

MODERN VACUUM

CHRISTOPH BERG

PGday.at, September 2025



SELECT author FROM talk;

- Senior PostgreSQL Engineer, CYBERTEC
- Debian Developer (PostgreSQL)
- PostgreSQL Major Contributor: apptools.postgresql.org (Debian)



AUSTRIA (HQ)

CYBERTEC POSTGRESQL
INTERNATIONAL (HQ)

ESTONIA

CYBERTEC POSTGRESQL
NORDIC

SWITZERLAND

CYBERTEC POSTGRESQL SWITZERLAND

POLAND

CYBERTEC POSTGRESQL
POLAND

URUGUAY

CYBERTEC POSTGRESQL
SOUTH AMERICA

SOUTH AFRICA

CYBERTEC POSTGRESQL
SOUTH AFRICA



VACUUM

- MVCC: Multi-Version Concurrency Control
- Update creates new row versions
- old versions have to be cleaned up: VACUUM

```
=# select xmin, xmax, ctid, * from foo;
```

xmin	xmax	ctid	id	t
476772	0	(0,1)	1	hello
476772	0	(0,3)	3	hello
476772	0	(0,4)	4	hello



Simple VACUUM

- scan table
- remove dead tuples
 - xmin, xmax, list of currently running transactions
 - dead = xmax < txid_current and xmax has committed

```
=# create extension pageinspect;
```

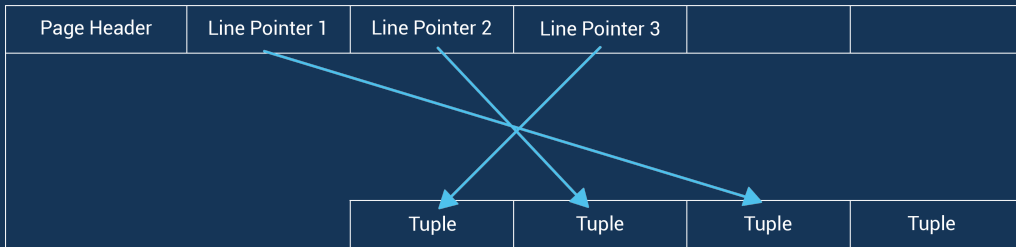
```
=# select * from heap_page_items(get_raw_page('foo', 0));
```

lp	lp_off	lp_flags	lp_len	t_xmin	t_xmax	t_field3	t_ctid
1	8152	1	34	476772	0	0	(0,1)
2	8112	1	34	476772	476774	0	(0,2)
3	8072	1	34	476772	0	0	(0,3)
4	8032	1	34	476772	0	0	(0,4)
5	0	3	0	(n)	(n)	(n)	(n)
6	7992	1	34	476772	0	0	(0,6)



VACUUM with indexes

- scan table
 - mark dead tuples with LP_DEAD (3), remember ctid
- scan indexes, identify dead entries by ctid, delete
- scan table again, delete dead tuples



Memory limits

- “remember ctid” = 6 Byte
- maintenance_work_mem (default 64 MB)
- multiple index cleanup passes



VACUUM with indexes and memory limits

- scan table
- loop:
 - mark dead tuples with LP_DEAD (3), remember ctid
 - scan indexes, identify dead entries by ctid, delete
 - scan table again, delete dead tuples
- VACUUM VERBOSE



Freezing

- xmin/xmax 32 bit, ring counter -> $2^{31} \sim 2$ billion transactions
- old tuples need to be frozen
- scan all tables and rewrite tuple header
- part of VACUUM



Modern VACUUM

How can we improve upon that?

- autovacuum
- speed
- visibility map
- memory limits
- parallelization
- run parts independently



The early days

```
commit 6e6016cb0875aa0f46eff0919892379062c6964a
```

```
Author: mao <mao>
```

```
Date:   Sun May 5 19:46:07 1991 +0000
```

```
'vacuum' is now a postquel command
```



7.4: pg_autovacuum

```
commit bd18c50ba87f12d1dc0aa65c1ff0507b2d1c5c41
Author:      Bruce Momjian <bruce@momjian.us>
AuthorDate:  Thu Mar 20 18:14:46 2003 +0000
```

I have updated my pg_autovacuum program (formerly pg_avd, the name changed as per discussion on the patches list).

