FUJITSU

Fujitsu
Enterprise Postgres 17

Userlog Operation Guide

Linux

J2UL-ULOG-01PEZ0(00)
December 2024

Preface

Purpose of this document

This book describes the userlog, an extension of PostgreSQL.

Reader of this book

This document is intended for those who use the userlog feature.

This manual assumes that you have a general knowledge of the following:

- PostgreSQL
- SQL
- Knowledge of Linux

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 1.0: December 2024
```


Copyright

Contents

Chapter 1 Userlog Feature Overview

This chapter describes the features of the userlog.

1.1 What is Userlog?

Arbitrary logging can be inserted into WAL (Write-Ahead Log), PostgreSQL's update log, using a dedicated function. The logs are automatically extracted from the archive file to an external file by a daemon process running on a background worker.

These functions and daemon processes come in the form of PostgreSQL extensions (EXTENSION).

Logs inserted within the same transaction are extracted as sequential log sequences. If there are multiple log sequences, they are extracted in order of transaction completion.

Completion of a transaction is either COMMIT or ROLLBACK. Also, if an instance crashes while a transaction is in progress, it is considered to have ROLLBACK at the time of standby promotion or crash recovery on the primary alone, and is subject to userlog extraction.

If the instance crashes, the userlogs of COMMIT transactions will not be lost, but some current userlogs of ROLLBACK transactions and running transactions may be lost. This is because PostgreSQL does not synchronize WAL writes except for COMMIT.

1.2 Daemon Behavior

The extraction daemon, whether primary or standby, starts when an instance is started by specifying userlog for shared_preload_libraries in postgresql.conf. The extraction control file must be initialized before starting. Refer to "Chapter 2 Setup" for detailed instructions.

The extraction daemon reads the user and transaction completion logs from the archive file location specified by userlog.archive_directory in postgresql.conf.

The read userlogs are organized for each transaction and stored in memory, and when the transaction completion log is read, all userlogs for that transaction are output to an extract file in the directory specified in userlog.directory in postgresql.conf.

1.3 Extract File

The name of the extracted file is the same as the WAL segment file. It consists of 24 hexadecimal digits,that number is always increasing.

For example, if there is a transaction's COMMIT log in a WAL segment file named 0000000100000000000001C, then all the transaction's userlog sequence will be output to an extract file named 0000000100000000000001C, even if the transaction inserted userlogs in an earlier segment file.

If no completion logs appear in a segment file for transactions that have inserted userlogs, an extract file with the same name as the segment file contains zero bytes.

This file has read permission only to the PostgreSQL user (the user who started the database cluster). This is because the contents of the userlog are equivalent to WAL segment files.

For details on the format of the extracted file, refer to "Appendix A Userlog File Format".

The most recent file that has been extracted can be seen from the latest_extracted_filename column of the pgx_stat_userlog view. For information about the pgx_stat_userlog view, refer to "Appendix B pgx_stat_userlog View"

Note

- Files newer than those listed in the latest_extracted_filename column of the pgx_stat_userlog view are still being created and should not be used.
- Use the information in the pgx_stat_userlog view of the instance that output the extract file you want to access to determine which extract file you have created. This is because information from views of other instances is staggered and can lead to incorrect decisions.

Once extracted, you can delete the extracted file at any time.

However, do not delete userlog_control and the file named userlog_control.tmp that is created during the creation of userlog_control to control extraction under the directory specified in userlog.directory.

This file contains the information needed to continue extracting across reboots. for example, which files were extracted up to, and where the oldest userlogs were emitted by the transaction that was running when the last extract was created (which archive file should be read over to create the next extract).

This directory also contains a file named pgx_userlog_control.log that contains a history of the extraction control file's init or set mode initialization commands. You can delete this file without any problems, but we recommend that you do not delete it because it may take longer to investigate if something goes wrong.

This file increases by about 50 bytes each time the initialization command is executed.

1.4 Relationship to Replication

Replication ensures that the extracted files are exactly the same between two directly connected instances.

There is a mechanism to accomplish this, which requires setting parameters. This is explained here.

The upstream replication instance should extract userlogs only to the extent that they were written to the downstream instance's WAL. Upstream and downstream refer to the case of a primary and its connected standby, and the relationship between two directly connected standbys in cascading replication. Without this constraint, the aforementioned identity cannot be guaranteed. For example, suppose you have two instances of synchronous replication that go down during transaction commit. With this configuration, there can be moments on disk where the primary is emitting a COMMIT log and the standby is not (Of course, from the application's point of view at this moment, the COMMIT has not been completed.). In this state, if the primary extraction daemon detects a COMMIT log and emits a userlog immediately, the userlog is in COMMIT state. However, if the primary goes down shortly afterwards and the standby is promoted, the promoted side will extract the userlog in ROLLBACK state because there will be no COMMIT log.

