FUJITSU

Fujitsu Enterprise Postgres 18SP1 for x86

General Description

Windows/Linux

J2UL-3053-02ENZ0(00)

March-2026

Preface

Purpose of this document

This document explains the Fujitsu Enterprise Postgres concepts to those who are to operate databases using it.

This document explains the features of Fujitsu Enterprise Postgres.

Intended readers

This document is intended for people who are:

- Considering installing Fujitsu Enterprise Postgres
- Using Fujitsu Enterprise Postgres for the first time
- Wanting to learn about the concept of Fujitsu Enterprise Postgres
- Wanting to see a functional overview of Fujitsu Enterprise Postgres

Readers of this document are also assumed to have general knowledge of:

- Computers
- Jobs
- Linux
- Windows(R)

Structure of this document

This document is structured as follows:

Export restrictions

Exportation/release of this document may require necessary procedures in accordance with the regulations of your resident country and/or US export control laws.

Issue date and version

```
Edition 2.0: March  2026
Edition 1.0: December  2025
```

Copyright

Contents

Chapter 1 Fujitsu Enterprise Postgres Basics

Fujitsu Enterprise Postgres maintains the operating methods, interfaces for application development and SQL compatibility of PostgreSQL, while providing expanded features for enhanced reliability and operability.

This chapter explains the functionality extended by Fujitsu Enterprise Postgres.

Refer to "Appendix A List of Features" for feature differences between editions.

Additionally, Fujitsu Enterprise Postgres supports various open source software (OSS). Refer to "Appendix B OSS Supported by Fujitsu Enterprise Postgres" for information on OSS supported by Fujitsu Enterprise Postgres.

Fujitsu Enterprise Postgres has the following features:

- Flexible database recovery
  Not only does Fujitsu Enterprise Postgres recover data to its most recent form when a failure occurs, which is essential for databases, but it can also recover to any point in time. Additionally, backup/recovery can be performed using any copy technology.

- Simple GUI-based installation and operation management
  Fujitsu Enterprise Postgres uses GUI to simplify cumbersome database operations, and allows databases to be used intuitively.

- High reliability by using database multiplexing
  Database multiplexing protects important data and enables highly reliable database operation.

L - High reliability by using the multi-master replication
  By using multi-master replication, where each node's database acts as a master, continue service even during disasters, minimizing the risk of system downtime and data loss, and achieving high reliability.

L - High reliability through a cluster system using Patroni
  Build a highly reliable cluster system in cooperation with Patroni.

L - High reliability by using the failover feature integrated with the cluster software
  Fujitsu Enterprise Postgres links with PRIMECLUSTER, thereby allowing highly reliable systems to be achieved by using failover.

- Seamless migration from Oracle databases
  Fujitsu Enterprise Postgres provides a compatibility feature with Oracle databases that localizes the correction of existing applications and allows easy migration to Fujitsu Enterprise Postgres.

- Storage data protection using transparent data encryption
  Information can be protected from data theft by encrypting data to be stored in the database.

L Also, you to choose an external key management system as the storage location for your encryption keys.

- Policy-based login security
  By defining login security policies as profiles and assigning profiles to database users, you can achieve login management and password operation according to the security policy.

- Management of access control by confidentiality management
  Confidentiality management feature supports the design/management of access privileges and optimizes access control, thereby enhancing security.

- Data masking for improved security
  The data masking feature changes the returned data for queries from applications, to prevent exposing actual data. This improves security for handling confidential data such as personal information.

- Audit logs for improved security
  Audit logs can be used to counter security threats such as unauthorized access and misuse of privileges for the database.

- Enhanced query plan stability
  Allows you to control query planning for SQL statements. This feature is used to prevent performance degradation due to changes in the query plan of SQL statements, such as in core business operations where stabilizing performance is more important than improving the processing performance of SQL statements.

- Increased aggregation performance using the in-memory feature
  The following features help speed up scans even when aggregating many rows.
  - Vertical Clustered Index (VCI)
  - In-memory data

- High-speed data load
  Data from files can be loaded at high speed into Fujitsu Enterprise Postgres tables using the high-speed data load feature.

- High availability by using Connection Manager
  With the Connection Manager features, replication operation can be continued without being aware of the connection destination of the applications.

- Memory usage reduction using Global Meta Cache
  The Global Meta Cache feature loads some of meta cahe information in shared memory. This reduces overall system memory usage.

- Memory usage reduction using Local Meta Cache Limit
  Reduce memory consumption by limiting the metacache that is kept in local memory.

- Protect and efficiently manage and use knowledge data for AI applications
  Knowledge Data Management enables Fujitsu Enterprise Postgres to securely store knowledge data used in RAG-based applications. It also automates part of the processing for knowledge data maintenance and retrieval, enabling efficient operation and utilization.

1.1 Flexible Database Recovery

Threats such as data corruption due to disk failure and incorrect operations are unavoidable in systems that use databases. The ability to reliably recover corrupted databases without extensive damage to users when such problems occur is an essential requirement in database systems.

Fujitsu Enterprise Postgres provides the following recovery features that flexibly respond to this requirement:

- Media recovery, which recovers up to the most recent point in time
- Point-in-time recovery, which can recover up to a specific point in time
- Backup/recovery that can integrate with various copy technologies

Media recovery, which recovers up to the most recent point in time

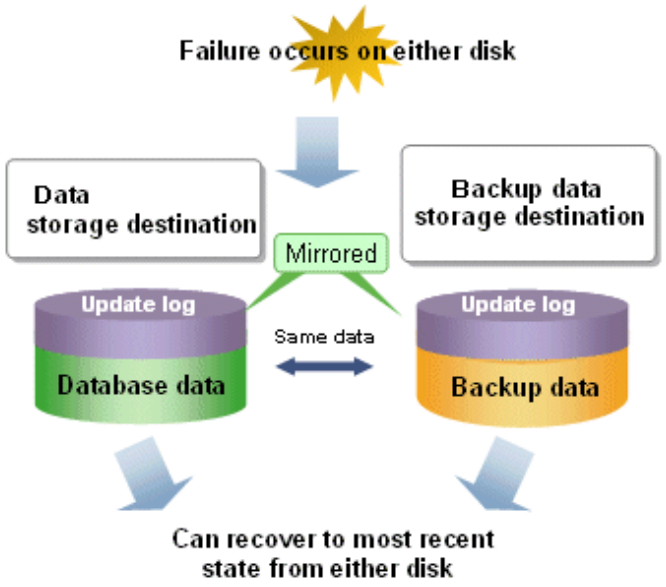
When a disk failure occurs, media recovery can recover data to how it was immediately before the failure.

In order to recover the database, Fujitsu Enterprise Postgres accumulates a history of database update operations, such as data additions and deletions, as an update log.

Fujitsu Enterprise Postgres retains a duplicate (mirror image) of the update log after backup execution on the data storage destination and on the backup data storage destination. Therefore, the data on one disk can be used to recover to the most recent state of the database even if a disk failure has occurred on the other.

Media recovery is executed using either a GUI tool provided with Fujitsu Enterprise Postgres (WebAdmin) or server commands.

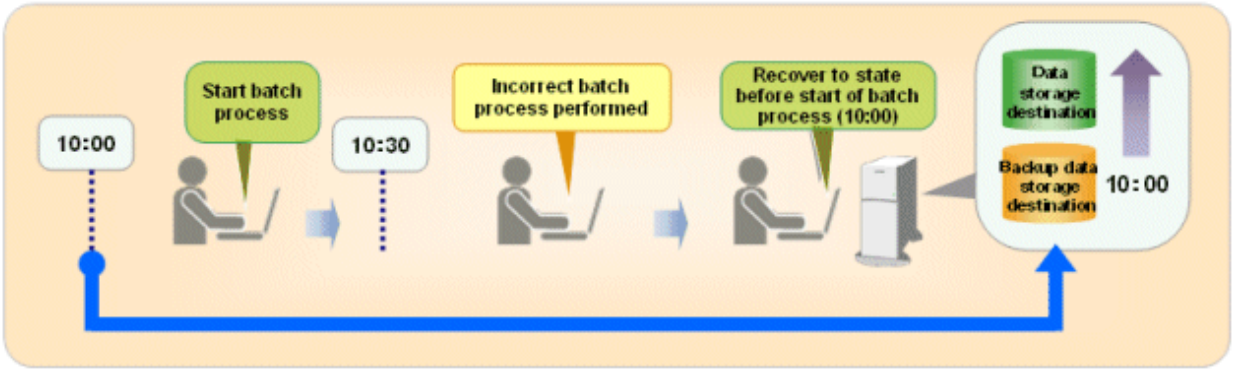
Recovery using WebAdmin requires less time and effort, since WebAdmin automatically determines the scope of the operation.



Point-in-time recovery, which can recover up to a specific point in time

Point-in-time recovery can be used to recover a database that has been updated by an incorrect operation, for example, by specifying any date and time before the incorrect operation.

Point-in-time recovery is executed using Fujitsu Enterprise Postgres server commands.



Backup/recovery that can integrate with various copy technologies

It is possible to back up to the backup data storage destination, or to any backup destination using any copy technology implemented by user exits.

L

For example, by using the high-speed copy feature of the storage device, the processing time for backup of large databases can be greatly reduced.

See

Refer to "Backup/Recovery Using the Copy Command" in the Operation Guide for information on backup/recovery using user exits.

1.2 Simple GUI-Based Installation and Operation Management

Fujitsu Enterprise Postgres provides WebAdmin, which is a GUI tool for a range of tasks, from database installation to operation management. This allows the databases to be used simply and intuitively.

