

Combining Fujitsu's High-Performance, High-Reliability DBMS Technology with Open Source DBMSs

● Takayuki Nakazawa ● Naohiro Ito ● Jo Ajisawa

(Manuscript received November 28, 2003)

There is an increasing demand for highly dependable, low-cost information systems, and the open source operating system Linux is gaining attention as a way of meeting this demand. Also, the database management system (DBMS) market has shown growing interest in open source DBMS technology due to its suitability for producing easy-to-build systems at low cost. From the point of view of performance and reliability, however, open source DBMSs lag behind their proprietary rivals. One way of meeting this market demand for an inexpensive, easy-to-build DBMS that nevertheless offers high performance and high reliability is to combine the open source DBMS PostgreSQL with the proprietary DBMS Symfoware Server in such a way as to exploit the advantages of both systems.

1. Introduction

These days, information systems in business network environments must feature low-cost, high reliability, and robustness in platforms that can cover the whole operating spectrum from the front-end system to the enterprise. Linux is very much in the public eye as such a platform.

Businesses around the world are adopting Linux for their enterprise systems because it is superior in terms of reliability, cost, performance, and openness and because support services for Linux are becoming increasingly available. Linux is also increasingly being chosen as the platform for Internet servers.

The effects of the open source revolution have also been seen in the database management system (DBMS) market, and the adoption of open source DBMSs on Linux platforms is becoming more and more prevalent.

As Internet business enters the broadband age, the need for systems to run 24 hours a day, 365 days a year while supporting large numbers of users and processing vast amounts of data is

paramount. For this reason, high performance and availability are constant requirements.

For an enterprise system constructed on a Linux platform to fulfill these conditions, it must have a DBMS that possesses superior processing ability, be able to manage large amounts of data, and be extremely robust.

It is now relatively easy and inexpensive to construct open source DBMSs for use with Linux. Because these systems support an assortment of Application Programming Interfaces (APIs) as well as the Structured Query Language (SQL) standard, they also compare favorably with systems that use existing proprietary databases in terms of functionality.

However, when viewed in terms of reliability and performance, deficiencies can be discerned in open source systems applied in backend situations (e.g., regarding criteria such as recovering damaged databases, large-scale data management, and processing speed).

The required solution is a system that can be easily designed and implemented at low cost,

supports a wide variety of APIs, and matches the performance and reliability of a proprietary DBMS.

2. Towards a solution

As stated in the Introduction, what is required is a high-performance, high-reliability DBMS that has the open source advantages of being easy and inexpensive to build.

Our approach is to integrate the storage management functions of a proprietary DBMS with an open source DBMS.

2.1 PostgreSQL

From a host of available open source DBMSs, we chose PostgreSQL and decided to improve on it.

PostgreSQL is a DBMS that includes a huge variety of APIs written in Java and PHP hyper-text preprocessor (PHP: a recursive acronym) and excels in terms of producing applications. Furthermore, because PostgreSQL was tried and tested in an objective relational database management system during its development, it readily conforms to the SQL standard and provides a variety of SQL functions required for expressing business logic in mission-critical systems.

Added to the above, there exists in Japan a vigorously active PostgreSQL user community and, accordingly, a wealth of related documentation and literature is available in Japanese. For this reason, as far as the systems integration market is concerned, there is a large number of engineers with PostgreSQL know-how, which in turn creates a favorable environment for the rapid progress of PostgreSQL.^{1,2)}

Consequently, in the current climate, where the need for IT systems is matched by the demand to cut initial costs (e.g., installation, setup, and customization costs), PostgreSQL is able to compete with proprietary DBMSs as a first-choice database system solution.

However, there are also problems with DBMSs that have led some businesses to put off

installation and adopt a wait-and-see attitude.