To avoid this, the downstream side extracts the userlog only within WAL records that have been written. To control this, the userlog.standby_names parameter should contain the names of instances that are at least downstream of the extraction daemon process. Refer to "Chapter 2 Setup" or a more detailed specification of this parameter.

1.5 Behavior when Extraction Fails

If an anomaly occurs that makes it impossible to extract userlogs, the instance is forcibly stopped.

There are two such cases. One case is that archiving by the archiver process fails. Write an archive_command in postgresql.conf to kill the instance if archiving fails.

The second is a failure of the extraction daemon to write the extraction file. If it fails, the extraction daemon shuts down the instance.

However, such abnormalities may be considered temporary. Therefore, the extraction daemon can specify the number of failures allowed. The default is never to allow. The behavior of the other archiver process is controlled by archive_command. To avoid interfering with the control of the archive_command, the extraction daemon logs a failure to read the archive file at level WARNING, but does not stop.

1.6 Features Not Supported

- Do not insert userlogs within local transactions enabled by XA. Inserts do not return an error, but do not extract userlogs for such transactions.
- Do not insert a userlog in a subtransaction that begins with a SAVEPOINT statement and ends with a RELEASE SAVEPOINT or ROLLBACK TO statement. If so, the insertion function returns SQLSTATE: 0A000 and the subtransaction is ABORT internally. Use the RELEASE SAVEPOINT or ROLLBACK TO statement to terminate the subtransaction.
- Do not insert a userlog inside a PL/pgSQL block using an EXCEPTION clause. This is because the start of such a block implicitly starts a subtransaction. If the userlog is inserted in a stored procedure that includes an EXCEPTION clause, SQLSTATE: 0A000 is returned.
- Do not insert userlogs in any PL other than PL/pgSQL. This is because such PLs invariably use subtransactions. If a userlog was inserted in a PL other than PL/pgSQL, SQLSTATE: 0A000 is returned.

Chapter 2 Setup

This chapter describes setting up userlogs.

2.1 Memory

The extraction daemon process maintains in memory all userlogs that have been emitted by all transactions running at a given time. Allow memory for this.

2.2 Primary Normal Shutdown

In a later step, you set the userlog extraction starting point to the last checkpoint, but if there are userlogs of running transactions after the last checkpoint, the correct extraction file cannot be generated. To avoid this, be sure to gracefully shut down the primary. This is because it guarantees that a checkpoint will be executed in the event of a normal shutdown and that there will be no more WAL in active transactions.

2.3 postgresql.conf Settings

Set the following PostgreSQL parameters to be able to extract the userlog:

These must be configured on all instances.

wal_level

Set logical or higher. However, it is not necessary to create a logical replication slot.

archive_mode

Set always. This is to extract userlogs even on the standby.

archive_command

Set up appropriate archiving commands according to the PostgreSQL documentation. However, the following two conditions must be met at this time.

- Run pg_ctl stop -m immediate if archiving (e.g., cp) fails.
- The destination file name must be "% f".

shared_preload_libraries

Add userlog. This allows the extraction daemon process to run. When adding userlog, the control file for userlog extraction must be initialized using the pgx_userlog_control command before the next instance restart. This procedure is described in the next step.

max_connections

The extraction daemon makes an internal connection to PostgreSQL using the Server Programming Interface (SPI) to obtain various internal aspects of PostgreSQL. This consumes one entry managed by max_connections. Add 1 to the estimate for this parameter.

Set the following parameters for userlog extraction:

Of these parameters, only userlog.standby_names can be affected by reload. All others require an instance restart.

The type of each parameter is shown in parentheses. The semantics of this type are the same as for PostgreSQL.

userlog.directory(string)

Specify the output directory of the extracted file.

Specify a directory different from userlog.archive_directory. If you specify the same directory, an error occurs.

userlog.archive_directory(string)

Specify the archive file output directory.

userlog.standby_names(string)

Specify the name of an instance that can be directly downstream of replication from an instance.

The name of the instance is the name specified in application_name in the replication connection string specified in primary_conninfo for each instance. All instances must have different names.

This parameter is referenced when the instance acts as the upstream side of replication. The upstream instance will extract up to the oldest write-synchronized log position reported by all downstream standbys. The effect of this is described in "1.4 Relationship to Replication".

If the replication connection to the downstream standby is broken, the old log location is remembered until it is reconnected and the new log location is advertised. That is, extraction will not proceed until the downstream standby is restarted and reconnected for replication, or until the standby is removed from userlog.standby_names and reloaded. This is useful when you want to temporarily stop the standby.

