PostgreSQL: Leader in Cost Reduction

Akiyoshi Kawada, Go Nishimura, Junji Teramoto, Etsuro Fujita, Takeshi Yamamuro, and Takuma Wakamori

Abstract

The NTT Open Source Software Center is conducting research and development on the PostgreSQL open-source database management system in order to develop and spread open source software (OSS). Recent advances in the development of PostgreSQL have brought it to a level of usability adequate for commercial systems, and due to its great cost-reducing effects relative to commercial products, it is receiving much attention and even being introduced for core systems. In this article, we introduce new functionality in the latest version, PostgreSQL 9.3, released in September 2013, and outline NTT's OSS development policies. We also introduce the SQL/MED (Structured Query Language/Management of External Data) indexing engine, which was developed by the authors as infrastructure for log analysis and an area for expanding applications of PostgreSQL.

Keywords: DBMS, PostgreSQL, open source

1. Introduction

Database management systems (DBMSs) store, manage, and make use of various types of data, and they have been under continuous research and development (R&D) for several decades. DBMSs hold a very important place in ICT (information and communication technology) systems and play many different roles. For example, in the area of data storage management, they provide synchronization control so that updates are processed correctly without creating inconsistencies if several people update the data at the same time, and recovery functions so that the results of completed operations are not lost if the system goes down.

In the area of data search, they provide a declarative language enabling applications to search data easily simply by describing what data they need and without having to implement detailed search algorithms. This simplifies application development.

A DBMS is a sophisticated mechanism that requires advanced technical ability to develop it. Furthermore, it will be used continuously by many users and systems for five, ten, or more years, so it must be maintained continuously to preserve stability and quality. PostgreSQL has been developed by technologists from around the world that have formed a community, and it currently has functionality and performance that compares favorably with commercial DBMS products. NTT is also a member of this community and has contributed by operating and supporting the portal site for the Japan PostgreSQL user's group (JPUG). We participate in development discussions and provide many patches for functionality and performance improvements. In fact, NTT was the top patch contributor from Japan for the current main versions, 9.x (9.0 to 9.3) (**Fig. 1**).

NTT also proposed the streaming replication function and developed it together with the community, to greatly improve PostgreSQL availability for use in internal and external mission-critical applications. This feature is currently used by many users around the world.