The first of these problems concerns performance scalability. With IT systems whose infrastructure is Internet-based, it is impossible to predict whether (or how quickly and by how much) processing requests to the database will increase after the system is up and running. Even in such cases, the common expectation is that, within the limits set by hardware capacity, the response time of the DBMS will not deteriorate once the system is in use. It is further expected that, even if the demands of processing do exceed the hardware capacity, a simple addition or expansion of hardware will improve performance proportionately (e.g., if the CPU is overloaded, increasing the number of CPUs will guarantee an improvement in performance scalability).

The second problem concerns fault tolerance. Although PostgreSQL may be easy on application developers, it raises a number of issues for system administrators. The biggest of these is how to preserve data when errors occur. PostgreSQL can only restore a database up to the point where the last backup was made, and the loss of the data updates that were done since that last backup can equate to a significant financial loss for a business. In addition, the VACUUM command must be run regularly in order to free areas in the database that are being used unnecessarily and thereby maintain system performance.

Fujitsu has been conducting development with the two-fold aim of 1) solving the above-mentioned problems in PostgreSQL by applying the technology of its proprietary DBMS Symfoware Server and 2) producing a DBMS that can be safely applied to business systems. The intention is also to achieve this aim without losing any of the advantages offered by PostgreSQL while maintaining full API compatibility.

2.2 Symfoware Server

In order to solve the problems described above and elevate the performance and reliability of PostgreSQL to a sufficiently high level,

Fujitsu loaded the storage management mechanism of Symfoware Server into PostgreSQL. This compensated for the perceived deficiencies in PostgreSQL by adding the functions described below and making it possible to apply the improved system in an enterprise system.^{3),4)}

1) Media recovery

Media recovery is a feature that recovers databases damaged by disk trouble. It recovers a database to its status immediately before the trouble occurred based on the backup data and stored archive logs.^{note 1)}

In Symfoware Server, it is possible to gather backup data without any detrimental effect on the applications that use Symfoware Server. The database recovery process makes it possible to recover the status immediately before the disk was damaged.

2) Stealth Sync Control

Even if a database system ends the updated transaction normally, data is generally not written immediately to the database by the simple act of reflecting the updated data on the buffer. Instead, it is usually written to the database either at a specified time or when the data to be updated exceeds a specified size.

However, because resources are concentrated on updating the database at this time, if there is a simultaneous access from an application, application throughput may be negatively affected.

Stealth Sync Control has been developed to stop this happening. It does so by constantly monitoring the log retrieval status and database update status and automatically writing updated data to the database at the optimum time. This ensures that, even if there is a large amount of data to update, a stable throughput can be achieved.

3) Pipeline Control

Pipeline Control is a feature that enables two or more database processes (e.g., reading data

from external files, writing data, and sorting data) to be run in parallel, thus ensuring a high-speed data processing flow. Normally, when a database is being created, one process must end before the next can begin. Pipeline Control avoids this by using the buffer on the memory to run processes simultaneously.

This has the advantage of reducing data preservation and maintenance tasks, because the time required to create the database is shortened.

4) Performance scalability

Symfoware Server's architecture makes optimum use of hardware (e.g., CPUs, memory, and disk drives).

Within the DBMS kernel, a resource managing background daemon is distributed to each resource in a multithread structure. These daemons run asynchronously and efficiently with application backend processes.

Because the parallel running of application backend processes in a multi-CPU environment is fully taken into account, a high level of performance scalability is realized.

2.3 Extending functions

By applying the storage management technology of its proprietary DBMS Symfoware Server to the open source DBMS PostgreSQL, Fujitsu has created a low-cost, high-performance, high-reliability DBMS called PostgreSQL Plus.

However, sufficient ease of operation cannot be secured through the simple combination of these technologies. Therefore, we added the function extensions described below to PostgreSQL Plus to improve operability.

1) All-in-one Backup

This is an extended function for simplifying the process of making backups.

All-in-one Backup copies backup data for the system catalog and all tables and operating environment files required for media recovery to a designated backup location. There is no need to stop business applications in order to perform a backup. In addition, All-in-one Backup deletes

note 1) A log that records details of database updates. This is the name used in Symfoware Server.

any archive logs that are not required for media recovery.