It excludes itself and any downstream standby that has never had a replication connection to the upstream side. So, although the parameter name says standby, you can also list yourself, all instances that might connect in the future, or names that are completely unrelated.

PostgreSQL allows both physical and logical replication connections from user applications. The application_name used by these applications must not be the same as the application_name of the replication connection between the servers. Also note the environment variables APPNAME and fallback_application_name. PostgreSQL will allow connections with the same name, but the extraction daemon cannot distinguish between them.

If an erroneous connection with the same name is made, wait until WAL writes are completed on all these replication destinations to avoid missing userlogs. However, be aware that this may be unexpected behavior due to the overall design of the system.

userlog.dbname(string)

The extraction daemon establishes an SQL connection via the Server Program Interface (SPI) to find out where WAL was written in the standby. Because you are connecting as a superuser on the instance, no credentials are required, but you must specify an existing database as the destination database. The default is "postgres". This database is created by default when an instance is created, but if you intentionally drop it, specify the name of another existing database in this parameter.

There are other parameters that control userlog extraction. However, these settings are not required.

userlog.limit_size_per_xact(integer)

This parameter can limit the total size of the userlog that can be emitted by a single transaction.

If you attempt to insert a userlog and the total size, including the userlog, exceeds the value specified for this parameter, you should explicitly execute a ROLLBACK statement to end the transaction, since the insertion function returns SQLSTATE: 54000 and raises an error, causing the transaction to ABORT internally and prevent further SQL execution. 54000 is the SQLSTATE we are defining as program_limit_exceeded. Specifying 0 for this parameter does not limit the total size. The default is 0. The unit (kB, MB, etc.) can be appended like any other PostgreSQL parameter.

For example, if you supply text data as an argument to pgx_emit_userlog(), it is limited by the length after conversion to the database character encoding.

userlog.extract_interval(integer)

The interval at which the extraction daemon monitors the archive directory. Detects that this monitoring has generated the following archive files and creates the following extracted files: The unit is seconds. The default is 1. The retry interval for the extracted file by userlog.write_retry_num also uses the value of this parameter.

userlog.write_retry_num(integer)

The number of times the extraction daemon can fail to create an extract file. The extraction daemon shuts itself down if it fails more than the specified number of times. A special value of -1 allows unlimited. A value of 0 simply means that no single failure is allowed. The default is 0.

2.4 Initializing the Extraction Control File

To extract userlogs from the same location on all instances, specify the extraction start point as follows:

1. Run the pgx_userlog_control command in init mode on the primary to initialize the userlog extraction control file. This will allow extraction to start from the userlog output on the next boot.
2. On all standbys, run the pgx_userlog_control command in set mode with the -l option to initialize the userlog extraction control file. In this case, for the -l option, specify the log location that was output to the standard output by the pgx_userlog_control command in step 1. This ensures that the primary and the extraction start point match.

For information about the pgx_userlog_control command, refer to "Appendix C pgx_userlog_control Command".

2.5 Starting Instance

Start all instances. Extract files start to be generated immediately after startup.

2.6 Introducing Userlog Extensions

Execute the CREATE EXTENSION statement on the primary with userlog specified. Replication affects all instances. This makes the userlog insertion function available.

Chapter 3 Operation

This chapter describes the operation of userlogs.

3.1 Deleting Archive Files

Many operations take an archive file for recovery and discard the previous archive file once it has been backed up.

However, if an archive file containing userlogs previously inserted by a transaction that is running at some point is destroyed, the complete extract file cannot be created.

To avoid this, see the required_filename column of the pgx_stat_userlog view and remove any earlier archive files.

Note

- Do not use the pgx_dumpall command. This command automatically deletes archived files prior to the backup after the backup.
- When deleting an instance's archive file, do not use the file name from the pgx_stat_userlog view of the other instance. There is a slight time difference and the archive file may still be needed.

3.2 When to Stopping an Instance

Even if you stop the instance normally, the latest userlogs are not necessarily extracted. For example, userlogs contained in the current WAL segment file that cannot be archived, or in the most recent archive file generated during a graceful shutdown. These userlogs will be extracted the next time you start them, but if you want to extract them when you stop them, do the following:

1. Complete all transactions that print userlogs and then do not start them.
2. Run the pg_switch_wal() function to switch the current WAL segment file. You can find the current segment file name before switching:

```
select pg_walfile_name(pg_switch_wal());
```

3. Wait for the extraction file with the same name as above to be completed. The waiting time takes the value of the userlog.extract_interval parameter plus a few seconds (time required for archiving and generating the extracted file).

3.3 When to Incorporate Standby

To incorporate a standby database, before starting the instance, initialize the extraction start position by executing the pgx_userlog_control command in set mode with the --follow-primary option specified.

Do not initialize with the -l option. This is because the -l option assumes that no transaction is currently running. --follow-primary has the effect of instructing the extraction daemon to wait for all such transactions to end before starting the extraction.

For information about the pgx_userlog_control command, refer to "Appendix C pgx_userlog_control Command".

3.4 When to Use Point In Time Recovery

Before starting the instance for recovery, remove userlog from shared_preload_libraries to prevent extraction of userlogs during recovery. Retry the setup steps before the recovery is complete and you are ready for operation. However, you do not need to "deploy userlog extensions".

For information about setup, refer to "Chapter 2 Setup".

3.5 Adding Userlog to shared_preload_libraries

If you have previously set up an instance, but you have deleted the userlog from shared_preload_libraries and started the instance at least once, you must repeat the setup steps. However, you do not need to “deploy userlog extensions ”.

This is because it will malfunction if there is any remaining information from a previous userlog extraction.

For information about setup, refer to "Chapter 2 Setup".

3.6 Notes

After you incorporate in a standby, it may take several tens of seconds before the first extraction file begins to be generated on the plugged in standby. Extract files generated on the primary side during this time are not generated on the standby side. In other words, redundancy of userlog extraction files will not start for a while after you make an instance redundant.

On the standby side, extraction begins after all transactions, such as those in progress at the time of inclusion, have completed. Note that if there are very long transactions, the non-redundant section of such userlogs will be lengthened accordingly.

The reason for this behavior is that making the right extract file requires WAL from the oldest to the latest of all userlogs inserted by all transactions that were running at the time the installation was complete, which is impractical. You have to go back through the archive file to get it, and you don't know how far back you have to go.

Chapter 4 Unsetup

This chapter describes unsetting up userlogs.

4.1 Check the Userlogs

Check that all required userlogs have been extracted. Once unset up, you cannot extract previous unextracted userlogs. To verify the extraction, refer to the procedure "3.2 When to Stopping an Instance". You can stop the instance after you remove the following userlog extensions:

4.2 Deleting Extensions

On the primary, execute DROP EXTENSION with userlog specified.

4.3 Stop Extract

Remove userlog from shared_preload_libraries and stop the instance.

The next time you reboot, the extraction daemon will not start. Note that you can leave parameters beginning with "userlog." in postgresql.conf and the instance will start correctly.

Chapter 5 How to Insert a Userlog (pgx_emit_userlog Function)

The following functions can be used to insert a userlog into WAL:

pgx_emit_userlog (content text) Return types: pg_lsn

pgx_emit_userlog (content bytea) Return types: pg_lsn

Arguments

Specify the userlog you want to insert in content. Except for the userlog.limit_size_per_xact parameter, which limits the total size of the data in a transaction, there is no special limit on the length of the data you specify.

Note that if text data is supplied as an argument to pgx_emit_userlog(), it is limited by the length after conversion to the database character encoding.

Privilege

Like the pg_logical_emit_message function, which also inserts a log into WAL, this function can be run by any user once introduced. Use the GRANT/REVOKE statement to adjust the privileges to suit your policy.

Schema

The userlog extension can relocate the schema in which it is registered.

Examples

select pgx_emit_userlog('some userlog')

Return value

Returns the WAL log location where the specified userlog was inserted.

Error

In addition to errors returned by PostgreSQL's pg_logical_emit_message function, this error is returned:

- If one of these functions is used within a subtransaction, it returns SQLSTATE: 0A000 and the transaction is ROLLBACK.
- function causes the total size of userlog inserts in the transaction to exceed userlog.limit_size_per_xact, return SQLSTATE: 54000 and internally ABORT the transaction, preventing further SQL execution, and explicitly terminate the transaction with a ROLLBACK statement.

Chapter 6 Messages

This chapter explains the messages that are output.

6.1 Message Numbers Beginning with 40000

6.1.1 40000

could not open file "@1@" for reading: @2@

[Description]

Open file for reading failed.

[System Processing]

Processing will be aborted.

[Action]

To investigate the cause of the occurrence from the message, and remove cause.

6.1.2 40001

could not read file "@1@": @2@

[Description]

Failure in reading the file.

[System Processing]

In case of error, the processing will be aborted. In case of warning, continue processing.

[Action]

To investigate the cause of the occurrence from the message, and remove cause.

6.1.3 40002

could not read file "@1@": read @2@ of @3@

[Description]

Could not read file read some bytes instead of actual bytes.

[System Processing]

In case of error, the processing will be aborted. In case of warning, continue processing.

