FUJITSU Enterprise Postgres

High-speed Data Load

FUJITSU Enterprise Postgres offers the ability to perform bulk data load using as many parallel processes as possible, according to the number of cores and CPU availability. This approach ensures that your data load process will make the best use of available resources, without the need to pre-configure and tune your environment beforehand.

About FUJITSU Enterprise Postgres

Founded on PostgreSQL, the world's most advanced open source relational database system, FUJITSU Enterprise Postgres extends base PostgreSQL functionality with a number of enhanced enterprise features. High-speed Data Load is one of those features.

Overview

In an era of mission-critical systems that are able to take advantage of multi-core technology, it is important that bulk data load tools do not use a fixed number of parallel processes, but instead optimally use as much available resources as possible.

FUJITSU Enterprise Postgres High-speed Data Load sends data from the input file to several parallel workers, each of which will simultaneously perform data conversion, table creation, and index creation, thus greatly reducing load time.

Bulk data load performance gain

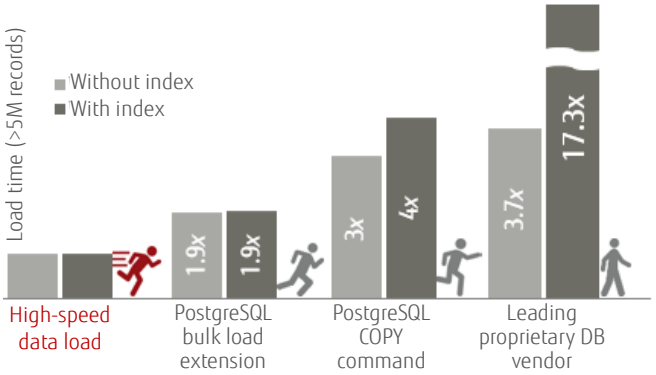
By definition, bulk data load tools are designed to offer improved performance over the execution of thousands or millions of INSERT statements.

This is achieved by employing a variety of techniques that efficiently read massive amounts of data and then convert the data and update the table and attached indexes, while maintaining data integrity and business continuity.

Faster bulk load amongst vendors

A variety of techniques and approaches can be used for bulk data load, which then result in different performance results.

When compared to other bulk data load solutions, FUJITSU Enterprise Postgres High-speed Data Load proved to be significantly faster both for load of indexed data and non-indexed data.



Flexible utilisation of idle resources

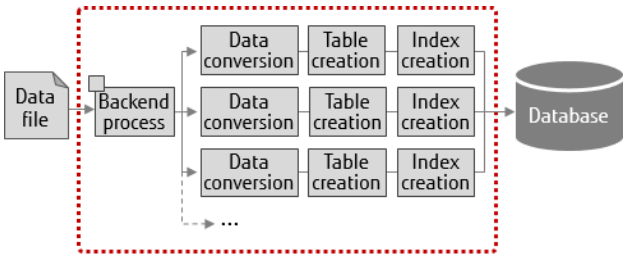
High-speed Data Load executes the PostgreSQL COPY command using multiple parallel workers - the degree of parallelism will be established by the tool itself at the time of execution, based on resource availability. Accordingly, data from files can be loaded at high speed into FUJITSU Enterprise Postgres tables.

shaping tomorrow with you

FUJITSU

Architecture

Each worker serially performs the tasks of data conversion, table creation, and index creation. This approach extends the performance gains provided by PostgreSQL, making it between 3 and 4 times faster than using the COPY command alone.



Dynamic creation of parallel workers according to resource availability

Additional performance gain

High-speed Data Load results in even greater performance gain if it is part of the same transaction as an earlier `CREATE TABLE` or `TRUNCATE` and the write-ahead log (WAL) parameter `wal_level` is set to `minimal`. This is the case because in this scenario no WAL records are generated.

Simple installation and use via extension

Install the extension

```
CREATE EXTENSION pgx_loader;
```

Update postgresql.conf

Change the parameters below by adding to their current value the number of instances where the High-speed Data Load may be executed multiplied by the maximum degree of parallelism.

- `max_parallel_workers` - Maximum number of background processes for parallel query.
- `max_prepared_transactions` - Maximum number of transactions that can be in prepared state for two-phase commit at the same time.
- `max_worker_processes` - Maximum number of background processes.

Example postgresql.conf

To configure the feature to run on 2 instances, with a degree of parallelism of 4, add 8 (2*4) to each of the parameters.

```
max_prepared_transactions = 13   # Initial value was 5
max_worker_processes = 16        # Initial value was 8
max_parallel_workers = 12        # Initial value was 4
```

Using High-speed Data Load

To load data from a file into a FUJITSU Enterprise Postgres table, execute the pgx_loader command in load mode, and then restart the instances.

The example below loads /path/sales.csv into table sales.

```
$ pgx_loader load
   -c "COPY sales FROM '/path/sales.csv' WITH CSV"
```

Checking the result

After the load is complete, you can check if all rows were successfully added.

```
SELECT gid FROM pg prepared xacts
WHERE gid IN
 (SELECT 'pgx_loader' || gid
  FROM  pgx loader.pgx loader state
  WHERE state = 'rollback'
    AND role oid IN (SELECT oid FROM pg roles
                     WHERE rolname = 'myrole')
    AND relation oid IN (SELECT relid
                         FROM pg_stat_all_tables
                         WHERE relname = 'sales');
```

In case there was a problem, run the command in recovery mode to resolve any in-doubt transaction.

```
$ pgx_loader recovery -t sales
```

Conclusion

When importing large amounts of data, it is especially important to choose the right approach, because any per-record overhead will be multiplied by thousands or millions. FUJITSU Enterprise Postgres High-speed Data Load leverages the PostgreSQL COPY command by running it in parallel using as many workers as your available resources allow.

About Fujitsu

Fujitsu is the 5th largest IT service provider in the world, offering a full range of technology products, solutions and services. Around 160,000 Fujitsu employees support customers in over 100 countries.

Contact
Fujitsu Australia Software Technology Pty Ltd
Address: 14 Rodborough Rd
Frenchs Forest NSW 2086
Australia
Email: postgresql@fast.au.fujitsu.com
Website: fast.fujitsu.com

Copyright 2020 FUJITSU AUSTRALIA SOFTWARE TECHNOLOGY. Fujitsu, the Fujitsu logo and Fujitsu brand names are trademarks or registered trademarks of Fujitsu Limited in Japan and other countries. Other company, product and service names may be trademarks or registered trademarks of their respective owners. All rights reserved. No part of this document may be reproduced, stored or transmitted in any form without prior written permission of Fujitsu Australia Software Technology. Fujitsu Australia Software Technology endeavours to ensure the information in this document is correct and fairly stated, but does not accept liability for any errors or omissions.

2020-02-03 WW EN