9. Upgrades

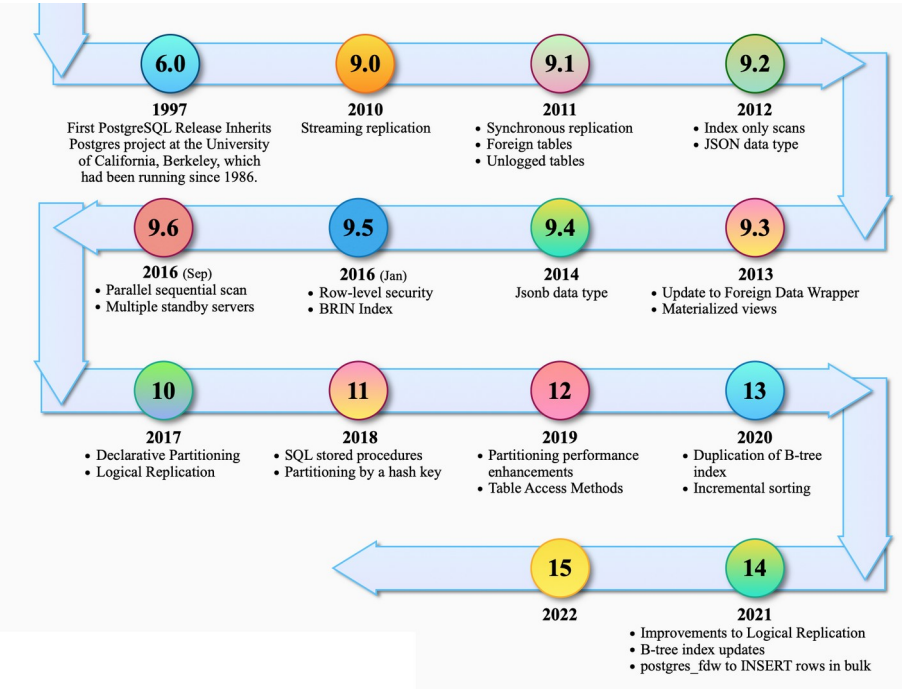
Introduction to PostgreSQL

AGENDA

- Major Versions
- Minor Versions
- Patches

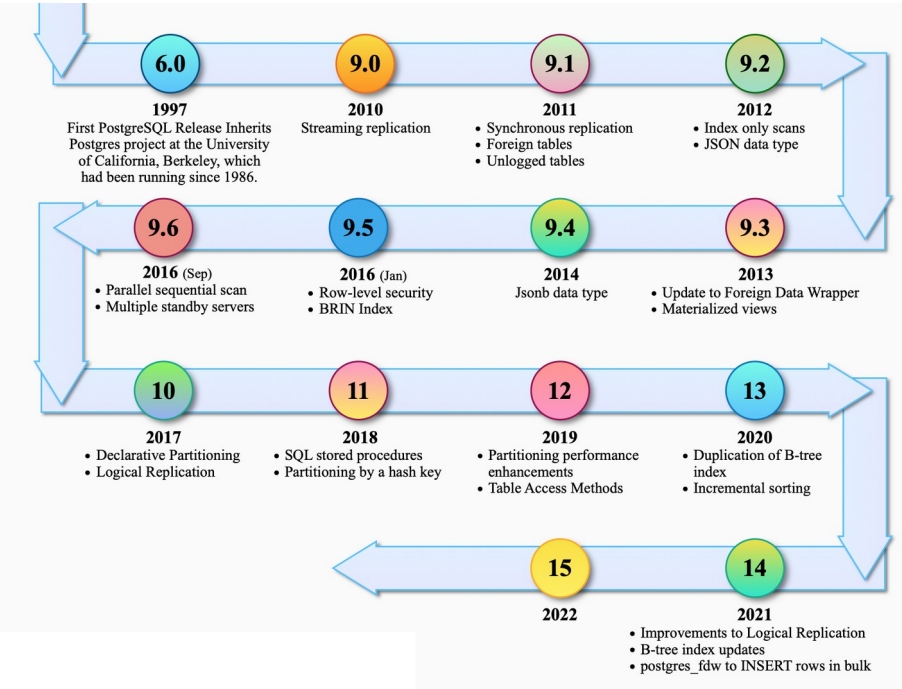
VERSIONING POLICY

- Major versions contain new features
  - Released about once a year.
  - Major version receives bug fixes and security fixes at least once every three months
  - This is called a "minor release."
- Starting with PostgreSQL 10
  - A major version is indicated by increasing the first part of the version, e.g. 10 to 11.
  - Before PostgreSQL 10, a major version was indicated by increasing either the first or second part of the version number, e.g. 9.5 to 9.6.



