Performance Tuning for Web Based Databases*

Wu Yue Qiu Huizhong Yu Shui Yu Yuanhui

(College of Computer Science and Engineering, UEST of China Chengdu 610054)

Abstract The traditional C/S database systems often suffer from the poor performance in the environment of Web applications because the number of users and the frequency or the times of accessing to a Web application are very difficult to be determined in advance. In this paper, a new method is proposed for the design, implementation and tuning of the Web based database system. This approach focuses on the configuration of RDBMS, the design of application model, and programming. By optimization on memory, on fragmentation, on disk parameters and on time related parameters, a better running environment of RDBMS can be set up. By reducing the CONNECT and DISCONNECT operations, creating suitable indexes, using more stored procedure and optimization on SQL, good performance can be got at programming step. By optimization on business logic and denormalization, the system performance can be greatly improved at design step.

Key words performance; optimization; Web database; tuning

Web数据库性能调优*

吴 跃** 邱会中 余 水 余元辉

(电子科技大学计算机科学与工程学院 成都 610054)

【摘要】提出了一种基于Web数据库系统设计、实现和性能调优的新方法，该方法包括由RDBMS配置优化、应用模式设计优化和编程优化。通过内存、分段、磁盘参数和时间相关参数的优化可以设置一个好的RDBMS运行环境。在编程时，通过减少与后台连接和断开操作、建立和使用索引、多用存储过程和优化SQL语句可以优化系统性能。在应用设计时，通过优化应用逻辑和降低规范化可以大大地改善系统性能。

关 键 词 性能; 优化; Web数据库; 调优

中图分类号 TP311

Nowadays, there are dramatic developments on the Web based database application, especially in the field of E-commerce. The developments not only change the mode of the applications, but also need more and more new progress in the analysis, design and implementation of the applications.

The database is at the core of any architecture meeting these needs. The new Web architecture needs a database that delivers:

1) A combination of object-relational technology and OLTP for updates and queries to text/

Received on August 30, 2001
2001年8月30日收稿
* The project supported by the National Natural Science Foundation of China, No: 69871005
国家自然科学基金资助项目，编号：69871005
** 男 44岁 在职博士生 教授

multimedia and relational data, as well as other dynamic data types such as geospatial and HTML;

2) A flexible architecture to integrate with existing systems and legacy data, while providing broad development support for Internet component standards such as Java, COM, and data-access gateways;

3) Scalability, availability, and rapid Web-application-development support[1].

More importantly, users of Web based databases need more efficient application; therefore the performance of a Web based database system is very critical. Unfortunately, the number of users and the frequency or the times of accessing to a Web based database are very difficult to be determined in advance. For this reason, the performance problems arise unpredictably and must be dealt with immediately.

There are a number of papers discussing the optimization of database. Some papers have discussed the technology of denormalization, by which the performance of database can be improved. Haus presented a list of normalization and denormalization types[2], and suggested that denormalization should be used according to the needs of data processing. Tupper developed two separate dimensions of denormalization[3]: one is to decrease the number of logical objects during the database logical design; another is to create entities or attributes for facilitating special requests. Date considered that denormalization should be processed at the physical storage level[4], rather at the logical or base relational level.

Some papers concentrated on the caching technology on Web based database. Divyesh discussed the technology of caching the database's data on Web to improve the performance[5]. Paul discussed the problem of cache consistency of Web based database[6].

All the papers, which we presented above, just discussed one or two aspects of the database performance. It is clear that the performance of the whole system depends on all parts of the system. The performance of the whole system may depend on the component with poor performance; therefore the optimization of the whole system involves every part of the system.

The paper is organized as follows: the optimization on relational DBMS is discussed in Section 1. In Section 2, the optimization on business logic and denormalization design is presented. Section 3 discuses the optimization on programming. In section 4, a method of Web based database design and implementation for the optimal performance is introduced, and finally, Section 5 summaries the paper.

1 Optimization on the RDBMS

RDBMS is the infrastructure of database application; therefore the performance of the database application is impacted by the configuration of the RDBMS. In order to meet the requirements of Web based database application, we need to setup the related configuration of the RDBMS, such as CPU affinity, process force residency, memory configuration, disk parameters, log buffer, log type, deadlock timeout, transaction timeout, and disk mirror, etc.

1.1 Optimization on Memory

Generally, memory is the critical component for system performance. Once the database server starts up, the memory is initiated, and it includes several parts, such as: stored procedure cache, lock queues, sort buffer, temporary buffer, logic log buffer, physical log buffer, and message portion, etc. There are about 100 parameters when database server software is started, which all need to be activated in memory, hence it is very important to set the suitable values to the memory related parameters. We can take use some methods as follows to optimize on memory.

