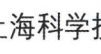
大数据工程技术与应用

数据库性能管理与调优

[韩] 金　范 —— 主编

上海科学技术出版社



大数据工程技术与应用

数据库性能管理与调优

[韩] 金　范　主编

上海科学技术出版社

图书在版编目(CIP)数据

数据库性能管理与调优 /（韩）金范主编. —上海：上海科学技术出版社，2016.10
（大数据工程技术与应用）
ISBN 978-7-5478-3239-4

Ⅰ.①数… Ⅱ.①金… Ⅲ.①关系数据库系统 Ⅳ.①TP311.138

中国版本图书馆 CIP 数据核字(2016)第 206351 号

数据库性能管理与调优
[韩]金 范 主编

上海世纪出版股份有限公司
上海科学技术出版社 出版
（上海钦州南路 71 号 邮政编码 200235）

上海世纪出版股份有限公司发行中心发行
200001 上海福建中路 193 号 www.ewen.co
苏州望电印刷有限公司印刷
开本 787×1092 1/16 印张 7
字数 140 千字
2016 年 10 月第 1 版 2016 年 10 月第 1 次印刷
ISBN 978-7-5478-3239-4/TP·43
定价：27.00 元

内容提要

本书重点介绍数据库性能管理与调试的理论及应用。首先强调任何系统的分析、设计、开发、测试、操作阶段都需要进行性能管理，并系统分析了各个阶段的性能管理应该关注的不同对象及目标。在以 Oracle 数据库为基础构建的系统中，将性能调试分为操作系统级调试和 DBMS 级调试分别进行详细介绍。在 DBMS 级调试阶段，为了在短时间内以低成本得到数据库性能提升效果，最好的方式是进行 SQL 调试，索引和连接就是 SQL 调试中必不可少的重要存在，是快速提取表中数据的手段，但是使用不当很容易适得其反，所以书中结合具体例子介绍索引和连接的使用方法及注意事项。

本书适合从事数据库管理和优化的研究人员及技术人员参考使用。

大数据工程技术与应用

编撰委员会

丛书序

“数支配着宇宙”——毕达哥拉斯。

大数据技术，使这句2000多年前的哲言如此形象、如此真切；大数据技术，正以前所未有的发展速度变革着人类的认知、产业和生活。

当前，我国正处于创新驱动发展、产业全面转型升级的关键阶段，大数据既是新的经济增长点，更是推进创新与发展的利器。上海产业技术研究院以服务于成果转化和产业化为使命，较早开始了大数据的应用研究和服务工作，构建了大数据应用技术平台，针对重点行业开展了一批大数据应用研究，涉及数据建模、数据分析、数据安全和数据库管理相关软件开发、测试、评价等多个方面。本丛书的出版既是前期工作探索的分享，更是进一步服务于成果转化和产业化的一个尝试。

大数据应用与产业化急需大量的工程应用技术人才。本丛书主要面向大数据工程应用的广大科技人员，在内容上汇聚了不同国家和区域、不同专业和领域的专家智慧，侧重大数据工程化知识、最佳实践和实用技巧，力求可操作性、实用性。

由于大数据技术研究和应用是一个新兴领域，发展方兴未艾，本丛书在编撰过程中因编者的知识和经验局限，必然存在许多不当之处，敬请广大读者提出宝贵意见。

2016年9月

前 言

企业通过数据库对大量的信息进行管理和使用，并正在创造出巨大的利益。然而，如何有效地管理每天都在快速增长的信息，这无疑是一个难题，随着数据的日益增加，企业正面临着与性能问题相关的很多困难。

数据库性能低下的问题会摧毁企业与客户之间的信任，然而，此类问题也很难在一两天解决。因此，只有投入大量的时间和费用，才能解决性能问题。如此一来，也给数据库性能低下的企业造成了巨大的损失。

数据库是运行系统、网络、程序等多种要素相结合运行的系统。为了确保这种复杂系统的稳定性，必须采取多方位的分析和监控，以及相应措施并行的处理方式。大多数企业为了进行期间业务与客户服务而使用数据库。作为支持此业务的重要系统，随着时间的推移、数据与用户的增多，性能会渐渐减弱，成为各企业的一个大麻烦，特别是执行管理任务的 DBA 或系统管理人员更是陷入困境。因此，DBA 和系统管理人员在了解性能管理重要性的同时，也需要熟知 OS 调试、网络调试、数据库调试、应用程序调试等的实战知识。

