FUJITSU Enterprise Postgres 9.5

General Description

Windows/Linux

J2UL-2115-02ENZ0(00)
June 2016

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 Control 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

```
Second edition: June 2016
First edition: March 2016
```

Copyright

Contents

Chapter 1 FUJITSU Enterprise Postgres Basics

FUJITSU Enterprise Postgres is a database that is fully compatible with PostgreSQL. The reliability, operability, and usability of FUJITSU Enterprise Postgres can be further improved by extending the PostgreSQL functionality.

This chapter explains the functionality extended 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.

- 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.

- 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.

- Linkage with integrated development environment
  FUJITSU Enterprise Postgres links with Visual Studio, thereby allowing a standard framework to be used to create applications.

- Storage Data Protection using Transparent Data Encryption
  Information can be protected from data theft by encrypting data to be stored in the database.

- 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.

- Enhanced query plan stability
  The following features can control SQL statement query plans:

  - Optimizer hints

  - Locked statistics

  These features are used for curbing performance deterioration caused by changes in SQL statement query plans, such as with mission-critical jobs that emphasize performance stability over improved SQL statement processing performance.

- Parallel scan
  Parallel scan uses available resources within the server to speed up scan processing for queries that require searching a large amount of data.

- Increased aggregation performance using the in-memory feature
  The following features help speed up scans even when aggregating many rows.

  - Vertical Columnar Index (VCI)

- In-memory data

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

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.

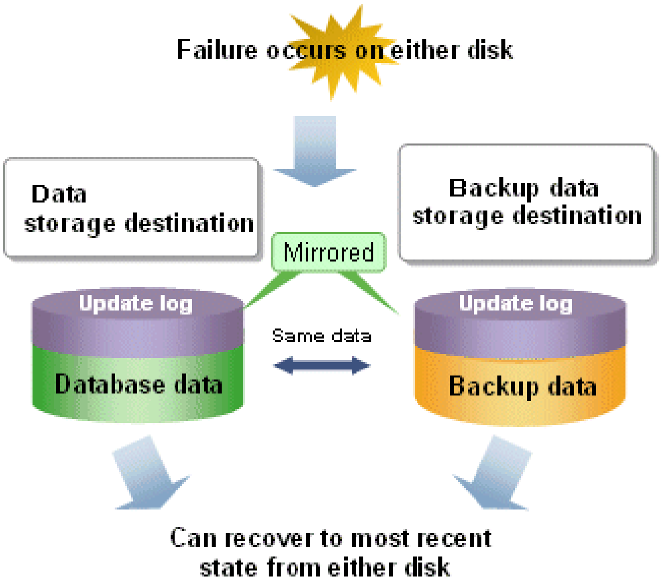
In order to recover the database, FUJITSU Enterprise Postgres accumulates a history of database update operations, such as data additions and deletions, as an update log.

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.

Information

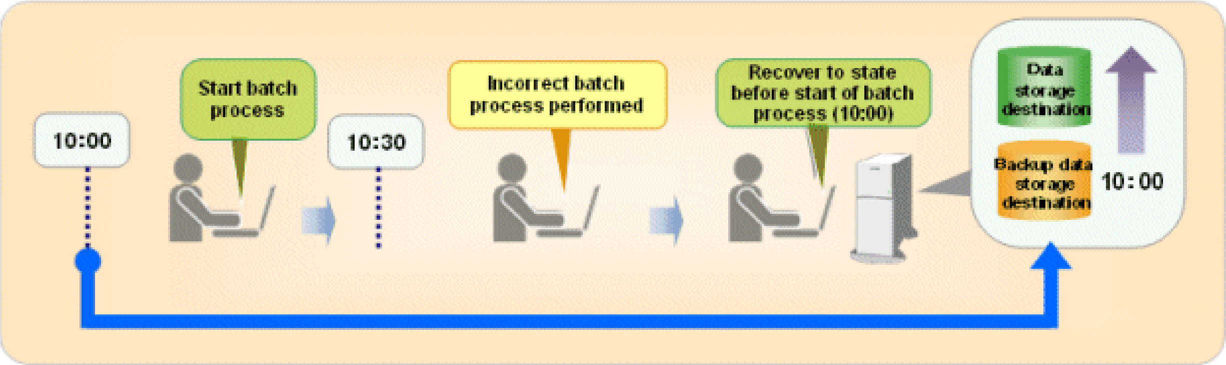
Recovery using WebAdmin requires less time and effort, since WebAdmin automatically determines the scope of the operation.


