Rails の PostgreSQL 18 対応

Rails meets PostgreSQL 18

Yasuo Honda

Kaigi on Rails 2025 - Sep 26, 2025

Partner with RubyStackNews

Independent Ruby & Rails publication for senior developers

Why RubyStackNews?

- Focused on Ruby and Ruby on Rails
- Long-form articles based on real conference talks
- Audience of senior developers and tech leads
- Readers from the US, Europe, and Asia

RubyStackNews turns conference talks and real-world experience into practical, production-focused technical articles.

Partnerships & Sponsorships

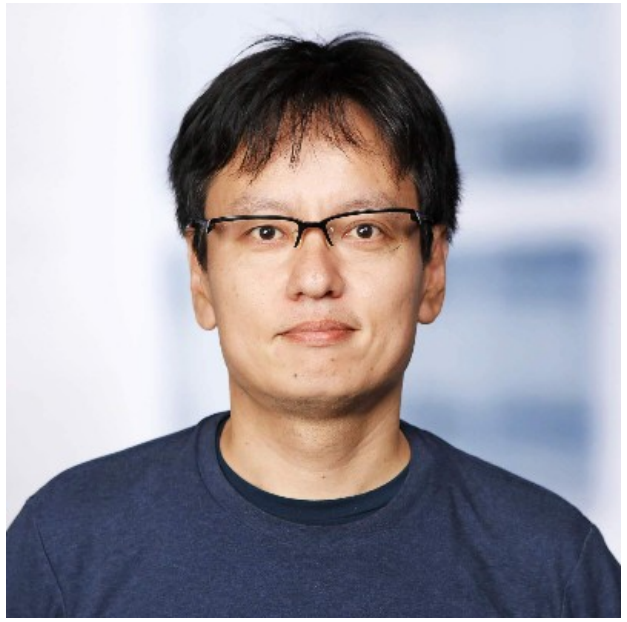
- Article sponsorships
- Inline placements inside articles
- Sidebar visibility

View partnership details

rubystacknews.com

Partnership info

About Me | Yasuo Honda



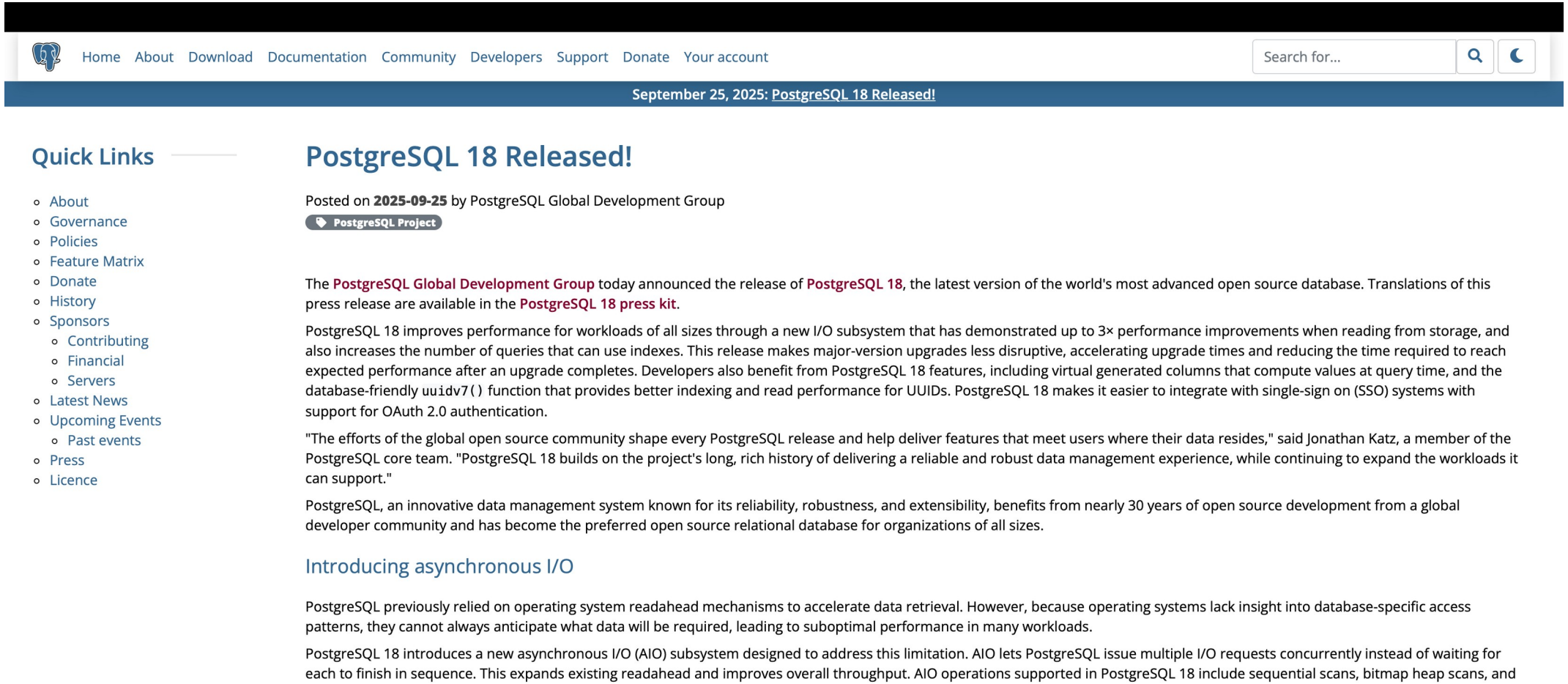
- Rails Committer
- Maintainer of Active Record Oracle enhanced adapter