If a Web database system needs no data security or weak data security, then we should set the logic

log buffer and the physical log buffer to a large value, and then we can use the memory completely to improve the performance.

If the Web database system needs to do a lot of SORT operations or JOIN operations, then by increasing the values of sort buffer or temporary buffer, so that the performance of the system will be improved because of its less disk I/O operations.

If a process in the memory of system hasn't been called for a long time, it will be stored to the disk, which is called as priority aging. We can disable the function of priority aging, and make the processes of database resident in memory even they are very "old". By setting this function on, we can also improve of the system performance.

1.2 Optimization on Fragmentation

Fragmentation is a technology to divide a table's content into several parts, and store every part in different disks. There are several kinds of methods to divide the content. One is to divide by the rows. We can divide the rows by an expression, a function, a hash function, and etc. In these ways, the DBMS can process the related database operations by parallel operation on I/Os, and then the system performance can be improved. Another method is to divide the table's content by columns. We can divide the columns into two or more parts, according to the using frequency of collectively accessed columns. In this case the query operations will not take all the columns of the table into memory, hence it reduces the I/O operations, and further, reduces the pressure to memory, and finally, the system performance can be improved.

1.3 Optimization on Disk Parameters

The disk system is very important for a database application system, and it has a critical role in the system performance. All data, such as user data, meta data, and program code, are stored in the disk. There are a lot of reading or writing operations on disk almost all the time once the database server is started, therefore how to configure the disk related parameters is important to system performance.

We can improve the system performance by distributing the system database and the application database into different disks. For any relational DBMS, there is a subsystem to keep the information of database itself, namely meta data. Generally, we name it as the system database, on the other hand, we name the database system, on which we design and implement for our customer, as the application database. System database is indispensable for a relational DBMS. Almost every database operation needs the help of system database; therefore system database is accessed most frequently. It is wise to dedicate a disk for system database to reduce the I/O bottleneck.

By assigning tables to different I/O can benefit to performance of the application database. All modern mainstream relational DBMSs support parallel processing. If we have several I/Os for the application database system, then we can group the tables into the I/Os according to the principle of I/O balance. Besides, we should assign the most frequently accessed table into the fastest I/O, and so on.

Page size is another important parameter for system performance. Page is the basic unit for I/O operation. In different applications, we should adjust the parameter, especially in a very large application database system. Sometimes we can get 3 or 4 times progression only by tuning the parameters.

The position of data on disk also has influence on system performance. As we know, the behavior of disk head to read or write data from or to disk is a mechanical movement, and the most efficient position of reading or writing is the middle of the disk. When we allocate the space for database system on hard disk, we should allocate the most frequently accessed table at the middle of hard disk.

Disk mirror can also improve the performance besides getting more security on data and reducing downtime. Because there is a copy of the application database's data in mirrored system, then if there is a reading operation, the relational DBMS, such as Informix Release 7, will divide the operation to the two mirrored I/Os, therefore we can obtain almost 2 times increment in query performance. Disk mirror technology fits the Web based database systems because most of operations on a Web based database system are query-oriented.

1.4 Optimization on Time Related Parameters

There are several time related parameters, such as CHECKPOINT INTERVAL, TRANSACTION TIMEOUT, and DEADLOCK TIMEOUT, etc. Suitable configuration of these time related parameters will improve the system performance.

Once there is a checkpoint, the response time of a database system decreases dramatically. To improve the performance, we should set the interval of checkpoint as long as possible, but on the other side, if we set the interval too long, it creates the problem of data security.

TRANSACTION TIMEOUT is a threshold value for transaction management, when the time of a transaction last arrive the threshold and the transaction is not committed, then the transaction will be rolled back. In an environment of limited bandwidth network, we should setup the threshold of TRANSACTION TIMEOUT longer; it is helpful to complete the users' requirements. DEADLOCK TIMEOUT is another threshold value for process management. If a process beyond the value of DEADLOCK TIMEOUT does not respond the time of a requirement, the system assumes the resource required by the process is occupied; therefore the system kills the process to release the resource. Adjusting the parameter will benefit system performance.

1.5 Startup Multi-instance on One Hardware Platform

Most of the mainstream relational DBMS products support multi-instance on one hardware platform. This can not only enhance the security of data accessing, but also improve system performance, because we can configure dedicate instances to do special jobs. In some cases, we can startup multi-instance on very limited hardware resources.

