DOI: 10.13657/j.cnki.gxkxyxb.2005.s1.036

Oracle9i数据库的性能优化方法
The Optimized Method of Oracle9i Database Performance

陈 迪,陈 琴
Chen Di, Chen Qin

(广西大学计算机与电子信息学院,广西南宁 530004)
(School of Comp., Elec. and Info., Guangxi Univ., Nanning, Guangxi, 530004, China)

摘要:介绍数据库性能优化的基本原则,并从设计阶段和运行阶段论述了Oracle9i数据库性能优化的方法,结合实例,给出Oracle9i数据库对系统全局区、磁盘输入输出和SQL语句的优化过程。Oracle9i数据库优化前后的效果比较表明,优化后的查询效率和CPU占用率明显改善,SQL语句执行时间减少。
关键词:数据库 性能优化 Oracle9i Cache命中率 SQL语句
中图法分类号:TP311.13 文献标识码:A 文章编号:1002-7378(2005)S0-0109-03

Abstract This paper introduces the basic principle of database performance optimizing. And the optimized method of database performance in design stage and run stage is discussed. The optimized process of Oracle9i database to System Global Area, input/output of disk and SQL statement is given by an example. The result of before and after optimizing verifies that after optimizing the query efficiency and CPU-take-up rate are improved obviously and the time of carrying out SQL statement is reduced.

Key words database, performance optimizing, Oracle9i, cache hits ratio, SQL statement

随着网络应用和电子商务的不断发展,各个站点的访问量越来越大,数据库规模也随之不断的扩大,数据库系统的性能问题越来越突出[1]。

Oracle数据库在我国电信、金融、证券等关键领域的计算机系统中有着非常广泛的应用,但由于Oracle系统结构复杂,性能受多方面因素影响,数据库管理员必须通过优化措施保证数据库以最优的性能运行。

本文介绍Oracle9i数据库的优化方法,通过Oracle数据库实例给出对系统全局区、磁盘输入输出,以及SQL语句的优化过程,并比较优化前后的效果。

1 Oracle9i数据库性能优化原则和方法

数据库性能优化的基本原则是:通过尽可能少的磁盘访问获得所需要的数据。对Oracle数据库进行性能调整时,应当按照一定的顺序进行,因为系统在前面步骤中进行的调整可以避免后面的一些不必要或者代价很大的调整。一般来说可以从两个阶段入手。

1.1 设计阶段

对其逻辑结构和物理结构进行优化设计,使之在满足需求条件的情况下,系统性能达到最佳,系统开销达到最小。这个阶段的优化包括调整应用程序结构设计;恰当使用分区、索引及存档功能;恰当编写访问数据的SQL语句;调整硬盘I/O;确定数据块大小和存储参数。

1.2 运行阶段

采取操作系统级、数据库级的一些优化措施来使系统性能最佳。这个阶段的优化包括:(1)操作系统级的调整:实施操作系统级调整的主要目的是减少内存交换,减少分页,使SGA(System Global Area)可留驻内存;(2)数据库级的调整,包括SGA的分配及使用效率;I/O和资源竞争。

另外,在Oracle中,需要采用一些机制来保证数据库对象在使用期间的稳定性和数据的一致性,如使用锁存器(latch)、锁(lock)等。因此争用与这些机

收稿日期:2005-09-11
作者简介:陈 迪(1980-),男,广西南宁人,本科生。

制相关的资源会影响数据库的性能。为了减少这种资源竞争,可以通过调整数据库的相关初始化参数来减少资源的争用,优化数据库性能。

2 Oracle9i数据库优化过程

现有一个采用 Oracle作为后台数据库的售后服务库存管理系统,系统运行一段时间后,随着数据量的上升,用户发现对表的查询的速度降低,为此我们对它进行如下的优化,以提高系统的性能。

本文选择我们的数据库 tongfang 中的 Sys_DepotOperateDetail 和 Sys_DepotOperateHeader 这两个表来进行查询并进行查询优化。这两个表的记录数都非常大,表 Sys_DepotOperateDetail中有 735039条记录, Sys_DepotOperateHeader 中有 195020条记录。

2.1 查看系统当前各项 Cache命中率

使用 SQL性能脚本管理文件来查看系统当前各项 Cache命中率: (1) DBCache Hits Ratio(数据库高速缓存命中率)为 94.55; (2) LibraryCache Hits Ratio(库高速缓存命中率)为 98.56; (3) Dictionary Cache Hits Ration(数据字典高速缓存命中率)为 94.44。

如果 LibraryCache Hits Ratio < 0.99 和 Dictionary Cache Hits Ration < 0.9,必须对 SHARED_POOL_SIZE进行调整以提高数据库性能。在实验中 Dictionary Cache Hits Ration可以保持不变,但是 LibraryCache Hits Ratio命中率小于 99%,所以我们对 SHARED_POOL_SIZE的大小进行调整,命令如下:

```
SQL> alter system set shared_pool_size= 48M scope= both
```

2.2 使用 fileio查看 I/O情况

通过查看发现读写量很多,影响了速度,为此我们建立 3个不同表空间的数据文件(分别建立到 D、E、F盘上),并把原来的数据转移到不同的磁盘,这样可以增加磁盘的 I/O并发,同时减少对同一磁盘的 I/O并发次数。

2.3 使用 SQL语句

使用 SQL语句方面的优化是常见和有效的方法[2],一个小的 SQL语句调整都会使数据库的性能有所提高,在实验中常用的优化方式是: WHERE 后面的条件顺序。

WHERE子句后面的条件顺序对大数据量表的查询会产生直接的影响,例如在 Sys_DepotOperateHead 表中查询所有 LotNum= 2和 OperateType> 3的记录,以下的两个方法中,方法一比方法二效率低。