[Action]

To investigate the cause of the occurrence from the message, and remove cause.

6.1.4 40003

incorrect checksum in userlog_control file

[Description]

The calculated checksum is different from the checksum that was saved in the userlog_control file.

[System Processing]

Processing will be aborted.

[Action]

To investigate the cause of the occurrence from the message, and remove cause.

6.1.5 40004

userlog_control was initialized by PostgreSQL version @1@, which is not compatible with this version @2@

[Description]

Incompatible server version.

[System Processing]

Processing will be aborted.

[Action]

To investigate the cause of the occurrence from the message, and remove cause.

6.1.6 40005

startup failed because userlog.directory is not set

[Description]

No userlog directory is configured.

[System Processing]

Processing will be aborted.

[Action]

To investigate the cause of the occurrence from the message, and remove cause.

6.1.7 40006

userlog directory "@1@" is too long

[Description]

userlog directory path is too long.

[System Processing]

Processing will be aborted.

[Action]

To investigate the cause of the occurrence from the message, and remove cause.

6.1.8 40007

startup failed because userlog.archive_directory is not set

[Description]

No archive directory is configured.

[System Processing]

Processing will be aborted.

[Action]

To investigate the cause of the occurrence from the message, and remove cause.

6.1.9 40008

archive directory "@1@" is too long

[Description]

archive directory path is too long.

[System Processing]

Processing will be aborted.

[Action]

To investigate the cause of the occurrence from the message, and remove cause.

6.1.10 40009

could not access directory "@1@": @2@

[Description]

could not access directory.

[System Processing]

Processing will be aborted.

[Action]

To investigate the cause of the occurrence from the message, and remove cause.

6.1.11 40010

userlog.directory and userlog.archive_directory must be different

[Description]

userlog.directory and userlog.archive_directory must be different.

[System Processing]

Processing will be aborted.

[Action]

To investigate the cause of the occurrence from the message, and remove cause.

6.1.12 40011

could not open directory "@1@": @2@

[Description]

Could not open directory.

[System Processing]

Processing will be aborted.

[Action]

To investigate the cause of the occurrence from the message, and remove cause.

6.1.13 40012

invalid list syntax in parameter "@1@"

[Description]

Invalid descriptions were found in userlog.standby_names.

[System Processing]

Processing will be aborted.

[Action]

To investigate the cause of the occurrence from the message, and remove cause.

6.1.14 40013

userlog extension can use only "logical" for wal_level

[Description]

The GUC parameter wal_level must be logical.

[System Processing]

Processing will be aborted.

[Action]

To investigate the cause of the occurrence from the message, and remove cause.

6.1.15 40014

userlog extension requires for archive_mode to be "on" or "always"

[Description]

The GUC parameter archive_mode must be on or always.

[System Processing]

Processing will be aborted.

[Action]

To investigate the cause of the occurrence from the message, and remove cause.

6.1.16 40015

no userlog is extracted if starting the PostgreSQL server as a standby and archive_mode is not "always"

[Description]

No userlog is extracted if starting the PostgreSQL server as a standby and archive_mode is not always.

[System Processing]

Continues processing.

[Action]

Check the message text and confirm that the issue does not affect the expected outcome.

6.1.17 40016

no userlog is extracted because archive_command is not set

[Description]

No userlog is extracted because archive_command is not set.

[System Processing]

Continues processing.

[Action]

Check the message text and confirm that the issue does not affect the expected outcome.

6.1.18 40017

@1@ (PID @2@) exited with exit code @3@

[Description]

Process terminated.

[System Processing]

Continues processing.

[Action]

If the instance and userlog daemon are not stopped together, investigate the cause of the stop and remove the cause.

6.1.19 40018

terminating postmaster process due to userlog daemon

[Description]

userlog daemon process terminated abnormally.

[System Processing]

Continues processing.

[Action]

Refer to this message together with the message that was output immediately beforehand.

6.1.20 40019

could not start as primary server because the userlog_control file was created with the follow-primary option

[Description]

Could not start as primary server because the userlog_control file was created with the follow-primary option.

[System Processing]

Processing will be aborted.

[Action]

To investigate the cause of the occurrence from the message, and remove cause.

6.1.21 40020

failed to update userlog_control

[Description]

Failed to update userlog_control file.

[System Processing]

Processing will be aborted.

[Action]

Refer to this message together with the message that was output immediately beforehand.

6.1.22 40021

out of memory

[Description]

Process runs out of memory.

[System Processing]

Processing will be aborted.

[Action]

To investigate the cause of the occurrence from the message, and remove cause.

6.1.23 40022

new extraction target timeline is @1@

[Description]

The extraction target timelineID has been changed.

[System Processing]

Continues processing.

[Action]

No action required.

6.1.24 40023

could not write extracted file

[Description]

Could not write extracted file.

[System Processing]

Processing will be aborted.

[Action]

Refer to this message together with the message that was output immediately beforehand.

6.1.25 40024

archive file may be corrupted

[Description]