WebAdmin can be used for Fujitsu Enterprise Postgres setup, creating and monitoring a streaming replication cluster, database backups, and for recovery. Depending on the configuration, WebAdmin can be used to manage Fujitsu Enterprise Postgres instances in a single server, or instances spread across multiple servers.

- Setup

  To perform setup using WebAdmin, you must create an instance. An instance is a set of server processes that manage a database cluster (database storage area on the data storage destination disk). Instances can be created easily and with only minimal required input, because the tool automatically determines the optimal settings for operation.

- Database backup/recovery

  Database backup and recovery can be performed using simple GUI operations.

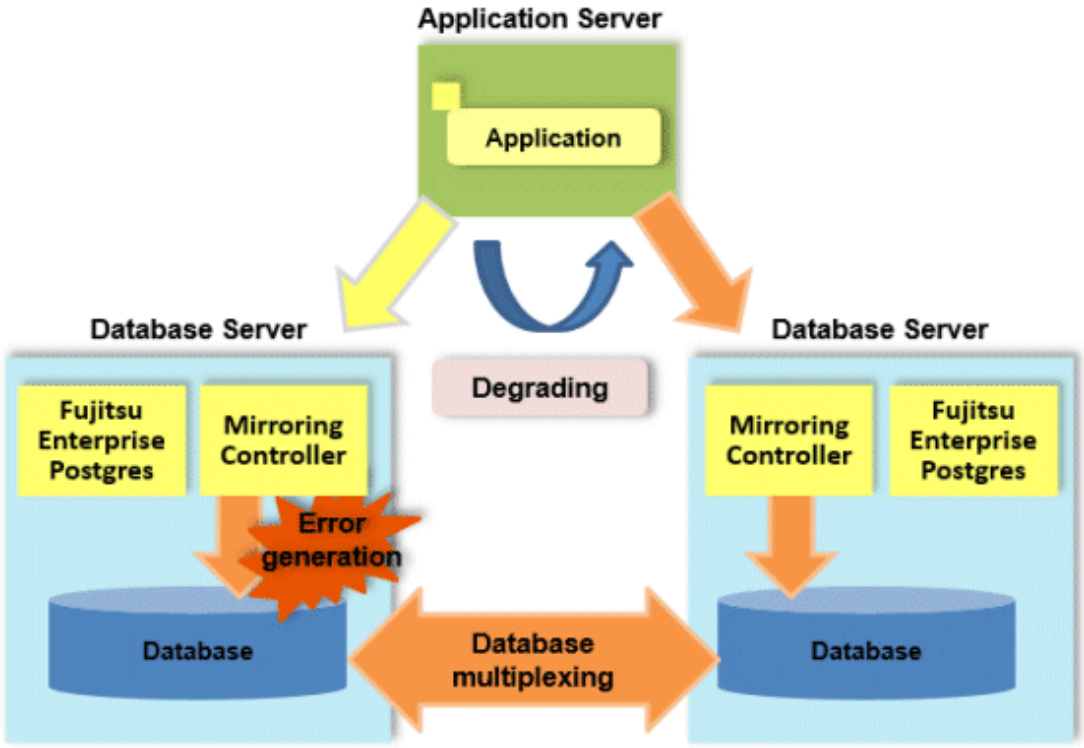
  In particular, Fujitsu Enterprise Postgres can automatically identify and isolate the location of errors. This simplifies the recovery process and enables faster recovery.

In addition, Fujitsu Enterprise Postgres provides the following expanded features in pgAdmin:

- NCHAR type
- Expanded trigger definition
  - REPLACE feature
  - Function call feature

1.3 High Reliability with Database Multiplexing

It is vital for systems that use databases to protect data from damage or loss caused by a range of factors such as hardware and software errors. Database multiplexing protects important data and enables highly reliable database operation.



Fujitsu Enterprise Postgres not only mirrors a database using the PostgreSQL streaming replication feature, but also provides simplified switchover and standby disconnection features as well as a feature to detect faults in elements that are essential for the continuity of database process, disk, network, and other database operations.

Even if a switchover is performed, the client automatically distinguishes between the primary and standby servers, so applications can be connected transparently regardless of the physical server.

The Mirroring Controller option enables the primary server (the database server used for the main jobs) to be switched automatically to the standby server if an error occurs in the former.

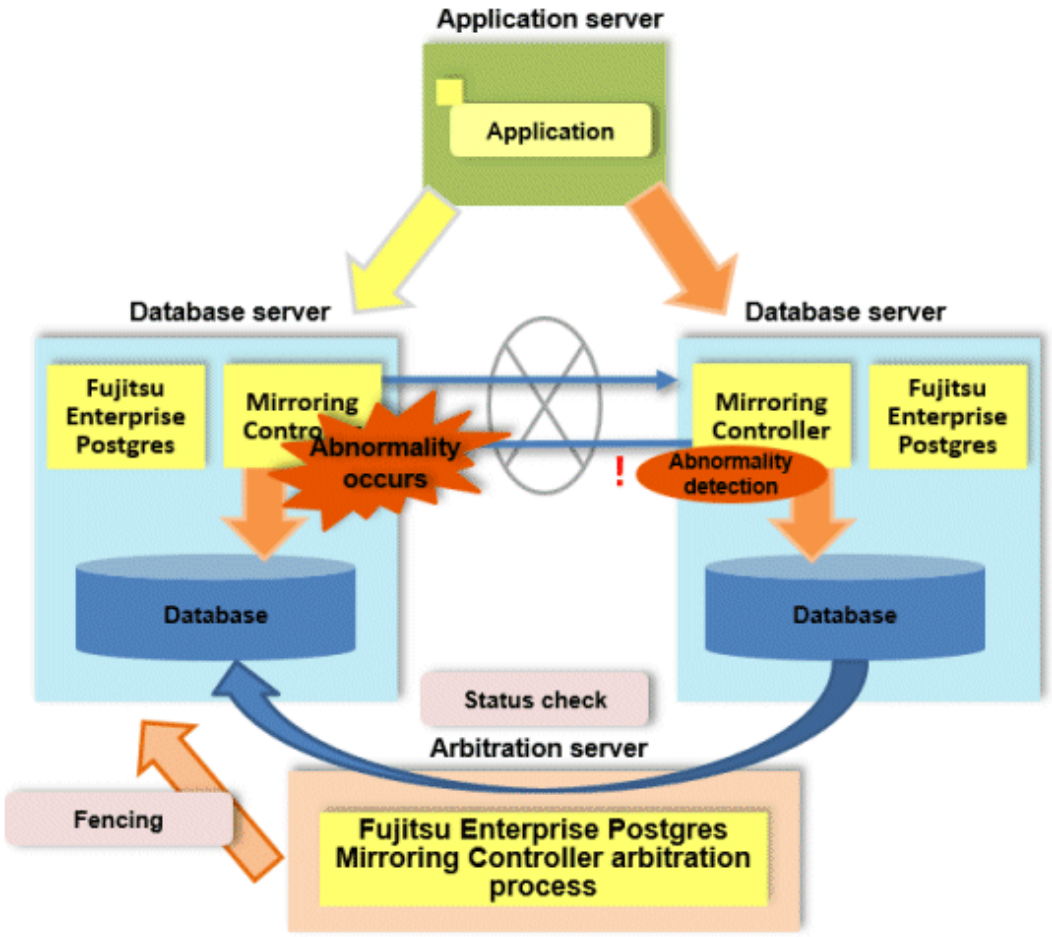
In addition, by using the data on the standby server, reference jobs such as data analysis and form output can be performed in parallel to the jobs on the primary server.

Operation using the arbitration server

Mirroring Controller may not be able to correctly determine the status of the other server if there is a network issue between database servers or a server is in an unstable state. As a result, both servers will temporarily operate as primary servers, so it may be possible to perform updates from either server.

The Server Assistant is a feature that objectively checks the status of database servers as a third party and isolates (fences) unstable servers in such cases.

In database multiplexing mode, the Server Assistant is made available by adding a new server (arbitration server) on which the Server Assistant is installed. Using an arbitration server can prevent the issue mentioned above (both servers temporarily operating as primary servers) and enables highly reliable operation.



See

Refer to the Cluster Operation Guide (Database Multiplexing) for information on the database multiplexing.