Find me on:

- https://github.com/yahonda
- https://rubyfriends.app/profiles/VRDX

PostgreSQL 18 Released!

- 2025 年 9 月 25 日
  https://www.postgresql.org/about/news/postgresql-18-released-3142/



Home About Download Documentation Community Developers Support Donate Your account

Search for...

September 25, 2025: PostgreSQL 18 Released!

Quick Links

- About
- Governance
- Policies
- Feature Matrix
- Donate
- History
- Sponsors
  - Contributing
  - Financial
  - Servers
- Latest News
- Upcoming Events
  - Past events
- Press
- Licence

PostgreSQL 18 Released!

Posted on **2025-09-25** by PostgreSQL Global Development Group

PostgreSQL Project

The **PostgreSQL Global Development Group** today announced the release of **PostgreSQL 18**, the latest version of the world's most advanced open source database. Translations of this press release are available in the **PostgreSQL 18 press kit**.

PostgreSQL 18 improves performance for workloads of all sizes through a new I/O subsystem that has demonstrated up to 3× performance improvements when reading from storage, and also increases the number of queries that can use indexes. This release makes major-version upgrades less disruptive, accelerating upgrade times and reducing the time required to reach expected performance after an upgrade completes. Developers also benefit from PostgreSQL 18 features, including virtual generated columns that compute values at query time, and the database-friendly `uuidv7()` function that provides better indexing and read performance for UUIDs. PostgreSQL 18 makes it easier to integrate with single-sign on (SSO) systems with support for OAuth 2.0 authentication.

"The efforts of the global open source community shape every PostgreSQL release and help deliver features that meet users where their data resides," said Jonathan Katz, a member of the PostgreSQL core team. "PostgreSQL 18 builds on the project's long, rich history of delivering a reliable and robust data management experience, while continuing to expand the workloads it can support."

PostgreSQL, an innovative data management system known for its reliability, robustness, and extensibility, benefits from nearly 30 years of open source development from a global developer community and has become the preferred open source relational database for organizations of all sizes.