...

I have not tested the xid wraparound code as I have to let my AthlonXP 1600 run select 1 in a tight loop for approx. two days in order to perform the required 500,000,000 xacts.

Matthew T. O'Connor



8.0: Cost based vacuum delay feature

```
commit f425b605f4e97a4571372b116a1ec81daf88dfc8
```

```
Author:      Jan Wieck <JanWieck@Yahoo.com>
```

```
AuthorDate:  Fri Feb 6 19:36:18 2004 +0000
```

Cost based vacuum delay feature.

Jan



8.1: Integrated autovacuum daemon

```
commit 29094193f526bf90671d71b59a2e007aad1fcae5
Author:      Tom Lane <tgl@sss.pgh.pa.us>
AuthorDate:  Thu Jul 14 05:13:45 2005 +0000
```

Integrate autovacuum functionality into the backend. There's still a few loose ends to be dealt with, but it seems to work. Alvaro Herrera, based on the contrib code by Matthew O'Connor.

- `autovacuum_vacuum_scale_factor` (default 0.4)



8.2: Make autovacuum behavior more aggressive

```
commit e0938c3f5b03e48bca32fe903f057e5777d43df8
Author:      Bruce Momjian <bruce@momjian.us>
AuthorDate:  Sat Sep 2 23:12:16 2006 +0000
```

Make autovacuum behavior more aggressive, per discussion on hackers list
--- was part of autovacuum default 'on' patch that was reverted, but we
want this part.

Peter Eisentraut

- `autovacuum_vacuum_scale_factor` (default 0.2)



8.3: Enable autovacuum in the default configuration

```
commit 10a5e3348eabc651fe89e93ec37bdbaefeba7a12
Author:      Álvaro Herrera <alvherre@alvh.no-ip.org>
AuthorDate:  Tue Jan 16 18:26:02 2007 +0000
```

Enable autovacuum in the default configuration, per discussion.

```
commit 53d2951be7687089885865f31949eda87439a80b
Author:      Álvaro Herrera <alvherre@alvh.no-ip.org>
AuthorDate:  Tue Jul 24 01:53:56 2007 +0000
```

Set a default autovacuum vacuum_cost_delay value of 20ms, to avoid excessive I/O utilization, per discussion.

While at it, lower the autovacuum vacuum and analyze threshold values to 50 tuples. It is a bit higher (i.e. more conservative) than what I originally proposed but much better than the old values for small tables.



8.3: Add a multi-worker capability to autovacuum

commit e2a186b03cc1a87cf26644db18f28a20f10bd739

Author: Álvaro Herrera <alvherre@alvh.no-ip.org>

AuthorDate: Mon Apr 16 18:30:04 2007 +0000

Add a multi-worker capability to autovacuum. This allows multiple worker processes to be running simultaneously. Also, now autovacuum processes do not count towards the `max_connections` limit; they are counted separately from regular processes, and are limited by the new GUC variable `autovacuum_max_workers`.

The launcher now has intelligence to launch workers on each database every `autovacuum_naptime` seconds, limited only on the max amount of worker slots available.

Also, the global worker I/O utilization is limited by the vacuum cost-based delay feature. Workers are "balanced" so that the total I/O consumption does



8.3: Allow an autovacuum worker to be interrupted automatically

```
commit acac68b2bcae818bc8803b8cb8cbb17eee8d5e2b
Author:      Álvaro Herrera <alvherre@alvh.no-ip.org>
AuthorDate:  Fri Oct 26 20:45:10 2007 +0000
```

Allow an autovacuum worker to be interrupted automatically when it is found to be locking another process (except when it's working to prevent Xid wraparound problems).