2 Optimization on Business Logic and Denormalization

Denormalization relations can improve performance, too. Generally, a relational database system design should be at least 3NF for minimizing data redundancy; but that is at the expense of performance. On the contrary, to be more efficient, we will break the rule in some degree, especially in the environment of a Web based database system.

In the environment of a Web based database system, query operations are used frequently; therefore we present several methods to denormalize the relation for query performance.

The first method is to calculate the intermittent result for frequently queried requirements beforehand, and save them in redundant tables, therefore it is not necessary to calculate the results for every query. For example, assuming a table in a communication company's database contains the detailed data of call quantities for one year, including call time, call date, and other information for the call. In fact, most of the customers just care about how many times he or she had in a certain month, therefore we can derive a table based on month, and obtain the calling number for each telephone number each month beforehand. So that it is not necessary to calculate in the detailed table every time for a customer's query. This method can greatly improve performance.

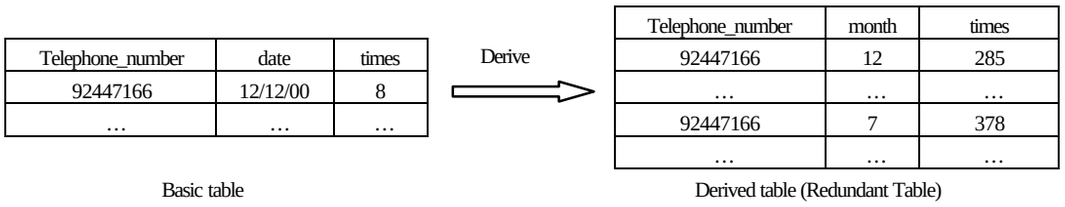

Figure1　Deriving redundant data to progress the performance

The second method is to add redundant attribute(s) to a relation to reduce calculation, and then improve the performance. For example, in order to keep with 3NF, sometimes a RDBMS needs to calculate some data for every row for a query. Therefore, we can add the transitively dependent attribute(s) to a relation for performance tuning. An example is showed in Figure 2.

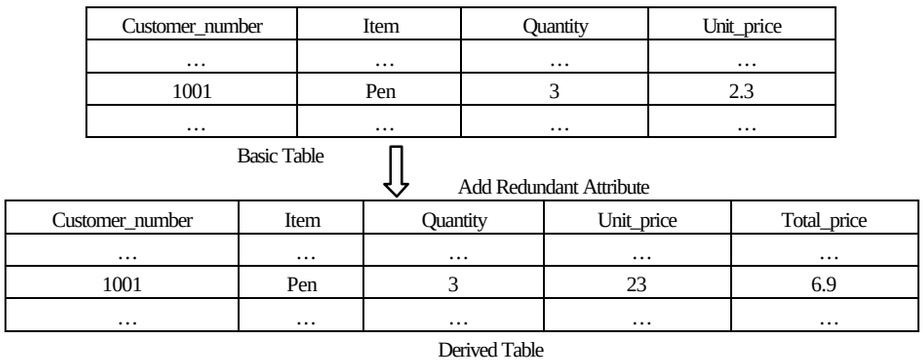

Figure 2　Adding redundant attribute(s) to improve performance

The third method of denormalization is to overlap attributes between different relations. The JOIN operation is a time consuming operation. If we concatenate the related attributes, which we hope to join together, into one relation, then it is not necessary to do the JOIN operation, and for this reason, we obtain better query performance. For example, assuming there are two relations, STUDENT (s_number, s_name, gender, height, telephone,) and COURSE(c_name, s_number, score). If we want to find a list of c_name, s_name, and score, we need to join the two relations. If we add the s_name attribute of relation STUDENT to relation COURSE, then it is just a project operation on relation COURSE to obtain what we want. Because of no JOIN operation, the query performance is better than the last one.

3　Optimization on Programming

In the circumstance of a Web based database system, most programming tasks are focused on middleware programming. We can build JAVA programs to access Web based databases through JDBC, or PHP programs to access Web based database through API, or even CGI programs to do so. Whichever model or programming language we choose, for performance consideration, we should pay attention to the following aspects.

3.1　Reducing the CONNECT and DISCONNECT Operations

It can improve the performance by reducing the CONNECT and DISCONNECT operations. Because the CONNECT and DISCONNECT operations are the time-consuming operations among all related database operations. Consequently in middleware programs, the developers should reduce the CONNECT and DISCONNECT operations as possible as they can.

3.2　Creating Suitable Indexes

Indexing is an efficient method to speed up the query operations. When there are more than 10 000

rows in a table, the query effect is obviously improved on the indexed table; sometimes we can get up to 15% increase in performance.