Introducing asynchronous I/O

PostgreSQL previously relied on operating system readahead mechanisms to accelerate data retrieval. However, because operating systems lack insight into database-specific access patterns, they cannot always anticipate what data will be required, leading to suboptimal performance in many workloads.

PostgreSQL 18 introduces a new asynchronous I/O (AIO) subsystem designed to address this limitation. AIO lets PostgreSQL issue multiple I/O requests concurrently instead of waiting for each to finish in sequence. This expands existing readahead and improves overall throughput. AIO operations supported in PostgreSQL 18 include sequential scans, bitmap heap scans, and

Rails の PostgreSQL 対応とは

- バージョン互換性
  - Rails 6.0 以降は PostgreSQL **9.3** 以上に対応
  - 「最小対応バージョン」のみを定義しており、上限は設けていない
    - つまり、現時点では PostgreSQL 18 も対応バージョンに含まれる
- データベース新機能対応
  - supports_< 機能名 >? メソッドによる判別
  - 例 : `supports_virtual_columns?` (PostgreSQL Adapter)

```
def supports_virtual_columns?
  database_version >= 12_00_00 # >= 12.0
end
```

Note: 「対応」という用語

本プレゼンテーションでは、 https://rubyonrails.org/maintenance より New Features ま
Bug Fixes でメンテナンスすることを「対応」と呼んでいます

- New Features
- Bug Fixes
- Security Fixes

Rails の PostgreSQL 対応を構成する要素

協調する 3 つの要素

- Rails : `PostgreSQLAdapter`
  - https://github.com/rails/rails/blob/main/activerecord/lib/active_record/connecti
    _adapters/postgresql_adapter.rb
- Database driver `pg` gem
  - https://github.com/ged/ruby-pg
- Client library : `libpq`
  - https://github.com/postgres/postgres/tree/master/src/interfaces/libpq
    https://git.postgresql.org/gitweb/?p=postgresql.git;a=tree;f=src/interfaces/libpq

Rails アプリケーション開発者への振る舞いを担保するのは **Rails**

Rails の PostgreSQL 18 バージョン互換性

- プロトコルバージョンの更新とキャンセルキー長の変更
- パーティション表の UNLOGGED サポートの削除

プロトコルバージョンの更新とキャンセルキー長の変更

- **3.0** : PostgreSQL 7.4 で追加、 PostgreSQL 18 でも利用可能
  - キャンセルキー (クエリーをキャンセルするのに必要なキー) の長さは 4byte 固定
  - 総当たりにより特定される可能性があるとの懸念
- **3.2** : PostgreSQL 18 で追加
  - 長いキャンセルキー (最大 256byte) 対応
  - 3.1 は PgBouncer が 3.1 を 3.0 として扱っていたため skip
- PostgreSQL(libpq) 18 は デフォルトでプロトコルバージョン 3.0 を利用
  https://www.postgresql.org/about/news/postgresql-18-released-3142/

> libpq still uses version 3.0 by default while clients (e.g., drivers, poolers, proxies) add support for the new protocol version.

pg gem と長いキャンセルキーの対応

- pg gem 1.5.9 以下のバージョンは長いキャンセルキーに対応していない
  - `PG::Connection#cancel` が `PG::Connection#backend_key` の取得に失敗
- pg gem 1.6.0 で PostgreSQL 17 の `PQcancelBlocking` と `PQcancelStart` 関数に対応
  - `PG::Connection#backend_key` の取得が不要に
  - https://github.com/ged/ruby-pg/pull/614
- 当時 v1.6.0.rc1 だったため、 1.6.0 をリリースしてほしいという issue
  - https://github.com/ged/ruby-pg/issues/639
  - 1.6.0.rc2 Rails の CI が green なことを確認し、 1.6.0 をリリースしてもらう