L

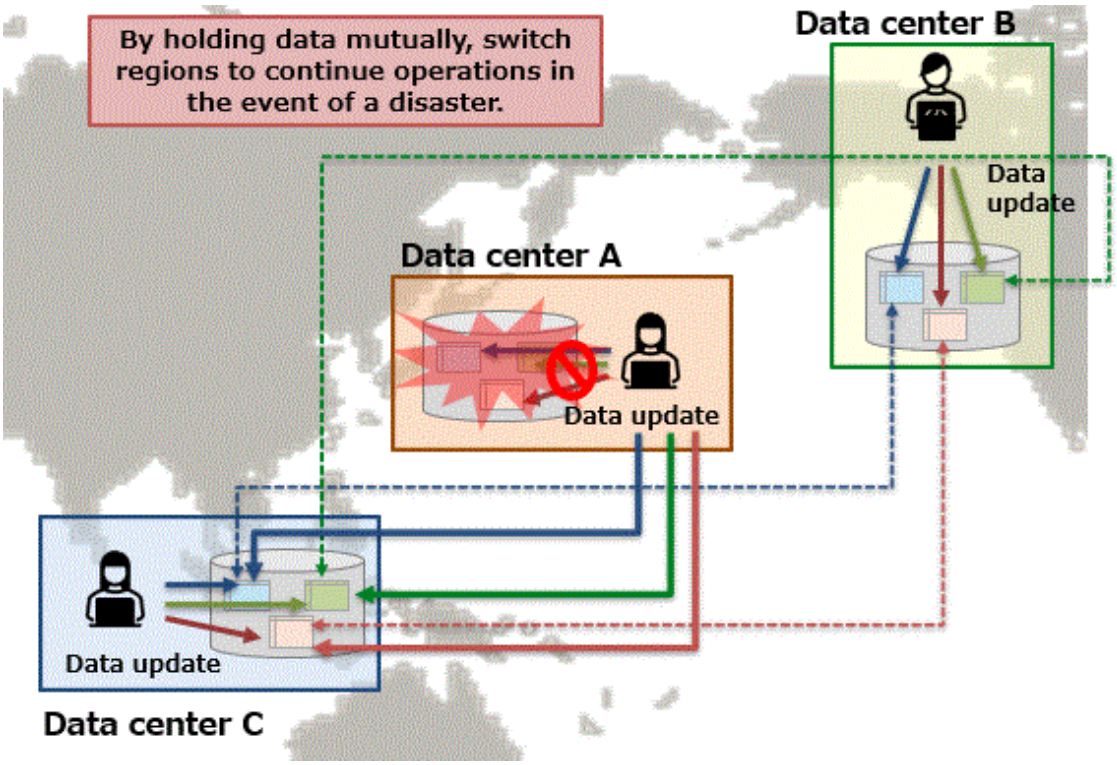
1.4 High Reliability Using the Multi-master Replication

With the multi-master replication feature provided by Fujitsu Enterprise Postgres, you can build a replication configuration that allows independent data update processing on multiple nodes. In this multi-master configuration, all nodes function as masters, allowing clients to write (update, insert, delete) data to any node, and those changes are automatically synchronized to all other nodes. By utilizing this multi-master replication feature, you can enhance business continuity in disaster recovery.

Enhancing business continuity in disaster recovery

In traditional single-master configurations, standby nodes are prepared in case of unforeseen circumstances, and if the master node stops due to a disaster, a failover process is required to promote the standby node to a new master. In this case, there are costs associated with preparing standby nodes, and there is a certain amount of time and risk of data loss involved in the failover process.

In a multi-master configuration, all nodes feature as independent masters, so even if one node stops, the remaining nodes can continue the service. For example, by building a multi-master replication configuration between geographically separated data centers, if one data center completely stops functioning due to a disaster such as an earthquake or a large-scale power outage, the other data center can quickly take over all processing. This minimizes system downtime and ensures business continuity while reducing the risk of data loss due to data transfer.



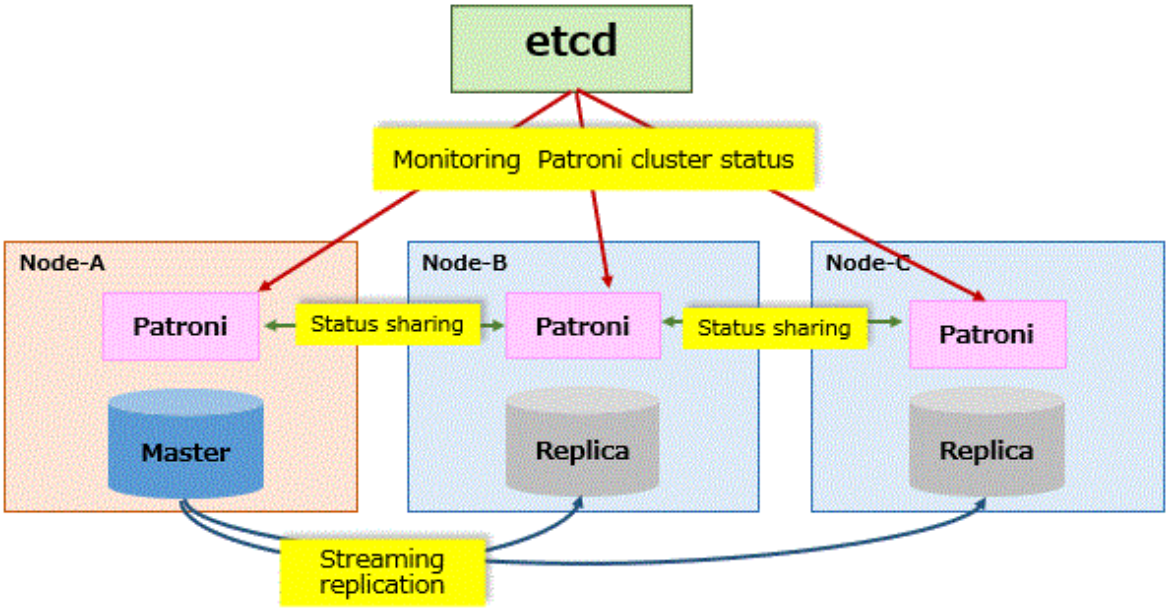
See

Refer to the Cluster Operation Guide (Multi-master Replication) for information on the multi-master replication.

L

1.5 High Reliability through a Cluster System Using Patroni

By combining Fujitsu Enterprise Postgres with Patroni and etcd, a highly available cluster system can be built. This cluster system provides automatic failover, replication management, cluster state monitoring, and automatic recovery feature, realizing highly reliable system operation. In particular, Patroni automates leader election and replica management of PostgreSQL instances by utilizing the etcd cluster as a Distributed Consensus Store (DCS), reducing the burden of operation management.



See

Refer to "Building a Cluster System Using Patroni" in the Operation Guide for information about cluster systems using Patroni.

L

1.6 High Reliability Using Failover Integrated with the Cluster Software

If the system stops, services are interrupted until recovery is complete. In large-scale systems, the interruption takes longer, and may cause significant disruption for many people receiving the services.

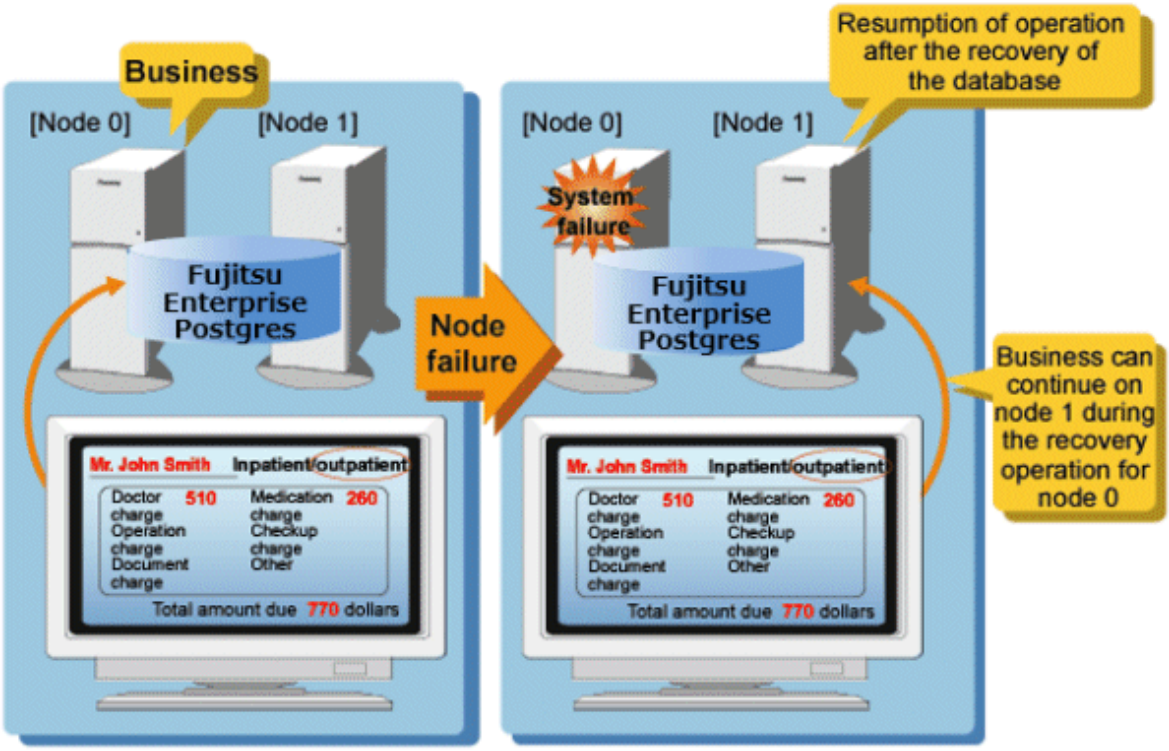
In Fujitsu Enterprise Postgres, the failover feature integrated with the cluster software can minimize the system stoppage time when an issue occurs.

Medical accounting system

Some hospitals with a large number of patients manage and operate the various data required for accounting in a database. If this accounting system stops and takes several minutes to recover, it is expected that this will have a significant impact.

But if failover is applied to this kind of system and an issue occurs on the operating server, it is quickly switched and the standby server takes over operation, so that services are provided without interruption.

The example below illustrates a medical accounting system using failover.



See

Refer to the Cluster Operation Guide (PRIMECLUSTER) for information on the failover feature integrated with the cluster software.

1.7 Seamless Migration from Oracle Databases

Fujitsu Enterprise Postgres supports Orafce, to provide compatibility with Oracle databases.

Using the compatibility feature reduces the cost of correcting existing applications and results in easy database migration.

See

Refer to "Compatibility with Oracle Databases" in the Application Development Guide for information on compatible features.