The NTT Open Source Software (OSS) Center is

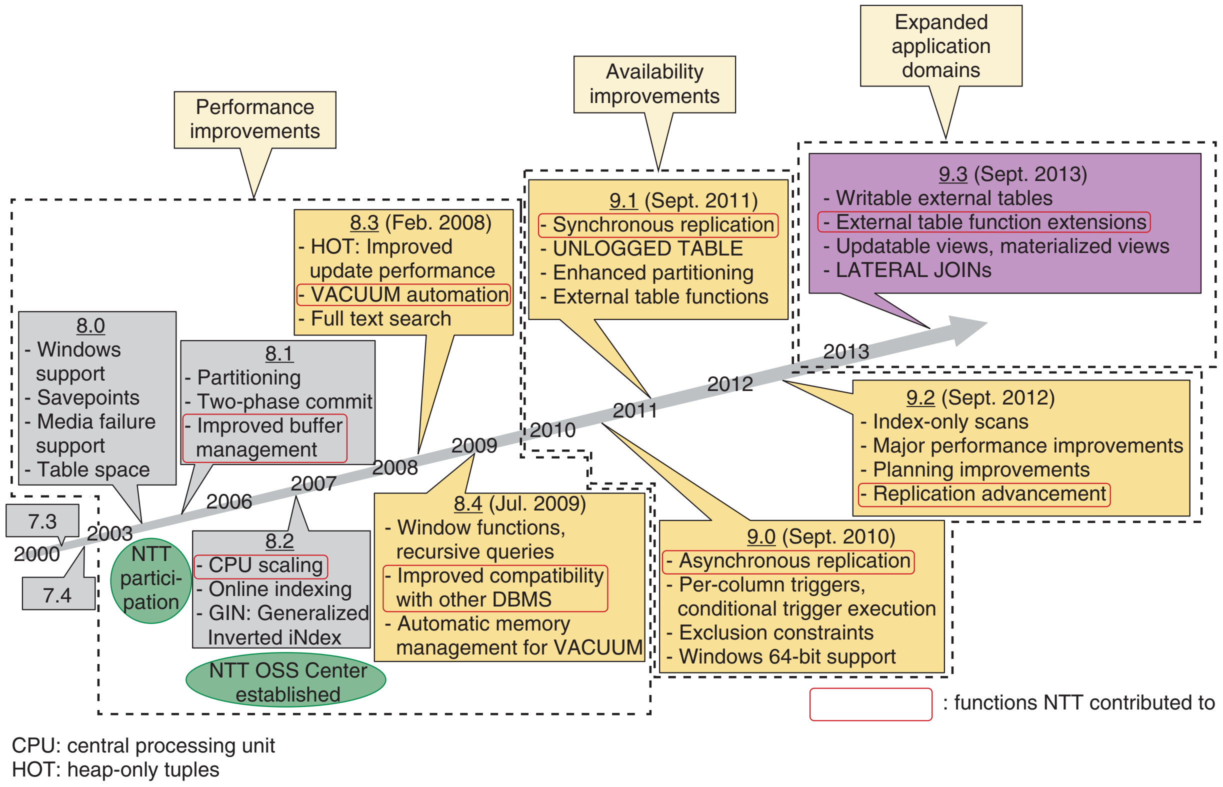

Performance improvements
Availability improvements
Expanded application domains
7.3
2000
7.4
2003
NTT partici-pation
2006
2007
2008
2009
2010
2011
2012
2013
8.0
- Windows support
- Savepoints
- Media failure support
- Table space
8.1
- Partitioning
- Two-phase commit
- Improved buffer management
8.2
- CPU scaling
- Online indexing
- GIN: Generalized Inverted iNdex
NTT OSS Center established
8.3 (Feb. 2008)
- HOT: Improved update performance
- VACUUM automation
- Full text search
8.4 (Jul. 2009)
- Window functions, recursive queries
- Improved compatibility with other DBMS
- Automatic memory management for VACUUM
9.0 (Sept. 2010)
- Asynchronous replication
- Per-column triggers, conditional trigger execution
- Exclusion constraints
- Windows 64-bit support
9.1 (Sept. 2011)
- Synchronous replication
- UNLOGGED TABLE
- Enhanced partitioning
- External table functions
9.2 (Sept. 2012)
- Index-only scans
- Major performance improvements
- Planning improvements
- Replication advancement
9.3 (Sept. 2013)
- Writable external tables
- External table function extensions
- Updatable views, materialized views
- LATERAL JOINs
: functions NTT contributed to
CPU: central processing unit
HOT: heap-only tuples

Fig. 1. Main functions contributed by NTT to each version of PostgreSQL.

addressing the themes of enhancing performance and functionality as databases increase in scale, and improving portability from commercial DBMSs. Initiatives include developing external table functions for use with large-scale databases, conducting R&D on distributed databases, and creating functions for improving compatibility that will enable Structured Query Language (SQL) in dialects particular to commercial DBMSs to run as-is on PostgreSQL as well.

2. PostgreSQL Version 9.3

A major revision of PostgreSQL is released once a year, and the most recent was Version 9.3, released in September 2013. This version includes many functional and performance enhancements. The three main enhancements are as follows.

(1) View function extensions

New additions to these functions include materialized views and updatable views. Views are virtual tables that could in fact consist of a query from multiple tables (a SELECT statement). A materialized view stores the result of executing the query as a table so it can be reused when the same query is called again to increase query processing speed. An updatable view allows updates to be applied to a view without creating complex definitions. Until now, view functions were a weak point when compared with commercial DBMSs, so implementation of these functions should simplify migration from commercial DBMSs.

(2) Replication function improvements

To improve fault tolerance in the system, PostgreSQL has a replication function that enables two servers to operate as active and standby servers. If the active server experiences a fault, the standby server must be restarted as the active server, but improvements have been made to reduce the time for this process. This has increased system availability and enabled operation rates to approach 100%. It is also one of the functions that NTT has put effort into developing and contributing.

(3) External table function extensions

PostgreSQL functions for managing external data as external tables have been enhanced. The external table functions enable data external to the DBMS (such as a comma-separated values (CSV) file) to be handled as a table, just like an internal table. This

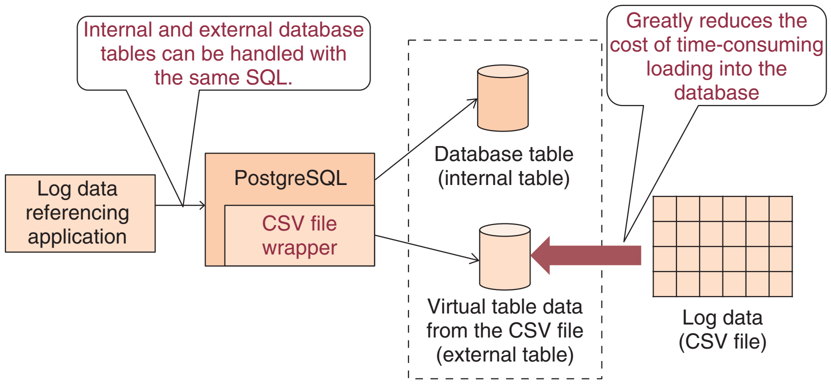

Internal and external database tables can be handled with the same SQL.
Greatly reduces the cost of time-consuming loading into the database
Log data referencing application
PostgreSQL
CSV file wrapper
Database table (internal table)
Virtual table data from the CSV file (external table)
Log data (CSV file)

