NAME

pg_recvlogical - control PostgreSQL logical decoding streams

SYNOPSIS

pg_recvlogical [*option*...]

DESCRIPTION

pg_recvlogical controls logical decoding replication slots and streams data from such replication slots.

It creates a replication-mode connection, so it is subject to the same constraints as **pg_receivewal**(1), plus those for logical replication (see Chapter 49).

pg_recvlogical has no equivalent to the logical decoding SQL interface's peek and get modes. It sends replay confirmations for data lazily as it receives it and on clean exit. To examine pending data on a slot without consuming it, use **pg_logical_slot_peek_changes**.

OPTIONS

At least one of the following options must be specified to select an action:

--create-slot

Create a new logical replication slot with the name specified by **--slot**, using the output plugin specified by **--plugin**, for the database specified by **--dbname**.

The **--two-phase** can be specified with **--create-slot** to enable decoding of prepared transactions.

--drop-slot

Drop the replication slot with the name specified by **--slot**, then exit.

--start

Begin streaming changes from the logical replication slot specified by **--slot**, continuing until terminated by a signal. If the server side change stream ends with a server shutdown or disconnect, retry in a loop unless **--no-loop** is specified.

The stream format is determined by the output plugin specified when the slot was created.

The connection must be to the same database used to create the slot.

--create-slot and **--start** can be specified together. **--drop-slot** cannot be combined with another action.

The following command-line options control the location and format of the output and other replication

behavior:

-E *lsn*
--endpos=*lsn*
In **--start** mode, automatically stop replication and exit with normal exit status 0 when receiving reaches the specified LSN. If specified when not in **--start** mode, an error is raised.

If there's a record with LSN exactly equal to *lsn*, the record will be output.

The **--endpos** option is not aware of transaction boundaries and may truncate output partway through a transaction. Any partially output transaction will not be consumed and will be replayed again when the slot is next read from. Individual messages are never truncated.

-f *filename*
--file=*filename*
Write received and decoded transaction data into this file. Use - for stdout.

-F *interval_seconds*
--fsync-interval=*interval_seconds*
Specifies how often pg_recvlogical should issue **fsync()** calls to ensure the output file is safely flushed to disk.

The server will occasionally request the client to perform a flush and report the flush position to the server. This setting is in addition to that, to perform flushes more frequently.

Specifying an interval of 0 disables issuing **fsync()** calls altogether, while still reporting progress to the server. In this case, data could be lost in the event of a crash.

-I *lsn*
--startpos=*lsn*
In **--start** mode, start replication from the given LSN. For details on the effect of this, see the documentation in Chapter 49 and Section 55.4. Ignored in other modes.

--if-not-exists
Do not error out when **--create-slot** is specified and a slot with the specified name already exists.

-n
--no-loop
When the connection to the server is lost, do not retry in a loop, just exit.

-o *name***[=***value***]**
--option=*name***[=***value***]**
Pass the option *name* to the output plugin with, if specified, the option value *value*. Which options exist and their effects depends on the used output plugin.

-P *plugin*
--plugin=*plugin*
When creating a slot, use the specified logical decoding output plugin. See Chapter 49. This option has no effect if the slot already exists.

-s *interval_seconds*
--status-interval=*interval_seconds*
This option has the same effect as the option of the same name in **pg_receivewal**(1). See the description there.

-S *slot_name*
--slot=*slot_name*
In **--start** mode, use the existing logical replication slot named *slot_name*. In **--create-slot** mode, create the slot with this name. In **--drop-slot** mode, delete the slot with this name.

-t
--two-phase
Enables decoding of prepared transactions. This option may only be specified with **--create-slot**

-v
--verbose
Enables verbose mode.

The following command-line options control the database connection parameters.

-d *dbname*
--dbname=*dbname*
The database to connect to. See the description of the actions for what this means in detail. The *dbname* can be a connection string. If so, connection string parameters will override any conflicting command line options. Defaults to the user name.

-h *hostname-or-ip*
--host=*hostname-or-ip*
Specifies the host name of the machine on which the server is running. If the value begins with a slash, it is used as the directory for the Unix domain socket. The default is taken from the

PGHOST environment variable, if set, else a Unix domain socket connection is attempted.

-p *port*
--port=*port*
Specifies the TCP port or local Unix domain socket file extension on which the server is listening for connections. Defaults to the **PGPORT** environment variable, if set, or a compiled-in default.

-U *user*
--username=*user*
User name to connect as. Defaults to current operating system user name.

-w
--no-password
Never issue a password prompt. If the server requires password authentication and a password is not available by other means such as a .pgpass file, the connection attempt will fail. This option can be useful in batch jobs and scripts where no user is present to enter a password.

-W
--password
Force pg_recvlogical to prompt for a password before connecting to a database.

This option is never essential, since pg_recvlogical will automatically prompt for a password if the server demands password authentication. However, pg_recvlogical will waste a connection attempt finding out that the server wants a password. In some cases it is worth typing **-W** to avoid the extra connection attempt.

The following additional options are available:

-V
--version
Print the pg_recvlogical version and exit.

-?
--help
Show help about pg_recvlogical command line arguments, and exit.

ENVIRONMENT

This utility, like most other PostgreSQL utilities, uses the environment variables supported by libpq (see Section 34.15).

The environment variable **PG_COLOR** specifies whether to use color in diagnostic messages. Possible values are always, auto and never.

NOTES

pg_recvlogical will preserve group permissions on the received WAL files if group permissions are enabled on the source cluster.

EXAMPLES

See Section 49.1 for an example.

SEE ALSO

pg_receivewal(1)