Although index is beneficial for query operation, unfortunately it is bad for the other database operations (such as, INSERT, DELETE and UPDATE). For example, once we attempt to insert a row to the table, we must do an INSERT operation in the index first, and then insert the data into the table itself.

3.3 Using More Stored Procedure

A stored procedure is a segment of program which is stored and executed on the database server and can be called by client programs. Using stored procedures can reduce the pressure on network, and its execution is efficient for it is executed on the powerful server machine, and it does not need to be compiled every time.

Another advantage of using stored procedure is that the stored procedure can isolate the application from the database schema. For example, if the name of an attribute is changed, the developers just need to modify the store procedure, instead of modifying applications.

3.4 Optimization on SQL

Currently, the mainstream relational DBMS have optimizers, which can execute the optimization on SQL sentences; therefore it is not necessary for the developers to pay attention to the SQL sentences in their programs. But the optimizer still needs the help of DBA. For example, in INFORMIX Release 7, it is necessary for the DBA to run UPDATE STATISTICS command from time to time to update the statistical data for optimization.

4 A Method for Web Based Database Design, Implementation and Perfor-mance Tuning

The Web based database model is different from the traditional Client/Server model, and posses many new features, including the following aspects,

1) Changing in database accessing method. The traditional two-tier model takes use of Client/Server architecture, but the Web based database model is based on three-tier model, it access the backend database with the help of Web server.

2) Changing in requirement analysis. In the traditional requirement analysising, the analysisor depends on all the business experts of the customer. For a Web based database project, it is almost impossible for the analysisor to depend on all the users over the world.

3) Changing in hardware environment. The hardware environment of Web based database model is based on Internet, a public communication platform. Most of the time, it is difficult to ensure the hardware environment to meet the requirement of the Web based database system.

4) Changing in data types. The Web based database application is an integrated system; usually it includes more data type, such as photo, picture, voice, text, and hypertext, etc.

Based on the above analysis, we introduce a new method for Web based database design, implementation and tuning. The details of the method are described as follow:

Step1 decide or optimize the hardware environment, and make sure that it does not form the bottleneck of the whole system.

Step2 design for the new requirements of the Web based database system according to the 3NF. Designers forecast the data volume and performance requirement, and based on the result, arrange the I/Os and divide the tables. Take use of all the methods, which we mentioned in the paper to implement or

perfect the system.

Step3 use performance diagnostic tool to supervise the performance from time to time when the Web based database system is running, and try to optimize the performance if it is necessary. And can process optimization according to the follow steps.

1) Start up database server by the default or current parameters.

2) Get the performance records by performance diagnostic tool.

3) Modify one parameter, if the performance becomes better, modify the parameter in this tendency, otherwise, modify the parameter in the vice direction, until there is no obvious progression.

4) Repeat 2), until there is no obvious improvement in performance for the whole system.

Step4 Add redundant data and denormalize the database system to improve the system perfor-mance.

Step5 If there is no improvement on the performance, begin the next cycle of the system design, implementation, and tuning.

5 Conclusion

Performance optimization acts an important role in the Web based database system. In traditional Client/Server database system, performance optimization is always one of the important problems that a database administrator should face. More over, in the Web based database system, there are more aspects cannot decide during the period of database design, such as data volume, accessing frequency, and the detail requirements of customers, etc. Consequently, the performance optimization is critical for the whole system to meet the performance requirements of customers.

In this paper, we present our conclusion of design, implementation, and optimization for Web based database system: Prototyping techniques are suit for the design and implementation of Web based database system. Besides, the system optimization for Web based database system is very important, it is indispensable in the software life cycle of Web based database system.

References

1 Wu Yue, Yu Shui, Fu Yan, *et al.* On Internet database accessing technology. Journal of University of Electronic Science and Technology of China, 2001,30(1):58~61 [吴 跃, 余 水, 傅 彦, 等. Internet 数据库访问技术. 电子科技大学学报, 2001, 30(1): 58-61]

2 Hanus M. To normalize or denormalize, that is the question. CMG Proceedings, 1 1994, ublby CMG, Chicago, IL, USA

3 Ricardo C. Database Systems: Principles, Design, & Implementation. Macmillan Publishing Company, 1990

4 Date C.J. The birth of the relational model. Intelligent enterprise magazine, 1998

5 Jadav D, Gupta M. Caching of large objects in Web servers. The 7th International Workshop on Research Issues in Data engineering, 1997

6 Francis P, Sato S. Design of a database and cache management strategy for a global information infrastructure, The 3rd Int' l Symposium on Autonomous Decentralized System, 1997