



Broadview®
www.broadview.com.cn

悦知文化
Delight Press

SQL Server 2005 Performance Tuning 性能调校



胡百敬 姚巧玫 刘承修 著



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
<http://www.phei.com.cn>



SQL Server 2005 Performance Tuning 性能调校

胡百敬 姚巧玫 刘承修 著

電子工業出版社

Publishing House of Electronics Industry

北京 • BEIJING



内 容 简 介

数据库系统经年累月地运行,日久便可能面临数据累积量大、使用人数增加、应用面扩增、当初系统设计有局限等问题,导致性能变差,这就需要调试人员进行性能调校。然而,他们进行性能调校的能力是需要培养的,一般来说,需要的不只是数据库方面的知识和经验,还要对商业领域知识、系统架构设计、应用程序撰写,以及对操作系统、网络环境架设、各种监控工具程序等都有一定的了解,才能在复杂的系统中,找到症结所在,完成调校任务。

本书正是为满足上述需要而编写的,适合 SQL Server DBA 阅读,书中提供了性能调校和错误处理的建议与提示,并通过实际案例,协助 DBA 建立正确的观念、充分了解系统架构,进而在阅读中传承功力,并打通任督二脉,领略其中运用之奥妙。

本书为精诚资讯股份有限公司-悦知文化授权电子工业出版社于大陆(台港澳除外)地区出版中文简体版本。本著作物之专有出版权为精诚资讯股份有限公司-悦知文化所有。该专有出版权受法律保护,任何人不得侵害。

版权贸易合同登记号 图字:01-2008-0589

图书在版编目(CIP)数据

SQL Server 2005 Performance Tuning 性能调校 / 胡百敬等著. —北京:电子工业出版社, 2008.6

ISBN 978-7-121-06296-4

I. S… II. 胡… , 姚… , 刘… III. 关系数据库—数据库管理系统, SQL Server 2005 IV. TP311.138

中国版本图书馆 CIP 数据核字(2008)第 041919 号

策划编辑:周 筠

责任编辑:梁 晶

印 刷:北京智力达印刷有限公司

装 订:三河市金马印装有限公司

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

开 本:787×980 1/16 印张:46.5 字数:900 千字

印 次:2008 年 6 月第 1 次印刷

定 价:80.00 元(含光盘 1 张)

凡所购买电子工业出版社图书有缺损问题,请向购买书店调换。若书店售缺,请与本社发行部联系,联系及邮购电话:(010) 88254888。

质量投诉请发邮件至 zlt@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线:(010) 88258888。

推荐序一

谈到数据库性能调校，在台湾百敬应坐第一把交椅，以他累积数十年的功力，可以预期这本书一定会成为人手一本的 DBA 圣经。

百敬一向是我最佩服的讲师，他对技术的专业、坚持与研究，让他在 SQL Server 的领域中树立了一块金字招牌。他把写书当成是一种生活乐趣，积极累积许多实际案例、协助读者建立正确的观念并让人充分了解系统架构，进而在阅读中传承功力并打通任督二脉，领略其中运用的奥妙。巧玫及承修在专业领域的成长也是大家有目共睹的，不管是对数据库的开发、管理、商业智能，还是数据转换等，都多有研究，字里行间皆可看出他们的用心，详尽的说明让读者可以快速了解且顺利上手。

拜读此书，百敬、巧玫及承修严谨并详细地将所有 DBA 最想知道的几大课题都涵盖进来，包含性能调校相关工具程序、索引及查询性能、交易及锁定管理，以及最新的 Visual Studio 2005 Team System for Database Professional 等，肯定能让所有 DBA 受用不尽。也感谢三位大师通力合作完成此本 SQL Server 2005 性能调校的大作，希望与大家一起分享！

台湾微软
资深产品行销经理
李玉秀
Cheryl Lee

推荐序二

数据库技术的最高议题，我想非性能调校莫属，对于 DBA 或是数据库系统开发人员来说，性能调校是“求好”，功力不够的人顶多“求有”个数据库即可，而功力高的人则会想在这个议题上多做一些深入的研究。性能调校的问题涉及的范围太广，不是三言两语可以道尽的，如果对整个数据库系统了解不够全面，或者不够深入，就很难找到性能不佳的原因，当然也就谈不上如何调校，以拥有较佳的性能了。