Archive file may be corrupted.

[System Processing]

Processing will be aborted.

[Action]

Refer to this message together with the message that was output immediately beforehand.

6.1.26 40025

could not seek in log file @1@ to offset @2@: @3@

[Description]

Could not seek in log file to offset and reason for failure also printed along with error.

[System Processing]

Continues processing.

[Action]

To investigate the cause of the occurrence from the message, and remove cause for the WARNING.

6.1.27 40026

could not fsync file "@1@": @2@

[Description]

Could not fsync file.

[System Processing]

Processing will be aborted.

[Action]

To investigate the cause of the occurrence from the message, and remove cause.

6.1.28 40027

could not close file "@1@": @2@

[Description]

Could not close file.

[System Processing]

Processing will be aborted.

[Action]

To investigate the cause of the occurrence from the message, and remove cause.

6.1.29 40028

could not create file "@1@": @2@

[Description]

Could not create file.

[System Processing]

Continues processing.

[Action]

To investigate the cause of the occurrence from the message, and remove cause for the WARNING.

6.1.30 40029

could not prepare SQL statement "@1@"

[Description]

An error occurred.

[System Processing]

Processing will be aborted.

[Action]

To investigate the cause of the occurrence from the message, and remove cause.

6.1.31 40030

failed to get flush_lsn of downstream standby server

[Description]

An error occurred.

[System Processing]

Processing will be aborted.

[Action]

To investigate the cause of the occurrence from the message, and remove cause.

6.1.32 40031

could not write file "@1@": @2@

[Description]

Failure in writing to a file.

[System Processing]

Continues processing.

[Action]

To investigate the cause of the occurrence from the message, and remove cause for the WARNING.

6.1.33 40032

pgx_emit_userlog() cannot be called in a subtransction

[Description]

pgx_emit_userlog() cannot be called in a subtransction.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.34 40033

sum of userlog exceeds userlog.limit_size_per_xact

[Description]

userlog size limit per transaction exceeded.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.35 40034

set-valued function called in context that cannot accept a set

[Description]

An error occurred during execution of the application or command.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.36 40035

materialize mode required, but it is not allowed in this context

[Description]

An error occurred during execution of the application or command.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.37 40036

return type must be a row type

[Description]

An error occurred during execution of the application or command.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.38 40037

could not open lock file "@1@" for reading: @2@

[Description]

Could not open lock file.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.39 40038

could not read lock file "@1@": @2@

[Description]

Could not read lock file.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.40 40039

lock file "@1@" is empty

[Description]

lock file is empty. Check if the lockfile is corrupt.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.41 40040

bogus data in lock file "@1@": "@2@"

[Description]

Check if the lockfile is corrupt.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.42 40041

command cannot be executed because there is the PostgreSQL server process (PID @1@) running in the data directory "@2@"

[Description]

There is a PostgreSQL server process running in the data directory.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.43 40042

out of memory

[Description]

Out of memory.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.44 40043

could not open file "@1@" for writing: @2@

[Description]

Could not open file for writing.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.45 40044

could not write file "@1@": @2@

[Description]

Could not write file.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.46 40045

clusters are not compatible with this version of pgx_userlog_control

[Description]

Clusters are not compatible with this version of pgx_userlog_control.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.47 40046

could not create file "@1@": @2@

[Description]

Could not create file.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.48 40047

could not fsync file "@1@": @2@

[Description]

Could not fsync file.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.49 40048

could not close file "@1@": @2@

[Description]

Could not close file.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.50 40049

could not rename file "@1@": @2@

[Description]

Could not rename file.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.51 40050

could not open file "@1@" for reading: @2@

[Description]

Could not open file for reading.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.52 40051

invalid WAL segment size

[Description]

WAL segment size is not valid.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.53 40052

argument of --wal-segsize must be a number

[Description]

The input parameter WAL segment size is wrong.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.54 40053

could not read file "@1@": @2@

[Description]

Failure in reading a file.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.55 40054

could not read file "@1@": read @2@ of @3@

[Description]

Could not read file read some bytes instead of actual bytes.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.56 40055

incorrect checksum in file "@1@"

[Description]

The calculated checksum is different from the checksum that was saved in the file.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.57 40056

userlog_control is not compatible with this version of pgx_userlog_control

