具中打开和使用。有一个 PowerShell 脚本必须通过 PowerShell 命令行运行。

联系作者

可以通过 grantedd@google.com 联系作者 Grant Fritchey，还可以在 <http://scarydba.wordpress.com> 访问他的博客。

目 录

第 1 章 SQL 查询性能调整	1	2.3 硬件资源瓶颈	18
1.1 性能调整过程	2	2.3.1 识别瓶颈	18
1.1.1 核心过程	2	2.3.2 瓶颈解决方案	19
1.1.2 迭代过程	4	2.4 内存瓶颈分析	19
1.2 性能 vs. 价格	7	2.4.1 SQL Server 内存管理	20
1.2.1 性能目标	7	2.4.2 Available Bytes	23
1.2.2 “足够好”的调整	7	2.4.3 Pages/sec 和 Page Faults/sec 计数器	23
1.3 性能基线	8	2.4.4 Buffer Cache Hit Ratio	24
1.4 工作的重点	9	2.4.5 Page Life Expectancy	24
1.5 SQL Server 性能杀手	10	2.4.6 Checkpoint Pages/sec	24
1.5.1 低质量的索引	10	2.4.7 Lazy writes/sec	24
1.5.2 不精确的统计	11	2.4.8 Memory Grants Pending	25
1.5.3 过多的阻塞和死锁	11	2.4.9 Target Server Memory (KB) 和 Total Server Memory (KB)	25
1.5.4 不基于数据集的操作	11	2.5 内存瓶颈解决方案	25
1.5.5 低质量的查询设计	12	2.5.1 优化应用程序工作负载	26
1.5.6 低质量的数据库设计	12	2.5.2 为 SQL Server 分配更多 内存	27
1.5.7 过多的碎片	12	2.5.3 增加系统内存	27
1.5.8 不可重用的执行计划	13	2.5.4 更换 32 位处理器为 64 位 处理器	27
1.5.9 低质量的执行计划	13	2.5.5 启用 3GB 进程空间	28
1.5.10 频繁重编译计划	13	2.5.6 在 32 位 SQL Server 中 使用 4GB 以上内存	28
1.5.11 游标的错误使用	13	2.6 磁盘瓶颈分析	29
1.5.12 错误配置数据库日志	14	2.6.1 磁盘计数器	30
1.5.13 过多使用或者错误 配置 tempdb	14	2.6.2 % Disk Time	30
1.6 小结	14	2.6.3 Current Disk Queue	
第 2 章 系统性能分析	15		
2.1 性能监视器工具	15		
2.2 动态管理视图	17		

Length	31	三级 (L3) 缓存	44
2.6.4 Disk Transfers/sec	31	2.9.5 运行更高效的控制器/ 驱动程序	44
2.6.5 Disk Bytes/sec	32	2.9.6 不运行不必要的软件	45
2.6.6 Avg. Disk Sec/Read和 Avg. Disk Sec/Write	32	2.10 网络瓶颈分析	45
2.7 磁盘瓶颈解决方案	32	2.10.1 Bytes Total/sec	45
2.7.1 优化应用程序工作负载	33	2.10.2 % Net Utilization	46
2.7.2 使用更快的磁盘驱动器	33	2.11 网络瓶颈解决方案	46
2.7.3 使用一个RAID阵列	33	2.11.1 优化应用程序工作 负载	46
2.7.4 使用SAN系统	35	2.11.2 增加网络适配器	47
2.7.5 恰当地对齐磁盘	35	2.11.3 节制和避免中断	47
2.7.6 使用电池后备的控制器 缓存	36	2.12 SQL Server 总体性能	47
2.7.7 添加系统内存	36	2.12.1 丢失索引	48
2.7.8 创建多个文件和文件组	36	2.12.2 数据库阻塞	49
2.7.9 将表和索引放在不同的 磁盘上	39	2.12.3 不可重用的执行计划	50
2.7.10 将日志文件保存到 独立的物理磁盘	39	2.12.4 总体表现	50
2.7.11 表的分区	40	2.13 创建一个基线	51
2.8 处理器瓶颈分析	40	2.13.1 创建性能计数器的 一个可重用列表	51
2.8.1 % Processor Time	41	2.13.2 使用性能计数器列表 创建一个计数器日志	54
2.8.2 % Privileged Time	41	2.13.3 最小化性能监视器 开销	55
2.8.3 Processor Queue Length	42	2.14 以基线为标准的系统 状态分析	56
2.8.4 Context Switches/sec	42	2.15 小结	57
2.8.5 Batch Requests/sec	42	第3章 SQL 查询性能分析	58
2.8.6 SQL Compilations/sec	42	3.1 SQL Profiler 工具	58
2.8.7 SQL Recompilations/sec	43	3.1.1 Profiler跟踪	59
2.9 处理器瓶颈解决方案	43	3.1.2 事件	60
2.9.1 优化应用程序工作负载	43	3.1.3 数据列	62
2.9.2 消除过多的编译/ 重编译	43	3.1.4 过滤器	64
2.9.3 使用更多或更快的 处理器	44	3.1.5 跟踪模板	65
2.9.4 使用大的二级 (L2) /			