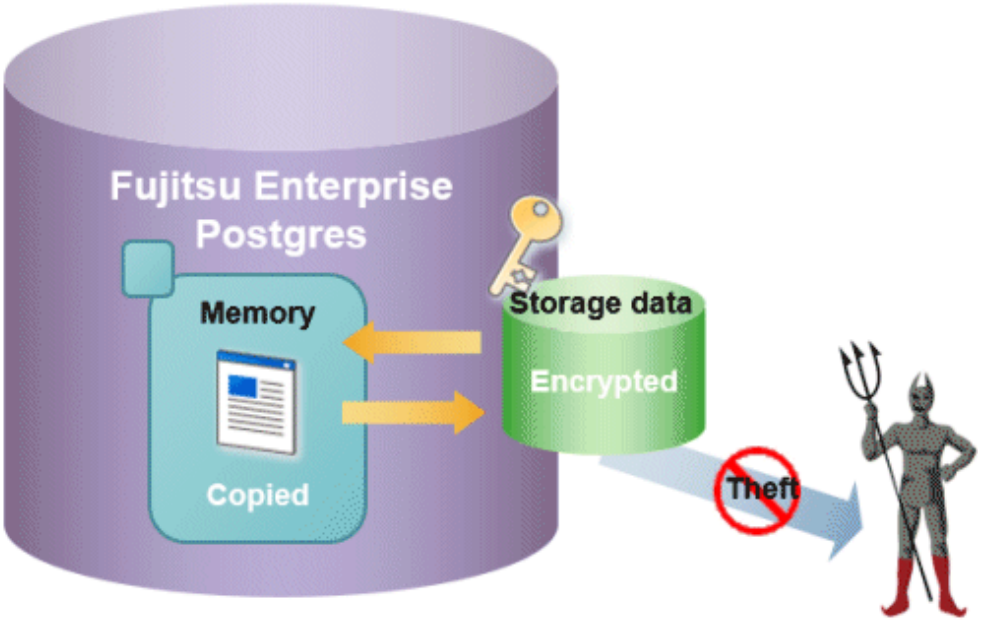
1.8 Storage Data Protection Using Transparent Data Encryption

The encryption of data to be stored in a database is essential under the following encryption requirements of PCI DSS (Payment Card Industry Data Security Standard), the data security standard of the credit industry:

- Confidential information (such as credit card numbers) can be encrypted.
- The encryption key and data are managed as separate entities.
- The encryption key is replaced at regular intervals.

To satisfy these requirements, Fujitsu Enterprise Postgres provides a transparent data encryption feature. Note that PostgreSQL uses an encryption feature called pgcrypto, which can also be used in Fujitsu Enterprise Postgres, but requires

applications to be modified. Therefore, we recommend using Fujitsu Enterprise Postgres's transparent data encryption feature.



The transparent data encryption feature also allows you to choose an external key management system as the storage location for your encryption keys.

See

Refer to "Protecting Storage Data Using Transparent Data Encryption" or "Using Transparent Data Encryption with Key Management Systems as Keystores" in the Operation Guide for information about transparent data encryption.

Refer to "Protecting Storage Data Using Transparent Data Encryption" in the Operation Guide for information about transparent data encryption.

If you use the --save-fullpage option of the pg_waldump command for a WAL file output by an instance that uses transparent data encryption, an error may occur. This option is a function that displays database pages that have been processed through compression or encryption included in WAL in their unprocessed state (expanded or decrypted). However, since the pg_waldump command does not have complete access to the database, it cannot obtain the information necessary for decryption. Therefore, if there is WAL that needs to be decrypted, executing a command with this option will result in an error.

1.9 Role-Based Access Control and Directory Service Integration for Security Management

Fujitsu Enterprise Postgres supports robust database operations by enabling fine-grained access control within the database based on user roles through Role-Based Access Control (RBAC). It also integrates with external directory services, allowing for centralized user management across the organization and automatic assignment of database privileges.

Access control using RBAC

RBAC is a method of managing privileges through roles, rather than directly granting privileges to users. Fujitsu Enterprise Postgres provides the following features based on this RBAC concept to achieve stronger access control.

- Policy-based Login Security
- Management of Access Control by Confidentiality Management

Centralized user management with directory services

Directory services centralize user information on a network and provide a foundation for authentication and authorization. Fujitsu Enterprise Postgres enhances its integration with directory services by supporting the following:

ldap2pg support

Fujitsu Enterprise Postgres supports ldap2pg, allowing you to synchronize Fujitsu Enterprise Postgres roles with an LDAP directory (a directory service protocol). This enables the integration of user information centrally managed in the directory service with Fujitsu Enterprise Postgres roles.

Refer to "ldap2pg" in the Installation and Setup Guide for Client for details.

LDAP authentication support

Fujitsu Enterprise Postgres supports user authentication via LDAP. In addition to managing users within the database itself, you can authenticate logins to the database using user information stored in the organization's directory service.

Refer to "LDAP Authentication File Settings" in the Installation and Setup Guide for Server for details.

1.9.1 Policy-based Login Security

By defining login security policies as profiles and assigning profiles to database users, you can automatically lock users who are not connected to the database for an extended period of time and enforce password authentication policies.

Profiles can have the following settings:

- Managing dormant users
- Managing policies when using password authentication
  - Set a password life time
  - Restrict password reuse
  - Lock users that have failed to log in continuously
  - Set a grace period after the password life time until the database can no longer be operated
  - If the account is locked due to repeated login failures, set the period during which the lock will be automatically unlocke.
  - Set a grace period (gradual password rollover time) for using old passwords after changing password

This enables login management and password management according to the security policy.

Refer to "Policy-based Login Security" in the Operation Guide for details.

1.9.2 Management of Access Control by Confidentiality Management

The confidentiality management feature supports the realization of access control according to the confidentiality level of data. We also support the work of confirming that operations are being performed according to access control.

This feature can only be used in Advanced Edition.

Access control

Access to sensitive data must be restricted in order to comply with the laws and rules governing data protection regulations. However, designing the access control is not easy in databases with diverse data and users performing diverse tasks. This is because for every combination of all database object and all role that can access the data, you must decide whether to allow access and define it in the database.

In our real world, we don't do that. For example, in a business group data with the same confidentiality level, and group several roles accessing to that data. After that, it makes more sense to consider whether access should be granted for combinations of group of data and group of roles. Because once data is added, deciding which group it belongs to naturally determines who can access it. When adding a role, it is enough to think about which role's group (called a confidentiality group) to add it to. The confidentiality management feature supports such a natural design.

Also, data belonging to a high level of confidentiality may need to be protected against unauthorized access to physical media and files, as well as access from users who can log into the database. The confidentiality management feature can force the encryption of tables belonging to high confidentiality levels. Similarly, you can force roles that belong to a confidentiality group to have attributes less than or equal to those set for the confidentiality group. Such table encryption and role management can also be designed naturally with this feature.

Inspection of operation

In order to comply with the laws and rules that define data protection regulations, it is necessary to ensure that the database is operated safely as designed. If you are using the confidentiality management feature, you do not have to worry about such things. However, if table or role definitions are changed without using this feature, it must be detected timely. The confidentiality management feature does not prohibit or detect such acts. Instead, use audit logs to detect such changes, etc.

However, even if they detect it, they may forget to deal with it. In order to prevent this, it is necessary to periodically check the differences between the confidentiality levels and confidentiality groups and the actual table and role definitions. At that time, you can use theconfidentiality management feature provided to obtain the difference.

See

Refer to "Confidentiality Management" in the Security Operation Guide for details.

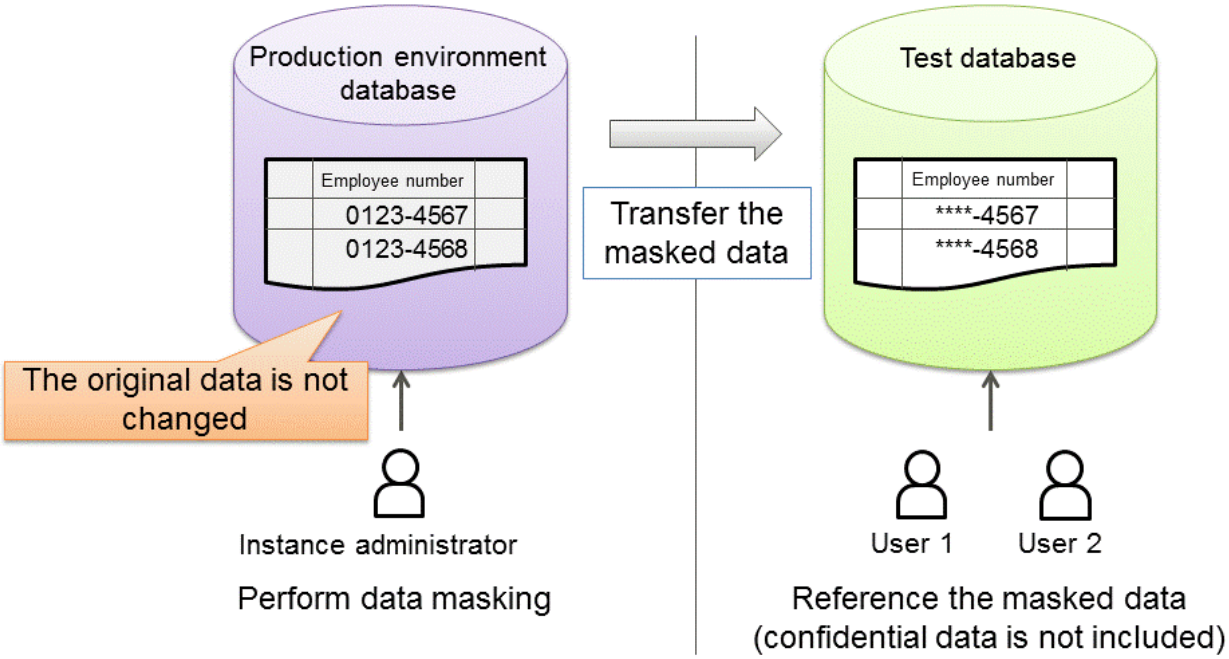
1.10 Data Masking for Improved Security

Fujitsu Enterprise Postgres provides a data masking feature that protects data to maintain security of data handled in systems.

