

凝聚名家技术典范 · 分享成功IT之路



DB2数据库性能 调整和优化(第3版)

张外借



牛新庄 著



清华大学出版社

DB2 数据库性能调整和优化

(第 3 版)

牛新庄 著

清华大学出版社

北 京

内 容 简 介

本书侧重于介绍 DB2 数据库的性能调优。性能调优是一个系统工程：全面监控分析操作系统、I/O 性能、内存、应用及数据库才能快速找到问题根源；深刻理解 DB2 的锁及并发机制、索引原理、数据库参数、优化器原理、统计分析和碎片整理等内部机理才能针对性地快速提出解决问题的方法；快照、db2pd、db2expln 以及各种管理视图和表函数等则是必须熟练掌握的工具。本书覆盖了进行 DB2 数据库性能调优所需的全部知识和工具，并提供了大量的性能调优的实际案例，这些案例都基于作者 10 多年积累的经验和总结，其中包括了近年来大型银行系统实际遇到的案例。本书还首次涵盖了针对 DB2 pureScale 及同城双活 GDPC(地理上分离的 pureScale 集群)的性能调优方法和实践。

本书适合有一定 DB2 数据库基础知识和经验的数据库工程师，以及希望深入、全面地掌握 DB2 数据库性能分析和调优知识的读者，同时可以成为数据库软件开发人员开发高性能数据库软件的参考书。

本书封面贴有清华大学出版社防伪标签，无标签者不得销售。

版权所有，侵权必究。侵权举报电话：010-62782989 13701121933

图书在版编目(CIP)数据

DB2 数据库性能调整和优化 / 牛新庄 著. —3 版. —北京：清华大学出版社，2017
ISBN 978-7-302-48118-8

I. ①D… II. ①牛… III. ①关系数据库系统 IV. ①TP311.132.3

中国版本图书馆 CIP 数据核字(2017)第 208131 号

责任编辑：王 军 李维杰

装帧设计：牛艳敏

责任校对：成凤进

责任印制：李红英

出版发行：清华大学出版社

网 址：<http://www.tup.com.cn>，<http://www.wqbook.com>

地 址：北京清华大学学研大厦 A 座 邮 编：100084

社 总 机：010-62770175 邮 购：010-62786544

投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn

质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn

印 装 者：清华大学印刷厂

经 销：全国新华书店

开 本：185mm×230mm

印 张：33.5

字 数：690 千字

版 次：2009 年 5 月第 1 版

2017 年 9 月第 3 版

印 次：2017 年 9 月第 1 次印刷

印 数：1~4000

定 价：98.00 元

序

关系型数据库已经走过整整半个世纪曲折而辉煌的历程，回顾 50 年的关系数据库发展史，我们心潮涌动，激情难抑。关系数据库始于 1970 年 IBM 公司研究员 E.F.Codd 博士，即“关系数据库之父”，发表的业界第一篇关于关系数据库理论的论文“A Relational Model of Data for Large Shared Data Banks”。从此关系型数据库如雨后春笋般四处萌发，各大厂商争先恐后，加入到关系型数据库的研发大潮中，而后又如大浪淘沙般去璞存真，时至今日，留下的是真正适用于客户、适应于潮流的关系型数据库产品。

IBM 公司作为关系数据库的推广先锋，为业界提供了一批优秀的数据库技术领域先驱科学家，他们所研发出的 DB2 数据库，经过近 50 年的发展，已经广泛应用于金融、电信、制造等多个行业，对日常的工作和生活带来了深远的影响。

在中国，DB2 数据库的兴起大概在 15 年前，当时国内鲜有应用，牛新庄先生作为国内首批接触研究 DB2 数据库的工程师，为业界提供了大量技术服务和专业培训，为 DB2 数据库的推广应用做出了积极杰出的贡献。同时，也为 IBM 公司反馈了很多很好的 DB2 研发建议，为关系型数据技术的长远发展贡献智慧。在数据库方向的精深造诣和丰富实践，就浓缩在他的 DB2 数据库著作中，为广大 IT 同仁授业解惑。

