

The Best Things About PostgreSQL 19

Christoph Berg

PGDay Lowlands, September 2026



Christoph Berg

- PostgreSQL Major Contributor (everything Debian in PostgreSQL)
- Debian Developer (everything PostgreSQL in Debian)
- Senior PostgreSQL Engineer at Cybertec (everything PostgreSQL)



Why CYBERTEC?



Pure-play
PostgreSQL
Company



Specialised,
International and
Global Team



25 years as a
Key Contributor to
the PostgreSQL
Community



Product &
Service Around
PostgreSQL



Fundamentally
Independent
Operated



AUSTRIA (HQ)

CYBERTEC POSTGRESQL
INTERNATIONAL (HQ)

SWITZERLAND

CYBERTEC POSTGRESQL
SWITZERLAND

URUGUAY

CYBERTEC POSTGRESQL
SOUTH AMERICA

ESTONIA

CYBERTEC POSTGRESQL
NORDIC

POLAND

CYBERTEC POSTGRESQL
POLAND

INDIA

CYBERTEC POSTGRESQL
INDIA PRIVATE LIMITED

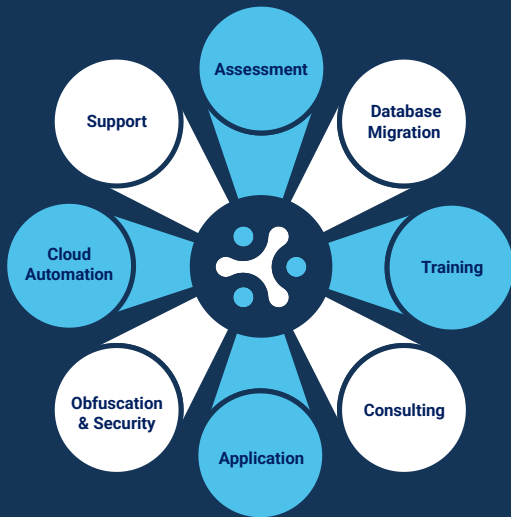
SOUTH AFRICA

CYBERTEC POSTGRESQL
SOUTH AFRICA



Database Services

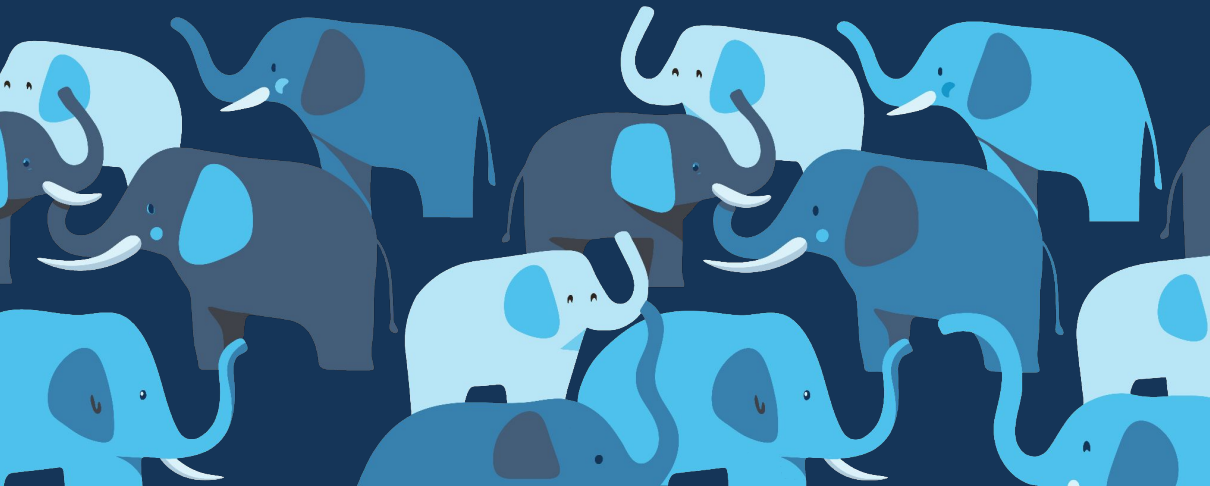
- 24/7 Support
- High Availability
- Consulting
- Performance Tuning
- Clustering
- Migration
- Compliance



Database Products & Tools



PostgreSQL is at the heart of everything we do, and we are proud to actively contribute to its community and project.



PostgreSQL 19 and the Community

- A few hundred unique contributors from many organizations
- 182 patches committed in the final commitfest alone (PG19-Final, April 9), plus more in the preceding ones
- Release expected for end of October
- Today: some spotlights from the perspective of Cybertec
- How the community worked on PostgreSQL 19



The big change: REPACK CONCURRENTLY

- pg_squeeze: tool that rewrites bloated tables online
 - uses logical replication instead of triggers
- Originally by Antonin Houska/Cybertec
- Álvaro Herrera asked: would you donate it to PostgreSQL itself?
- Part 1 (March): new SQL command REPACK
 - “REPACK table” replaces VACUUM FULL and CLUSTER
 - Old commands still exist, but the documentation emphasizes REPACK now
- Part 2 (April): REPACK CONCURRENTLY
 - Create new table, set up logical replication, copy, swap tables at the end
- Two years to get committed, many preparatory commits along the way
- Lesson: community tools can become core features