Rails と長いキャンセルキーの対応

- Rails は pg gem 1.1 以上、 2.0 未満に対応している `gem "pg", "~> 1.1"`
  - pg gem 1.5.9 以下のユーザーへの対応が必要
- Rails の対応 : libpq が 18 以上かつ pg gem 1.6.0 未満で `ActiveRecord::ConnectionAdapters::DatabaseStatements#cancel_any_running_query` で `PG::Connection#cancel` を呼ばない
  - https://github.com/rails/rails/pull/55540
- 理由 `:cancel_any_running_query` は private メソッド
  - `exec_rollback_db_transaction` と `exec_restart_db_transaction` (いずれも非公開 API) からのみ呼ばれる
  - クエリーがキャンセルされなくてもトランザクションはいずれロールバックされる

Rails と長いキャンセルキーの対応

- Rails 8.1.0 に入る予定、 Rails 8.0.3 で修正ずみ
  - Bug Fixes としての対応 (PostgreSQL 9.3 以上をサポートするということ)
  - Rails 7.2 へのバックポートはしない
    - 新しい PostgreSQL バージョンの対応は Security Fixes ではない
- 個人的な推奨
  - PostgreSQL (libpq) 18 を使う場合は、 pg gem 1.6 以上 を利用
  - pg gem 1.6.0 以降は Linux でも fat gem(例 : 1.6.2-x86_64-linux)
    - 1.6.2 では libpq 17.6 を同梱
  - fat gem であっても PostgreSQL client は必要

    `config.active_record.schema_format = :sql` では `pg_dump`

    `dbconsole` では `psql` が必要なため、同じ libpq を使いたい場合は non-fat gem

パーティション表の UNLOGGED サポートの削除

- UNLOGGED : テーブルへの書き込み時に WAL(Write Ahead Log) を書かない
  - 高速な書き込みとクラッシュセーフではないトレードオフ
- Rails フレームワークのテストの高速化のため UNLOGGED テーブルを利用
  - https://github.com/rails/rails/pull/47499
- PostgreSQL 18 はパーティション表に UNLOGGED を指定するとエラーになる
- Rails フレームワークテストではパーティション表を LOGGED で作成するようにした
  - https://github.com/rails/rails/pull/53439
- rails/rails#47499 が入った 7.1(7-1-stable ブランチ) に遡ってバックポート

バージョン互換性対応の pull request を PostgreSQL 正式版リリース前にマージした (できた) 理由

- PostgreSQL 18 RC 1 の動作を確認した上でマージしている
  - https://www.postgresql.org/about/news/postgresql-18-rc-1-released-3130/
- パーティション表の UNLOGGED サポートの削除対応
  - 仮に revert されても Rails フレームワークのユニットテスト内部の変更のみでユーザーに影響がない
- Rails と長いキャンセルキーの対応
  - 過去にリリースされた (変えられない) pg gem 1.5.9 以下の対応

Rails の PostgreSQL 18 データベース新機能対応

- pg_stat_statements の変更
- 仮想生成列 (virtual generated columns)

pg_stat_statements の変更

- pg_stat_statements とは
  https://www.postgresql.jp/docs/17/pgstatstatements.html

  > pg_stat_statements モジュールは、サーバで実行されたすべての SQL 文のプラン生成時と実行時の統計情報を記録する手段を提供します。

- Kaigi on Rails 2024 "Rails の Pull requests のレビューの時に私が考えていること " で話した「 pg_stat_statements の " 汚染 " 」への改善が PostgreSQL 18 に入った
- https://git.postgresql.org/gitweb/?p=postgresql.git;a=commitdiff;h=62d712ecf
  - Patch をテストしたりユースケースを pgsql-hackers メーリングリストに投稿した
- リリースノートの Acknowledgments に名前が載る
  - https://www.postgresql.org/docs/current/release-18.html