性能问题的产生可能有多种原因，其中包括两个主要原因：一种是构建数据库时，由于以结果为主的构建与时间的计算，设计出的数据库性能欠佳，或未能构建出优化的数据库，这种情况下，随着时间的推移，便会渐渐出现性能低下的问题；另一种是访问数据库的 SQL 未能实现优化，从而导致性能的低下。性能管理并非只单纯地出现在系统的运行和使用中，在对任意系统进行分析、设计、开发、测试、运行的阶段中，性能管理都是不可或缺的步骤。此外，以数据库为基础构建的系统，其各阶段均需要在考虑数据库相关事项后再进行操作。

本系列丛书阐述了操作现场可能出现的数据库性能管理方法，希望能够成为帮助初学者和专业人士的性能管理指南。

本书由金范编写，张青对本书进行了认真校对，周兆明、王一帆、邱雯参与了资料的

收集、整理、录入等工作。此外，本书的编写得到了上海产业技术研究院大数据专家委员会等相关单位的大力支持和指导，上海产业技术研究院的组织协调也使本书得以顺利出版，在此一并表示衷心感谢。

金　范

2016 年 6 月

目 录

第1章

数据库性能管理

1.1 对数据库性能管理的访问

数据库系统是由操作系统(operation system，OS)、数据库管理系统、网络、应用程序等多种要素构成的应用系统。为使这种复杂的系统保持稳定的性能，须从多方面展开分析和监测，同时采取适当的措施。为了业务线和客户服务，大部分企业使用了数据库。随着数据和用户的增加，支持这种业务的系统性能不断降低，成为各企业的难题。管理此系统的DBA或者系统管理员尤其如此。在此，本书将为DBA或系统管理员介绍性能管理的重要性和OS调优、网络调优、数据库调优、应用程序调优等实战知识。

数据库性能出现问题时，一般有两种解决方法：

(1) 提高系统配置。例如，增加CPU和内存等OS中有限的资源，更换性能良好的磁盘，用性能更好的机器替换系统本身等方法。

(2) 进行调优。即在保持原本有限资源的前提下，多方面调优内存、CPU、磁盘、数据库、应用程序，以保证工作效率。

性能下降的主要原因有应用程序的结构问题、数据和并发用户数量持续增加、新应用程序增加等。由用户增加导致性能下降时，可以通过增设硬件资源轻松解决，但如果是随着时间增加而产生的数据、系统、数据库结构等方面的问题，则无法通过增加硬件解决。这种情况下，则需要进行调优，即性能管理。通常经数据库性能管理专家的调优，性能可以提高20%～50%，不用增设特殊硬件便可达到满意的结果。

1.2 按项目阶段进行性能管理

并不只在运行和使用系统时才需要性能管理。任何系统的分析、设计、开发、测试、操作阶段都需要进行性能管理。此外，以Oracle数据库为基础构建的系统，所有阶段均需要参考数据库相关的内容进行操作。

1) 分析阶段

在分析阶段进行分析时需考虑整体性能和稳定性，此时业务流程优化、系统结构(technical architecture)设置、容量计算(capacity)非常重要。在业务流程优化期间，系统进行电算化的同时改善低效率流程，以提高整体性能。系统结构要先考虑事务处理量、稳定

性、维护等再确定结构，容量计算要先通过应用分析出待构建业务的事务、并发用户数、数据的增加值等的预期值再进行计算。需要了解的是，并不需要准确地计算出容量，只需使用大致的经验值进行计算即可。开发与过去 BMT(Bench Mark Testing)结果类似的系统时，大致的经验可以成为测试资料等宝贵的参考资料。此外，计算容量时最好留出多余的空间。将来开放系统后，相对于空间不足，有多余的空间将更有利于计算出更准确的容量。

2) 设计阶段

相比逻辑性设计，在进行数据物理设计时需要考虑与性能相关的操作。以逻辑性设计时导出的 ERD(entity relationship diagram)为基础，构建系统结构和性能，并进行物理设计。此时须考虑请求响应时间、分布式数据库环境、并发用户数、数据大小、批量处理等。当然，应用程序设计同样与数据库息息相关，因此设计应用程序时要使其发挥最佳性能。

