What's new in PG18

Tomas Vondra <tomas@vondra.me> / https://vondra.me

OpenAlt 2025, November 1-2, Brno

Tomas Vondra

- Postgres engineer @ Microsoft
- https://vondra.me
- vondratomas@microsoft.com
- tomas@vondra.me
- office hours
- ...

Prague events

Prague PostgreSQL Developer Day 2026

January 27-28

CfP (closes November 14)
https://cfp.p2d2.cz/2026/

looking for sponsors & partners

Prague PostgreSQL Meetup

https://www.meetup.com/prague-postgresql-meetup

Agenda

- Development overview
- Breaking changes
- New features
  - DBA and administration
  - SQL and developer
  - Backup and replication
  - Performance

https://www.hagander.net/talks/PostgreSQL%2018.pdf

https://www.postgresql.org/docs/current/release-18.html

https://www.youtube.com/@pgeu/videos

Development schedule

- July 2024 - branch 17
- July 2024 - CF1
- September 2024 - CF2
- November 2024 - CF3
- January 2025 - CF4
- March 2025 - CF5
- September 2025 - Release!

REL_18_STABLE

```
commit e26810d01d441a457217a6eae9c2989fba29b80f
Author: Michael Paquier <michael@paquier.xyz>
Date:   Mon Jul 1 07:56:10 2024 +0900

    Stamp HEAD as 18devel.

    Let the hacking begin ...
```

current status (2025/10/30):

3119 commits

3997 files changed, 413639 insertions(+), 211453 deletions(-)

Breaking changes

- Remove support for HPPA
- Remove support for lack of spinlocks
- Remove support for lack of atomics
- Remove support for OpenSSL older than 1.1.1

DBA and administration

DBA and administration

- data checksums enabled by default
  - Finally!
  - By initdb
  - --no-data-checksums to disable
  - NOT on upgrades (pg_upgrade)
- upgrades & stats
  - Stats are transferred on pg_upgrade
  - pg_stats, not pg_stat
  - ready to use much faster after upgrade!
  - actually in pg_dump
  - only basic stats (not extended)

DBA and administration / Authentication

- md5 deprecated
  - Can set md5_password_warnings=off
  - But don't!
- OAUTHBEARER
  - Log in using OAUTH bearer token
  - Requires server side provider
  - Written in C
  - No default provided
- SCRAM pass-through
  - In postgres_fdw and dblink
  - No need for clear-text password
  - use_scram_passthrough=true on SERVER
  - Must have same salt and iteration count!

DBA and administration / TLS

- Support for TLSv1.3 cipher suites
- Support for multiple ECDH curves
- pg_crypto can disable built-in crypto

autovacuum

- autovacuum_max_threshold
  - for large tables (rows * vacuum_scale_factor) too large
  - upper bound on calculated threshold, default 100M
- autovacuum_max_workers
  - change without restart
  - up to autovacuum_worker_slots
  - probably not what you need

VACUUM / ANALYZE

- EXPLAIN ANALYZE
  - BUFFERS enabled by default
  - Show parallel bitmap scan stats
  - Show memory/disk use for Materialize nodes
- VACUUM [ONLY]
  - For both VACUUM and ANALYZE
  - Specify ONLY to not recurse into partitions
  - ANALYZE particularly useful for partitioned tables

COPY

```
log_verbosity = 'silent'


COPY a FROM '/tmp/test.csv'
       WITH (FORMAT csv, ON_ERROR ignore);
NOTICE: 2 rows were skipped due to data type incompatibility
COPY 4


COPY a FROM '/tmp/test.csv'
       WITH (FORMAT csv, ON_ERROR ignore, LOG_VERBOSITY silent);
COPY 4
```

Parallel worker stats

- New fields
  - parallel_workers_to_launch
  - parallel_workers_launched
- Per db or statement
  - pg_stat_database
  - pg_stat_statements

VACUUM stats

- Per table time spent
  - total_vacuum_time
  - total_autovacuum_time
  - total_analyze_time
  - total_autoanalyze_time
- Time spent delaying
  - pg_stat_progress_vacuum
  - pg_stat_progress_analyze

WAL stats

- Now tracked in pg_stat_io
  - Much more granular
  - Per backend-type
- Removed from pg_stat_wal
- wal_buffers_full
  - Added to pg_stat_statements
  - In VACUUM/ANALYZE VERBOSE
  - In EXPLAIN (WAL)
- Still globally in pg_stat_wal

GUC changes

- effective_io_concurrency
- maintenance_io_concurrency
  - new default is 16

SQL and developer

UUIDv7

- New generation function
- Sortable
- Standard says milliseconds
- PostgreSQL does 12-bit sub-millisecond
- Better for indexes

OLD/NEW for RETURNING

- OLD/NEW for RETURNING
- Ability to access both old and new value
  - In UPDATE
  - And MERGE
- But also for ON CONFLICT
  - Determine INSERT or UPDATE

Virtual generated columns

- Like STORED virtual columns
- Except not.. stored.
- Re-calculated on each read
- Cannot be indexed
- "Partial view"

Temporal keys

- PRIMARY and FOREIGN
- You probably want btree_gist

Backup and replication

pg_verifybackup

Can now verify tar format

(previously only plain)

logical replication

Replicate generated columns

- Logical replication of generated columns
- Only stored!

pg_stat_subscription_stats

- Collects conflict stats
- INSERT / UPDATE conflicts
- Origin conflicts
- UPDATE / DELETE missing

Performance

Many different

- Lots of infrastructure
- Often not directly exposed
- Use streaming I/O
- More eagerly vacuum all-visible pages
  - To make aggressive vacuum cheaper
- (... more)

Parallel CREATE INDEX

- Now also for GIN
- (in addition to btree and brin)
- hstore, pg_trgm, tsvector, json, jsonb, ...

btree index skip-scan

Use multi-column index for non-prefix scans

Not as fast as dedicated index

But fewer indexes!

Typically with few distinct values in early columns

https://www.youtube.com/watch?v=DpeGBfxg4yc

pg_upgrade

Much more parallel

Previously just pg_dump and copy/link

--swap mode

Move data directory, then overwrite catalog

Fast, but no rollback

General queries

- Detect redundant GROUP BY based on UNIQUE
  - Previously only PRIMARY KEY
- Proper row estimates for generate_series
  - now also numeric and timestamp
- Optimized tuplestore for recursive CTE
  - Much faster for some queries (25+%)

General queries

- Reduced memory usage on partitionwise join
- JSON escaping using SIMD
- Right Semi Join
- Faster numeric multiplication and division

General queries

- enable_self_join_elimination
  - remove unnecessary joins (table already joined)
  - ... can be proven to be the same output
  - often caused by VIEWs or ORMs
  - has to be cheap not to hurt "good" queries
- enable_distinct_reordering
  - remove unnecessary sort for DISTINCT
- incremental sorts, partition-wise joins
  - allow in more cases

Asynchronous I/O

- worker or io_uring
- default: worker
- faster prefetching
- foundation for direct I/O
- but not there yet
  - only reads (for now)

pgconf.eu
- https://anarazel.de/talks/2025-10-23-pgconf-eu-aio-in-PG-18-and-beyond/aio-in-PG-18-and-beyond.pdf

pgconf.dev
- https://anarazel.de/talks/2025-05-15-pgconf-dev-what-went-wrong-aio/what-went-wrong-aio.pdf
- https://www.youtube.com/watch?v=GR5v9DHiS8w

Q & A