Case Studies of Extending Open-Source Database Software PostgreSQL and Contributing to Open-Source Communities

Authors: *Masashi Tatedoko*, Shingo Oidate**

**Information Technology R&D Center*

Abstract

Open-Source Software (OSS) has become increasingly important in recent years. In addition to using OSS, Mitsubishi Electric emphasizes contribution to OSS communities to improve their value and achieve sustainability, and we plan to expand those activities. By deepening understanding of internal operation of PostgreSQL*1, an OSS Relational Database Management System (RDBMS), through efforts toward management of external data with PostgreSQL, we have identified opportunities for accelerating aggregation processing in distributed configurations, and proposed some development results to the OSS community. To advance the dissemination and development of OSS, and we are also contributing to various OSS communities through activities such as patch submissions, documentation translation, and conference presentations.

1. Introduction

To achieve sustainable growth, Mitsubishi Electric has established a management strategy of promoting open innovation based on technical capabilities and creativity. The aim is to design the future and create new value in a timely fashion through fusion of knowledge and co-creation both within and outside the group. One important initiative within this overall strategy is contributing to OSS communities. Use of OSS is growing throughout the world. It is said that 96% of the world's software projects include OSS, and 77% of their source code is OSS(1). OSS is essential to global software development, including at Mitsubishi Electric. We believe it is our social responsibility to participate in and contribute to OSS communities in order to improve the value of OSS and make it sustainable. Mitsubishi Electric has promoted the use of OSS in accordance with its philosophy and licenses, formulating OSS Usage Guidelines in 2014, and establishing an Internal OSS Manager Liaison Committee in 2018.

PostgreSQL, an OSS RDBMS, supports the standard SQL used for database operation and definition, and is highly extendable. In connecting our database products by using PostgreSQL's foreign database management feature, it was necessary to understand PostgreSQL's internal operation in order to overcome limitations due to the interface of our database products. When, as part of this process, we performed parallel aggregation processing in a distributed configuration by connecting PostgreSQL instances using the foreign database management feature, we found there was still room for significant performance improvement, and that led us to create a patch for PostgreSQL core functionality, and propose it to the community.

2. Initiatives for PostgreSQL Feature Extensions

Mitsubishi Electric is engaged in various initiatives relating to PostgreSQL, such as feature extensions. An example is shown in Fig. 1. The feature extensions in the figure are explained in section 2.1 and 2.2 below.

*1 PostgreSQL is a registered trademark of the PostgreSQL Community Association of Canada.

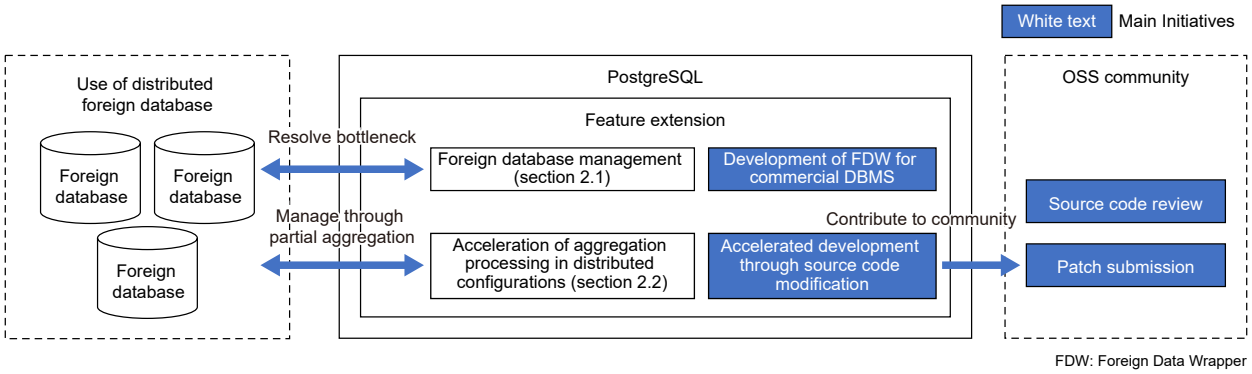

Fig. 1 An example of various initiatives, including PostgreSQL feature extensions

2.1 Foreign database management

Here we present two examples of extension in application of the FDW, the external data management feature of PostgreSQL to a commercial Database Management System (DBMS).