在台湾，对数据库技术能够研究得既广又深的，百敬当属第一人，而性能这个议题又是他最深感兴趣，也是投入最多时间研究的议题。要谈性能调校，各项基本功不到炉火纯青，恐怕是没有资格谈的，而且要把性能调校技术有组织地教授传达出来，也要花相当的功夫。由百敬来编写本书，是最恰当的人选。

在百敬出版的上本书中，我曾经说过，数据库本身有其生命周期，并不是一生成好就可以放着永远不用再去理它的，若是你对性能的要求高，在生成数据库之前，就得考虑将来可能会发生性能瓶颈的问题。性能是全面性的问题，每个性能不好的数据库都有不同的原因，没有办法一概而论，而这也正是性能调校之所以困难的原因，托尔斯泰说：“All happy families resemble each other, while each unhappy family is unhappy in his own way.”把“family”代换成“database”，在性能调校来讲，也是相当贴切的。

如果性能的问题一直困扰着你，请你也不要期待会有一颗万灵丹，只需开口一问，就会得到你想要的答案，性能调校的问题，终究还是要你亲自去寻找答案，但是在找答案的过程中，你必须要有正确、完整而且有组织的指引，否则这答案你将永远找不到。按图索骥虽然辛苦，然而你心里却会有底，知道离答案有多近。读者若能站在本书作者的肩膀上，将会省去很多寻找解决方案的力气。

精诚信息恒逸教育训练中心

知识产品事业部资深处长

张智凯

Richard Chang

SQL Server 2005 Performance Tuning 性能调校

作者序一

Microsoft SQL Server 已是世界上应用最普遍的大型数据库之一，在许多系统中都有它的踪迹。就笔者所触及的系统：上 tera 的数据量者有之；全球近亿条记录需要在多国多服务器间复写者有之；在一般硬件上每秒钟新增数百条记录，而需要调整到上千条者有之；以上百个 SQL Server 服务器为骨干，负责庞大企业集团时刻的营运者有之；凡此种种，皆彰显了 SQL Server 无与伦比的运算力。

每个使用数据库的人，都可能面临数据累积、使用人数增加、应用方面扩增、当初系统设计或程序编写不良等各种原因，导致性能变差。让用户无法忍受系统响应的时间，待处理的工作长期停滞在系统的队列中，甚至程序无法执行、系统崩溃、数据大乱。信息系统已经是一般企业不可或缺的竞争力之一，而数据库系统又往往是信息系统的核心，跑不动的数据库对企业绝对是莫大的伤害。

数据库管理员（DBA）或系统开发人员在发生性能问题的初期，大都不知如何解决。市面上有一堆书籍教人如何利用 SQL Server 建立数据库，或是以 .NET 等程序语言访问数据库。数据库建成容易，程序经过个把月也写完了，但以后数年都需要对数据库、对系统进行性能调校。

性能调校的能力是需要培养的，不是做了多年的 DBA 就一定可以应对各式各样层出不穷的状况的，因为现在的系统都已经太过复杂，任何一个小环节发生问题，在交互影响下，呈现出来的现象往往会误导判断，让调校人员耗费非常多的时间与精力，却往往是做的虚功。

性能调校不是简单的事，一般来说，需要有广泛的经验与知识，不单是数据库的经验，还要对商业领域运作、体系结构设计、应用程序编写、操作系统运行、网络环境架设、各种检测与监控工具程序的使用等，都要有基本的了解，才能在复杂的系统中找到症结所在。知道问题后，并不一定有解，如何廉价而有效地解决问题，又是另一项挑战。

信息技术不断更迭创新，而应用广泛多变，笔者碰到来自各方的朋友讨论系统的瓶颈所在，但往往因为需求不明，应用领域不熟，很难有具体的结果。西方谚语 “One man’s meat is another man’s poison（你喜好的肉糜对我却是毒药）”，恰能描绘性能调校的技巧难以普遍适用的