Failure occurs on either disk
Data storage destination
Backup data storage destination
Mirrored
Update log
Database data
Same data
Update log
Backup data
Can recover to most recent state from either disk

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.


10:00
Start batch process
10:30
Incorrect batch process performed
Recover to state before start of batch process (10:00)
Data storage destination
Backup data storage destination
10:00

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.

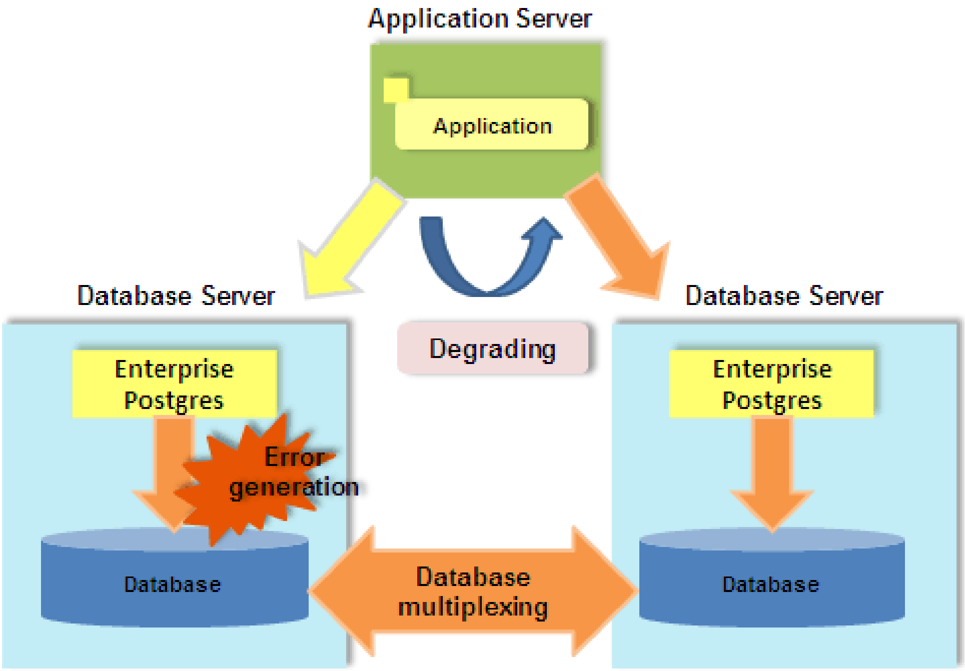
  In particular, FUJITSU Enterprise Postgres can automatically identify and isolate the location of errors. This simplifies the recovery process and enables faster recovery.

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 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.

See

Refer to "Database Multiplexing Mode" in the Cluster Operation Guide for information on the database multiplexing.

1.4 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.

Information

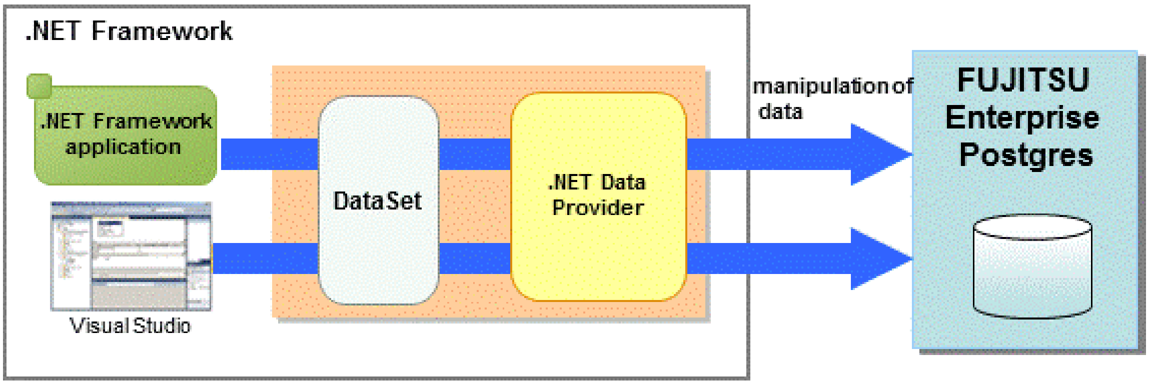
The features compatible with Oracle databases are enabled by default.