本套图书可以说是伴随着 DB2 数据库的成长，从第一版主讲 DB2 V8，到第二版的 DB2 V9，再到第三版的 DB2 V10。基本上每出一个版本我都会仔细品味，每个版本作者都很用心，都会删减对当前不适用的章节，加入很多新的功能和他近期实际经历的案例。经典性、权威性、实用性是本套书籍锁定的主要目标。

第一，本套书籍涵盖了 DB2 的几乎所有功能，是业界最大规模的系统梳理与总结。从理论知识到最佳实践，无所不包，无所不有，同时还总结了几十个最佳案例。

第二，本套书籍理论讲解深入浅出，案例多样详实，无论是对零基础还是拥有多年 DBA 工作经验的人都非常适用。

第三，精品的价值在于传世久远，经典的意义在于常读常新。我认为，只有被广泛阅读，受到大家喜爱、接受的作品，才具有经典的资质与意义。希望作者继续努力，将本套书籍打磨成关系数据库的经典图书。

其中第一本书《循序渐进 DB2 DBA 系统管理、运维与应用案例(第3版)》，是 DB2 学习的入门书籍。该书包含了从入门到中级阶段的知识技能的介绍，全面展示了 DB2 的主要功能和日常的工作技巧，尤其是实例和案例部分，这部分内容从实际出发为读者列举了常见的案例场景和处理办法，非常实用。在新一版中，所有的内容、示例都基于 DB2 V10.5 版本进行了修订，并介绍了 DB2 V11.1 中的新功能、新特性。

第二本书《DB2 高级管理、系统设计与诊断案例(第3版)》是 DB2 学习的高级进阶，该书从 DB2 体系结构入手，介绍了 DB2 各个内部组件的层次与功能、内存内部结构、存储内部结构、锁和并发原理等。在理解 DB2 内部基本原理的基础上进一步介绍了 DB2 的高级功能，包括分区功能、高级压缩功能等。此外，系统介绍了 OLTP 和 OLAP 系统的设计方法和管理技术、高可用和容灾方案，以及集群技术，其中涉及 HADR、DPF 和 pureScale 技术，以及地理上分离的 pureScale 集群(GDPC)技术。该书还介绍了 DB2 各种监控和诊断方法，通过精选的诊断案例使读者在学习知识的同时积累实践经验。

第三本书《DB2 数据库性能调整和优化(第3版)》专门介绍 DB2 性能调整和优化，从 DB2 数据库性能有关的基础知识和原理入手，从数据库所处的运行环境(OS、存储等)开始介绍，并对 DB2 的进程和内存进行深入讲解。在全面了解性能相关知识后，开始逐步展开，从设计到监控，从配置参数到调优工具，从锁和并发到优化器统计信息，最后列出了几个完整的性能调优案例以增加技术理解。

祝愿每一位读者能有所得、有所悟，成长为新一代的数据技术专家，也祝愿牛新庄先生在数据技术领域这条康庄大道上走得更宽更远。

IBM 前大中华区总经理
IBM 大中华区高级顾问

王天义

前 言

自 1999 年左右我开始从事数据库有关的技术工作到现在已近 20 年时间，此期间信息技术飞速发展，从无纸化办公和数据大集中到移动互联网和大数据、人工智能、云计算等信息技术改变了生活，并颠覆了传统商业模式。信息科技的发展离不开数据处理技术的进步，在这一轮信息化浪潮中，数据处理技术也发生了翻天覆地的变化，对企业经营发展和对外服务的意义越来越重要。一方面，传统企业级数据库的能力，在原有的道路和方向上不断地持续提升演进，以满足企业市场不断迸发的各类需求。另一方面，互联网场景孕育的各种新兴的数据处理技术亦不断涌现，例如 NewSQL、NoSQL、Hadoop 等大数据处理技术，这些技术成为传统数据库产品的必然补充，同时也对传统数据库产品产生了一定的冲击。但是以我长期从事企业数据处理相关工作的经验看，在企业级市场尤其是金融企业市场里面，传统数据库产品的能力依然是解决企业主要业务需求的不二选择。因此，传统数据库技术的研究和应用仍然是信息科技工作的重点。