State in 8.3 (2008)

- autovacuum
- threshold = 20%
- 3 workers
- cost-based IO limit
- 1 GB memory per index pass



8.3: \h VACUUM

```
$ /usr/lib/postgresql/8.3/bin/psql
```

```
postgres=# \h VACUUM
```

```
Command:      VACUUM
```

```
Description:  garbage-collect and optionally analyze a database
```

```
Syntax:
```

```
VACUUM [ FULL ] [ FREEZE ] [ VERBOSE ] [ table ]
```

```
VACUUM [ FULL ] [ FREEZE ] [ VERBOSE ] ANALYZE [ table [ (column [, ...] ) ] ]
```



18: \h VACUUM

```
postgres=# \h VACUUM
```

```
VACUUM [ ( option [, ...] ) ] [ table_and_columns [, ...] ]  
    FULL [ boolean ]  
    FREEZE [ boolean ]  
    VERBOSE [ boolean ]  
    ANALYZE [ boolean ]  
    DISABLE_PAGE_SKIPPING [ boolean ]  
    SKIP_LOCKED [ boolean ]  
    INDEX_CLEANUP { AUTO | ON | OFF }  
    PROCESS_MAIN [ boolean ]  
    PROCESS_TOAST [ boolean ]  
    TRUNCATE [ boolean ]  
    PARALLEL integer  
    SKIP_DATABASE_STATS [ boolean ]  
    ONLY_DATABASE_STATS [ boolean ]  
    BUFFER_USAGE_LIMIT size
```



<https://www.postgresql.org/about/featurematrix/#vacuum-and-maintenance>

VACUUM and Maintenance

	17	16	15	14	13	12	11	10	9.6	9.5	9.4	9.3	9.2	9.1	9.0	8.4	8.3	8.2	8.1
Inserted data can trigger autovacuum	Yes	Yes	Yes	Yes	Yes	No	No	No	No	No	No	No	No	No	No	No	No	No	No
Integrated autovacuum daemon	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Page freezing optimizations	Yes	Yes	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No
Parallelized VACUUM for Indexes	Yes	Yes	Yes	Yes	Yes	No	No	No	No	No	No	No	No	No	No	No	No	No	No
Parallel vacuumdb jobs	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No	No	No	No	No	No	No
Radix tree memory structure for vacuum	Yes	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No
Vacuum "emergency mode"	Yes	Yes	Yes	Yes	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No
Visibility Map for Vacuuming	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No



Timeline

- 8.4: Visibility Map for Vacuuming
- 9.4: New `autovacuum_work_mem` parameter
- 9.5: Parallel vacuumdb jobs (`vacuumdb --jobs`)
- 9.6: Don't vacuum all-frozen pages
- 9.6: Add VACUUM (`DISABLE_PAGE_SKIPPING`) for emergencies
- 12: Add option `SKIP_LOCKED` to VACUUM and ANALYZE
- 12: Reduce default value of `autovacuum_vacuum_cost_delay` to 2ms
- 12: Allow VACUUM to be run with index cleanup disabled
- 12: Add `TRUNCATE` parameter to VACUUM



Timeline

- 13: Parallelized VACUUM for Indexes (`vacuumdb --parallel`)
- 13: Inserted data can trigger autovacuum
- 14: Add option `PROCESS_TOAST` to VACUUM
- 14: Vacuum “emergency mode”
- 14: Teach VACUUM to bypass unnecessary index vacuuming
- 15: Default to `log_checkpoints=on`, `log_autovacuum_min_duration=10m`
- 16: Page freezing optimizations
- 16: Add `VACUUM/ANALYZE BUFFER_USAGE_LIMIT` option
- 17: Radix tree memory structure for vacuum
- 18: Allow changing `autovacuum_max_workers` without restarting
- 18: Track per-relation cumulative time spent in `[auto]vacuum` and `[auto]analyze`
- 18: Introduce `autovacuum_vacuum_max_threshold`



8.4: Visibility Map for Vacuuming

commit 608195a3a3656145a7eec7a47d903bc684011d73

Author: Heikki Linnakangas <heikki.linnakangas@iki.fi>

AuthorDate: Wed Dec 3 13:05:22 2008 +0000

Introduce visibility map. The visibility map is a bitmap with one bit per heap page, where a set bit indicates that all tuples on the page are visible to all transactions, and the page therefore doesn't need vacuuming. It is stored in a new relation fork.