Fig. 2. Overview of PostgreSQL SQL/MED technology.

enables many large logs to be analyzed using SQL without loading them into the DBMS; instead, they are kept externally. External tables were introduced in PostgreSQL 9.1, and in the latest version 9.3, they have been extended to handle data on other PostgreSQL servers as external tables in addition to data in files. Update functions to add and delete records from external tables have also been added, so large amounts of data can now be easily distributed over multiple PostgreSQL servers. NTT's contributions have helped to advance the distributed query optimization technology since external table functions were first added, and these results were used in the latest functional enhancements.

3. SQL/MED indexing engine development initiatives

We now introduce the SQL/MED (Management of External Data) indexing engine developed by the authors as a log analysis infrastructure using PostgreSQL external table functions.

Conventionally, DBMSs have primarily been used for processing online transactions. However, management decision-making using business intelligence has taken center stage, and DBMSs are being used more and more for analyzing large-scale data such as logs. Service history data (log data) are now increasingly being managed as CSV files for the purpose of analyzing customer preferences. Even on information systems operating in the background of services, more detailed system operation and error logs are being collected, and these logs are often being analyzed in order to improve operations.

In some cases, most of the analysis of acquired logs is done manually using editors and scripts, but this manual analysis work can be difficult if the logs being processed are large. If the logs are placed in a DBMS, the large volume of data can be summarized and analyzed from various perspectives, and sophisticated searches with complex conditions can be done by the DBMS, leaving the developer to focus on the task of analysis. There are also issues with this approach, however. Loading many large logs into a DBMS incurs a great cost in terms of time and management. Using external tables is one way of resolving this issue (**Fig. 2**).

At NTT, we have implemented SQL/MED as defined in SQL2003 (the SQL language international standard set in 2003) for PostgreSQL, as well as an extended implementation of external table indexing not in the SQL standard, called SQL/MED indexing [1].

Currently, NTT is also conducting R&D to implement partitioning technology and index-only scan technology to further increase processing speed (**Fig. 3**). These technologies were introduced into a contact-center voice-mining platform for call analysis (a voice mining system), which had required quite a long time to process database updates with earlier technology. The results indicated that the technologies reduced the time to one-half to one-third that of earlier values. The analyst was able to allocate the time saved on processing to other types of analysis, and to provide more data.

We now introduce the two key technologies to increase speed: partitioning technology and index-only scan technology.

(1) Partitioning technology

Partitioning technology partitions tables in order to

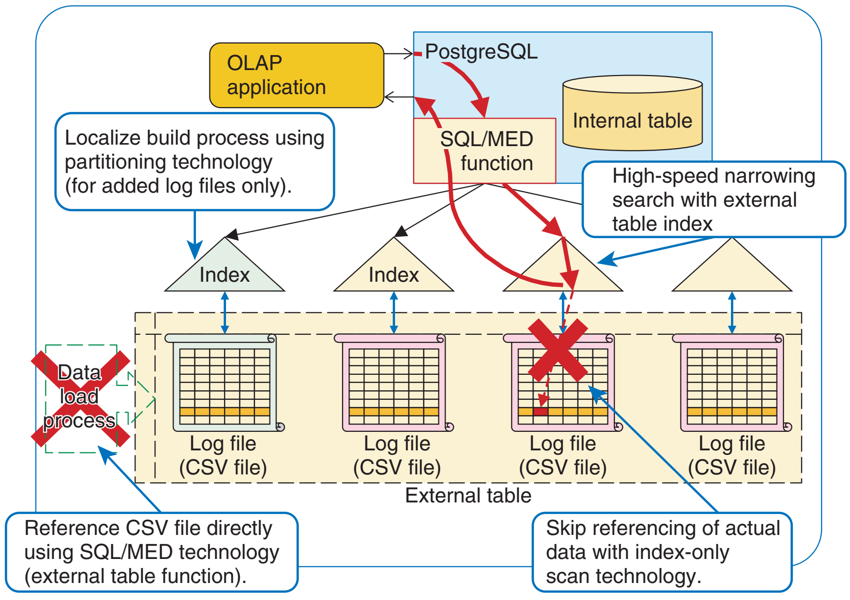

OLAP application
PostgreSQL
SQL/MED function
Internal table
Localize build process using partitioning technology (for added log files only).
High-speed narrowing search with external table index
Index
Index
Data load process
Log file (CSV file)
Log file (CSV file)
Log file (CSV file)
Log file (CSV file)
External table
Reference CSV file directly using SQL/MED technology (external table function).
Skip referencing of actual data with index-only scan technology.
OLAP: online analytical processing