近年来传统数据库产品在不断改进升级，以支持更快的处理能力和更高的可用性，满足不同场景下的用户需求。DB2 作为一款主流数据库产品，在这些方面也都进步明显，例如 Purescale 集群技术、跨数据中心的 GDPC 技术、列存储的 BLU 技术等创新功能就表现不俗，满足了特定业务场景需求，给企业带来了很大的价值提升。特别是 GDPC 技术，帮助企业搭建关键业务系统同城对等全双活生产架构，为最终用户提供高等级容灾的连续服务，对企业对外服务的提升意义非凡，也使数据库从业者们领略了 DB2 产品创新的精华。

基于 DB2 产品的演进以及近些年的思考和实践，我重新梳理了之前编写的第 2 版的 3 本 DB2 系列技术图书，对其进行了大篇幅的修改和重写，力图对近些年实践的精华和 DB2

产品的新趋势进行总结。在此奉献给各位数据库从业的同仁，在技术的路上共勉。

由于本人水平有限，时间有限，书中不免有这样或者那样的错误，希望广大读者朋友不吝赐教指正！

最后，感谢我的家人和同事在本书重写过程中的帮助，谢谢你们！

牛新庄

目 录

第 1 章 性能调整概述	1
1.1 性能概述	2
1.2 性能评估	4
1.3 建立性能目标	7
1.4 什么时候需要做性能调整	8
1.5 性能调整准则	9
1.6 性能调整的方法和过程	10
1.6.1 性能调整的步骤	10
1.6.2 性能调整的限制	11
1.6.3 向客户了解情况	11
1.6.4 性能调整流程图	12
1.7 性能模型	15
1.7.1 输入	17
1.7.2 处理	17
1.7.3 输出	23
1.8 本章小结	24
第 2 章 操作系统及存储的性能调优	27
2.1 AIX 性能监控综述	29
2.1.1 监控工具	29
2.1.2 监控系统总体运行状态	30
2.1.3 监控 CPU 性能	34

2.1.4 监控内存使用	38
2.1.5 监控存储系统状态	40
2.1.6 监控网络状态	42
2.2 操作系统性能优化	43
2.2.1 直接 I/O 和并发 I/O	44
2.2.2 异步 I/O 和同步 I/O	45
2.2.3 minpout 和 maxpout	47
2.2.4 文件系统和裸设备	47
2.2.5 负载均衡及条带化(Striping)	48
2.3 逻辑卷和 lvmo 优化	53
2.3.1 使用 lvmo 进行优化	54
2.3.2 卷组 pbuf 池	55
2.3.3 pbuf 设置不合理导致性能 问题调整案例	56
2.3.4 使用 ioo 进行优化	59
2.4 操作系统性能调整总结	64
2.5 存储 I/O 设计	65
2.6 存储基本概念	65
2.6.1 硬盘	65
2.6.2 磁盘阵列技术	67
2.6.3 存储的 Cache	67
2.6.4 网络存储技术	68