困境，我们所讨论的功能、技法往往是对某些系统应用得好，但在另一些情境下则适得其反。例如：

- 原本以为通过 ADO.NET 的 Dataset 可以解决 SQL Server 的瓶颈，没想到先拖垮了 IIS。
- 本想将数据缓存到 COM+减轻 SQL Server 的存取，没想到 COM+先挂了。
- 本以为减少数据库联机的程序编写方式可以节省服务器资源，没想到反而发挥不出多线程、多 CPU 的运算能力。
- 已经做了 10 倍于在线存取量的压力测试，系统都运行平稳，没想到未测到重点，当实际系统上线需求涌入时，才发生一大堆的 Lock/Dead Lock。
-

诸此种种，不可胜数。系统设计的好坏，存乎一心，架构好，系统可大可久，架构不好，处处捉襟见肘。但何谓好何谓不好，没有定论。只有开发团队小心翼翼地测试采用的技术，看它在自己的商业领域中，应用得是否恰当来检验。

本书尝试定义、寻找与解决问题。深入探究 SQL Server 的运行原理，如索引、光标、数据存取接口、交易与锁定等，我们将查看与仿真 SQL Server 运行时与性能相关的议题，通过工具来探索问题的来龙去脉，例如，大量的用户同时存取时，系统上有太多的锁定/被锁定/死锁（lock/block/deadlock），其形成原因、如何跟踪、如何解决等，诸如此类大部分数据库管理员可能面临的问题。

系统设计时应注意什么事项、程序该如何编写、各数据库对象使用方式的优劣也是本书的重点之一，务求有好的数据库设计，也有好的应用程序来访问它。笔者希望能以多年使用 SQL Server 的经验，参考并整理起散在诸多微软制式教材、网站、书籍与文献中的数据，期待能协助你解决性能调校这个恼人的问题。

本书所示范的各种性能调校的技巧都是执行在 MS Windows 2000、XP 及 Windows 2003 操作系统上的，执行的 SQL Server 版本以 2005 为主。你所使用的操作系统或 SQL Server 的版本也许不同，但如何建立一个有效率的系统，当应用程序性能不足时，要如何调校？其中的概念、使用工具，与大部分的技巧都是相通的。因此，就算你使用的环境与前述软件版本不同，本书应该仍具有参考价值。

这是一本 SQL Server 进修提高的书，提供给已经在使用 SQL Server 的工程师一些性能调校或错误处理的建议与提示，因此，一些基本的操作，或是应有的数据库基本观念我们都未解释，仅列出可供参考的资源。

笔者常常对听课的朋友谈到：“技术可以教，而艺术不能教”。性能调校比较偏向艺术，因

为技巧是双刃剑，可能有益于系统，但条件稍有不同时，却反而对系统有害。由于小而零散的规则极多，笔者企盼能从更深的层面找到共通运行的原理，让你可以收提纲挈领之效。无奈才疏学浅，看到了一堆的点，却没有贯穿的线。这让笔者一直犹豫要如何完成这本书，眼见时间匆匆流逝，新技术滚滚而来，既要深入已有的技术，又要融合新的趋势。唉，凭一己之力在信息长河里只能取一瓢饮，难有宏观擘画的能力，此时完成此书让它面市，是希望对当前的技术完成一个快照，提供给大家现行系统性能分析的基础。

有幸在大大小小的公司内当顾问，碰到过形形色色的商务，接触过各种企业文化和团队。深觉人是一切系统的基础，文化是孕育成功的摇篮。切莫急功近利、私心自用。乐观进取、互助分享才能造就好系统。在无形的气氛中，我们可化腐朽为神奇，也可浪掷千金。

在此感谢台湾微软美丽的资深产品行销经理李玉秀，有你的资源与技术指导，才有本书。也同时感谢悦知文化的叶怡慧处长和编辑 Vicky，有你们的敦促与校阅，本书才得以实时而正确地面市。

搬至桃园近十个寒暑，在相同的书桌上，整理出第十本书。大小儿已十岁与九岁，一切晃眼而过，不变的是越追越远的信息技术和孩子们的喧哗，所幸身畔有妻相伴，让我得以专心搜集，一点一点地累积。