方法一: SELECT * FROM Sys_DepotOperateHeader WHERE OperateType > 3 AND LotNum= 2;

方法二: SELECT * FROM Sys_DepotOperateHeader WHERE LotNum= 2 AND OperateType> 3

因为上面两个 SQL中 LotNum及 OperateType两个字段没有进行索引,所以执行时为全表扫描,第一条 SQL的 OperateType> 3条件在记录集内比率在 90%以上,而 LotNum= 2的比率只为 1%左右。在进行第一条 SQL时, 90%条记录都进行 LotNum及 OperateType的比较,而在进行第二条 SQL时,只有 1%条记录进行 LotNum及 OperateType的比较。由此可以看出,第二条 SQL的 CPU占用率明显比第一条低。

2.4 使用索引

使用索引可以极大地提高系统检索性能[3],建立索引主要是对经常出现在索引条件中的数据建立索引,在数据库中我们为 Sys_DepotOperateHeader 和 Sys_DepotOperateDetail表建立相关的索引,并将索引与用户表所在的表空间分开,用户表在 tongfang 表空间,索引在 index表空间上,这样可以减少 I/O并发次数,加快查询速度。

3 性能优化分析

本文选择一个数据量较大的表 Sys_DepotOperateDetail,通过执行查询语句 SELECT * FROM SA.SYS_DEPOTOPERATEDETAIL来比较优化前后的效果见表 1。

表 1 优化前后的效果比较

优化前后	SQL语句执行时间(s)	CPU占用率(%)	软解析(%)	解析实际运行效率(%)	查询实际运行效率(%)
优化前	72	22	90.45	85.28	90.37
优化后	46	16	93.45	89.28	93.37

表 1中软解析表示 SQL在共享区的命中率;解析实际运行效率表示解析实际运行时间 /(解析实际运行时间+ 解析中等待资源时间);查询实际运行效率表示查询实际运行时间 /(查询实际运行时间+ SQL解析时间)。

通过比较可以看到,经过优化后, SQL语句执

行时间和 CPU 占用率两项的优化效果比较明显,SQL语句执行时间减少 26s,CPU 占用率也减少了 6%,其他三项参数也都有一定的提高。比较结果表明,Oracle9i数据库性能优化的效果较好。

4 结束语

优化数据库整体的应用性能是提高计算机系统处理速度的一种行之有效的办法,本文从对 SGA(系统全局区)、磁盘 I/O 情况以及 SQL语句的优化等方面的优化进行讨论,结果表明优化后的数据库性能有所提高,但在实际应用中要根据系统的实际情况具体分析,对优化的方法反复实验,最后再确定最终的优化方法,这样才能达到较好的优化效果。

参考文献:

[1] 佚 名. Oracle 数据库性能优化技术 [EB/OL]. http://www.ddvip.net/database/oracle/index4/34.htm, 2005-03-18.

[2] 王秋生.基于 PL/SQL的 Oracle数据库性能优化 [J].微机发展, 2003, 13: 47-52.

[3] 薛 铭. Oracle性能调整及优化技术 [J].长春工程学院学报, 2002, 3: 70-72.

(责任编辑: 黎贞崇)

(上接第 108页)

声明,以及一个事务注册表,用于登记事务发起者和事务使用的连接的对应关系。通过该表,可以将使用事务的部分和连接管理部分隔离开来,因为该表是在运行时根据实际的调用情况动态生成的,事务使用的连接在该事务运行中不能被复用。当使用者需要使用事务方法时,首先调用连接管理服务提供的 beginTrans方法,该方法主要代码如下:

```
public void beginTrans( ) {
  ...
  conn = getFreeConnection( );    //从连接池中获取一个空闲连接
  userId = getUserId( );    //获取用户标识,它是唯一的
  registerTrans( userId, conn);    //将用户标识 ID和其获取的连接对应地放入注册表中
}
```

在实现过程中,用户标识用使用者所在的线程号来标识。所有对数据库的访问都通过查找该注册表和使用已分配的连接来完成。当事务结束时,从注册表中删除相应表项。

我们把连接池模块与执行数据库各种 SQL操作的模块分别封装在 DBPool.java和 DBsql.java中,以进一步提高代码的重用性与可扩展性,有利于进一步开发。

3 结束语

本文介绍了数据库连接的 JavaBean实现方法。在 JavaBean中我们用一般的数据库连接方法实现了 SQL的数据库连接,实现了代码重用,提高了开发效率。为了使系统有高效,可靠的数据库连接,我们在 JavaBean的基础上引入了连接池技术。通过高效的连接池管理策略,使数据库连接效率和安全性得到了极大提升,从而为企业应用系统的开发提供一个高效、方便、可靠、安全的数据库连接解决方案。

参考文献:

[1] 张晓东. JAVA数据库高级教程 [M].北京: 清华大学出版社, 2004.

[2] 王强兵,刘广钟.基于 J2EE的 Web企业计算 [J].计算机工程, 2002, 28(1): 262-264.

[3] 马廷淮,赵亚伟,刘 忠.用 EJB开发 J2EE应用 [J].计算机应用, 2002, 22(4): 111-113.

[4] Art Taylor[美]. JDBC数据库编程与 J2EE[M].李东升译.北京: 电子工业出版社, 2004.

[5] 崔莹峰,王 洪.数据库连接池技术及其在 web系统开发中的应用 [J].铁路计算机应用, 2005, 14(2): 48-50.

(责任编辑: 邓大玉)