Lazy vacuum uses the visibility map to skip pages that don't need vacuuming. Vacuum is also responsible for setting the bits in the map. In the future, this can hopefully be used to implement index-only-scans, but we can't currently guarantee that the visibility map is always 100% up-to-date.



9.4: New autovacuum_work_mem parameter

```
commit 8693559cacf1765697c32fc38574af3c19ce61c1
```

```
Author: Simon Riggs <simon@2ndQuadrant.com>
```

```
Date: Thu Dec 12 11:42:39 2013 +0000
```

New autovacuum_work_mem parameter

If autovacuum_work_mem is set, autovacuum workers now use this parameter in preference to maintenance_work_mem.

Peter Geoghegan



9.5: Parallel vacuumdb jobs (vacuumdb --jobs)

```
commit a17923204736d8842eade3517d6a8ee81290fca4
Author:      Álvaro Herrera <alvherre@alvh.no-ip.org>
AuthorDate:  Fri Jan 23 15:02:45 2015 -0300
```

```
vacuumdb: enable parallel mode
```

This mode allows vacuumdb to open several server connections to vacuum or analyze several tables simultaneously.

Author: Dilip Kumar. Some reworking by Álvaro Herrera
Reviewed by: Jeff Janes, Amit Kapila, Magnus Hagander, Andres Freund

- VACUUM macht immer nur eine Tabelle gleichzeitig



9.6: Don't vacuum all-frozen pages

```
commit fd31cd265138019dcccc9b5fe53043670898bc9f
Author:      Robert Haas <rhaas@postgresql.org>
AuthorDate:  Thu Mar 10 16:12:10 2016 -0500
```

Don't vacuum all-frozen pages.

This should greatly reduce the cost of anti-wraparound vacuuming on large clusters where the majority of data is never touched between one cycle and the next, because we'll no longer have to read all of those pages only to find out that we don't need to do anything with them.

Patch by me, reviewed by Masahiko Sawada.



9.6: Add VACUUM (DISABLE_PAGE_SKIPPING) for emergencies

commit ede62e56fbe809baa1a7bc3873d82f12ffe7540b

Author: Robert Haas <rhaas@postgresql.org>

Date: Fri Jun 17 15:48:57 2016 -0400

Add VACUUM (DISABLE_PAGE_SKIPPING) for emergencies.

If you really want to vacuum every single page in the relation, regardless of apparent visibility status or anything else, you can use this option. In previous releases, this behavior could be achieved using VACUUM (FREEZE), but because we can now recognize all-frozen pages as not needing to be frozen again, that no longer works. There should be no need for routine use of this option, but maybe bugs or disaster recovery will necessitate its use.

Patch by me, reviewed by Andres Freund.



12: Add option SKIP_LOCKED to VACUUM and ANALYZE

commit 803b1301e8c9aac478abeec62824a5d09664ffff

Author: Michael Paquier <michael@paquier.xyz>

Date: Thu Oct 4 09:00:33 2018 +0900

Add option SKIP_LOCKED to VACUUM and ANALYZE

When specified, this option allows VACUUM to skip the work on a relation if there is a conflicting lock on it when trying to open it at the beginning of its processing.

Similarly to autovacuum, this comes with a couple of limitations while the relation is processed which can cause the process to still block:
...

Author: Nathan Bossart

Reviewed-by: Michael Paquier, Andres Freund, Masahiko Sawada



12: Reduce default value of autovacuum_vacuum_cost_delay to 2ms

```
commit cbccac371c79d96c44fcd8c9cbb5ff4dedaaa522
```

```
Author: Tom Lane <tgl@sss.pgh.pa.us>
```

```
Date: Sun Mar 10 15:16:21 2019 -0400
```

Reduce the default value of autovacuum_vacuum_cost_delay to 2ms.

This is a better way to implement the desired change of increasing autovacuum's default resource consumption.