This means that, after preparing a backup location and an archive log file large enough to hold a full day's updated data, daily backups can be done just by running All-in-one Backup.

2) Area re-use

MultiVersion Concurrency Control^{note 2)} (MVCC) is an advanced technology for improving database performance. By allowing multiple versions of data to exist, MVCC guarantees that data is viewed consistently within a transaction and prevents interference between transactions, thus improving data concurrency.

However, the disadvantage arises that, for the duration of each transaction, the size of the accessed data increases with each new version that is created, so there is a corresponding detrimental effect on performance.

In PostgreSQL, the VACUUM command^{note 3)} must be run to manage the unnecessary areas created in the database due to the existence of multiple versions. This frees up the areas for re-use in other operations. Running the VACUUM command on a regular basis also prevents the database from becoming too large. Unfortunately, the VACUUM command can be difficult to run in systems that run constantly, with the result that a lot of unnecessary areas are created in databases that are repeatedly added to or updated. This can also lead to the database file size becoming unreasonably large.

In PostgreSQL Plus, this type of problem is solved because data is stored in such a way that it does not have to be reorganized. This means that unnecessarily used areas can be automatically freed without stopping the system. Consequently, there is no need to run the VACUUM command

and the time and effort required to perform the most burdensome maintenance tasks in PostgreSQL are reduced.

2.4 Improving usability

As described above, applying Symfoware Server technology in PostgreSQL makes it possible to reduce costs, simplify system construction, and improve performance and reliability, resulting in an overall improvement in system operability. In PostgreSQL Plus, there is an additional improvement in terms of usability.

In PostgreSQL, a wide variety of tools are available for defining the database and accessing data (e.g., open source software such as phpPgAdmin,^{note 4)} as well as free software). Because database access is standardized using SQL, it is easy to transplant and "borrow" tools from other DBMSs. However, each DBMS has its own specifications, meaning that system-specific tools are required for tasks such as environment modification and everyday operation. This causes a lack of tools for improving operability. This is also true of PostgreSQL.

In order to solve this problem, PostgreSQL Plus provides tools to support tasks from installation, through building the environment, to all aspects of the everyday running of a system. PostgreSQL Plus Setup is a tool for building a database environment that facilitates the tasks of creating instances and making environment settings (**Figure 1**). PostgreSQL Plus Administrator is a tool for database administrators that allows operations to be performed using generic Web protocols (**Figure 2**).

These tools not only make it easier to use the PostgreSQL Plus storage management functions, but also improve overall usability by making it

note 2) A method of maintaining data consistency that allows multiple versions of data to be retained.

note 3) A PostgreSQL-specific command that frees up area held by deleted data, allowing it to be re-used.

note 4) Software written in PHP that allows PostgreSQL to be managed from a Web browser. Distributed under license by GPL.

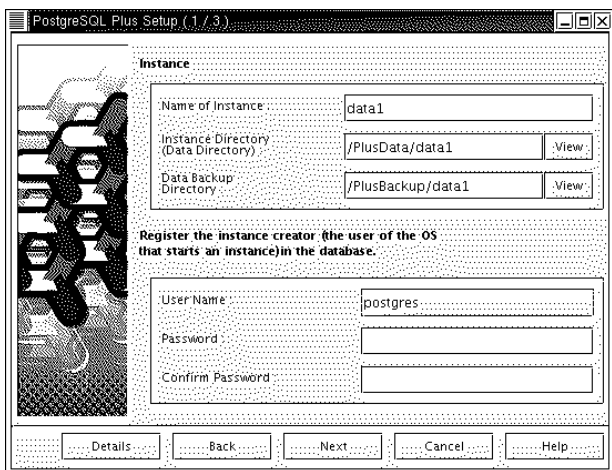


Figure 1
PostgreSQL Plus Setup window.

easier to execute applications and work with the database. In addition, PostgreSQL Plus Setup automatically optimizes environment settings, enabling PostgreSQL Plus's already high performance to be enhanced to the maximum.