每本书的写成是源于我对技术的脚注，也累积了对家人的轻忽怠慢。感谢慧无私的宽容与支持，轻巧地推动着家。仅将此成果献与妻。

胡百敬

作者序二

SQL Server 2005 大幅改版后，增加了许多新功能及工具，让我们不论在设计开发或是维护管理中，都有多种解决方案可选择。例如管理维护时，可使用 SQL Server 2005 提供的“Database Engine Tuning Advisor”，协助用户在尚未深入了解索引、数据分布统计、索引查看等原理前，就能有效地建立这些对象，以提升系统性能。分析死锁（Deadlock）时，可利用 SQL Profiler 提供的新功能，图形化地呈现。利用 SQL Profiler 可集成操作系统的性能监视器跟踪结果，并提供交互参照，让用户能更广泛直观地分析所搜集到的数据。若要观察整体系统运行状况，则可搭配“Performance Dashboard Reports”。

此外，以设计开发来说，单是 T-SQL 语句就增加了不少指示符，例如，使用 CTE 处理递归地查询，通过 PIVOT 或 UNPIVOT 指示符呈现以往较难产出的枢纽分析，以及获取排名计算的函数 RANK、DENSE_RANK、ROW_NUMBER 和 NTILE 等。而使用 ADO.NET 2.0 存取数据库时，也有新的功能可选用，诸如异步存取架构、批处理更新与大量数据复制、单一联机同时多执行结果集（MARS）等。若能有效使用这些 SQL Server 2005 提供的新功能，将对系统性能有或多或少的帮助。

要成为性能调校高手并非一蹴而就，不论是开发或管理层面，都要有硬功夫及丰富的实战经验，才能在面对问题时，快、狠、准地解决。也因为性能调教涉及的技术相当广，以致笔者在着手编辑此书时，常常面临技术上的瓶颈。在此，我要特别感谢百敬的鼓励，才不致中途放弃。另外，我们也邀请了实践经验非常丰富的承修参与。有他的协助，相信此书的内容更能符合 DBA 的需求。

本书是众人引颈期盼的一本书，在百敬的监督下，我们期待每位读者都能从中获取丰富的实践经验。在此也一并感谢催生者，悦知文化的叶怡慧处长和编辑 Vicky，有你们殷切地督促进度，并牺牲假期校稿，此书才能及早完成。

最后，感谢细心呵护我的家人，有你们的支持，我才能无后顾之忧地钻研这无止境的学问。

姚巧玫

作者序三

Microsoft SQL Server 如今已成为众多企业使用的大型数据库之一，并维系整个企业集团信息系统的营运，诸此种种皆彰显了 SQL Server 的可用性与稳定性的要求。但数据库管理员（DBA）或开发人员面临数据库性能等相关问题时，却无所依据，一旦他们不知如何解决问题，数据库的可用性就会降低。而且当今的信息系统都太过庞杂，任何一个地方发生故障，在相互影响之下，会使 DBA 在解决问题时无从下手、耗费时日，甚至徒劳无功。

基于以上状况，本书尝试以定义、架构为出发点，深入探讨 SQL Server 的运行原理，并说明如何通过适当的工具去探索数据库有关性能的相关问题。笔者期望能以在业界管理 SQL Server 的相关经验，并参考文献中的相关信息，能为读者在面临数据库性能问题时，提供一些有效的解决方法。

本次是承修初次写作，在此感谢百敬老师、巧玫能给我机会参与此书的编写；也感谢日盛金控的同事们——副处长、正斌、柏松、志伟、子鸿、启翰、奇泽、佩忠给我的鼓励；还有恒逸信息的俊宇教授我数据库管理员的专业素养；也同时感谢悦知文化的叶怡慧处长和编辑 Vicky 细心的校稿。

最后感谢我的家人给我无忧的环境，特别是我的爱妻——亚文，细心帮我照顾调皮的喵喵，让我无后顾之忧。

刘承修

联系博文视点

您可以通过如下方式与本书的出版方取得联系。

读者信箱: reader@broadview.com.cn

投稿信箱: bvtougao@gmail.com

北京博文视点资讯有限公司 (武汉分部)

湖北省 武汉市 洪山区 吴家湾 邮科院路特 1 号 湖北信息产业科技大厦 1402 室

邮政编码: 430074

电话: (027) 87690813 传真: (027) 87690813 转 817

若您希望参加博文视点的有奖读者调查, 或对写作和翻译感兴趣, 欢迎您访问:

<http://bv.csdn.net>

关于本书的勘误、资源下载及博文视点的最新书讯, 欢迎您访问博文视点官方博客:

<http://blog.csdn.net/bvbook>

目 录

第 1 章 性能调校概观	1
1.1 什么是性能调校	4
1.2 建立性能的基线	5
1.3 性能调校的步骤——DETECT	8
1.3.1 各阶段重点说明	9
1.3.2 练习 DETECT 方法	12
1.3.3 二分查找	14
1.3.4 定义瓶颈	15
1.4 结语	16
第 2 章 SQL Server 架构简介	19
2.1 SQL Server 运行架构	20
2.1.1 SQL Server 的访问架构	24
2.1.2 SQL Server 的核心引擎	27
2.1.3 SQL Server 动态自我管理	30
2.2 各项硬件使用剖析	32
2.2.1 内存管理	33
2.2.2 中央处理器	50
2.2.3 磁盘子系统	57
2.3 仿真系统运行	63
2.4 结语	65
第 3 章 性能调校相关工具程序	67
3.1 综观的工具	71
3.1.1 SQLDiag 公用程序概述	71
3.1.2 观察影响效率的内容	96

3.1.3 性能监视器.....	101
3.2 进一步的分析工具	111
3.2.1 Management Studio	111
3.2.2 SQL Profiler 概述.....	117
3.3 针对特定对象的工具	135
3.3.1 Database Engine Tuning Advisor	135
3.3.2 查询编辑器.....	138
3.3.3 网络监视器.....	141
3.3.4 DBCC.....	146
3.3.5 跟踪标记.....	152
3.4 Performance Dashboard Reports	155
3.4.1 SQL Server 2005 Performance Dashboard Reports.....	155
3.4.2 Performance Dashboard Reports 主要的分析途径.....	161
3.4.3 Performance Dashboard Reports 所提供的各式报表.....	164
3.4.4 Blocking 报表.....	166
3.4.5 General Wait 报表	166
3.4.6 其他细节报表.....	168
3.4.7 扩展 Performance Dashboard Reports 报表功能.....	169
3.5 压力测试工具程序	174
3.5.1 Microsoft Application Center Test	175
3.5.2 Load Simulator.....	178
3.5.3 自行编写压力测试程序.....	181
3.6 结语	184
第 4 章 动态管理视图和函数	185
4.1 动态管理视图和函数简介	186
4.2 动态管理视图和函数的使用范例	188
4.3 观察各种资源的使用情况	196
4.3.1 内存缓存区.....	196
4.3.2 CPU 的使用	198
4.3.3 执行计划重用.....	202
4.3.4 锁定与被锁定的关系.....	203
4.3.5 I/O 的使用	206
4.3.6 tempdb 系统数据库的使用.....	208

第5章 数据库设计	213
5.1 数据库设计	214
5.2 使用分割数据表切割和平行运行	222
5.2.1 分割数据表的使用范例	224
5.2.2 分割数据表与大量数据加载的集成	234
5.3 TEMPDB 系统数据库的规划	235
5.3.1 tempdb 的用途	235
5.3.2 SQL Server 2005 针对 tempdb 所做的改良	237
5.3.3 监控 tempdb 的使用	238
5.3.4 性能考虑	240
5.4 备份与还原	242
5.4.1 数据库恢复模式	242
5.4.2 数据库恢复模式之间的切换	246
5.5 大量数据加载	247
5.6 设计磁盘子系统	251
5.7 结语	257
第6章 索引	259
6.1 索引概观	261
6.1.1 建立索引与相关的属性设置	262
6.1.2 平行建立索引	269
6.1.3 在线索引	271
6.1.4 集群索引与非集群索引	274
6.1.5 排序	278
6.1.6 与索引相关的系统视图	282
6.1.7 是否值得建索引	295
6.2 索引维护	304
6.2.1 观察数据不连续	304
6.2.2 使用动态管理函数观察数据不连续	308
6.2.3 重组、重建与停用索引	310
6.3 优化执行计划的各阶段	312
6.4 统计	314
6.4.1 更新统计	321
6.5 覆盖索引	326
6.6 在视图与计算字段上建立索引	331