2.7	存储架构	69	3.4.12	监控全表扫描的 SQL	108
2.7.1	存储 I/O 处理过程	69	3.4.13	检查页清理器是否足够	108
2.7.2	RAID IOPS	70	3.4.14	监控 prefetcher 是否足够	109
2.7.3	RAID 10 和 RAID 5 的比较	71	3.4.15	监控数据库内存使用	110
2.8	良好存储规划的目标	74	3.4.16	监控日志使用情况	111
2.9	良好存储规划的设计原则	75	3.4.17	监控占用日志空间最旧的 事务	111
2.10	存储相关性能调整案例	76	3.4.18	监控存储路径	112
2.11	存储 I/O 性能调整总结	79	3.4.19	追踪监控历史	113
2.12	本章小结	80	3.5	db2pd	113
第 3 章	DB2 性能监控	81	3.5.1	常用 db2pd 监控选项和 示例	114
3.1	快照监视器案例	81	3.5.2	使用 db2pd 监控死锁 案例	126
3.1.1	监控动态 SQL 语句	81	3.5.3	db2pd 使用问题总结	132
3.1.2	监控临时表空间使用	84	3.6	内存监控	133
3.2	事件监视器及监控案例	87	3.6.1	db2pd 内存监控	133
3.3	利用表函数监控	93	3.6.2	db2mtrk 内存监控	137
3.4	性能管理视图及案例	97	3.7	本章小结	139
3.4.1	监控缓冲池命中率	99	第 4 章	DB2 配置参数调整	141
3.4.2	监控 Package Cache 大小	100	4.1	初识 DB2 配置参数	141
3.4.3	监控执行成本最高的 SQL 语句	100	4.2	监控和调优实例级(DBM) 配置参数	143
3.4.4	监控运行时间最长的 SQL 语句	101	4.2.1	代理程序相关配置参数	143
3.4.5	监控 SQL 准备和预编译 时间最长的 SQL 语句	101	4.2.2	sheapthres	145
3.4.6	监控执行次数最多的 SQL 语句	102	4.2.3	fcm_num_buffers	145
3.4.7	监控排序次数最多的 SQL 语句	103	4.2.4	sheapthres_shr	146
3.4.8	监控锁等待时间	103	4.2.5	intra_parallel	146
3.4.9	监控 Lock Chain	103	4.2.6	mon_heap_sz	147
3.4.10	监控锁内存的使用	106	4.3	监控和调优数据库级配置 参数	147
3.4.11	监控锁升级、死锁和 锁超时	107	4.3.1	缓冲池大小	147
			4.3.2	日志缓冲区大小(logbufsz)	152

4.3.3	应用程序堆大小 (applheapsz)	153
4.3.4	sortheap 和 sheapthres_shr	154
4.3.5	锁相关配置参数	157
4.3.6	活动应用程序的最大数目 (maxappls)	160
4.3.7	pckcachesz	161
4.3.8	catalogcache_sz	161
4.3.9	异步页清除程序的数目 (num_iocleaners)	161
4.3.10	异步 I/O 服务器的数目 (num_ioservers)	163
4.3.11	avg_appls	163
4.3.12	chnngpgs_thresh(DB)	164
4.3.13	maxfilop	164
4.3.14	logprimary、logsecond 和 logfilisz	164
4.3.15	stmtheap	165
4.3.16	dft_queryopt	165
4.3.17	util_heap_sz (DB)	165
4.4	调整 DB2 概要注册变量	166
4.4.1	db2_parallel_io	166
4.4.2	db2_evaluncommitted	168
4.4.3	db2_skipdeleted	168
4.4.4	db2_skipinserted	168
4.4.5	db2_use_page_container_tag	168
4.4.6	db2_selectivity	169
4.4.7	db2maxfscsearch	169
4.5	内存自动调优	169
4.5.1	内存自动调优示例	170
4.5.2	启用内存自动调优及 相关参数	171
4.5.3	内存配置参数的配置原则	173
4.6	本章小结	174
第 5 章	锁和并发	175
5.1	锁的概念	176
5.1.1	数据一致性	176
5.1.2	事务和事务边界	176
5.1.3	锁的概念	178
5.2	锁的属性、策略及模式	183
5.2.1	锁的属性	183
5.2.2	加锁策略	183
5.2.3	锁的模式	184
5.2.4	如何获取锁	186
5.2.5	锁的兼容性	188
5.3	隔离级别(Isolation Levels)	189
5.3.1	可重复读(RR—Repeatable Read)	189
5.3.2	读稳定性(RS—Read Stability)	191
5.3.3	游标稳定性(CS—Cursor Stability)	192
5.3.4	当前已提交(Currently Committed)	194
5.3.5	未提交读(UR—Uncommitted Read)	194
5.3.6	隔离级别的摘要	196
5.4	锁转换、锁等待、锁升级和 死锁	198
5.4.1	锁转换及调整案例	198
5.4.2	锁升级及调整案例	200
5.4.3	锁等待及调整案例	203
5.4.4	死锁及调整案例	205
5.5	锁相关的性能问题总结	209
5.6	锁与应用程序设计	211
5.7	锁监控工具	214