Discussion: <https://postgr.es/m/28720.1552101086@sss.pgh.pa.us>



12: Allow VACUUM to be run with index cleanup disabled

`commit a96c41feec6b6616eb9d5baee9a9e08c20533c38`

Author: Robert Haas <rhaas@postgresql.org>

Date: Thu Apr 4 14:58:53 2019 -0400

Allow VACUUM to be run with index cleanup disabled.

This commit adds a new reloption, `vacuum_index_cleanup`, which controls whether index cleanup is performed for a particular relation by default. It also adds a new option to the VACUUM command, `INDEX_CLEANUP`, which can be used to override the reloption. If neither the reloption nor the VACUUM option is used, the default is true, as before.

Masahiko Sawada, reviewed and tested by Nathan Bossart, Alvaro Herrera, Kyotaro Horiguchi, Darafei Praliaskouski, and me.

The wording of the documentation is mostly due to me.



12: Add TRUNCATE parameter to VACUUM

commit b84dbc8eb80b43e554891c459a19969d9fbdefe5

Author: Fujii Masao <fujii@postgresql.org>

Date: Wed May 8 02:10:33 2019 +0900

Add TRUNCATE parameter to VACUUM.

This commit adds new parameter to VACUUM command, TRUNCATE, which specifies that VACUUM should attempt to truncate off any empty pages at the end of the table and allow the disk space for the truncated pages to be returned to the operating system.

This parameter, if specified, overrides the vacuum_truncate reloption. If neither the reloption nor the VACUUM option is used, the default is true, as before.

Author: Fujii Masao

Reviewed-by: Julien Rouhaud, Masahiko Sawada



13: Parallelized VACUUM for Indexes (vacuumdb --parallel)

commit 40d964ec997f64227bc0ff5e058dc4a5770a70a9

Author: Amit Kapila <akapila@postgresql.org>

AuthorDate: Mon Jan 20 07:57:49 2020 +0530

Allow vacuum command to process indexes in parallel.

This feature allows the vacuum to leverage multiple CPUs in order to process indexes. This enables us to perform index vacuuming and index cleanup with background workers. This adds a PARALLEL option to VACUUM command where the user can specify the number of workers that can be used to perform the command which is limited by the number of indexes on a table. Specifying zero as a number of workers will disable parallelism. This option can't be used with the FULL option.

Author: Masahiko Sawada and Amit Kapila

Reviewed-by: Dilip Kumar, Amit Kapila, Robert Haas, Tomas Vondra,

Mahendra Singh and Sergei Kornilov



13: Inserted data can trigger autovacuum

```
commit b07642dbcd8d5de05f0ee1dbb72dd6760dd30436  
Author:      David Rowley <drowley@postgresql.org>  
AuthorDate:  Sat Mar 28 19:20:12 2020 +1300
```

Trigger autovacuum based on number of INSERTs

Traditionally autovacuum has only ever invoked a worker based on the estimated number of dead tuples in a table and for anti-wraparound purposes. For the latter, with certain classes of tables such as insert-only tables, anti-wraparound vacuums could be the first vacuum that the table ever receives. This could often lead to autovacuum workers being busy for extended periods of time due to having to potentially freeze every page in the table. This could be particularly bad for very large tables. New clusters, or recently pg_restored clusters could suffer even ...



13: Inserted data can trigger autovacuum

...

There are additional benefits to triggering autovacuum based on inserts, as tables which never contain enough dead tuples to trigger an autovacuum are now more likely to receive a vacuum, which can mark more of the table as "allvisible" and encourage the query planner to make use of Index Only Scans.