The first is an example which resolved the restriction on the number of connections with foreign databases. FDW needs to maintain the connection to a foreign database until the query is complete. Therefore, when multiple queries are executed simultaneously to a foreign database or when executing queries containing numerous foreign tables, and the number of connections exceeds the limit, a connection-waiting deadlock may occur. In this case, processing of the entire query halts. To solve this issue, we added a connection management feature for foreign databases to FDW, allowing new connections to be initiated while continuing existing query processing when the limit on the number of connections has been exceeded (Fig. 2).

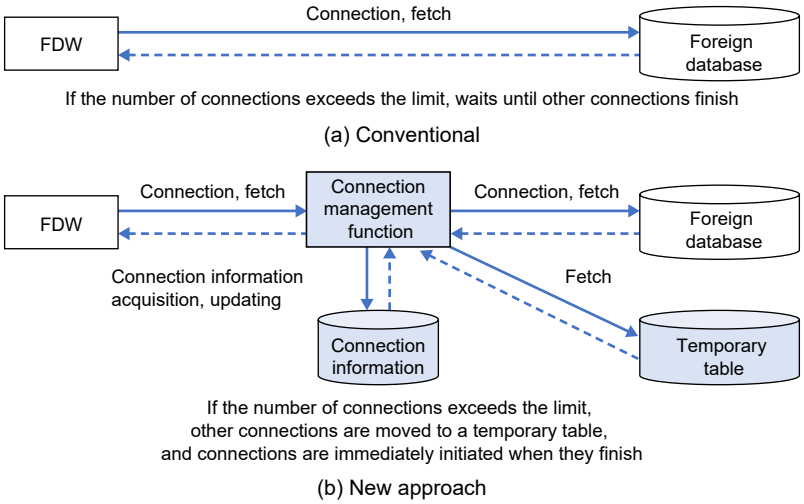

Fig. 2 Removing foreign database connection restrictions

The second example resolves processing bottlenecks of FDW. Since PostgreSQL executes queries with a single process, PostgreSQL and FDW are executed synchronously, resulting in inefficiency because FDW processing cannot be executed during execution of PostgreSQL processing. When there is a discrepancy between PostgreSQL's internal data format and the foreign database's output format, transform processing may take a longer time. To address this issue, we achieved acceleration by executing PostgreSQL processing, fetch processing, and transform processing asynchronously, and parallelizing the transform processing (Fig. 3).

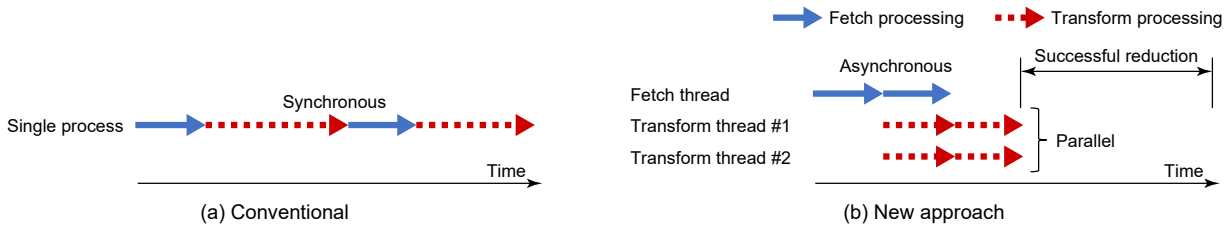

Fig. 3 Acceleration of querying foreign database

2.2 Acceleration of aggregation processing in distributed configurations

The PostgreSQL development community is advancing the development of built-in sharding to achieve scale-out using only standard features such as FDW. In the existing built-in sharding, operations that can be processed in parallel are limited to row selection, joins, and the like, and parallel processing of aggregation operations is not supported. Mitsubishi Electric regards built-in sharding as a collaborative area, and we offer improvement proposals based on know-how in foreign database management that we have accumulated through the development described in Section 2.1.