3) 开发阶段

在开发阶段，为有效构思 SQL、PL/SQL 等，需要提高开发者的能力。此外，开发者对数据库优化的理解并非仅仅是通过 SQL 得出结果，而是构思可最小化内部处理量的 SQL，以便对整个系统的性能产生良好影响。

4) 测试和运行阶段

最后的测试和运行阶段可执行的操作包括应用程序调优、数据库调优、OS 调优等。开发和测试阶段的调优非常重要，足以决定系统开放的成败。开放之前调优得越多，开放后就越稳定。开放系统时，很多客户时常因应用程序完成度、结构上的问题、性能问题等原因而推迟开放系统，这是由于分析和设计阶段未能完美执行。以上介绍了各个阶段需要考虑的调优重点。现在开始介绍开发、测试、运行阶段要考虑的与数据库联动的 OS 和网络调优。

按项目阶段进行数据库性能管理如图 1-1 所示。

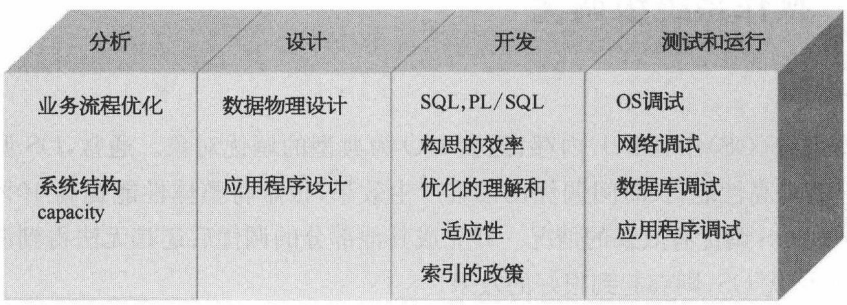


图 1-1　按项目阶段进行数据库性能管理

1.3　定期数据库调优

系统管理员必须了解“要持续定期执行性能管理，即调优”。一次性的调优可以改善当

前状态，但随着时间的推移，还是会因数据库增加和使用环境变化而出现性能下降的情况。因此管理员要定期执行 OS、数据库、应用程序调优。调优时必须谨记以下事项：

(1) 设定准确的目标；

(2) 在项目阶段中，越早的完成阶段调优费用越低；

(3) 根据环境变化，反复定期执行调优；

(4) 充分了解系统环境和功能；

(5) 逐渐更改要变更的部分并记下变更内容，然后比较变更前后的情况；

(6) 负责性能管理的人员必须具有充分的权限；

(7) 开放系统前应进行充分的性能测试；

(8) 调优并非片面，而是综合的技术。需要考虑各个方面。

当然，从项目一开始就考虑整体性能最有效，且成本更低。但大多数客户都是在开发结束向用户开放系统后才意识到性能问题并且查找解决方法。此时进行无缝调优和有效调优为时已晚，只能花费较多费用进行设计阶段的调优，或者进行应用程序的调优(根据作者的经验，90%以上的客户遇到此类问题)。在调优过程中，应用程序调优是最为有效且易于访问的部分，通过修改无效访问路径，生成优化的数据库对象，创建有效索引策略等操作，可将性能提高数十倍至数万倍。图 1-2 所示为各类别调优的效果。

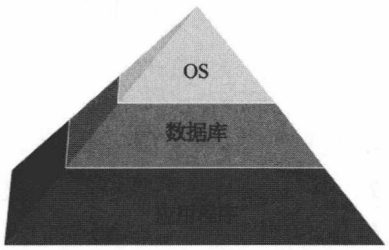


图 1-2　各类别数据库调优效果

1.4　操作系统级调优

在操作系统(OS)中，CPU、内存、磁盘 I/O 为典型的调优对象。通常，OS 调优一般情况下很难显著提高性能，长时间调优也很难产生效果，很难将整体性能提高 10%以上。但有时常会出现 OS 调优有效果的情况。即完成其他部分的调优后还是无法得到满意的结果时，通过仔细调优 OS，则会起到很好的效果。