Author: Laurenz Albe, Darafei Praliaskouski

Reviewed-by: Alvaro Herrera, Masahiko Sawada, Chris Travers, Andres Freund, Ju

Discussion: https://postgr.es/m/CAC8Q8t%2Bj36G_bLF%3D%2B0iMo6jGNWnLnWb1tujXu



14: Add option PROCESS_TOAST to VACUUM

commit 7cb3048f38e26b39dd5fd412ed8a4981b6809b35

Author: Michael Paquier <michael@paquier.xyz>

Date: Tue Feb 9 14:13:57 2021 +0900

Add option PROCESS_TOAST to VACUUM

This option controls if toast tables associated with a relation are vacuumed or not when running a manual VACUUM. It was already possible to trigger a manual VACUUM on a toast relation without processing its main relation, but a manual vacuum on a main relation always forced a vacuum on its toast table. This is useful in scenarios where the level of bloat or transaction age of the main and toast relations differs a lot.

A new option switch, called `--no-process-toast`, is added to `vacuumdb`.



14: Vacuum “emergency mode”

commit 1e55e7d1755cefb44982fbacc7da461fa8684e6

Author: Peter Geoghegan <pg@bowt.ie>

Date: Wed Apr 7 12:37:45 2021 -0700

Add wraparound failsafe to VACUUM.

Add a failsafe mechanism that is triggered by VACUUM when it notices that the table's relfrozenxid and/or relminmxid are dangerously far in the past. VACUUM checks the age of the table dynamically, at regular intervals.

Author: Masahiko Sawada <sawada.mshk@gmail.com>

Author: Peter Geoghegan <pg@bowt.ie>

- no indexes, no vacuuming, no speed limit
- vacuum_failsafe_age = 1,6 Mrd



14: Teach VACUUM to bypass unnecessary index vacuuming

commit 5100010ee4d5c8ef46619dbd1d17090c627e6d0a

Author: Peter Geoghegan <pg@bowt.ie>

Date: Wed Apr 7 16:14:54 2021 -0700

Teach VACUUM to bypass unnecessary index vacuuming.

...

We only skip index vacuuming when 2% or less of the table's pages have one or more LP_DEAD items -- bypassing index vacuuming as an optimization must not noticeably impede setting bits in the visibility map.

Author: Masahiko Sawada <sawada.mshk@gmail.com>

Author: Peter Geoghegan <pg@bowt.ie>

- <https://www.cybertec-postgresql.com/en/when-the-dead-wont-die/>



15: Default log_checkpoints=on, log_autovacuum_min_duration=10m

commit 64da07c41a8c0a680460cdafc79093736332b6cf

Author: Robert Haas <rhaas@postgresql.org>

AuthorDate: Mon Dec 13 09:48:04 2021 -0500

Default to log_checkpoints=on, log_autovacuum_min_duration=10m

The idea here is that when a performance problem is known to have occurred at a certain point in time, it's a good thing if there is some information available from the logs to help figure out what might have happened around that time.

Bharath Rupireddy, reviewed by Fujii Masao and by me. Many other people commented on the thread, but as far as I can see that was discussion of the merits of the change rather than review of the patch.



16: Page freezing optimizations

```
commit c3ffa731a5f99c4361203015ce2219d209fea94c
```

```
Author: Peter Geoghegan <pg@bowt.ie>
```

```
Date:   Wed Aug 31 11:37:35 2022 -0700
```

Derive freeze cutoff from nextXID, not OldestXmin.

```
commit 9e5405993c1e242ae6fb7561353e437241244ac1
```

```
Author: Peter Geoghegan <pg@bowt.ie>
```

```
Date:   Tue Nov 15 07:48:41 2022 -0800
```

Deduplicate freeze plans in freeze WAL records.

```
commit 1de58df4fec7325d91f5a8345757314be7ac05da
```

```
Author: Peter Geoghegan <pg@bowt.ie>
```

```
Date:   Wed Dec 28 08:50:47 2022 -0800
```



16: SKIP_DATABASE_STATS and ONLY_DATABASE_STATS

commit a46a7011b27188af526047a111969f257aaf4db8

Author: Tom Lane <tgl@sss.pgh.pa.us>

Date: Fri Jan 6 14:17:25 2023 -0500

Add options to control whether VACUUM runs `vac_update_datfrozenxid`.