The premise of built-in sharding is that the coordinator that controls the workers expresses the processing requested by FDW to each worker as SQL statements, issues them to the workers, and generates the final result using the SQL statement results from each worker. To maintain consistency with this premise, the content of the state value generation processing (partial aggregation processing) of the aggregation function that workers pass to the coordinator needs to be expressed using SQL aggregation functions. However, there is a problem that partial aggregation of some existing aggregation functions cannot be expressed by the format of existing aggregation functions. For example, in the case of numerical averaging processing, the state value that a worker passes to the coordinator is an array containing the total and the count of data items, but there is no existing aggregation function that returns such a state value.

For this issue, we submitted a patch to the community that installs aggregation functions (partial aggregation functions) that execute partial aggregation processing in workers, and those functions are then executed by each worker. For all aggregation functions that can be parallelized, we implemented their partial aggregation processing as functions based on existing mechanisms, so that partial aggregation can be installed without coding but only with simple definition statements. Performance was evaluated using TPC-H, an international standard benchmark for analytic queries of large-scale databases, and the results confirmed that the aggregation processing speed increased proportionally with the number of workers. With five workers, it was confirmed that the aggregation processing speed became 12 times faster compared to conventional PostgreSQL due to aggregation parallelization and the elimination of data transfer overhead through partial aggregation.

3. Activities Contributing to Various OSS Communities

Aside from the PostgreSQL described in section 2, Mitsubishi Electric participates in various OSS communities and contributes to the dissemination and development of OSS through activities such as feature enhancement and quality improvement through patch submissions, document translation, and conference presentations. These contribution activities of Mitsubishi Electric have only just started, but we are formulating an “OSS Contribution Policy” and “OSS Contribution Guidelines,” and we plan to broaden our activities in line with those frameworks. In addition to PostgreSQL, we have been contributing to communities by submitting patches for feature enhancements and bug fixes, and reviewing patch submissions, for various types of open source software including PyTorch[*2] (one of the most widely used AI frameworks), Apache TVM[*3] (a compiler that optimizes and accelerates AI processing), and Hummingbird. In Hummingbird, our company’s engineer Masahiro Hiramori ranks No. 3 in number of commits (as of November 2024). He contributes to the development and sustainability of OSS communities as a committer with the authority to integrate source code provided by contributors into the mainline. We were one of the few Japanese companies that made a presentation at TVMCon, a conference on deep learning compilers.

*2 PyTorch and Hyperledger are registered trademarks of the Linux Foundation.
*3 Apache TVM is a registered trademark of the Apache Software Foundation.

In addition to submitting patches, we contribute in the form of translating official documentation from English to Japanese for Hyperledger[*2], a blockchain platform. We gave a presentation on this translation work at the Hyperledger Conference.

We are also contributing to the overall OSS ecosystem by participating in the Linux Foundation, a non-profit organization that brings together numerous major OSS communities including PyTorch and Hyperledger mentioned above to promote the development of OSS.

4. Conclusion

This paper has discussed examples of PostgreSQL extension development, our involvement with OSS communities, and our previous efforts to contribute in the fields of AI frameworks and data management. Going forward, we will establish internal OSS contribution policies and OSS contribution guidelines, conduct high-quality and speedy development using OSS, and fulfill our social responsibilities through contributions to OSS communities.

Reference

(1) Black Duck Software, Inc.: 2024 Open Source Security and Risk Analysis Report
https://www.blackduck.com/en-us/resources/analyst-reports/open-source-security-risk-analysis.html