5.8	最大化并发性	218	6.6.1	设置影响索引性能的配置参数	250
5.8.1	选择合适的隔离级别	218	6.6.2	为索引指定不同的表空间	250
5.8.2	尽量避免锁等待、锁升级和死锁	218	6.6.3	确保索引的集群度	251
5.8.3	设置合理的注册变量	218	6.6.4	使表和索引统计信息保持最新	251
5.9	本章小结	227	6.7	索引维护	251
第6章	索引设计与优化	229	6.7.1	异步索引清除(AIC)	252
6.1	索引概念	229	6.7.2	联机索引整理碎片	254
6.1.1	索引优点	229	6.7.3	查找使用率底下的索引	254
6.1.2	索引类型	231	6.7.4	索引压缩	256
6.2	索引结构	231	6.8	DB2 Design Advisor (db2advis)	256
6.3	理解索引访问机制	234	6.9	本章小结	260
6.4	索引设计	237	第7章	DB2 优化器	265
6.4.1	创建索引	237	7.1	DB2 编译器介绍	266
6.4.2	创建集群索引	238	7.2	SQL 语句编译过程	268
6.4.3	创建双向索引	239	7.3	优化器组件和工作原理	271
6.4.4	完全索引访问	240	7.3.1	查询重写示例: 谓词移动、合并和转换	271
6.4.5	与创建索引相关的问题	241	7.3.2	优化器成本评估	276
6.4.6	创建索引示例	241	7.3.3	本地谓词基数(cardinality)估计	277
6.5	索引创建原则与示例	242	7.3.4	连接基数(cardinality)估计	279
6.5.1	索引与谓词	242	7.3.5	分布统计信息	283
6.5.2	根据查询使用的列建立索引	244	7.3.6	列组统计信息对基数的影响	287
6.5.3	根据条件语句中谓词的选择度创建索引	245	7.4	数据访问方式	297
6.5.4	避免在建有索引的列上使用函数	246	7.4.1	全表扫描	297
6.5.5	在那些需要被排序的列上创建索引	246	7.4.2	索引扫描	298
6.5.6	合理使用 INCLUDE 关键词创建索引	248	7.4.3	扫描共享	301
6.5.7	指定索引的排序属性	249	7.5	连接方法	302
6.6	影响索引性能的相关配置	250	7.5.1	嵌套循环连接	303

7.5.2	合并连接	305	8.2.1	自动 RUNSTATS 的基本 概念	357
7.5.3	哈希连接	306	8.2.2	如何打开 auto runstats	359
7.5.4	选择最佳连接的策略	307	8.2.3	如何监控 auto runstats	361
7.6	优化级别	307	8.2.4	自动收集统计视图的统计 信息	362
7.6.1	优化级别概述	308	8.3	碎片整理	363
7.6.2	选择优化级别	311	8.3.1	碎片产生机制和影响	363
7.6.3	设置优化级别	312	8.3.2	确定何时重组表和索引	364
7.7	基于规则的优化	314	8.3.3	执行表、索引检查是否需要 做 REORG	367
7.7.1	优化器概要文件概述	314	8.3.4	REORG 的用法和使用 策略	368
7.7.2	启用优化概要文件	316	8.4	重新绑定程序包	371
7.7.3	优化概要文件使用示例	317	8.5	本章小结	373
7.8	如何影响优化器来提高性能	324	第 9 章	SQL 语句调优	375
7.8.1	使 DB2 统计信息保持 最新	324	9.1	通过监控找出最消耗资源 的 SQL 语句	376
7.8.2	构建适当的索引	324	9.2	通过解释工具分析 SQL 语句 执行计划	376
7.8.3	配置合理的数据库配置 参数	325	9.2.1	解释表	377
7.8.4	选择合适的优化级别	326	9.2.2	db2expln	378
7.8.5	合理的存储 I/O 设计	326	9.2.3	db2exfmt	380
7.8.6	良好的应用程序设计和 编码	327	9.2.4	各种解释工具的比较	382
7.9	本章小结	329	9.2.5	如何从解释信息中获取有 价值的建议	382
第 8 章	统计信息更新与碎片整理	331	9.3	理解 SQL 语句如何工作	383
8.1	统计信息更新	332	9.3.1	理解谓词类型	383
8.1.1	统计信息的重要性	332	9.3.2	排序和分组	387
8.1.2	如何更新统计信息	333	9.3.3	连接方法	388
8.1.3	统计信息更新示例	335	9.3.4	扫描方式	389
8.1.4	LIKE STATISTICS 统计信息 更新	339	9.4	SQL 调优案例	390
8.1.5	列组统计信息更新	340			
8.1.6	分布统计信息更新	349			
8.1.7	统计信息更新策略	355			
8.2	自动统计信息更新	357			