Note : データベース固有 (specific) と非依存 (agnostic) の両立

- バランス
  - 特定データベース固有機能であっても追加されることはある
  - 遅延制約 (PostgreSQL のみ) など
- 複数のデータベースで類似した機能が存在する場合
  - Rails からは共通の API 、名称になるように努めている
  - 例 : Rails 8.1 での Disabling index 対応
    - https://github.com/rails/rails/pull/54332
  - MySQL では Invisible Indexes, MariaDB では Ignored Indexes と呼ばれるもの

生成列 (generated columns) とは

- https://www.postgresql.jp/docs/17/ddl-generated-columns.html

> 生成列は常に他の列から計算される特別な列です。 ですから、これは列におけるテーブルに対するビューのようなものです。

- Rails での生成列の例 (`t.virtual`)

```
create_table :users do |t|
  t.string :name
  t.virtual :name_upcased, type: :string, as: 'upper(name)', stored: true
end
```

データベースアダプタごとの生成列対応

- MySQL 5.7.5+ and MariaDB 5.2.0+.
  - Rails 5.1
  - https://github.com/rails/rails/commit/65bf1c60053e727835e06392d27a2fb49665484c
- SQLite 3.31.0+
  - Rails 7.2
  - https://github.com/rails/rails/pull/49346
- PostgreSQL 12+
  - Rails 7.0
  - https://github.com/rails/rails/pull/41856

PostgreSQL と生成列

- PostgreSQL 12 では、格納生成列 (stored generated columns) のみ対応
  - 他のカラムから演算した結果をディスクに保存する
  - 演算は書き込み時に行われる
- PostgreSQL 18 から、仮想生成列 (virtual generated columns) 対応追加
  - 他のカラムから演算した結果をディスクに保存しない
  - 演算は読み込み時に行われる
  - 仮想生成列がデフォルト
- MySQL/MariaDB, SQLite では仮想生成列がデータベースとしてのデフォルトの動作
  - オプションで格納生成列も利用可能

Rails 8.0 以下の PostgreSQL 12+ での生成列対応

- 格納生成列のみ対応

```
create_table :users do |t|
  t.string :name
  t.virtual :name_upcased, type: :string, as: 'upper(name)', stored: true
end
```

- `stored: true` を必須とし、それ以外は `ArgumentError` を raise していた

```
PostgreSQL currently does not support VIRTUAL (not persisted) generated columns.
Specify 'stored: true' option for '#{options[:column].name}'
```

- https://github.com/rails/rails/pull/41856#issuecomment-920933731

PostgreSQL 18+ での生成列対応 (rails/rails#55142)

- PostgreSQL 18 以上の場合に仮想生成列に対応
  - `stored: false` を指定可能 (Rails のデフォルトの動作)

```
create_table :users do |t|
  t.string :name
  t.virtual :lower_name, type: :string, as: "LOWER(name)", stored: false
  t.virtual :name_length, type: :integer, as: "LENGTH(name)"
end
```

- https://github.com/rails/rails/pull/55142 で open 中
  - 入る場合は main ブランチのみ (New Features として)

データベース新機能を利用する Rails の新機能

- PostgreSQL 正式版リリース前は追加された機能の revert や変更の可能性を想定
- 仮想生成列、セキュリティや振る舞いなど PostgreSQL 開発コミュニティで議論
  - "pg18: Virtual generated columns are not (yet) safe when superuser selects from them" https://www.postgresql.org/message-id/flat/156542c6fb54bfadf2e67bcb419749bba6e65149.camel%40j-davis.com#993fe6ec9cf7bf2ab5ab7eb8c9fbc580
- パッチが提供され、議論はおさまった

PostgreSQL 18 Released!

- 2025 年 9 月 25 日に PostgreSQL 18 リリース
  https://www.postgresql.org/docs/current/release-18.html#RELEASE-18-HIGHLIGHTS

  > Virtual generated columns that compute their values during read operations. This is now the default for generated columns.

- https://github.com/rails/rails/pull/55142 ?

Thank you for listening - enjoy PostgreSQL 18 with Rails 8.1!