3.1.6 跟踪数据	65	4.1.1 索引的好处	97
3.2 跟踪的自动化	66	4.1.2 索引开销	98
3.2.1 使用GUI捕捉跟踪	66	4.2 索引设计建议	100
3.2.2 使用存储过程捕捉跟踪	67	4.2.1 检查WHERE子句和 连接条件列	100
3.3 结合跟踪和性能监视器输出	68	4.2.2 使用窄索引	102
3.4 SQL Profiler 建议	69	4.2.3 检查列的唯一性	103
3.4.1 限制事件和数据列	69	4.2.4 检查列数据类型	106
3.4.2 丢弃性能分析所用的 启动事件	70	4.2.5 考虑列顺序	107
3.4.3 限制跟踪输出大小	70	4.2.6 考虑索引类型	109
3.4.4 避免在线数据列排序	71	4.3 聚簇索引	109
3.4.5 远程运行Profiler	71	4.3.1 堆表	110
3.4.6 限制使用某些事件	71	4.3.2 与非聚簇索引的关系	110
3.5 没有 Profiler 情况下的查询 性能度量	71	4.3.3 聚簇索引建议	112
3.6 开销较大的查询	72	4.4 非聚簇索引	117
3.6.1 识别开销较大的查询	73	4.4.1 非聚簇索引维护	117
3.6.2 识别运行缓慢的查询	77	4.4.2 定义书签查找	117
3.7 执行计划	78	4.4.3 非聚簇索引建议	118
3.7.1 分析查询执行计划	80	4.5 聚簇索引 vs. 非聚簇索引	118
3.7.2 识别执行计划中开销 较大的步骤	82	4.5.1 聚簇索引相对于非聚簇 索引的好处	119
3.7.3 分析索引有效性	83	4.5.2 非聚簇索引相对于聚簇 索引的好处	120
3.7.4 分析连接有效性	84	4.6 高级索引技术	121
3.7.5 实际执行计划 vs. 估算 执行计划	88	4.6.1 覆盖索引	122
3.7.6 计划缓存	89	4.6.2 索引交叉	124
3.8 查询开销	90	4.6.3 索引连接	125
3.8.1 客户统计	90	4.6.4 过滤索引	126
3.8.2 执行时间	91	4.6.5 索引视图	128
3.8.3 STATISTICS IO	92	4.6.6 索引压缩	132
3.9 小结	94	4.7 特殊索引类型	134
第 4 章 索引分析	95	4.7.1 全文索引	134
4.1 什么是索引	95	4.7.2 空间索引	135
		4.7.3 XML	135
		4.8 索引的附加特性	135

4.8.1 不同的列排序顺序	135	7.2.1 更新统计的好处	162
4.8.2 在计算列上的索引	136	7.2.2 过时统计的缺点	164
4.8.3 BIT数据类型列上的索引	136	7.3 在非索引列上的统计	165
4.8.4 作为一个查询处理的CREATE INDEX语句	136	7.3.1 在非索引列上统计的好处	166
4.8.5 并行索引创建	136	7.3.2 丢失非索引列上的统计的缺点	169
4.8.6 在线索引创建	137	7.4 分析统计	172
4.8.7 考虑数据库引擎调整顾问	137	7.4.1 密度	174
4.9 小结	137	7.4.2 多列索引上的统计	174
第5章 数据库引擎调整顾问	139	7.4.3 过滤索引上的统计	175
5.1 数据库引擎调整顾问机制	139	7.5 统计维护	176
5.2 数据库引擎调整顾问实例	143	7.5.1 自动维护	177
5.2.1 调整一个查询	143	7.5.2 人工维护	179
5.2.2 调整一个跟踪工作负载	146	7.5.3 统计维护状态	181
5.3 数据库引擎调整顾问的局限性	148	7.6 为查询分析统计的有效性	182
5.4 小结	149	7.6.1 解决丢失统计问题	182
第6章 书签查找分析	150	7.6.2 解决过时统计问题	184
6.1 书签查找的目的	150	7.7 建议	186
6.2 书签查找的缺点	152	7.7.1 统计的向后兼容性	186
6.3 分析书签查找的起因	153	7.7.2 自动创建统计	186
6.4 解决书签查找	155	7.7.3 自动更新统计	187
6.4.1 使用一个聚簇索引	155	7.7.4 自动异步更新统计	189
6.4.2 使用一个覆盖索引	155	7.7.5 收集统计的采样数量	189
6.4.3 使用索引连接	158	7.8 小结	190
6.5 小结	160	第8章 碎片分析	191
第7章 统计分析	161	8.1 碎片的成因	191
7.1 统计在查询优化中的角色	161	8.1.1 UPDATE语句引起的页面分割	193
7.2 索引列上的统计	162	8.1.2 INSERT语句引起的页面分割	196
		8.2 碎片开销	197
		8.3 分析碎片数量	200
		8.4 碎片解决方案	204