- 9.4.1 尽量使用单条语句完成逻辑 390
- 9.4.2 合理使用 NOT IN 和 NOT EXISTS 391
- 9.4.3 利用子查询进行优化 392
- 9.4.4 调整表连接顺序使 JOIN 最优 394
- 9.4.5 数据非均匀分布时手工指定选择性 395
- 9.4.6 使用 UDF 代替查询中的复杂部分 396
- 9.4.7 合并多条 SQL 语句到单个 SQL 表达式 397
- 9.4.8 使用 SQL 一次处理一个集合语义 398
- 9.4.9 在无副作用的情况下使用 SQL 函数 400
- 9.4.10 小结 401
- 9.5 提高应用程序性能 401
 - 9.5.1 良好的 SQL 编码规则 401
 - 9.5.2 提高 SQL 编程性能 403
 - 9.5.3 改进游标性能 405
 - 9.5.4 根据业务逻辑选择最低粒度的隔离级别 406
 - 9.5.5 通过 REOPT 绑定选项来提高性能 406
 - 9.5.6 统计信息、碎片整理和重新绑定 407
 - 9.5.7 避免不必要的排序 408
 - 9.5.8 在 C/S 环境中利用 SQL 存储过程降低网络开销 408
 - 9.5.9 在高并发环境下使用连接池 408
 - 9.5.10 使用 Design Advisor(db2advis) 建议索引 408
 - 9.5.11 提高批量删除、插入和更新速度 409
 - 9.5.12 提高插入性能 409
 - 9.5.13 高效的 SELECT 语句 410
- 9.6 高性能 SQL 语句注意事项 412
 - 9.6.1 避免在搜索条件中使用复杂的表达式 412
 - 9.6.2 将 OPTIMIZE FOR *n* ROWS 子句与 FETCH FIRST *n* ROWS ONLY 子句配合使用 412
 - 9.6.3 避免使用冗余的谓词 412
 - 9.6.4 避免使用多个带有 DISTINCT 关键字的聚集操作 413
 - 9.6.5 避免连接列之间数据类型不匹配 414
 - 9.6.6 避免对表达式使用连接谓词 414
 - 9.6.7 避免在谓词中使用空操作表达式来更改优化器估算 415
 - 9.6.8 确保查询符合星型模式连接的必需条件 415
 - 9.6.9 避免使用非等式连接谓词 416
 - 9.6.10 避免使用不必要的外连接 417
 - 9.6.11 使用参数标记来缩短动态查询的编译时间 418
 - 9.6.12 使用约束来提高查询优化程度 418
- 9.7 本章小结 419