1.4.1　CPU 调优

使用 UNIX 时，可通过 sar 命令(图 1-3)确认 CPU 的使用量。正常运行时，CPU 建议使用程度为 70%～80%，其中需维持 20%～30%的空闲时间。如果空闲未达到 20%～30%，系统使用量增加时(最后期限、结算、预算编制等)，空闲可能为 0%(此时空闲为 0%可能表示 100%使用 CPU，但使用超过 100%会判定为发生 CPU 瓶颈现象)。

如图 1-3 所示，使用 sar 指令监测 CPU 的使用量时，通常%idle 要闲置 20%～30%，

但%idle 比该值低时，则判定为当前 OS 的 CPU 发生不足现象。判断为 CPU 不足时，可以找到问题所在并加以解决，或者考虑增设 CPU。CPU 使用量过大时，则先比较%usr、%sys、%wio 的比率。使用数据库应用程序运营的系统，正常情况下显示顺序为%usr＞%sys＞%wio。如果%wio 的值最高，则可以视为 I/O 等待在占用 CPU。因此，消除 I/O 瓶颈现象或提高 UNIX 文件系统的工作缓冲区缓存的利用率，会减少整个 CPU 的使用量，这比提高缓冲区缓存的利用率，分配 I/O 更为有效。如果%sys 的使用量高，可能是异常进程占用太多 CPU。在这种情况下，找出当前系统上大量占用 CPU 的进程并分析原因。图 1－4 所示为在 sun micro systems 设备上使用 Berkeley ps 指令按顺序查看大量占用 CPU 进程的方法。判断方法分别是 HP 为 top 或 glance，IBM 为 monitor，DEC 为 ps aux command。

```
$ sar 3 5
11:01:10    %usr    %sys    %wio   %idle
11:01:13      41       6      13      40
11:01:16      43       7      12      38
11:01:19      43       6      10      40
11:01:22      41       8      11      39
11:01:25      41       7      13      39
11:01:28      41       7      12      40

Average       41       7      12      40
```

图 1－3　sar 指令

```
$/usr/ucb/ps -aux | head -10
USER   PID  %CPU %MEM  SZ     RSS    TT    S START    TIME    COMMAND
oracle 23064 6.0  36.1 1471248 1450904 ?     S 10:30:37  0:08    ora805 (DESC
root       3 0.6  0.0       0       0 ?     S Nov 24    2124:24 fsflush
oracle 13598 0.4  35.8 1471192 1439136 ?     S Jan 19    0:16    ora_lgwr_805
oracle 13618 0.3  35.8 1471136 1439856 ?     S Jan 19    0:02    ora_arc0_805
oracle 13600 0.3  35.8 1471216 1438928 ?     S Jan 19    14:24   ora_ckpt_805
oracle 13596 0.1  35.8 1471688 1439000 ?     S Jan 19    0:03    ora_dbw0_805
root   23093 0.1  0.1    1656    1240 pts/3 0 10:31:13  0:00    /usr/ucb/ps -aux
root     586 0.1  0.3 1415210     416 ?     S Nov 24    126:51  /opt/VRTSvmsa/jre/
oracle 23063 0.1  0.2   10376    5928 pts/4 S 10:30:37  0:00    sqlplus scott/tiger
oracle 22859 0.1  36.1 1471808 1450568 ?     S 10:21:57  0:01    ora805 (LOCA
root     605 0.1  0.1    2744    1600 ?     S Nov 24    53:04   mibiisa -r -p 3278
oracle 22861 0.1  36.1 1471736 1450488 ?     S 10:21:58  0:01    ora805 (LOCA
oracle 23070 0.1  0.1    1032768 pts/3 S     10:30:58    0:00    /bin/sh top
root     452 0.0  0.1    4592    3664 ?     S Nov 24    3:17    /usr/sbin/nscd
```

图 1－4　查看大量占用 CPU 的进程

在前面的结果中，需要仔细查看%CPU、%MEM、TIME，它们分别为 CPU 占用率、内存占用率、启动后 CPU 的累积使用值。此外，最上方显示的进程占用 CPU 最多。如果一个进程占用一个 CPU 的 99%或 100%并使用较长时间(1 min 以上)，需要确认该进程是哪种进程并且进行何种操作。这种进程肯定为异常进程，属于不必要地占用系统资源。要在 OS 上分析任何进程，需要确认该进程的系统调用(system call)，此时可使用 truss(sun OS)、tusc(HP)指令等。图 1－5 所示为使用 truss 指令监测之前进程中占用 CPU 最多的 23064 进程的情况。该进程工作正常，无异常。