The data masking feature changes the returned data for queries from applications and makes it available for reference without exposing the actual data.

For example, for a query of employee data, digits except the last four digits of an eight-digit employee number can be changed to "*" so that it can be used for reference.

Also, the data changed by the data masking feature can be transferred to a test database so that users who perform testing or development can reference the data. As production data should not be used in a test or development environment because of the risk of data leakage, this feature enables data that is similar to actual production data to be safely used in those environments.



See

Refer to "Data Masking" in the Operation Guide for information on data masking.

1.11 Security Enhancement Using Audit Logs

Details relating to database access can be retrieved in audit logs. The audit log feature can be used to counter security threats such as unauthorized access to the database and misuse of privileges.

In PostgreSQL, logs output as server logs can be used as audit logs by using the log output feature. There are, however, logs that cannot be analyzed properly, such as SQL runtime logs, which do not output the schema name. Additionally, because the output conditions cannot be specified in detail, log volumes can be large, which may lead to deterioration in performance.

The audit log feature of Fujitsu Enterprise Postgres enables retrieval of details relating to database access as an audit log by extending the feature to pgaudit. Additionally, audit logs can be output to a dedicated log file or server log. This enables efficient and accurate log monitoring.

This feature can only be used in Advanced Edition.

See

Refer to "Audit Log Feature" in the Security Operation Guide for details.

1.12 Enhanced Query Plan Stability

Fujitsu Enterprise Postgres estimates the cost of query plans based on SQL statements and database statistical information, and selects the least expensive query plan. However, like other databases, Fujitsu Enterprise Postgres does not necessarily select the most suitable query plan. For example, it may suddenly select unsuitable query plan due to changes in the data conditions.

In mission-critical systems, stable performance is more important than improved performance, and changes in query plans case to be avoided. In this situation, pg_hint_plan and pg_dbms_stats can be used to stabilize the query plan.

See

Refer to "Enhanced query plan stability" in the Operation Guide for information on enhanced query plan stability.

Note

For pg_hint_plan and pg_dbms_stats, take advantage of features introduced when installing Fujitsu Enterprise Postgres. Fujitsu Enterprise Postgres does not support other similar open-source features.

1.13 Increased Aggregation Performance Using the In-memory Feature

Fujitsu Enterprise Postgres provides the in-memory feature, which uses columnar index and memory-resident data. This reduces disk I/Os and enhances aggregation performance.

This feature can only be used in Advanced Edition.

Columnar index

Many aggregation processes may require a large portion of data in a particular column. However, traditional row data structure reads unnecessary columns, resulting in inefficient use of memory and CPU cache, and slower processing. Fujitsu Enterprise Postgres provides a type of columnar index, VCI (Vertical Clustered Index). This addresses the above issues, and enhances aggregation performance.

VCI provides the following benefits:

- Minimizes impact on existing jobs, and can perform aggregation using job data in real time.
- Provided as an index, so no application modification is required.
- Stores data also on the disk, so aggregation jobs can be quickly resumed using a VCI even if a failure occurs (when an instance is restarted).
- If the amount of memory used by VCI exceeds the set value, aggregation can still continue by using VCI data on the disk.

It also provides the features below:

- Disk compression
  Compresses VCI data on the disk, minimizing required disk space. Even if disk access is required, read overhead is low.
- Parallel scan
  Enhances aggregation performance by distributing aggregation processes to multiple CPU cores and then processing them in parallel.

In-memory data

The following features keep VCI data in memory and minimize disk I/Os on each aggregation process.

- Preload feature
  Ensures stable response times by loading VCI data to memory before an application scans it after the instance is restarted.
- Stable buffer feature
  Reduces disk I/Os by suppressing VCI data eviction from memory by other job data.

Purposes of this feature

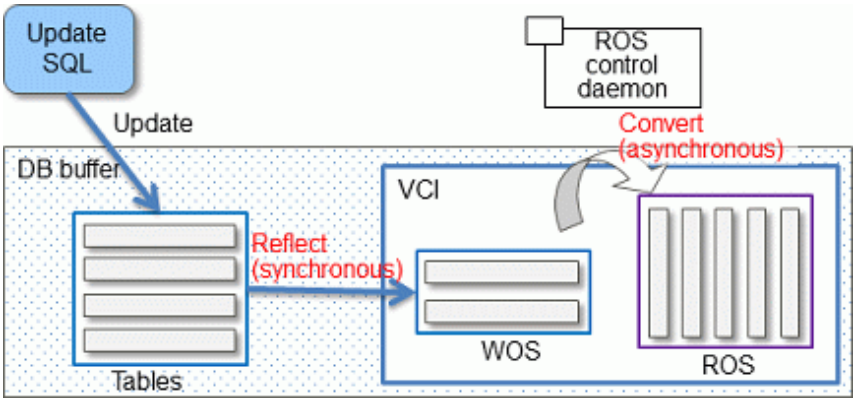
This feature has a data structure that can efficiently use the newly added resources, and aims to enhance the existing aggregation processing in normal operations to be faster than parallel scan. It shares the same purpose of enhancing aggregation performance with the parallel scan feature that is provided separately, but differs in that it speeds up nightly batch processes by utilizing available resources.

VCI architecture

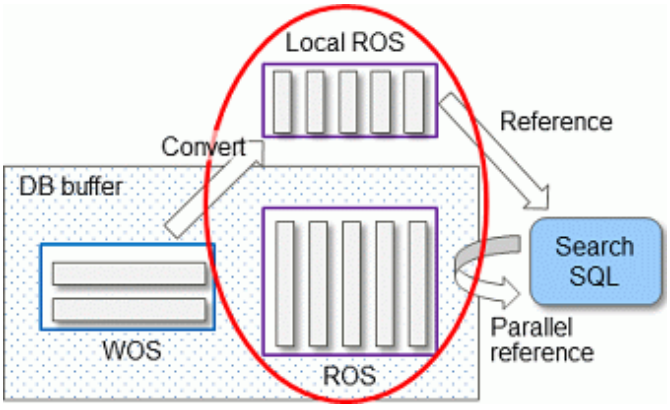
This section briefly explains VCI architecture as it contains basic terminology required, for example, when setting parameters.

Update and aggregation operations to enable real time use of job data are described.

VCI has write buffer row-based WOS (Write Optimized Store) in addition to the columnar data structure ROS (Read Optimized Store). Converting each update into a columnar index has a significant impact on the update process response times. Therefore, data is synchronously reflected to the row-based WOS when updating. After a certain amount of data is stored in WOS, the ROS control daemon asynchronously converts it to ROS. As above, the entire VCI is synchronized with the target table column, minimizing update overhead.



The same scan results can be obtained without a VCI by using WOS in conjunction with ROS. More specifically, WOS is converted to Local ROS in local memory for each aggregation process, and aggregated with ROS.



See

Refer to "Installing and Operating the In-memory Feature" in the Operation Guide for information on installation and operation of VCI.

Refer to "Scan Using a Vertical Clustered Index (VCI)" in the Application Development Guide for information on scan using a VCI.

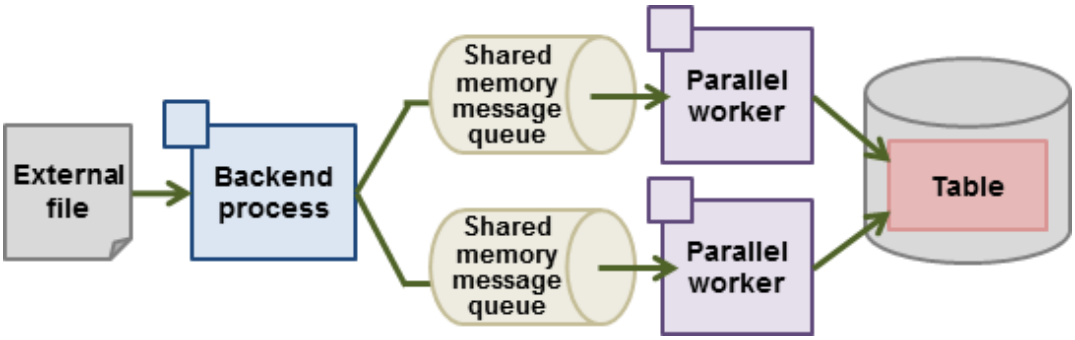
1.14 High-Speed Data Load

High-speed data load executes COPY FROM commands using multiple parallel workers. Because conversion of data from the external file to the appropriate internal format, table creation, and index creation are performed in parallel, it is possible to load large volumes of data at high speed.

This feature can only be used in Advanced Edition.

Architecture of high-speed data load

High-speed data load is required for parameter setting and resource estimation, so a brief description of its architecture is provided below.



High-speed data load uses a single backend process collaborating with multiple parallel workers to perform data load in parallel. Data is exchanged between the backend process and parallel workers via shared memory message queues. The backend process distributes the loaded data of external files to multiple parallel workers. Each parallel worker then converts the data loaded from the shared memory message queue into the appropriate internal format, and inserts it into the table. If the table has indexes, their keys are extracted and inserted into the index page.

See

Refer to "High-Speed Data Load" in the Operation Guide for details.

1.15 High availability by using Connection Manager

The Connection Manager provides the following features. You can use these features to increase system availability.