第 10 章 DB2 集群调优	421	10.6.2 减少通信影响	447
10.1 DB2 集群介绍	421	10.6.3 热点表调优案例	447
10.2 DB2 集群参数解析	423	10.7 本章小结	450
10.2.1 组缓冲池	423	第 11 章 DB2 调优案例、问题总结和技巧	451
10.2.2 全局锁管理器	425	11.1 调优案例 1: 某移动公司存储设计不当和 SQL 引起的 I/O 瓶颈	451
10.2.3 DB2 pureScale 集群相关参数	425	11.2 调优案例 2: 某银行知识库系统锁等待、锁升级引起性能瓶颈	458
10.3 DB2 集群性能监控	429	11.3 调优案例 3: 某汽车制造商 ERP 系统通过调整统计信息提高性能	466
10.3.1 查看 CF 资源利用	429	11.4 调优案例 4: 某农信社批量代收电费批处理慢调优案例	476
10.3.2 查看各个成员的负载情况	430	11.5 调优案例 5: 某银行系统字段类型定义错误导致 SQL 执行时间变长	480
10.3.3 查看缓冲池命中率	430	11.6 调优案例 6: 某银行客户回单系统 CPU 使用率高	483
10.3.4 查看全局锁性能	438	11.7 调优案例 7: 某银行手机银行系统 latch 竞争导致 active session 高、性能慢问题	488
10.3.5 查看页回收(Page Reclaiming)行为	441	11.8 调优学习案例: 利用压力测试程序学习 DB2 调优	492
10.4 DB2 集群设计调优	442		
10.4.1 使用小的 pagesize	442		
10.4.2 使用大的 extentsize	442		
10.4.3 使用 LOB inline 方法	442		
10.4.4 使用大的 pctfree 设置	443		
10.4.5 巧用 CURRENT MEMBER	443		
10.4.6 巧用随机索引	444		
10.5 同城双活集群介绍	444		
10.6 同城双活集群调优	446		
10.6.1 减少存储影响	446		

性能调整概述

为什么要进行性能调优呢？因为我们的应用系统在运行一段时间后，用户报告系统运行会变慢，使他们不能完成所有的工作，完成事务和处理查询花费了过长的时间，或者应用程序在一天中的某些时段变慢。要确定造成问题的本质原因，必须评估系统资源的实际使用情况并进一步地分析资源使用的瓶颈所在。

用户通常报告以下性能问题：

- 事务或查询的响应时间比预期的长
- 事务吞吐量不足以完成必需的工作负载
- 事务吞吐量减少

为了维持数据库应用程序的最优性能，应该制定一个计划用于评估系统性能，以便在性能出现问题时，该计划可以根据性能问题的情况对数据库做出调整，以维持良好的性能。定期的、特定的评估能够帮助您预见并纠正性能问题。通过尽早识别出问题，可以有效地防止这些问题严重地影响用户。