8.4.1 卸载并重建索引	204	9.8.3 使用sp_executesql编程 以避免存储过程维护	252
8.4.2 使用DROP_EXISTING 子句重建索引	205	9.8.4 实现准备/执行模式以 避免重传查询字符串	253
8.4.3 执行ALTER INDEX REBUILD语句	205	9.8.5 避免即席查询	253
8.4.4 执行ALTER INDEX REORGANIZE语句	207	9.8.6 对于动态查询sp_ executesql优于 EXECUTE	253
8.5 填充因子的重要性	209	9.8.7 小心地参数化查询的 可变部分	254
8.6 自动维护	212	9.8.8 不要允许查询中对象的 隐含解析	254
8.7 小结	217	9.9 小结	254
第9章 执行计划缓冲分析	218	第10章 存储过程重编译	256
9.1 执行计划生成	218	10.1 重编译的好处和缺点	256
9.1.1 解析器	219	10.2 确认导致重编译的语句	258
9.1.2 代数化器	220	10.3 分析重编译起因	260
9.1.3 优化	221	10.3.1 架构或绑定变化	261
9.2 执行计划缓冲	227	10.3.2 统计变化	261
9.3 执行计划组件	227	10.3.3 延迟对象解析	264
9.3.1 查询计划	227	10.3.4 SET选项变化	266
9.3.2 执行上下文	227	10.3.5 执行计划老化	266
9.4 执行计划的老化	228	10.3.6 显式调用sp_ recompile	267
9.5 分析执行计划缓冲	228	10.3.7 显式使用RECOMPILE 子句	268
9.6 执行计划重用	229	10.4 避免重编译	269
9.6.1 即席工作负载	230	10.4.1 不要交替使用DDL和 DML语句	270
9.6.2 预定义工作负载	231	10.4.2 避免统计变化引起的 重编译	271
9.6.3 即席工作负载的计划 可重用性	231	10.4.3 使用表变量	273
9.6.4 预定义工作负载的计划 可重用性	239	10.4.4 避免在存储过程中 修改SET选项	275
9.7 查询计划 Hash 和查询 Hash	248		
9.8 执行计划缓冲建议	251		
9.8.1 明确地参数化查询的 可变部分	252		
9.8.2 使用存储过程实现 业务功能	252		

10.4.5 使用OPTIMIZE FOR 查询提示	276	局部变量	306
10.4.6 使用计划指南	277	11.6.6 小心地命名存储过程	309
10.5 小结	281	11.7 减少网络传输数量	311
第 11 章 查询设计分析	282	11.7.1 同时执行多个查询	311
11.1 查询设计建议	282	11.7.2 使用SET NOCOUNT	311
11.2 在小结果集上操作	283	11.8 降低事务开销	312
11.2.1 限制选择列表中的 列数	283	11.8.1 减少日志开销	312
11.2.2 使用高选择性的 WHERE子句	284	11.8.2 减少锁开销	314
11.3 有效地使用索引	284	11.9 小结	315
11.3.1 避免不可参数化的 搜索条件	285	第 12 章 阻塞分析	316
11.3.2 避免WHERE子句列 上的算术运算符	289	12.1 阻塞基础知识	316
11.3.3 避免WHERE子句列 上的函数	290	12.2 理解阻塞	317
11.4 避免优化器提示	292	12.2.1 原子性	317
11.4.1 连接提示	293	12.2.2 一致性	320
11.4.2 索引提示	295	12.2.3 隔离性	320
11.5 使用域和参照完整性	296	12.2.4 持久性	321
11.5.1 非空约束	297	12.3 数据库锁	321
11.5.2 声明参照完整性	299	12.3.1 锁粒度	322
11.6 避免资源密集型查询	301	12.3.2 锁升级	325
11.6.1 避免数据类型转换	301	12.3.3 锁模式	326
11.6.2 使用EXISTS代替 COUNT(*)验证数 据存在	303	12.3.4 锁兼容性	332
11.6.3 使用UNION ALL 代替UNION	304	12.4 隔离级别	332
11.6.4 为聚合和排序操作 使用索引	305	12.4.1 未提交读	333
11.6.5 避免在批查询中的		12.4.2 已提交读	333
		12.4.3 可重复读	335
		12.4.4 可序列化 (Serializable)	338
		12.4.5 快照 (Snapshot)	343
		12.5 索引对锁的作用	343
		12.5.1 非聚簇索引的作用	344
		12.5.2 聚簇索引的作用	346
		12.5.3 索引在可序列化隔离 级别上的作用	346
		12.6 捕捉阻塞信息	347