This feature can only be used in Advanced Edition.

Heartbeat monitoring feature

Detects kernel panics between the server running the client and the server running the instance, physical server failures, and inter-server network link downs, and notifies the client or instance. The client is notified as an error event through the SQL connection, and the instance will be notified in the form of a force collection of SQL connections with clients that are out of service.

Transparent connection support feature

When an application wants to connect to an instance of an attribute (Primary/Standby) configured with replication, it can do so without knowing which server the instance is running on.

Load balancing and read connection redistribution feature

With load balancing, if selection is possible, connections can be preferentially made to the server accepting the fewest connections.

Using read connection redistribution, when a standby server is promoted, Connection Manager disconnects existing connections, allowing the promoted server to focus on the workload of the newly added primary server. Only connections using Connection Manager are disconnected, and the disconnection occurs when a new query is executed.

Additionally, when the number of connections between standby servers becomes uneven, existing connections can be disconnected. This gives applications an opportunity to reconnect, thereby balancing the number of connections. For example, this can handle changes in the number of connections due to the addition or removal of standby servers, or the suspension of specific operations. Only non-update connections using Connection Manager are disconnected, and the disconnection occurs when a new query is executed.

Initial data synchronization waiting feature for standby servers

You can exclude standby servers that are not synchronized with the primary server on a transaction-by-transaction basis from the connection destination candidates.

Information

The available client drivers for Connection Manager are libpq (C language library), ECPG (embedded SQL in C), ECOBPG (embedded SQL in COBOL), JDBC driver, ODBC driver, Python language package (psycopg) and Go language.

See

Refer to the Connection Manager User's Guide for details.

1.16 Memory Usage Reduce with Meta cache Reduction and Limit

When executing SQL, you must refer to the definition of the table or index you want to access. These definitions are cached in the local memory of the backend process separately from the shared memory buffer of the database on shared_buffers because they are referenced each time SQL is executed. The direct definition is a tuple of system catalogs. The cache for this tuple is called "Catalog Cache". A structure that makes the definition easy to use is called a "Relations Cache". And in Fujitsu Enterprise Postgres, these two are collectively called "Meta Cache".

The meta cache will be kept indefinitely for performance reasons. In a large database, a single backend process accesses a large number of tables and so on, which results in a large meta-cache for the backend process. As a result, the sum of the local memory of the backend process may exceed the realistic memory size.

On the other hand, the feature to reduce the meta cache for the entire instance by sharing the meta cache between backend processes is the Global Meta Cache feature. The current Global Meta Cache feature only shares the catalog cache. Therefore, the metacache in Global Meta Cache now refers to the catalog cache.

What you still cannot share using the current Global Meta Cache feature and need to keep in local memory is the information (Meta cache header) and relation cache to access Global Meta Cache in shared memory. If you do not use the Global Meta Cache feature, keep the catalog and relation caches in local memory. The meta cache held in local memory is called the "Local Meta Cache". The feature to limit the size of a Local Meta Cache by removing it if it has not been accessed for a long time is the Local Meta Cache limit feature.

The Global Meta Cache feature and the Local Meta Cache limit feature can be used together to provide the strictest control over memory consumption. Of course, you can use only one or the other.

However, the Global Meta Cache feature has several percent overhead to access shared memory. The Local Meta Cache limit feature also causes the overhead of reholding the metacache because it may discard the previously held metacache. Therefore, consider using these features when your estimates do not allow for memory consumption.

1.16.1 Memory Usage Reduction Using Global Meta Cache

The Global Meta Cache feature cache the meta cache in shared memory. The meta cache on shared memory is called the Global Meta Cache (GMC).

Without this feature, the meta cache was cached in per-process memory. Therefore, there was a problem increase in memory usage in environments with large databases and large numbers of connections. The Global Meta Cache feature enables sharing of meta caches on shared memory, thereby reducing overall system memory usage.

This feature can only be used in Advanced Edition.

Meta cache

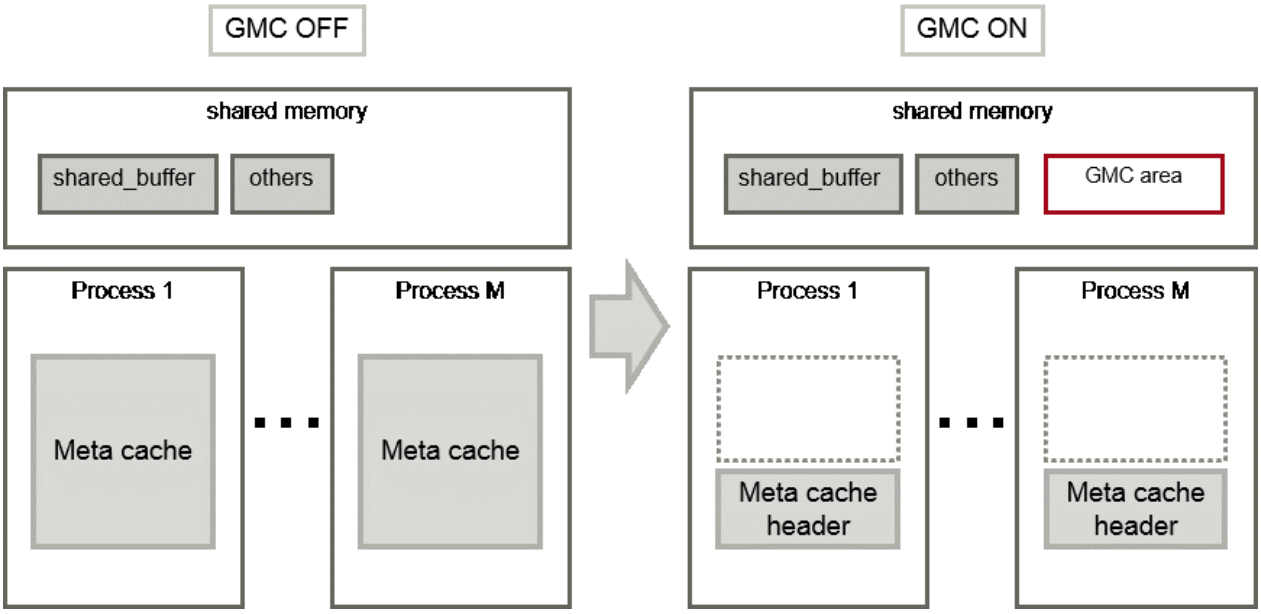
Processing a query involves parsing the query, creating the plan, executing the plan, and so on. PostgreSQL process accesses the system catalog to perform these steps. Once accessed, the system catalog tuples are cached in per-process memory. The

direct definition is one tuple of system catalogs. Each process performs faster query processing by searching the meta cache instead of searching for the required tuples in the system catalog each time.

The meta cache usage increases in proportion to the number of tables and columns accessed. It is cached on a per-process basis, so the system's overall meta cache usage increases in proportion to the number of connections.

Architecture of Global Meta Cache feature

Describes the architecture of the Global Meta Cache feature.



When the GMC feature is on, the per-process meta cache is cached in the GMC area on shared memory. Reference to the GMC area and process-specific work information is cached in the memory of each process. PostgreSQL process searches the meta cache for each process and accesses the GMC based on the reference information. If there is no reference information in the process's memory, it searches the GMC area. If the GMC area also does not have a corresponding meta cache, it accesses the system catalog to create meta cache.

Also, sharing the meta cache does not cause any loss of data consistency. If the system catalog or table definition changes while a transaction is running, the cache deletion or creation does not affect outside of the process running the transaction. After the transaction commits, the GMC area cache is deleted or created. If other transactions are referencing the cache when GMC is tried to be deleted, the deletion is deferred until there are no more references. After a commit, a new transaction sees the new cache instead of the old one.

 See

Global Meta Cache feature is disabled by default. Refer to "Global Meta Cache" in the Operation Guide for information how to decide whether introduce it or not and usage.

1.16.2 Memory Usage Reduction Using Local Meta Cache Limit

Local Meta Cache Limit feature limits the size of a Local Meta Cache by removing it if it has not been accessed for a long time.

Of the definitions that SQL accesses, the main factors that make the Local Meta Cache bloat are tables and indexes. In addition, table column definitions are also maintained as a catalog cache.

For example, in a system where one long-lived connection is shared by various businesses, one connection (that is, backend process) will access many tables. If there are 3,000 such connections, and each connection accesses a table of 50,000, the total amount of memory consumed by the 3,000 backend processes may be a few terabytes.

In such a case, using this feature may reduce it to about several tens of gigabytes.

This feature can only be used in Advanced Edition.

Architecture of Local Meta Cache Limit feature

When this feature is enabled, the caching strategy is to keep the cache as long as possible within the specified upper limit. If holding a new cache exceeds the limit, consider locality of reference and delete the cache from the one with the longest unreferenced time.

However, because the cache used by active transactions cannot be deleted, if a transaction uses a large number of caches, the cache may be held above the limit. In this case, delete the cache at the end of the transaction.

See

Local Meta Cache limit feature is disabled by default. Refer to "Local Meta Cache Limit" in the Operation Guide for information how to decide whether introduce it or not and usage.

1.17 Protect and Efficiently Manage and Use Knowledge Data for AI Applications

Generative AI applications, where accuracy and reliability are critical, often use an approach called RAG that leverages external data sources to improve the accuracy of the generated AI responses and make use of expertise and up-to-date information. Fujitsu Enterprise Postgres provides knowledge data management capabilities for using Fujitsu Enterprise Postgres as an external data source in the RAG approach.