本章我们将通过一个实际生产中的客户案例，展开对性能调整的讲解。本章主要内容包括：

- 性能概述
- 性能目标
- 性能评估
- 性能模型

- 什么时候需要性能调整
- 性能调整准则
- 性能调整的方法和过程
- 保持良好性能

1.1 性能概述

首先让我们以一个真实银行的应用案例来展开性能调整的话题，图 1-1 和图 1-2 分别为该行手机银行系统的逻辑架构部署图和物理架构部署图。

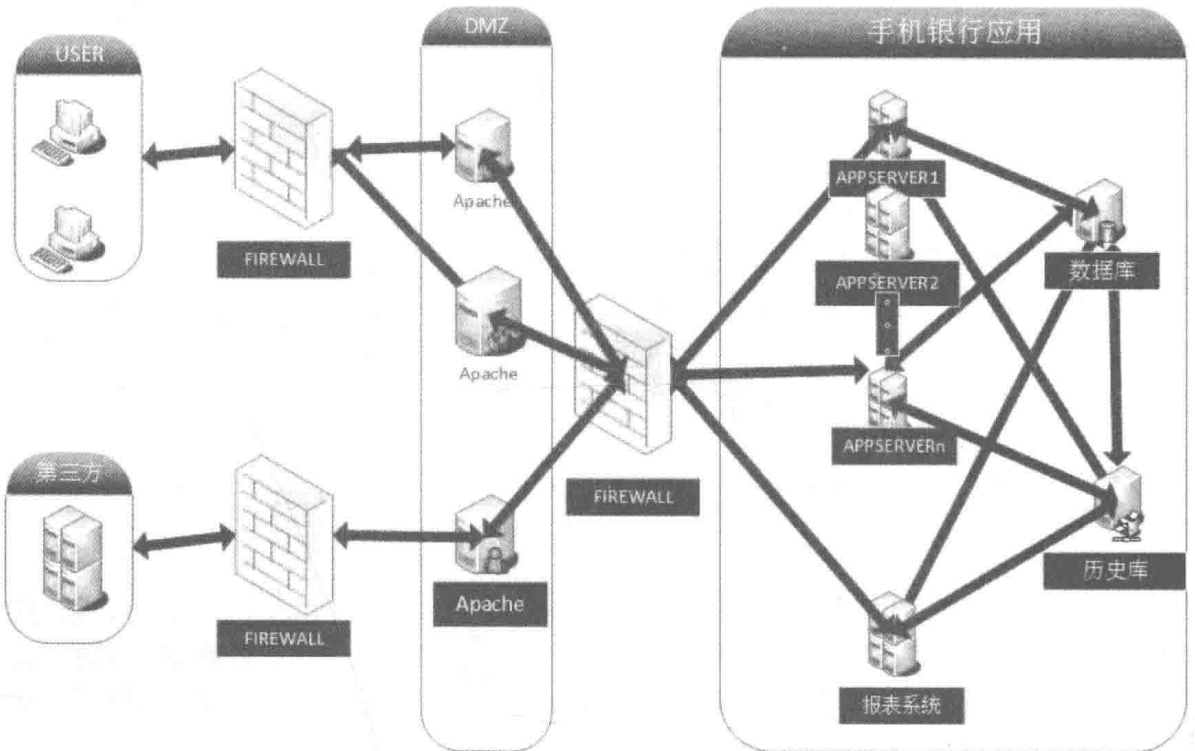


图 1-1 某手机银行系统的逻辑结构图

图 1-2 中的手机银行系统是一个大型的复杂系统。在这个系统中，从上至下包括以下几个层次：应用程序、中间件应用服务器、数据库、主机系统(操作系统)、光纤交换机和 SAN 存储网络(EMC VMAX)。在系统发生性能问题时，性能问题的定位和调优很复杂。

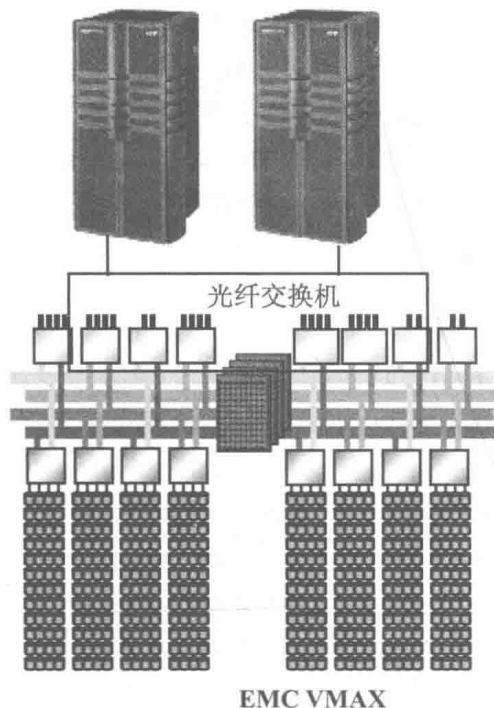


图 1-2 某手机银行系统的物理结构图

该手机银行系统的信息架构如下：

- 存储：采用的是 EMC VMAX 存储系统
- 操作系统：IBM AIX 7.1 (p750 lpar)
- 数据库：IBM DB2 9.7
- 中间件：Oracle Weblogic
- Web 服务器：Linux Apache Server
- 应用：采用基于 J2EE 的 Java 应用

该系统自从 2013 年上线以来，业务量逐年增长，运行至今，对最终用户的交互响应速度不如预期。在主机系统上观察，CPU 较忙，在业务繁忙时一直保持在 50% 左右。手机银行系统由业务应用程序、DB2 数据库、AIX 主机、EMC 存储多个部分组成，因此性能瓶颈的定位和性能的优化都比较复杂。

那么我们如何来解决这些性能问题呢？有两种方法：

- 第一种方法是我们可以通过扩容硬件物理资源(增加 CPU、内存以及购买更快的存储系统)来实现。
- 第二种方法是我们试图对应用系统做出相应的调整来优化系统以改善目前的情况。