[Description]

Incompatible server version.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.58 40057

could not parse start WAL location "@1@"

[Description]

An error occurred during execution of the application or command.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.59 40058

userlog directory "@1@" is too long

[Description]

userlog directory path is too long.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.60 40059

too many command-line arguments (first is "@1@")

[Description]

Too many command-line arguments.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.61 40060

unrecognized operation mode "@1@"

[Description]

Unrecognized operation mode was specified.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.62 40061

no operation specified

[Description]

No operation mode was specified.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.63 40062

at least one -l option or --follow-primary option must be specified

[Description]

Option specification is incorrect.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.64 40063

cannot specify both -l option and --follow-primary option

[Description]

Option specification is incorrect.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.65 40064

-l option can be used only in set operation mode

[Description]

Option specification is incorrect.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.66 40065

--follow-primary option can be used only in set operation mode

[Description]

Option specification is incorrect.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.67 40066

--wal-segsize option can be used only in show operation mode

[Description]

Option specification is incorrect.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.68 40067

no userlog directory specified

[Description]

No userlog directory specified.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.69 40068

no database directory specified and environment variable PGDATA unset

[Description]

No data directory specified.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.70 40069

calculated CRC checksum does not match value stored in pg_control

[Description]

An error occurred reading the pg_control file.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.71 40070

user name lookup failure: error code @1@

[Description]

An error occurred during execution of the application or command.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.72 40071

could not look up local user ID @1@: @2@

[Description]

An error occurred during execution of the application or command.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

6.1.73 40072

local user with ID @1@ does not exist

[Description]

An error occurred during execution of the application or command.

[System Processing]

Processing will be aborted.

[Action]

Check the message text and confirm that the application is written correctly and the command is being used correctly.

Appendix A Userlog File Format

The extracted file has no file header, only userlog records. Each userlog record has a record header, followed by the userlog.

Userlog record headers are placed on 8-byte boundaries.

The header of a userlog record has the following format: Note that the endianness of each numeric field follows the endianness of the machine on which PostgreSQL is running.

A.1 Userlog Header Format

The format of the userlog record header is as follows:

Field name	Description	Remarks
total_len	8-byte unsigned integer	
lsn	8-byte unsigned integer	
tli	4-byte unsigned integer	
xact_stat	single-byte character	
is_last	1 byte boolean	
content_type	single-byte character	
reserved	1 byte	
txid	8-byte unsigned integer	
content_len	8-byte unsigned integer	
content	variable length	Userlog

A.2 Field Meaning

total_len

The total size of the log record.

Size includes this field. In addition, the size includes padding up to the 8-byte boundary placed after the userlog record.

That is, the next log record header is at a total_len offset from the beginning of the log record header.

lsn

Log sequence number.

A sequence number that will be appended to all WAL records, not just userlogs. The following tli and sets can uniquely identify the log.

tli

The timeline ID.

See

For more information about timeline IDs, see “Continuous Archiving and Point-In-Time Recovery (PITR) ” in the“ PostgreSQL Documentation ”. Although not documented in this manual, the timeline also changes when a switchover occurs in a hot standby configuration.

xact_stat

Transaction state.

"c" (for commit) or "a" (for abort).

is_last

"true" (integer 1) if this is the last log in the same transaction. Otherwise, it is "false" (integer 0).

content_type

The data type of the userlog.

"t" for text type and "b" for bytea type.

reserved

Reserved area.

txid

The transaction ID.

content_len

The length of content specified to the pgx_emit_userlog function.

content

The content specified to the pgx_emit_userlog function.

Even type text does not include the terminator of the string.

The character encoding is the database encoding. The database encoding is specified, for example, by the -E option of initdb, or by the WITH ENCODING clause of the CREATE DATABASE statement.

Appendix B pgx_stat_userlog View

Describes the pgx_stat_userlog view.

Column	Type	Description
required_filename	text	The name of the segment file that contains the oldest userlog output by transactions that the extraction daemon knows are running. If the archive files after this segment are lost, the userlogs cannot be extracted correctly.
latest_extracted_filename	text	The name of the most recent extraction file that was completed. Newer extract files are still being created and should not be used.

Appendix C pgx_userlog_control Command

Name

pgx_userlog_control - View or initialize userlog_control file contents

Summary