Fig. 3. Partitioning and index-only scan technologies.

speed up table referencing, index creation, and other operations. Partitioning tables reduces the range of key values handled when processing a table reference. In a scenario where newly arrived log entries are always added to the end of the log, the creation of indices can be completed by processing only the latest log file that caused the update. Speed is increased by processing only some of the partition tables. Table partitioning in PostgreSQL uses a function called table inheritance, which can now also be used with external tables. For example, consider a table that is partitioned by months and operated cyclically, with tables added for the latest month and deleted when they are a year or more old. When table partitioning is introduced, operations such as deletion, addition, and index creation only apply to tables for particular months, so the work and processing time for these operations is reduced. When large logs are analyzed by processing external tables, the disk I/O (input/output)—rather than the CPU (central processing unit)—tends to be the bottleneck, so using partitioning technology to reduce disk I/O can be particularly effective in increasing speed.

(2) Index-only scan technology

Index-only scan technology speeds up the processing of table references by only scanning an index (and not accessing the data itself) to obtain search results in cases when the query satisfies certain conditions. Normally, index sizes are small and can be kept in memory, so disk I/O can be reduced dramatically when processing external tables. This results in dramatic increases in the speed of table reference processing. The query condition for this technology to be effective is that the columns specified in the SELECT list and WHERE condition in the query must all be present in the columns used to create the index. (Multiple columns can be used.) It is possible to consider that all of the columns needed for a search result are included when designing an index, so index-only scan technology can be effective in many circumstances.

By combining the above technologies, the SQL/MED indexing engine can access large amounts of data at high speed without loading it into the DBMS and can be used in business for applications such as data analysis.

4. Future developments

This article presented some trends in the latest version of PostgreSQL and the development of a log analysis infrastructure at the NTT OSS Center. In addition, the NTT OSS Center is putting effort into providing support for introduction and operation of PostgreSQL so that it can be confidently used for business. The technical expertise gained through development is used in providing support, and the requests for improvements received when providing support are fed back into development, contributing to further improvements in PostgreSQL. We will continue to contribute to reducing business costs and expanding business domains through the two-pronged approach of support and development.

References

[1] T. Hitaka, N. Kotani, Y. Suga, E. Fujita, A. Kawada, and T. Yamamuro: "PostgreSQL SQL/MED Technology—Infrastructure for Efficient Analysis of Service and System Logs," NTT Technical Journal, Vol. 23, No. 8, pp. 30–34, 2011 (in Japanese).

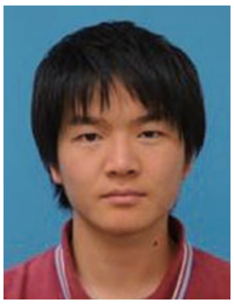
Akiyoshi Kawada
Expert, Infrastructure Technology Unit, NTT Open Source Software Center.
He received the M.S. in theoretical physics from Hokkaido University, Sapporo, in 1996. He joined NTT Comware in 1996. He then moved to NTT Open Source Software Center in 2012. His research interests include database as platforms of log analysis and open source software.

Etsuro Fujita
Senior Expert, Infrastructure Technology Unit, NTT Open Source Software Center.
He received the PhD. in informatics from The Graduate University for Advanced Studies (SOKENDAI), Kanagawa, in 2013. He joined NTT in 1996. He then moved to NTT Open Source Software Center in 2012. His research interests include database systems, information integration, information retrieval, and open source software.

Go Nishimura
Senior Manager, Infrastructure Technology Unit, NTT Open Source Software Center.
He received the M.M.G. from Keio University, Tokyo, in 1996. He joined NTT in 1996. He then moved to NTT Open Source Software Center in 2013. His research interests include information systems and applications.

Takeshi Yamamuro
Researcher, Distributed Computing Technology Project, NTT Software Innovation Center (since April, 2014).
He received the M.E. in science and engineering from Sophia University, Tokyo, in 2008. He joined NTT in 2008, and then moved to NTT Open Source Software Center in 2012. He is interested in columnar-db topics such as vectorized pipelining, type-specific compression, and hardware-aware techniques.

Junji Teramoto
Manager, Infrastructure Technology Unit, NTT Open Source Software Center.
He received the M.E. in electronics, information and communications engineering from Waseda University, Tokyo, in 1998. He joined NTT in 1998. He then moved to NTT Open Source Software Center in 2012. His research interests include relational database management systems and open source software.



Takuma Wakamori
Researcher, Distributed Computing Technology Project, NTT Software Innovation Center (since April, 2014).
He received the M.E. in electronic and information engineering from Yokohama National University, Kanagawa, in 2013. He joined NTT Open Source Software Center in 2013. He then moved to NTT Software Innovation Center in 2014. His research interests include DBMS, distributed computing, and open source software.