NAME

vacuumdb - garbage-collect and analyze a PostgreSQL database

SYNOPSIS

vacuumdb [*connection-option*...] [*option*...] [**-t** | **--table** *table* [(*column* [,...])]]... [*dbname*]

vacuumdb [*connection-option*...] [*option*...] **-a** | **--all**

DESCRIPTION

vacuumdb is a utility for cleaning a PostgreSQL database. vacuumdb will also generate internal statistics used by the PostgreSQL query optimizer.

vacuumdb is a wrapper around the SQL command **VACUUM**. There is no effective difference between vacuuming and analyzing databases via this utility and via other methods for accessing the server.

OPTIONS

vacuumdb accepts the following command-line arguments:

-a
--all

Vacuum all databases.

[-d] *dbname*
[--dbname=]*dbname*

Specifies the name of the database to be cleaned or analyzed, when **-a**/**--all** is not used. If this is not specified, the database name is read from the environment variable **PGDATABASE**. If that is not set, the user name specified for the connection is used. The *dbname* can be a connection string. If so, connection string parameters will override any conflicting command line options.

--disable-page-skipping

Disable skipping pages based on the contents of the visibility map.

> **Note**
> This option is only available for servers running PostgreSQL 9.6 and later.

-e
--echo

Echo the commands that vacuumdb generates and sends to the server.

-f

--full

Perform "full" vacuuming.

-F
--freeze

Aggressively "freeze" tuples.

--force-index-cleanup

Always remove index entries pointing to dead tuples.

> **Note**
> This option is only available for servers running PostgreSQL 12 and later.

-j *njobs*
--jobs=*njobs*

Execute the vacuum or analyze commands in parallel by running *njobs* commands simultaneously. This option may reduce the processing time but it also increases the load on the database server.

vacuumdb will open *njobs* connections to the database, so make sure your max_connections setting is high enough to accommodate all connections.

Note that using this mode together with the **-f** (FULL) option might cause deadlock failures if certain system catalogs are processed in parallel.

--min-mxid-age *mxid_age*

Only execute the vacuum or analyze commands on tables with a multixact ID age of at least *mxid_age*. This setting is useful for prioritizing tables to process to prevent multixact ID wraparound (see Section 25.1.5.1).

For the purposes of this option, the multixact ID age of a relation is the greatest of the ages of the main relation and its associated TOAST table, if one exists. Since the commands issued by vacuumdb will also process the TOAST table for the relation if necessary, it does not need to be considered separately.

> **Note**
> This option is only available for servers running PostgreSQL 9.6 and later.

--min-xid-age *xid_age*

Only execute the vacuum or analyze commands on tables with a transaction ID age of at least *xid_age*. This setting is useful for prioritizing tables to process to prevent transaction ID wraparound (see Section 25.1.5).

For the purposes of this option, the transaction ID age of a relation is the greatest of the ages of the main relation and its associated TOAST table, if one exists. Since the commands issued by vacuumdb will also process the TOAST table for the relation if necessary, it does not need to be considered separately.

> **Note**
> This option is only available for servers running PostgreSQL 9.6 and later.

--no-index-cleanup

Do not remove index entries pointing to dead tuples.

> **Note**
> This option is only available for servers running PostgreSQL 12 and later.

--no-process-toast

Skip the TOAST table associated with the table to vacuum, if any.

> **Note**
> This option is only available for servers running PostgreSQL 14 and later.

--no-truncate

Do not truncate empty pages at the end of the table.

> **Note**
> This option is only available for servers running PostgreSQL 12 and later.

-P *parallel_workers*
--parallel=*parallel_workers*

Specify the number of parallel workers for parallel vacuum. This allows the vacuum to leverage multiple CPUs to process indexes. See **VACUUM**(7).

> **Note**
> This option is only available for servers running PostgreSQL 13 and later.

-q
--quiet

Do not display progress messages.

--skip-locked

Skip relations that cannot be immediately locked for processing.

> **Note**
>
> This option is only available for servers running PostgreSQL 12 and later.

-t *table* **[(***column* **[,...])]**
--table=*table* **[(***column* **[,...])]**

Clean or analyze *table* only. Column names can be specified only in conjunction with the **--analyze** or **--analyze-only** options. Multiple tables can be vacuumed by writing multiple **-t** switches.

> **Tip**
>
> If you specify columns, you probably have to escape the parentheses from the shell. (See examples below.)

-v
--verbose

Print detailed information during processing.

-V
--version

Print the vacuumdb version and exit.

-z
--analyze

Also calculate statistics for use by the optimizer.

-Z
--analyze-only

Only calculate statistics for use by the optimizer (no vacuum).

--analyze-in-stages

Only calculate statistics for use by the optimizer (no vacuum), like **--analyze-only**. Run three stages of analyze; the first stage uses the lowest possible statistics target (see default_statistics_target) to produce usable statistics faster, and subsequent stages build the full statistics.

This option is only useful to analyze a database that currently has no statistics or has wholly incorrect ones, such as if it is newly populated from a restored dump or by **pg_upgrade**. Be aware that running with this option in a database with existing statistics may cause the query optimizer choices to become transiently worse due to the low statistics targets of the early stages.

-?
--help

Show help about vacuumdb command line arguments, and exit.

vacuumdb also accepts the following command-line arguments for connection parameters:

-h *host*
--host=*host*

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.

-p *port*
--port=*port*

Specifies the TCP port or local Unix domain socket file extension on which the server is listening for connections.

-U *username*
--username=*username*

User name to connect as.

-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 vacuumdb to prompt for a password before connecting to a database.

This option is never essential, since vacuumdb will automatically prompt for a password if the server demands password authentication. However, vacuumdb 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.

--maintenance-db=*dbname*

Specifies the name of the database to connect to to discover which databases should be vacuumed, when **-a**/**--all** is used. If not specified, the postgres database will be used, or if that does not exist, template1 will be used. This can be a connection string. If so, connection string parameters will override any conflicting command line options. Also, connection string parameters other than the database name itself will be re-used when connecting to other databases.

ENVIRONMENT

PGDATABASE
PGHOST
PGPORT
PGUSER

Default connection parameters

PG_COLOR

Specifies whether to use color in diagnostic messages. Possible values are always, auto and never.

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

DIAGNOSTICS

In case of difficulty, see **VACUUM**(7) and **psql**(1) for discussions of potential problems and error messages. The database server must be running at the targeted host. Also, any default connection settings and environment variables used by the libpq front-end library will apply.

NOTES

vacuumdb might need to connect several times to the PostgreSQL server, asking for a password each time. It is convenient to have a ~/.pgpass file in such cases. See Section 34.16 for more information.

EXAMPLES

To clean the database test:

```
$ vacuumdb test
```

To clean and analyze for the optimizer a database named bigdb:

```
$ vacuumdb --analyze bigdb
```

To clean a single table foo in a database named xyzzy, and analyze a single column bar of the table for the optimizer:

```
$ vacuumdb --analyze --verbose --table='foo(bar)' xyzzy
```

SEE ALSO

VACUUM(7)