12.6.1 使用SQL捕捉阻塞 信息	347	14.2.1 游标位置的开销比较...	378
12.6.2 Profiler跟踪和被阻塞 进程报告事件	349	14.2.2 游标并发性上的开销 比较	380
12.7 阻塞解决方案	351	14.2.3 在游标类型上的开销 比较	381
12.7.1 优化查询	352	14.3 默认结果集	383
12.7.2 降低隔离级别	352	14.3.1 好处	384
12.7.3 分区争用的数据	353	14.3.2 缺点	384
12.7.4 争用数据上的覆盖 索引	354	14.4 分析 SQL Server 游标开销	386
12.8 减少阻塞的建议	354	14.5 游标建议	390
12.9 自动化侦测和收集阻塞 信息	355	14.6 小结	392
12.10 小结	359	第 15 章 数据库工作负载优化	393
第 13 章 死锁分析	360	15.1 工作负载优化基础知识	393
13.1 死锁基础知识	360	15.2 工作负载优化步骤	394
13.2 使用错误处理来捕捉死锁	361	15.3 捕捉工作负载	397
13.3 死锁分析	362	15.4 分析工作负载	399
13.3.1 收集死锁信息	362	15.5 识别开销最大的查询	400
13.3.2 分析死锁	364	15.6 确定开销最大的查询的 基线资源使用	402
13.4 避免死锁	368	15.6.1 总体资源使用	402
13.4.1 按照相同的时间顺序 访问资源	368	15.6.2 详细资源使用	402
13.4.2 减少被访问资源的 数量	369	15.7 分析和优化外部因素	405
13.4.3 最小化锁的争用	369	15.7.1 分析应用程序使用的 批级别选项	405
13.5 小结	370	15.7.2 分析统计有效性	406
第 14 章 游标开销分析	372	15.7.3 分析碎片整理需求	406
14.1 游标基础知识	372	15.8 分析开销最大的查询的 内部行为	410
14.1.1 游标位置	373	15.8.1 分析查询执行计划	410
14.1.2 游标并发性	374	15.8.2 识别执行计划中开销 较大的步骤	412
14.1.3 游标类型	376	15.8.3 分析处理策略的效率	412
14.2 游标开销比较	378	15.9 优化代价最大的查询	412
		15.9.1 修改现有索引	413

15.9.2	分析连接提示的应用	415	16.2.9	采用重用执行计划的 最佳实践	435	
15.9.3	避免聚簇索引扫描 操作	417	16.2.10	采用数据库事务 最佳实践	436	
15.9.4	修改过程	418	16.2.11	消除或减少数据库 游标开销	437	
15.10	分析对数据库工作负载的 影响	420	16.3	配置设置	437	
15.11	迭代各个优化阶段	421	16.3.1	Affinity Mask	437	
15.12	小结	424	16.3.2	内存配置选项	437	
第 16 章 SQL Server 优化检查列表			425	16.3.3	并行性开销阈值	438
16.1	数据库设计	425	16.3.4	最大并行度	438	
16.1.1	平衡不足和过多的 规范化	426	16.3.5	优化即席工作负载	438	
16.1.2	从实体完整性约束中 得利	427	16.3.6	查询调控器开销 限制	439	
16.1.3	从域和参照完整性 约束中得利	428	16.3.7	填充因子 (%)	439	
16.1.4	采用索引设计最佳 实践	430	16.3.8	被阻塞过程阈值	439	
16.1.5	避免在存储过程名称中 使用 sp_前缀	431	16.3.9	数据库文件布局	439	
16.1.6	最小化触发器的使用	431	16.3.10	数据库压缩	440	
16.2	查询设计	432	16.4	数据库管理	440	
16.2.1	使用 SET NOCOUNT ON 命令	432	16.4.1	保持统计最新	440	
16.2.2	显式定义对象所有者	432	16.4.2	保持最小数量的索引 碎片数量	441	
16.2.3	避免不可参数化的 搜索条件	432	16.4.3	循环使用 SQL 错误 日志文件	441	
16.2.4	避免 WHERE 子句列 上的算术运算符	433	16.4.4	避免像 AUTO_CLOSE 或 AUTO_SHRINK 这样的 自动化数据库功能	441	
16.2.5	避免优化器提示	434	16.4.5	最小化 SQL 跟踪 开销	442	
16.2.6	远离嵌套视图	434	16.5	数据库备份	442	
16.2.7	确保没有隐含的数据 类型转换	435	16.5.1	增量和事务日志备份 频率	442	
16.2.8	最小化日志开销	435	16.5.2	备份分布	443	
			16.5.3	备份压缩	444	
			16.6	小结	444	