Knowledge in the RAG approach is expressed in text and graph form, and vector data is also leveraged for fast retrieval. Knowledge data management allows you to centralize the management of knowledge data such as vectors and graphs in Fujitsu Enterprise Postgres. In addition to eliminating the need for a dedicated database for each data type, you can protect your knowledge data with the reliable and secure features of Fujitsu Enterprise Postgres.

In addition, the knowledge data management supports:

- Vector data management feature

  This feature stores and retrieves vector data.

L - Semantic text search and automatic vectorization

  Text Semantic Search is a feature that retrieves semantically similar text data from Fujitsu Enterprise Postgres without requiring the application to process vector data. You can also use hybrid search that combines semantic text search and full-text search.

  Automated vectorization is a feature to automatically generate and manage semantic vectors used for semantic search of text, based on declarative specifications.

  Vectorization is performed using pre-trained models called embedding models. At this time, not only external vector embedding models such as Ollama and OpenAI can be used, but also model managed in the database can be used.

- MCP server feature for business data utilization

  Generative AI applications are increasingly dominated by Agentic AI, which aims to achieve goals autonomously. Business data stored in Fujitsu Enterprise Postgres can be utilized by using MCP (Model Context Protocol), which is a standard interface in Agentic AI.

- Graph management feature

  This feature stores and retrieves graph data.

- LangChain linkage

  Knowledge data stored in Fujitsu Enterprise Postgres is available to RAG-based applications using LangChain.

See

Refer to the Knowledge Data Management Feature User's Guide for details.

Appendix A List of Features

The following table lists the main features provided by Fujitsu Enterprise Postgres.

Category	Feature	Linux		Windows	
		AE	SE	AE	SE
Operational management tools	WebAdmin	Y	Y	Y	Y
Improved reliability and availability	Database multiplexing	Y	N	Y	N
	Multi-master replication	Y	N	N	N
	Cluster system using Patroni	Y	N	N	N
	Failover (integration with PRIMECLUSTER)	Y (*1)	Y (*1)	N	N
	Backup/recovery using user exits	Y	N	Y	N
	Connection Manager	Y	N	Y	N
Application development	Embedded SQL integration Java integration ODBC integration .NET Framework integration Python language (psycopg) integration	Y	Y	Y	Y
	Go language integration	Y	Y	N	N
	Features compatible with Oracle databases	Y	Y	Y	Y
Security	Transparent data encryption	Y	Y	Y	Y
	Transparent data encryption using a key management system	Y	Y	N	N
	Data masking	Y	Y	Y	Y
	Audit log	Y	N	Y	N
	Confidentiality management	Y	N	Y	N
	Policy-based Login Security	Y	Y	Y	Y
Support for AI application development	Knowledge data management	Y	Y	Y(*2)	Y(*2)
Performance	In-memory feature	Y	N	Y	N
	High-speed data load	Y	N	Y	N
	Global Meta Cache	Y	N	Y	N
	Local Meta Cache Limit	Y	N	Y	N
Cloud integration	Database Monitoring with Amazon CloudWatch	Y	Y	Y	Y
	Autoscale of Replica Servers through Amazon EC2 Auto Scaling Integration	Y	N	N	N
Performance tuning	Enhanced query plan stability	Y	Y	Y	Y

Y: Provided

N: Not provided

*1: Supported on RHEL only. Also, it cannot be used simultaneously with database multiplexing.

*2: The following features are not available on Windows
- Speed up retrieval processing of vector data management feature
- Semantic text search and automatic vector conversion
- Full-text search
- MCP server feature for business data utilization
- Model management in the database
- Graph management feature

See

For more information on third-party tools that can work with Fujitsu Enterprise Postgres, refer to the "Third-party tools that work with Fujitsu Enterprise Postgre" available at:

https://www.postgresql.fastware.com/resource-center

Appendix B OSS Supported by Fujitsu Enterprise Postgres

The OSS supported by Fujitsu Enterprise Postgres is listed below.

OSS name	Version and level	Platform		Description	Reference
		Linux	Windows		
PostgreSQL	18.3	Y	Y	Database management system	PostgreSQL Documentation
orafce	4.16.3	Y	Y	Oracle-compatible SQL features	"Compatibility with Oracle Databases" in the Application Development Guide
Pgpool-II	4.6.5	Y	N	Failover, connection pooling, load balancing, etc.	"Pgpool-II" in the Installation and Setup Guide for Server
oracle_fdw	2.8.0	Y	Y	Connection to the Oracle database server	"oracle_fdw" in the Installation and Setup Guide for Server
pg_statsinfo	17.0-2	Y	N	Collection and accumulation of statistics	"pg_statsinfo" in the Installation and Setup Guide for Server
pg_hint_plan	18.1.8.0	Y	Y	Tuning (statistics management, query tuning)	- "pg_hint_plan" in the Installation and Setup Guide for Server - "Enhanced Query Plan Stability" in the Operation Guide
pg_dbms_stats	14.0	Y	Y		- "pg_dbms_stats" in the Installation and Setup Guide for Server - "Enhanced Query Plan Stability" in the Operation Guide
pg_repack	1.5.3	Y	N	Table reorganization	"pg_repack" in the Installation and Setup Guide for Server
pg_rman	1.3.18	Y	N	Backup and restore management	"pg_rman" in the Installation and Setup Guide for Server
pgBackRest	2.57.0	Y	N		"pgBackRest" in the Installation and Setup Guide for Server
pgAdmin4	9.11	N	Y (*1)	Operation and development GUI	Operation Guide
pgBadger	13.2	Y	N	Log analysis	"pgBadger" in the Installation and Setup Guide for Server

OSS name	Version and level	Platform		Description	Reference
		Linux	Windows		
pg_bigm	1.2-20250903	Y	N	Full-text search (multibyte)	"pg_bigm" in the Installation and Setup Guide for Server
ldap2pg	6.5.1	Y	N	Managing user	- "LDAP Authentication File Settings" in the Installation and Setup Guide for Server - "ldap2pg" in the Installation and Setup Guide for Client
PostgreSQL JDBC driver	42.7.8	Y	Y	JDBC driver	"JDBC Driver" in the Application Development Guide
psqlODBC	17.00.0007	Y	Y	ODBC driver	"ODBC Driver" in the Application Development Guide
Npgsql	10.0.1 8.0.8	Y (*2)	Y	.NET data provider (*3)	".NET Data Provider" in the Application Development Guide
psycopg	3.2.13	Y	Y	Python driver	"Python Language Package (psycopg)" in the Application Development Guide
pgx	5.8.0	Y	N	Go language	"Go Language" in the Application Development Guide
pgvector	0.8.1	Y	Y	Vector data storage, operation, and search (*4)	"pgvector" in the Installation and Setup Guide for Server
pgvectorscale	0.9.0	Y	N	Accelerating vector data search processing (*5)	"pgvectorscale" in the Installation and Setup Guide for Server
Apache AGE	1.6.0 for PG17	Y	N	Graph data storage and search (*6)	"Apache AGE" in the Installation and Setup Guide for Server
pgai	0.9.2	Y	N	Vectorization of text data using external embedding model and declarative vectorization specification (*7)	Knowledge Data Management Feature User's Guide
pg_cron	1.6.7	Y	N	SQL Job scheduler	"pg_cron" in the Installation and Setup Guide for Server
pgactive	2.1.7	Y	N	Multi-master replication	Cluster Operation Guide (Multi-master Replication)

OSS name	Version and level	Platform		Description	Reference
		Linux	Windows		
Patroni	4.1.0	Y	N	High availability management tool (*8)	"Building a Cluster System Using Patroni" in the Operation Guide
etcd	3.6.7	Y	N	Distributed consensus store (*9)	"Building a Cluster System Using Patroni" in the Operation Guide
PostGIS	3.5.4	L (*10)	N	Geographic information system extension	PostGIS Build Instructions (*11)

Y: Supported

L: Limited support

N: Not supported

*1: Server mode is not supported.
*2: In order for Npgsql to operate, an .NET provided by Microsoft is required, but check the support version of the provider for the OS version on which .NET can operate.

*3: Multiple Npgsql module versions are available. Use the one compatible with your project's .NET target framework.

*4: pgvector by itself cannot generate vector data. pgai allows you to generate vector data for text data using an external embedded model. To use vector representations in other data formats, use software or services that generate vector data and store the generated vector data in a database.
The manual may refer to pgvector documentation. Refer to the documents published at the following site:
https://github.com/pgvector/pgvector/blob/v0.8.1/README.md

*5: The manual may refer to pgvectorscale documentation. Refer to the documents published at the following site:
https://github.com/timescale/pgvectorscale/blob/0.9.0/README.md

*6: The manual may refer to Apache AGE documentation. Refer to the documents published at the following site:
https://age.apache.org/age-manual/master/index.html

*7: The manual may refer to pgai documentation. Refer to the documents published at the following site:
https://github.com/timescale/pgai/blob/pgai-v0.9.2/README.md

*8: The manual may refer to Patroni documentation. Refer to the documents published at the following site:
https://patroni.readthedocs.io/en/latest/

*9: The manual may refer to etcd documentation. Refer to the documents published at the following site:
https://etcd.io/docs/v3.6/

*10: OSS is not included in the Fujitsu Enterprise Postgres program and does not provide a fix. Therefore, you should refer to the PostGIS Build Instructions and download the required OSS.
The PostGIS Build Instructions will be updated once a version with new fixes is released by the community and verified by Fujitsu. A revision of the PostGIS Build Instructions will be notified when the fix for server feature is provided. Build according to the updated PostGIS Build Instructions to take advantage of the new modified version.