如果该进程异常占用较多 CPU，则系统调用反复循环的情况非常多，须更加注意。有时使用 truss 不会输出或显示任何对象。此种情况为不经过系统调用而在用户模式下执行

```
$truss -fp  23064
23064:  semop(2031616, 0xFFBE669C, 1)   (sleeping...)
23064:      Received signal #14, SIGALRM, in semop() [caught]
23064:  semop(2031616, 0xFFBE669C, 1)              Err#91 ERESTART
23064:  sigprocmask(SIG_BLOCK, 0xFFBE62A0, 0x00000000)  = 0
23064:  times(0xFFBE6210)                                = 707179313
23064:  sigprocmask(SIG_UNBLOCK, 0xFFBE62A0, 0x00000000) = 0
23064:  getcontext(0xFFBE6060)
23064:  setcontext(0xFFBE6060)
23064:  sigprocmask(SIG_BLOCK, 0xFFBE657C, 0x00000000)  = 0
23064:  setitimer(ITIMER_REAL, 0xFFBE6504, 0x00000000)  = 0
23064:  sigprocmask(SIG_UNBLOCK, 0xFFBE657C, 0x00000000) = 0
23064:  semctl(2031616, 8, 8, 1)                         = 0
23064:  getcontext(0xFFBE6468)
23064:  sigprocmask(SIG_BLOCK, 0xFFBE657C, 0x00000000)  = 0
23064:  times(0xFFBE6500)                              = 707179313
23064:  setitimer(ITIMER_REAL, 0xFFBE6504, 0x00000000)  = 0
23064:  sigprocmask(SIG_UNBLOCK, 0xFFBE657C, 0x00000000) = 0
23064:  semop(2031616, 0xFFBE669C, 1)                    = 0
23064:  sigprocmask(SIG_BLOCK, 0xFFBE657C, 0x00000000)  = 0
23064:  setitimer(ITIMER_REAL, 0xFFBE6504, 0x00000000)  = 0
23064:  sigprocmask(SIG_UNBLOCK, 0xFFBE657C, 0x00000000) = 0
23064:  semctl(2031616, 8, 8, 1)
```

图 1-5　根据进程确认系统调用

的情况(例如,for 语句在同一个地方不停地执行)。truss 命令会显示该进程执行的所有系统调用,因此该命令执行操作时会增加负载,只能在短时间内使用 truss 监测进程。如果对任何系统调用都有问题,则可以使用 UNIX man 指令确认系统调用执行了何种操作(例如:$ man semop)。如果占用较多 CPU 并且分析结果显示进程有异常时,则确认该进程关联的业务后先停止该进程。这种进程为常进程,多数为只占用系统资源的进程。因此为保证系统正常运行,如果已收集了各种信息则停止进程。如果系统包含四个 CPU 并且有两个活跃的异常进程,则已占用系统 50%的 CPU,系统不可能正常工作。

1.4.2　内存调优

为克服物理内存的限制,UNIX 系统使用虚拟(virtual)内存的概念运行内存,虚拟内存的大小为"物理内存+虚拟磁盘(交换磁盘)"。虚拟内存由一组 page 构成,page 的大小通常为 4 KB(8 KB)。Page 为内存的分页单位,由 UNIX 的 page daemon 管理,为保持尽可能多的可用内存,将当前不使用的 page 保存到磁盘(页面调出)中。如果当前的某个进程请求物理内存中不存在的 page(缺页错误),则会将磁盘中保存的 page 作为内存读取(页面调进)。简单来说,交换操作比分页的使用范围更广泛。如果说分页是以 page 为单位的管理,交换则管理进程使用的所有 page。因此,要重新恢复该进程,所有相关的 page 都需要重新调进到内存中,对性能有很大的不良影响。为保证系统正常运行,绝对不能发生交换情况,并要避免同时发生分页的情况。因此,监测内存时,分析的重点主要是分页和交换的有无