1.5 Linkage with Integrated Development Environment

You can link with Microsoft Visual Studio to create application. And, you can automatically generate applications to access database resources by linking to Visual Studio.

Relationship between .NET Framework and FUJITSU Enterprise Postgres

FUJITSU Enterprise Postgres provides .NET Data Provider, which is an interface for ADO.NET of .NET Framework. This enables you to select FUJITSU Enterprise Postgres as the connection destination database of ADO.NET and use the intuitive and efficient application development features of Visual Studio.

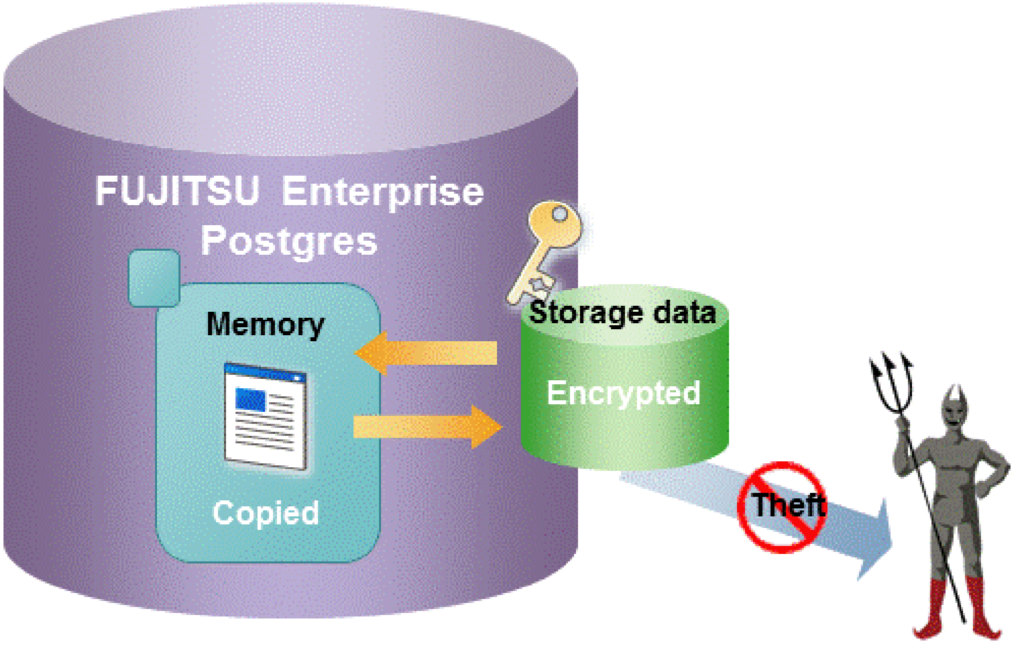


1.6 Storage Data Protection using Transparent Data Encryption

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.