6.6.1 如何有效地建立 Indexed View	334
6.6.2 Indexed View 的适用范围	340
6.7 单一查询使用多个索引	341
6.8 结语	342
第 7 章 T-SQL 语法	343
7.1 有效地查询参数	344
7.1.1 不要对数据域做运算	345
7.1.2 勿负向查询	347
7.1.3 勿在 Where 子句对字段使用函数	348
7.1.4 小心使用 OR 操作	350
7.2 连接	352
7.2.1 连接	352
7.2.2 Join 语句	353
7.2.3 嵌套循环连接	357
7.2.4 合并连接	357
7.2.5 哈希连接	358
7.2.6 连接与子查询	360
7.3 其他注意事项	363
7.3.1 INSERT、DELETE 和 UPDATE	365
7.3.2 子查询	366
7.3.3 搭配 EXISTS 与 IN 的子查询	368
7.3.4 通过连接更新数据	370
7.3.5 查询提示	374
7.4 新的 DML 语句	379
7.4.1 Common Table Expression	379
7.4.2 获取排名或顺序的函数	388
7.5 SQL Server 提供的公共变量	395
7.6 结语	404
第 8 章 重用执行计划	405
8.1 编译与高速缓存执行计划	406
8.2 影响计划重用的因素	414
8.3 执行计划与 Execution Context	419
8.4 观察执行计划的使用	420

8.5 需要重新编译计划	423
8.5.1 不同参数使用相同执行计划可能引发的问题	426
8.5.2 以提示影响查询引擎所建立的执行计划	428
第9章 交易与锁定管理	435
9.1 锁定	436
9.1.1 锁定的种类及范围	436
9.1.2 锁定的兼容性	440
9.1.3 可锁定的资源	441
9.1.4 锁定与交易隔离等级	443
9.1.5 动态的锁定管理	457
9.1.6 锁定超时	458
9.2 数据行版本控制	460
9.2.1 数据行版本控制基本运行行为	461
9.2.2 “数据行版本控制”使用时机	469
9.2.3 “数据行版本控制” TEMPDB 数据库资源管理	470
9.3 交易	472
9.3.1 交易行为概述	472
9.3.2 批处理与交易	475
9.3.3 嵌套交易	480
9.3.4 存储点	485
9.3.5 锁定提示	488
9.3.6 使用交易之注意事项	492
9.4 死锁状况	494
9.4.1 发生 Cycle 死锁	494
9.4.2 发生 Conversion 死锁	495
9.4.3 分布式死锁	496
9.4.4 SQL Server 无法侦测的死锁实例	498
9.5 观察与分析系统的锁定状况	502
9.5.1 观察 SQL Server 当前执行的状况	503
9.5.2 观察与分析系统的锁定状况	509
9.6 锁定的原因及相关处理	512
9.6.1 费时的查询或交易	512
9.6.2 不正确的交易或交易隔离等级设置	514
9.6.3 交易未正确处理	514

9.6.4	未检测到的分布式死锁.....	515
9.6.5	锁定数据粒度 (Lock Granularity) 太高或太低.....	516
9.6.6	Compile Blocking	516
9.6.7	基本原则.....	517
9.7	结语	519
第 10 章	前端应用程序设计	521
10.1	程序架构	522
10.2	用户端与 SQL 服务器的交互.....	524
10.2.1	用户端访问 SQL Server 的模式.....	527
10.2.2	准备再执行的模式.....	531
10.2.3	测试各种执行 SQL 语句方式的性能	535
10.2.4	Connection Pooling.....	536
10.3	多数据结果集	545
10.3.1	SQL Server 数据访问与结果集.....	546
10.3.2	工作阶段内容信息与 MARS	546
10.3.3	前端程序经由 MARS 访问	548
10.3.4	MARS 的执行方式	554
10.3.5	同时读取与更新数据.....	556
10.4	光标	561
10.4.1	光标概观.....	562
10.4.2	默认结果集.....	564
10.4.3	服务器端光标.....	566
10.4.4	使用光标的 T-SQL 语句	571
10.4.5	与光标相关的系统存储过程.....	583
10.4.6	通过前端程序访问四种类型的光标.....	585
10.4.7	异步使用光标.....	589
10.4.8	使用光标时应注意的事项.....	592
10.5	数据高速缓存	605
10.5.1	使用高速缓存的原因.....	605
10.5.2	.NET Framework 与 Microsoft patterns & Practices Enterprise Library 提供的高速缓存.....	608
10.5.3	访问 Caching Application Block.....	612
10.6	应用程序错误处理	620