```
pgx_userlog_control init -p userlog_directory [-D data_directory]

pgx_userlog_control set -p userlog_directory {-l lsn | --follow-primary} [-D data_directory]

pgx_userlog_control show -p userlog_directory [-D data_directory]
```

Description

When run in init mode, the userlog extraction start point is initialized to the last checkpoint point, and this point is output to standard output. The output format is a non-whitespace character concatenation of the timeline ID and the string representation of the log location, which is of type pg_lsn. Run this command only on a successful primary. The command does not result in an error if this condition is violated.

When run in set mode, the -l option sets the userlog extraction location to the specified location, and the --follow-primary option sets the extraction to start when the log location is reached, where the correct extraction file can be output. Run this command only on a stopped standby. If this condition is not met, the command does not result in an error.

In show mode, the contents of the userlog extraction control file are displayed. You may not use this mode unless you are investigating something.

Options

-p,--path=userlog_directory

Specify the directory specified by the userlog.directory parameter in postgresql.conf.

-l,--lsn=*lsn*

Specify the character string that was output to the standard output when pgx_userlog_control was executed in init mode on the primary. The string format is a non-whitespace character concatenation of the timeline ID and its string representation of type pg_lsn.

-D, --pgdata=data_directory

Specify the instance data directory. If this parameter is omitted, the directory specified by the PGDATA environment variable is referenced. You cannot omit both this option and the environment variable PGDATA.

Diagnosis

If there is an error, a nonzero value is returned. Returns 0 on success.

Presently, 1 is returned if the user_control or pg_control file does not exist, 2 if it exists but the contents are invalid, and 3 for any other error, but this might change in the future, so only use zero or non-zero information.

Annotation

In the show mode display, new fields may be added and the order in which they appear may change.

Privilege

Run the pgx_userlog_control command as the user who created the PostgreSQL instance. Otherwise, the command will succeed if you have privilege to move to the directory specified by the -p option and read and write to the userlog_control file in that directory.

pgx_userlog_control show display contents

The field name is followed by a colon (:), as in the following example:

The information displayed may be added or changed.

```
LatestExtractedFilename:000000010000000000000001C   ... (1)
OldestUserlogLSN:0/16EF3A8                          ... (2)
```

(1) LatestExtractedFilename: The name of the extracted file that was last extracted.
(2) OldestUserlogLSN : Oldest WAL log location currently needed