3. Conclusion

In the DBMS market, a trend towards adoption of the Linux open source operating system is apparent, and this is accompanied by a surge in interest in open source DBMSs. However, currently available open source DBMSs leave much to be desired in terms of performance and reliability, such that they are not yet ready to compete with proprietary DBMSs.

There is therefore a demand for a DBMS product that combines the open source DBMS benefit of simple system construction at low cost with the proprietary DBMS benefits of high performance and reliability. Fujitsu has combined

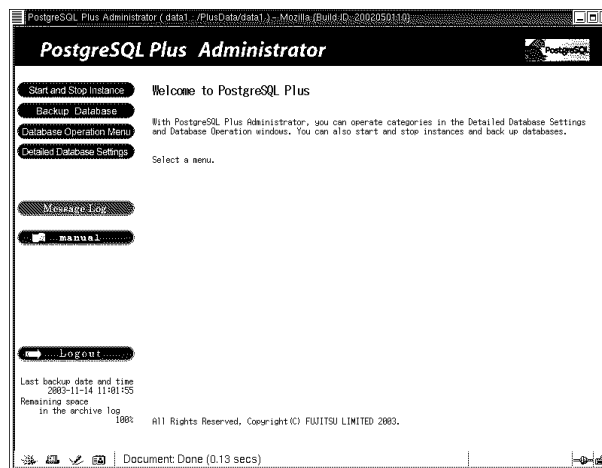


Figure 2
PostgreSQL Plus Administrator window.

the superior technology of its established DBMS Symfoware Server with the highly regarded open source DBMS PostgreSQL to create such a product. It must be pointed out that this product, PostgreSQL Plus, is not simply the result of bundling together two existing products. Instead, the current state of the DBMS market was studied and new features were included in accordance with users' requirements.

We have no doubt that PostgreSQL Plus will satisfy this demand in the DBMS market.

References

- 1) PostgreSQL.
<http://www.postgresql.org/index.html>
- 2) Japan PostgreSQL User Conference.
<http://www.postgresql.jp/>
- 3) Nikkei BP Plan: Fujitsu's Symfoware database. (in Japanese), Edition 1, Tokyo, Nikkei BP Publishing Center, 2002.
- 4) Symfoware Server V5 technical white paper.
<http://software.fujitsu.com/jp/symfoware/catalog/wp/pdf/symfowp.pdf>



Takayuki Nakazawa graduated from Tochigi Prefectural Utsunomiya Technical High School, Utsunomiya, Japan in 1977. He joined Fujitsu Ltd., Numazu, Japan in the same year and worked on development of the mainframe database software AIM/DB. Later, he worked on development of the relational database Symfoware. He is currently working on development to strengthen the storage layer of the open

source software PostgreSQL.

E-mail: t.nakazawa@jp.fujitsu.com



Naohiro Ito received the B.S. degree in Mathematical Sciences from the University of Tokyo, Tokyo, Japan in 1990. He joined Fujitsu Ltd., Numazu, Japan in the same year and worked on development of a database engine for fault-tolerant systems. Later, he worked on development of the Symfoware relational database for open systems. He is currently working on development to strengthen the storage layer of the

open source software PostgreSQL.

E-mail: itou.naohiro@jp.fujitsu.com



Jo Ajisawa received the B.S. degree in Political Science from Waseda University, Tokyo, Japan in 1992. He joined Fujitsu Software Engineering Laboratory Ltd., Odawara, Japan in the same year (this later changed its name to Fujitsu Hyper Software Technologies Limited) and worked on development of the Symfoware database access infrastructure. He left Fujitsu Hyper Software Technologies Limited in 2003 and joined

Fujitsu Ltd., Numazu, Japan in the same year. He is currently working on development to strengthen the storage layer of the open source software PostgreSQL.

E-mail: ajisawa@jp.fujitsu.com