See

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

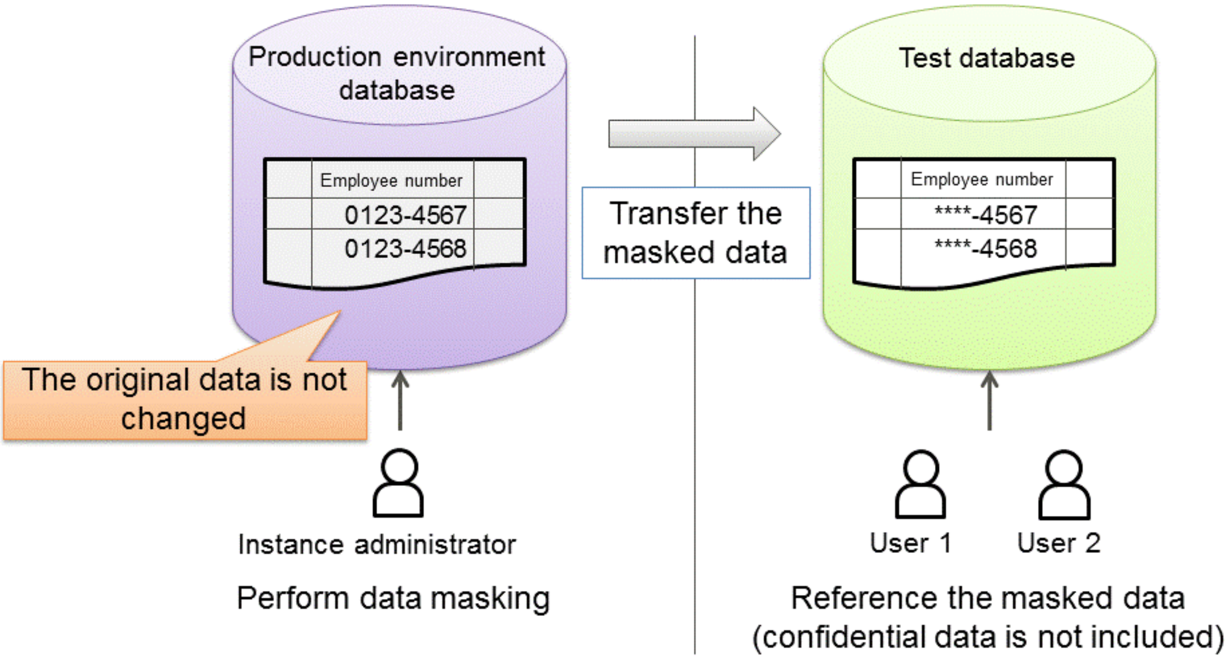
1.7 Data Masking for Improved Security

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.

1.8 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, the following features can stabilize query plans:

- Optimizer hints

  You can use pg_hint_plan to specify a query plan in each individual SQL statement.

- Locked statistics

  You can use pg_dmbs_stats to lock statistical information per object, such as a database, schema, or table.

See

Refer to "Optimizer Hints" in the Application Development Guide for information on optimizer hints.

Refer to "Locked Statistics" in the Application Development Guide or information on locked statistics.

Note

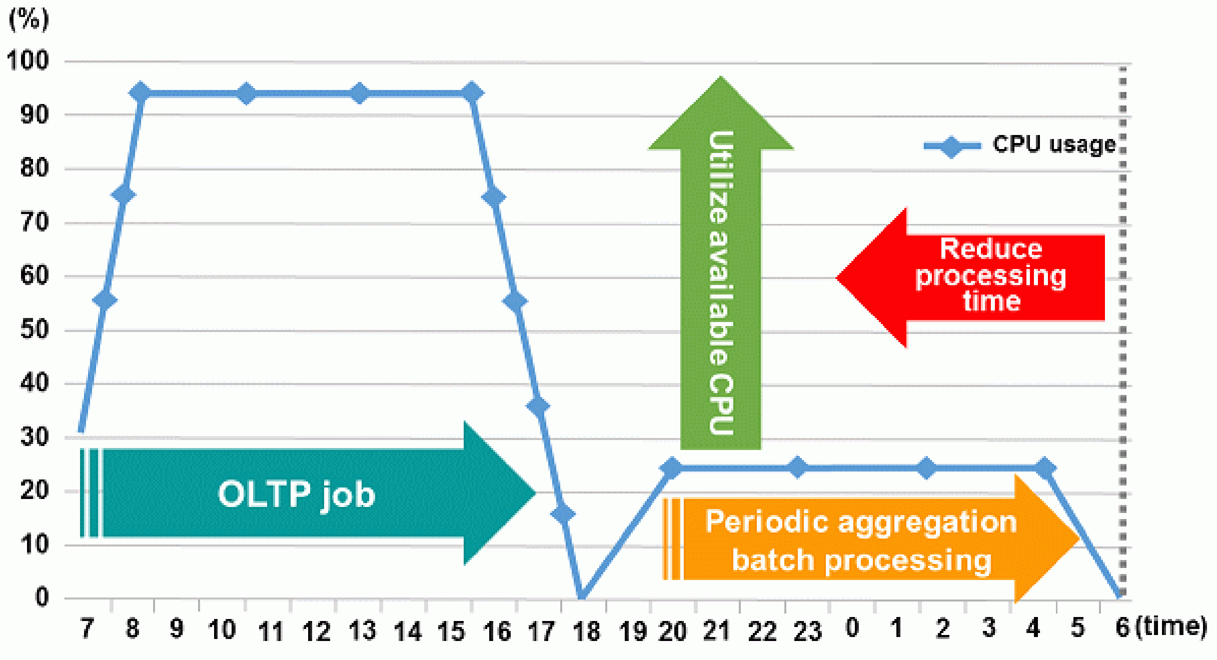
Use the features provided when FUJITSU Enterprise Postgres is installed for optimizer hints and locked statistical information. FUJITSU Enterprise Postgres does not support other similar open-source features.

