PostgreSQL Upgrade Strategies

data egret

Your remote PostgreSQL DBA team

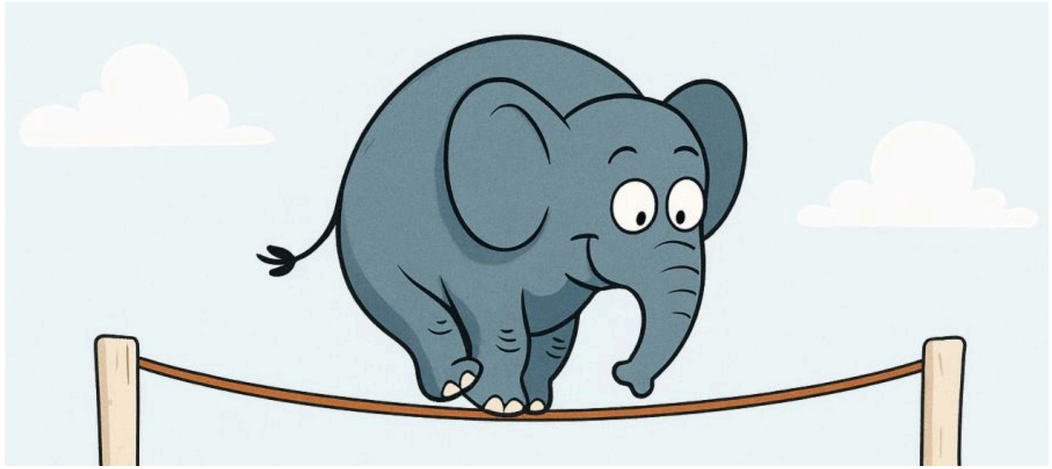
Stefan FERCOT

stefan.fercot@dataegret.com

Stefan Fercot

- PostgreSQL Expert **@Data Egret**
  - PostgreSQL consulting, support, and training
- pgBackRest fan & contributor
- Recognized PostgreSQL contributor
- aka. pgstef
- https://pgstef.github.io

PostgreSQL Upgrade Strategies



Today’s agenda

- PostgreSQL release cycle and milestones
- Major upgrade approaches
- Comparing strategies: downtime, space, complexity
- Handling extensions (with focus on PostGIS)

PostgreSQL Versions

- `X`: major version (10, 11, … 17)
  - One major version is released each year and supported for five years
  - **Sep 25, 2025 (planned):** PostgreSQL 18 general release
  - **Nov 13, 2025:** End of Life (EOL) for PostgreSQL 13
- `X.Y`: minor version (e.g., 14.8, 17.4)
  - Released typically every quarter
- Always read the release notes!
  - https://why-upgrade.depesz.com/

Major Upgrade

- Upgrade process:
  - Update the binaries
  - Update or process the data files
- Thoroughly test your application with the new version
  - Check for regressions in features and performance
  - Consider the operating system, extensions, and external tools
  - Read release notes carefully

Upgrade Strategies

- PostgreSQL documentation:
  - https://www.postgresql.org/docs/current/upgrading.html
- Data upgrade options:
  - `pg_dumpall`
  - `pg_upgrade`
  - Logical replication

Logical Export

> *The traditional method for moving data to a new major version is to dump and restore the database, though this can be slow.*

- Simple procedure
- Benefits:
  - Reduced bloat
  - Allows enabling features such as checksums on the new cluster
- Suitable for small to medium databases
- Allows restore on same or another server

`pg_upgrade`

- In-place migration, both versions must be installed on the server
  - Create an empty database for the new PostgreSQL version
  - Run `pg_upgrade` with `--check` for a dry run
  - Stop the database running the old PostgreSQL version
  - Run the upgrade with the `pg_upgrade` command
  - Start the database with the new PostgreSQL version

> *`pg_upgrade` can be very fast, especially in `--link` mode.*

`pg_upgrade --link`

> *If you did start the new cluster, it has written to shared files and it is unsafe to use the old cluster. The old cluster will need to be restored from backup in this case.*

Logical Replication

- Set up logical replication from the old cluster to the new one
- Perform a switchover once the new cluster is fully synchronised
- Flexible but more complex than the previous methods
- Initial synchronisation can take a long time
- Subject to logical replication limitations

Short Summary

Method	Downtime	Extra disk space	Complexity	Riskiness
dump/restore	high	double	low	low
pg_upgrade (copy)	high	double	medium	low
pg_upgrade (link)	low	low	medium	high
logical replication	low	double	high	low

Useful Resources

- PGConf.DE 2023 - An ultimate guide to upgrading your PostgreSQL installation
- PGConf India 2024 - pg_upgrade like a boss!
- PGDay BE 2025 - Tips and tools for minimal downtime in PostgreSQL upgrades
- Examining Postgres Upgrades with pg_upgrade
- Upgrading and updating PostgreSQL
- Upgrading Postgres major versions using Logical Replication