VACUUM normally ends by running `vac_update_datfrozenxid()`, which requires a scan of `pg_class`. Therefore, if one attempts to vacuum a database one table at a time --- as `vacuumdb` has done since v12 --- we will spend $O(N^2)$ time in `vac_update_datfrozenxid()`. That causes serious performance problems in databases with tens of thousands of tables, and indeed the effect is measurable with only a few hundred. To add insult to injury, only one process can run `vac_update_datfrozenxid` at the same time per DB, so this behavior largely defeats `vacuumdb`'s `-j` option.



16: Add PROCESS_MAIN to VACUUM

commit 4211fbd8413b26e0abedbe4338aa7cda2cd469b4

Author: Michael Paquier <michael@paquier.xyz>

Date: Mon Mar 6 16:41:05 2023 +0900

Add PROCESS_MAIN to VACUUM

Disabling this option is useful to run VACUUM (with or without FULL) on only the toast table of a relation, bypassing the main relation. This option is enabled by default.

So this feature is basically a shortcut to be able to run VACUUM or VACUUM FULL on a toast relation, using only the name of the parent relation.

Author: Nathan Bossart



16: Add VACUUM/ANALYZE BUFFER_USAGE_LIMIT option

commit 1cbbee03385763b066ae3961fc61f2cd01a0d0d7

Author: David Rowley <drowley@postgresql.org>

Date: Fri Apr 7 11:40:31 2023 +1200

Add VACUUM/ANALYZE BUFFER_USAGE_LIMIT option

Add new options to the VACUUM and ANALYZE commands called BUFFER_USAGE_LIMIT to allow users more control over how large to make the buffer access strategy that is used to limit the usage of buffers in shared buffers. Larger rings can allow VACUUM to run more quickly but have the drawback of VACUUM possibly evicting more buffers from shared buffers that might be useful for other queries running on the database.

Per idea by Andres Freund.

Author: Melanie Plageman



17: Radix tree memory structure for vacuum

commit 667e65aac354975c6f8090c6146fceb8d7b762d6

Author: Masahiko Sawada <msawada@postgresql.org>

Date: Tue Apr 2 10:15:37 2024 +0900

Use TidStore for dead tuple TIDs storage during lazy vacuum.

Since the backing radix tree makes small allocations as needed, the 1GB limit is now gone. Further, the total memory used is now often smaller by an order of magnitude or more, depending on the distribution of blocks and offsets. These two features should make multiple rounds of heap scanning and index cleanup an extremely rare event. TID lookup during index cleanup is also several times faster, even more so when index order is correlated with heap tuple order.

Reviewed-by: John Naylor, (in an earlier version) Dilip Kumar



18: Allow changing autovacuum_max_workers without restarting

commit c758119e5bfb47b38cf957f9a5a37ceae96fa9b3

Author: Nathan Bossart <nathan@postgresql.org>

AuthorDate: Mon Jan 6 15:01:22 2025 -0600

Allow changing autovacuum_max_workers without restarting.

This commit introduces a new parameter named `autovacuum_worker_slots` that controls how many autovacuum worker slots to reserve during server startup. Modifying this new parameter's value does require a server restart, but it should typically be set to the upper bound of what you might realistically need to set `autovacuum_max_workers`. With that new parameter in place, `autovacuum_max_workers` can now be changed with a SIGHUP (e.g., `pg_ctl reload`).

Reviewed-by: Sami Imseih, Justin Pryzby, Robert Haas, Andres Freund, Yogesh Sha



18: Track time spent in [auto]vacuum and [auto]analyze

commit 30a6ed0ce4bb18212ec38cdb537ea4b43bc99b83

Author: Michael Paquier <michael@paquier.xyz>

Date: Tue Jan 28 09:57:32 2025 +0900

Track per-relation cumulative time spent in [auto]vacuum and [auto]analyze

This commit adds four fields to the statistics of relations, aggregating the amount of time spent for each operation on a relation:

- total_vacuum_time, for manual vacuum.
- total_autovacuum_time, for vacuum done by the autovacuum daemon.
- total_analyze_time, for manual analyze.
- total_autoanalyze_time, for analyze done by the autovacuum daemon.