A simple query

```
select gender, count(*)  
  from person join gender on gender_id = gender.id  
 group by gender;
```

```
gender | count  
-----+-----  
female | 499846  
male   | 500154  
(2 rows)
```



A simple plan

```
explain select gender, count(*)  
  from person join gender on gender_id = gender.id group by gender;
```

```
HashAggregate  (cost=29548.54..29548.56 rows=2 width=40)  
  Group Key: gender.gender  
    -> Hash Join  (cost=1.04..24548.54 rows=1000000 width=32)  
        Hash Cond: (person.gender_id = gender.id)  
        -> Seq Scan on person  (cost=0.00..16360.00 rows=1000000 width=4)  
        -> Hash  (cost=1.02..1.02 rows=2 width=36)  
            -> Seq Scan on gender  (cost=0.00..1.02 rows=2 width=36)
```

Execution Time: 256.730 ms

- Let the planner push aggregates through a join, so they are evaluated earlier



```
set enable_eager_aggregate = on;
```

```
explain select gender, count(*)
```

```
from person join gender on gender_id = gender.id group by gender;
```

```
Finalize GroupAggregate (cost=21360.32..21360.36 rows=2 width=40)
```

```
Group Key: gender.gender
```

```
-> Sort (cost=21360.32..21360.33 rows=2 width=40)
```

```
Sort Key: gender.gender
```

```
-> Nested Loop (cost=21360.13..21360.31 rows=2 width=40)
```

```
-> Partial HashAggregate (cost=21360.00..21360.02 rows=2 width=12)
```

```
Group Key: person.gender_id
```

```
-> Seq Scan on person (cost=0.00..16360.00 rows=1000000 width=4)
```

```
-> Index Scan using gender_pkey on gender (cost=0.13..0.15 rows=1 width=3)
```

```
Index Cond: (id = person.gender_id)
```

Execution Time: 106.591 ms



Patience pays off: eager aggregation

- Antonin's original patch was posted in 2017
- Richard Guo picked it up in late 2025, landed in PostgreSQL 19
- Contributions have a long tail, persistence is rewarded



Better defaults: log_lock_waits = on

- PG log file historically very quiet
- That changed with log_checkpoints (14)
- log_lock_waits = on was suggested multiple times in the past
- Lots of +1 from various people
- Laurenz Albe finally pushed it through for 19, committed by Peter Eisentraut
- Perhaps my favorite “feature” of the release



How the community decides: JIT is off by default

- JIT since PostgreSQL 11 (2018), based on estimated execution cost
- Estimates can be wildly off: 10ms OLTP queries become 1s hogs
- Report after report that turning JIT off was the fix - and other community members and the major cloud providers recommend the same default
- FOSDEM developer meeting: I asked for a raise of hands, everyone was in favor of off by default
- Jelte Fennema-Nio converted the discussion into a pgsql-hackers thread: alternatives - raise the JIT cost limit, tweak the cost model?
- Andres Freund (original author of JIT): since JIT was added, LLVM itself has gotten better but slower to execute, cost-model trigger unfixable for now
- Committed by Tomas Vondra: one default, changed by people from several different companies



The PostgreSQL community contributes to Linux kernel development

- NUMA observability (18 feature by Tomas Vondra, problems found in 19 cycle)
- 32-bit builds on a 64-bit kernel reported weird NUMA node numbers
 - 32 vs 64-bit confusion in the kernel syscall
 - In the end a Linux kernel bug
 - My first kernel patch got merged, and a workaround on the PostgreSQL side
- Builds in Docker: restricted syscalls plus an old libnuma that did not recognize it
 - Another workaround
- Later, under load on real hardware: the kernel can swap out pages
 - Touching all pages to force allocation is not reliable
 - We made the code more robust



allocated



Is “allocated” a word?

```
bool isReset;          /* T = no space allocated since last reset */  
bool allowInCritSection; /* allow palloc in critical section */
```

- Is “allocated” a proper word?
- Rafia Sabih sent a short patch
- Discussion: is “allocated” or “allocated” better?
- Tom Lane (original author of that line): it may have been shortened to keep the line from overflowing
- But the full word fits



COMMIT;

Author: David Rowley <drowley.postgresql.org>

Date: Sat Oct 18 12:50:14 2025 +1300

Englishify comment wording

Switch to using the English word here rather than using a verbified function name.

The full word still fits within a single comment line, so it's probably better just to use that instead of trying to shorten it, which might cause confusion.

Author: Rafia Sabih <rafia.pghackers.gmail.com>



The best thing about PostgreSQL is the Community

- One comment nitpick and a two-year feature went through the identical pipeline
- Post on the list, discuss, review, commit
- We care about details: the “allocated” fix and REPACK CONCURRENTLY were held to the same standard
- Everyone working together to improve PostgreSQL
- This (and everything else) is why PostgreSQL works so well



Get involved

- PostgreSQL 19 ships at the end of October
- PostgreSQL 20 is next
- Join the next thread that catches your eye - even one-liners
- Review patches in the commitfest
- Or bring a tool: REPACK started as a community project
- Help out at conferences, user group meetings, ...