*11: Refer to the PostGIS Build Instructions to complete the build and setup. The PostGIS Build Instructions can be obtained from the following site:
https://www.postgresql.fastware.com/knowledge-base/how-to/installing-postgis

Appendix C Features that can be Used on Servers Other than the Database Server

This chapter explains the configuration and operating environment of features to be installed and used on servers other than the database server when used in conjunction with the Fujitsu Enterprise Postgres database server.

In this chapter, Fujitsu Enterprise Postgres programs are referred to as server programs.

Below are features to be installed and used on servers other than the database server:

- WebAdmin
- Server Assistant
- Pgpool-II (failover, connection pooling, and load balancing)

C.1 WebAdmin

If there is only one database server, WebAdmin is normally installed on the same server as the database (the WebAdmin program can be installed at the same time as the server program).

If there are multiple database servers, database server instances can be managed collectively if a dedicated WebAdmin server is used. In this case, the WebAdmin program is installed on the WebAdmin server, and the server program and WebAdmin program are installed on the database server.

See

- Refer to "1.2 Simple GUI-Based Installation and Operation Management" for information on WebAdmin.
- Refer to "Determining the Preferred WebAdmin Configuration" in the Installation and Setup Guide for Server for information on the server configuration when using WebAdmin.
- Refer to "Required Operating System" in the Installation and Setup Guide for Server for information on the operating environment of WebAdmin.

C.2 Server Assistant

To use the Server Assistant, the Server Assistant program is installed on a dedicated server (arbitration server).

See

- Refer to "Overview of Database Multiplexing Mode" in the Cluster Operation Guide (Database Multiplexing) for information on the Server Assistant and the server configuration.
- Refer to "Required Operating System" in the Installation and Setup Guide for Server Assistant for information on the operating environment of the Server Assistant.

C.3 Failover, Connection Pooling, and Load Balancing Features of Pgpool-II

Pgpool-II is software that is placed between the database server and database client to relay the connection.

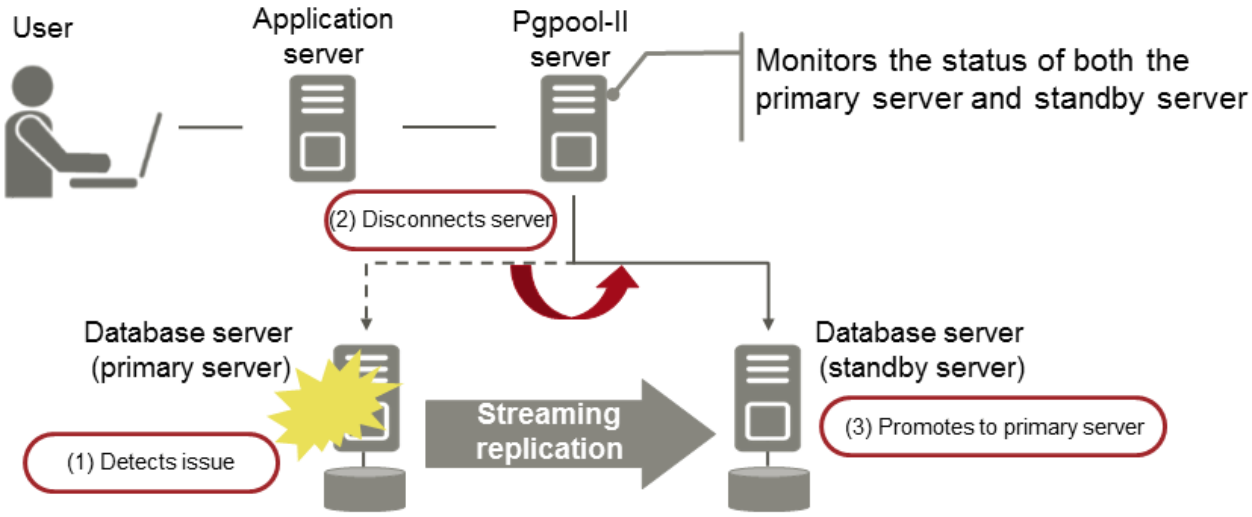
Pgpool-II provides the failover, connection pooling, and load balancing features for use during streaming replication.

Failover

In PostgreSQL, a database can be made redundant (building a high availability system) using synchronous streaming replication.

If the database server of either the primary server or standby server fails or is no longer accessible when using synchronous streaming replication, jobs will stop.

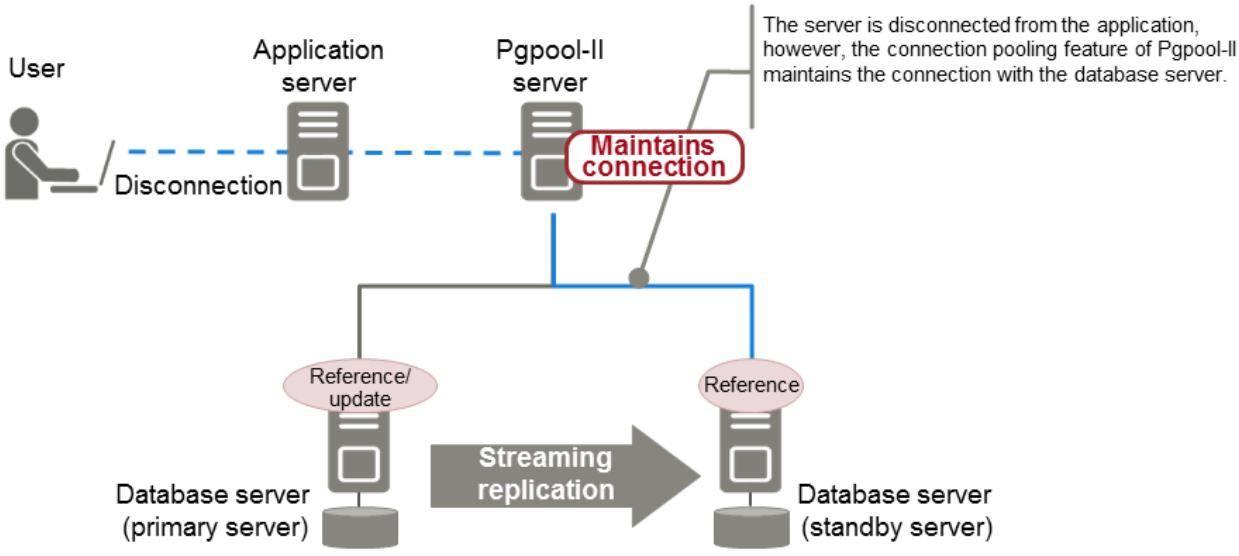
Failover monitors the status of each database and automatically disconnects the server when an error occurs. As a result, jobs can continue uninterrupted on the remaining server.



Connection pooling

This feature maintains (pools) the connection established with the database server, and reuses that connection each time a new connection with the same properties (user name, database, and protocol version) arrives.

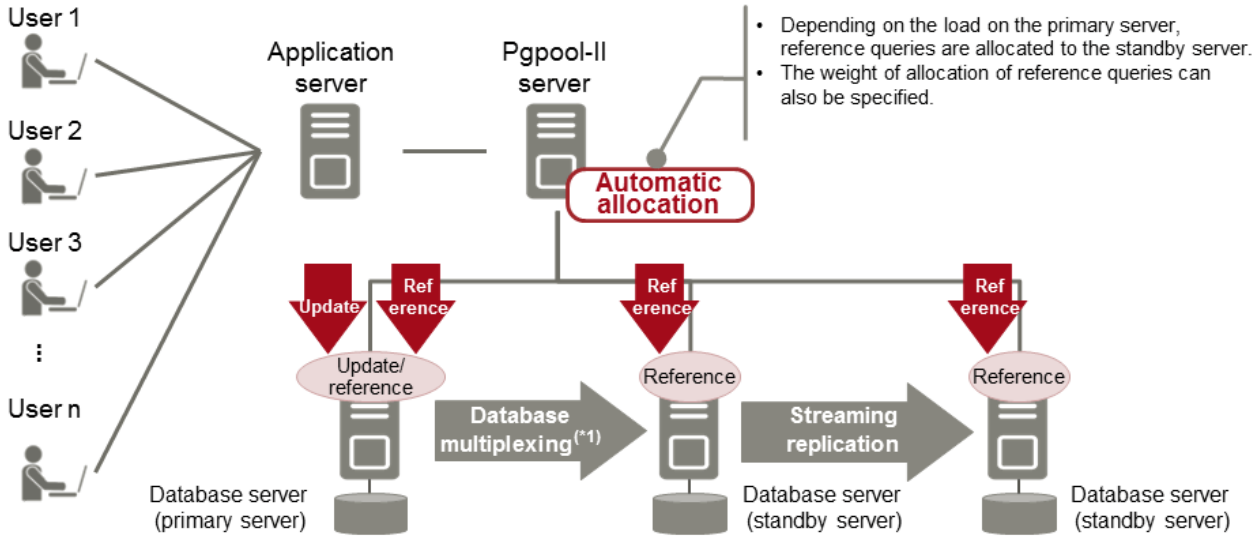
Connection pooling reduces the connection overhead for the database server, improving throughput of the whole system.



Load balancing

This feature distributes reference queries to multiple database servers, improving throughput of the whole system.

By combining load balancing with the Fujitsu Enterprise Postgres database multiplexing feature or the PostgreSQL streaming replication feature, load on the database server is reduced.



*1: The arbitration server used during database multiplexing has been omitted from this document.

See

- Refer to "System configuration when using Pgpool-II" in the Installation and Setup Guide for Server for information on the server configuration when using Pgpool-II.
- Refer to "Required Operating System" in the Installation and Setup Guide for Server for information on the operating environment of Pgpool-II.

Index