1.9 Parallel Scan

FUJITSU Enterprise Postgres enhances scan performance by distributing scan processes that contain a high number of items to multiple CPU cores (according to their load) and then processing them in parallel.

Periodic aggregation, in periods such as night time

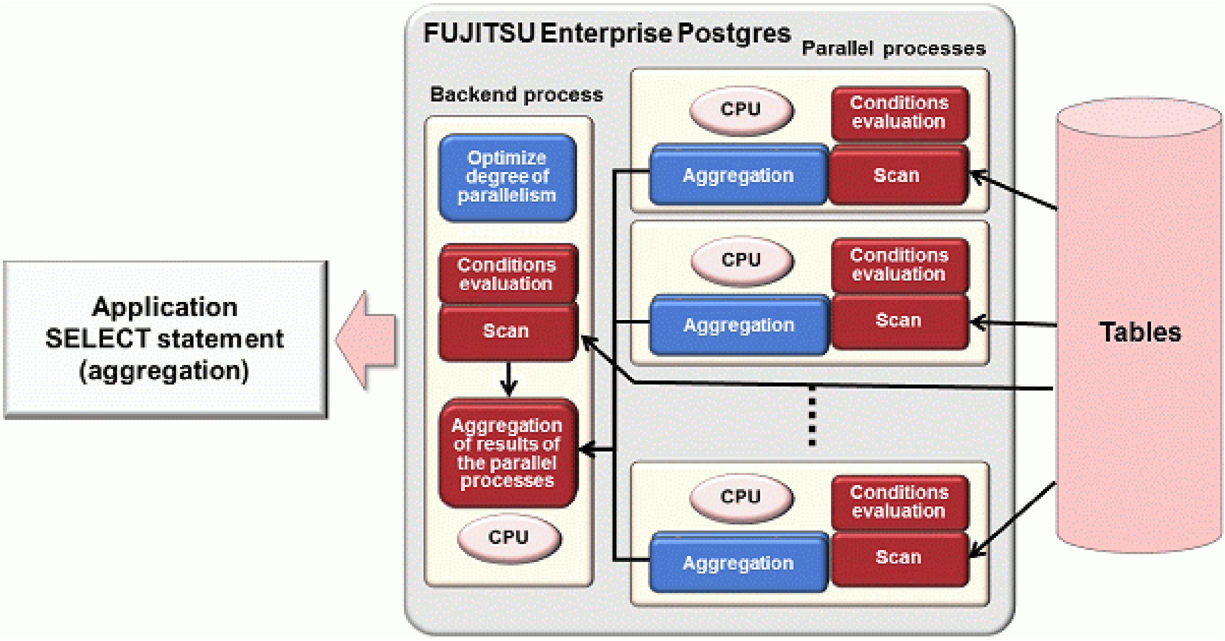
As shown in the figure below, processing time can be significantly reduced by taking full advantage of available server resources if online processes are executed separately from traditional SQL applications (grouped in periodic aggregation batch processing).