Extensions

- Verify that all required extensions are available for the new PostgreSQL version
- Check availability in documentation or with:
  - `SELECT * FROM pg_available_extensions;`
- Cycle through all extensions and run:
  - `ALTER EXTENSION extension_name UPDATE;`
- Some extensions need special care (e.g. PostGIS)

PostGIS

https://postgis.net/workshops/postgis-intro/upgrades.html

- PostGIS should be updated before performing a PostgreSQL upgrade
- The exact version pairs available are dictated by the PGDG packages
  - PostGIS Support Matrix

```
SELECT postgis_extensions_upgrade();
-- verify you are running latest now
SELECT postgis_full_version();
```

Aligned Versions, Flexible Upgrade

```
$ dnf search postgis*_13
========================== Name Matched: postgis*_13 ==========================
postgis33_13.x86_64 : Geographic Information Systems Extensions to PostgreSQL
postgis34_13.x86_64 : Geographic Information Systems Extensions to PostgreSQL
postgis35_13.x86_64 : Geographic Information Systems Extensions to PostgreSQL
postgis36_13.x86_64 : Geographic Information Systems Extensions to PostgreSQL

$ dnf search postgis*_17
========================== Name Matched: postgis*_17 ==========================
postgis33_17.x86_64 : Geographic Information Systems Extensions to PostgreSQL
postgis34_17.x86_64 : Geographic Information Systems Extensions to PostgreSQL
postgis35_17.x86_64 : Geographic Information Systems Extensions to PostgreSQL
postgis36_17.x86_64 : Geographic Information Systems Extensions to PostgreSQL
```

Constrained Upgrade Path

```
$ dnf search postgis*_11
========================== Name Matched: postgis*_11 ==========================
postgis30_11.x86_64 : Geographic Information Systems Extensions to PostgreSQL
postgis31_11.x86_64 : Geographic Information Systems Extensions to PostgreSQL
postgis32_11.x86_64 : Geographic Information Systems Extensions to PostgreSQL
postgis33_11.x86_64 : Geographic Information Systems Extensions to PostgreSQL

$ dnf search postgis*_17
========================== Name Matched: postgis*_17 ==========================
postgis33_17.x86_64 : Geographic Information Systems Extensions to PostgreSQL
postgis34_17.x86_64 : Geographic Information Systems Extensions to PostgreSQL
postgis35_17.x86_64 : Geographic Information Systems Extensions to PostgreSQL
postgis36_17.x86_64 : Geographic Information Systems Extensions to PostgreSQL
```

postgis_extensions_upgrade()

```
SELECT name, default_version, installed_version FROM pg_available_extensions
  WHERE installed_version <> default_version;
          name           | default_version | installed_version
-------------------------+-----------------+-------------------
 postgis                 | 3.6.0           | 3.3.8
 postgis_sfcgal          | 3.6.0           | 3.3.8
 postgis_tiger_geocoder | 3.6.0           | 3.3.8
 postgis_topology        | 3.6.0           | 3.3.8
```

```
SELECT postgis_extensions_upgrade();
NOTICE:  Updating extension postgis 3.3.8
NOTICE:  Updating extension postgis_sfcgal 3.3.8
NOTICE:  Updating extension postgis_topology 3.3.8
NOTICE:  Updating extension postgis_tiger_geocoder 3.3.8
                            postgis_extensions_upgrade
-------------------------------------------------------------------------------
 Upgrade to version 3.6.0 completed, run SELECT postgis_full_version(); for details
```

Conclusion

> *Don’t be scared, be prepared*

- Using proper tools helps manage downtime
  - Both when things go right and when they don’t
- Upgrading on time makes it easier
- Extensions can surprise you
  - Some of them (e.g., PostGIS) require special care

PostgreSQL <3 Belgium

https://www.meetup.com/postgresbe

- **PgBE PostgreSQL Users Group Belgium** meetup group
  - **October 14**: Google, Brussels
  - **November 25**: Idewe, Leuven

Thank You!

contact@dataegret.com

Securing your PostgreSQL database availability and high performance.

- Performance audit
- Backup & restore
- Migration
- Cloud Cost Management
- Architecture review
- DataOps/ CDC projects
- 24/7 Incident support

on premises & cloud

EXPERTISE

Senior DBA team with **10+ years of PostgreSQL experience** each.

DEVELOPMENT

Involved in **new feature and extension development**.

TAILORED APPROACH

Dedicated DBA team that focused on success of your project.

COMMUNITY

Recognised significant **contributing sponsor to PostgreSQL**.