Author: Sami Imseih

Reviewed-by: Bertrand Drouvot, Michael Paquier

Discussion: https://postgr.es/m/CAA5RZ0uVOGBYmPEeGF2d1B_67tgNjKx_bKDUL+olftu



18: Introduce autovacuum_vacuum_max_threshold

commit 306dc520b9dfd6014613961962a89940a431a069

Author: Nathan Bossart <nathan@postgresql.org>

AuthorDate: Wed Feb 5 15:48:18 2025 -0600

Introduce autovacuum_vacuum_max_threshold.

This new parameter provides a way to set a limit on the value calculated with autovacuum_vacuum_threshold and autovacuum_vacuum_scale_factor so that very large tables are vacuumed more frequently. By default, it is set to 100,000,000 tuples, but it can be disabled by setting it to -1. It can also be adjusted for individual tables by changing storage parameters.

Author: Nathan Bossart <nathandbossart@gmail.com>

Co-authored-by: Frédéric Yhuel <frederic.yhuel@dalibo.com>



18: \h VACUUM

```
postgres=# \h VACUUM
```

```
VACUUM [ ( Option [, ...] ) ] [ Tabelle-und-Spalten [, ...] ]
```

```
    FULL [ boolean ]
```

```
    FREEZE [ boolean ]
```

```
    VERBOSE [ boolean ]
```

```
    ANALYZE [ boolean ]
```

```
    DISABLE_PAGE_SKIPPING [ boolean ]
```

```
    SKIP_LOCKED [ boolean ]
```

```
    INDEX_CLEANUP { AUTO | ON | OFF }
```

```
    PROCESS_MAIN [ boolean ]
```

```
    PROCESS_TOAST [ boolean ]
```

```
    TRUNCATE [ boolean ]
```

```
    PARALLEL ganze_Zahl
```

```
    SKIP_DATABASE_STATS [ boolean ]
```

```
    ONLY_DATABASE_STATS [ boolean ]
```

```
    BUFFER_USAGE_LIMIT Größe
```



postgres=# \dconfig *vacuum*

Liste der Konfigurationsparameter

Parameter	Wert	Parameter	Wert
autovacuum	on	log_autovacuum_min_duration	10min
autovacuum_analyze_scale_factor	0.1	vacuum_buffer_usage_limit	2MB
autovacuum_analyze_threshold	50	vacuum_cost_delay	0
autovacuum_freeze_max_age	200000000	vacuum_cost_limit	200
autovacuum_max_workers	3	vacuum_cost_page_dirty	20
autovacuum_multixact_freeze_max_age	400000000	vacuum_cost_page_hit	1
autovacuum_naptime	1min	vacuum_cost_page_miss	2
autovacuum_vacuum_cost_delay	2ms	vacuum_failsafe_age	1600000000
autovacuum_vacuum_cost_limit	-1	vacuum_freeze_min_age	50000000
autovacuum_vacuum_insert_scale_factor	0.2	vacuum_freeze_table_age	150000000
autovacuum_vacuum_insert_threshold	1000	vacuum_max_eager_freeze_failure_rate	0.03
autovacuum_vacuum_max_threshold	100000000	vacuum_multixact_failsafe_age	1600000000
autovacuum_vacuum_scale_factor	0.2	vacuum_multixact_freeze_min_age	5000000
autovacuum_vacuum_threshold	50	vacuum_multixact_freeze_table_age	150000000
autovacuum_worker_slots	16	vacuum_truncate	on
autovacuum_work_mem	-1		

(31 Zeilen)



Defaults

- overall pretty ok, much more performant
- vacuum cost limit (12: autovacuum_vacuum_cost_delay=2ms)
- freezing much less frightening (9.6: VM, 14: emergency, 16: page freezing)
 - watching pg_database.datfrozenxid still makes sense
- autovacuum_max_workers=3 isn't much
- autovacuum_vacuum_scale_factor=0.2
 - solution autovacuum_vacuum_max_threshold?
- automatic index skip (2%) is surprising (14)
- `alter table foo set ( vacuum_truncate = off );`