FUJITSU Enterprise Postgres parallel scan supports the following:

- Sequential scan
- Aggregation (aggregation functions, GROUP BY)

As shown in the figure below, processing is partitioned for each read range and allocated to parallelized background processes (called parallel processes) according to the CPU load. FUJITSU Enterprise Postgres uses the scan result of each parallel process and performs aggregation. Lastly, it aggregates all results and returns the result to the application.



It performs parallel processing using more processes if the CPU load is low, and redundant processes are not parallelized if the CPU load is high. This type of automatic determination enables more efficient use of CPUs and improved scan performance.

See

Refer to "Parallel Scan" in the Application Development Guide for details.

1.10 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.

Vertical Columnar Index (VCI)

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 Columnar 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).

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

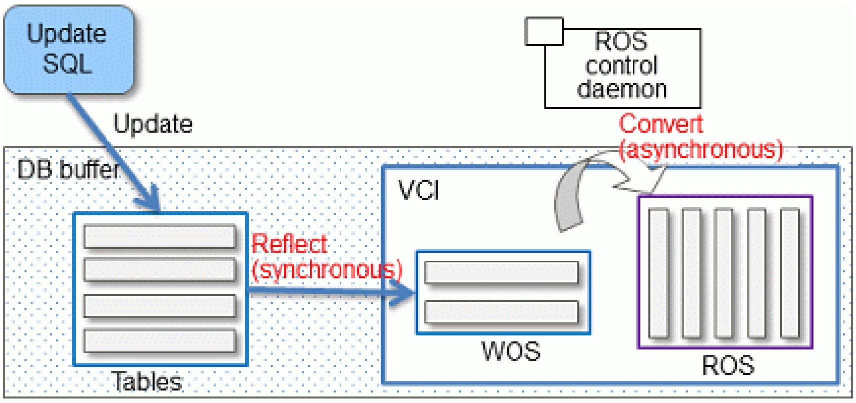
This feature has a data structure that can efficiently use the newly added resources, and aims to enhance the existing aggregation processing in normal operations to be faster than parallel scan. It shares the same purpose of enhancing aggregation performance with the parallel scan feature that is provided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

VCI architecture

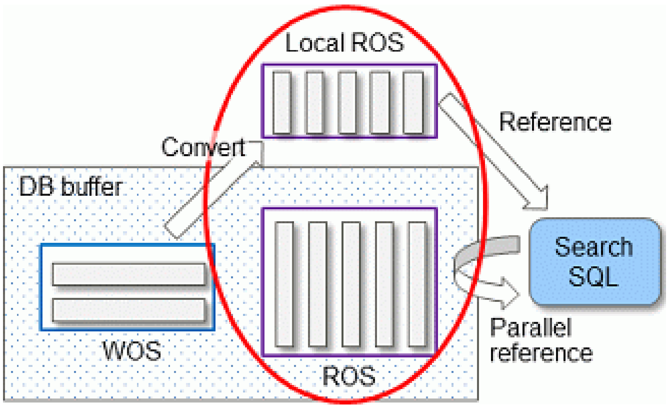
This section briefly explains VCI architecture as it contains basic terminology required, for example, when setting parameters.

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 Columnar Index (VCI)" in the Application Development Guide for information on scan using a VCI.

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
Fujitsu-developed software technology	WebAdmin (Rapid setup, One-click recovery)	Y	Y	Y	Y
Improved reliability	Database multiplexing	Y	N	Y	N
Application development	Embedded SQL integration Java integration ODBC integration .NET Framework integration	Y	Y	Y	Y
	Features compatible with Oracle databases	Y	Y	Y	Y
Security	Storage data encryption	Y	Y	Y	Y
	Data masking	Y	Y	Y	Y
Performance	Parallel scan	Y	N	Y	N
	In-memory feature	Y	N	Y	N
Performance tuning	Optimizer hints	Y	Y	Y	Y
	Fixed statistical information	Y	Y	Y	Y

Y: Provided

N: Not provided

Index