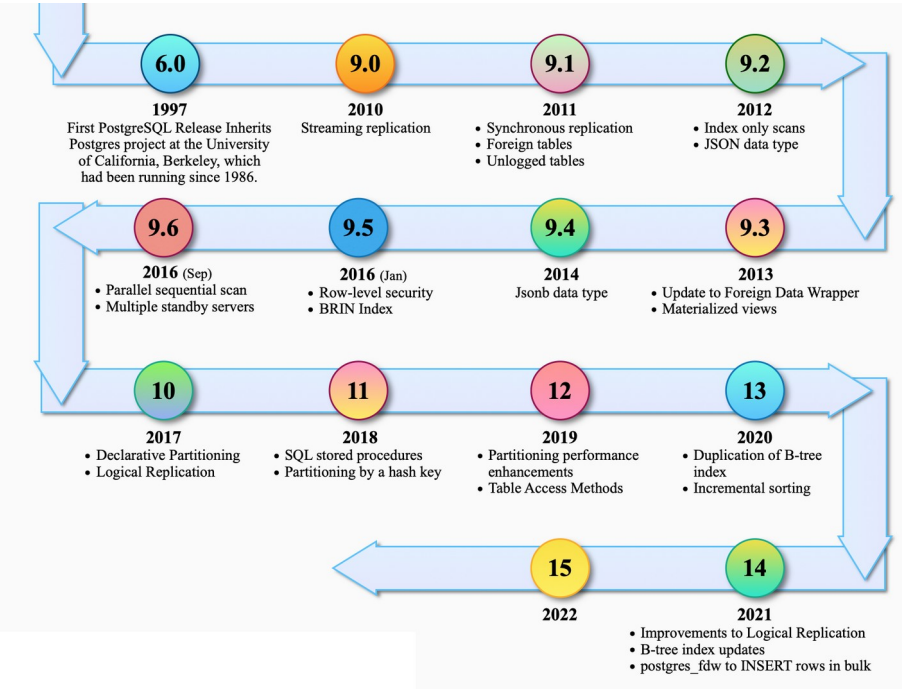
VERSIONING POLICY

- Minor releases
  - Numbered by increasing the last part of the version number.
  - Beginning with PostgreSQL 10, this is the second part of the version number, e.g. 10.0 to 10.1;
  - For older versions this is the third part of the version number, e.g. 9.5.3 to 9.5.4
- Minor releases do not alter features
  - Upgrades to minor release involve
  - Stopping the server
  - Installing the new binaries
  - Restarting the server



VERSIONING POLICY

- Upcoming minor releases
  - November 14th, 2024
  - February 13th, 2025
  - May 8th, 2025
  - August 14th, 2025
- Next major is version 17 release.
  - Planned for September 2024.
- Patches
  - Usually applied as a minor version upgrade



MAJOR RELEASES

- Major versions make complex changes
  - The contents of the data directory cannot be maintained in a backward compatible way.
  - A logical dump/reload of the database or use of the pg_upgrade application is required for major upgrades.
  - Reading the upgrading section of the major version you are planning to upgrade to is critical.
  - You can upgrade from one major version to another without upgrading to intervening versions
  - Physical backups and restores do not work for major version upgrades

PG_UPGRADE

- Allows data stored in PostgreSQL data files to be upgraded to a later major version
  - Does not require a dump and restore of data typically required for major version upgrades
  - It is not required for minor version upgrades, e.g., from 12.7 to 12.8 or from 14.1 to 14.5.
- Major releases add new features that often change the layout of the system tables
  - But the internal data storage format rarely changes.
  - pg_upgrade uses this fact to perform rapid upgrades by creating new system tables and simply reusing the old user data files.
  - f a future major release ever changes the data storage format in a way that makes the old data format unreadable, pg_upgrade will not be usable for such upgrades.
  - The PostgreSQL community attempts to avoid such situations.
- Does its best to make sure the old and new clusters are binary-compatible
- Supports upgrades from 9.2.X and later to the current major release

PG_UPGRADE STEPS

- Optionally move the old cluster
  - If there is version-specific installation directory, e.g., /opt/PostgreSQL/16, the cluster does not have to be moved
  - If the installation directory is not version-specific, e.g., /usr/local/pgsql, it is necessary to move the current install directory so it does not interfere with the new installation.
  - Once the current PostgreSQL server is shut down, it is safe to rename the PostgreSQL installation directory
- For source installs, build the new version
  - Build the new PostgreSQL source with configure flags that are compatible with the old cluster.
  - pg_upgrade will check *pg_controldata* to make sure all settings are compatible before starting the upgrade.

PG_UPGRADE STEPS

- Install the new PostgreSQL binaries
- Initialize the new PostgreSQL cluster
  - Initialize the new cluster using initdb.
  - use compatible initdb flags that match the old cluster.
  - Many prebuilt installers do this step automatically. T
  - There is no need to start the new cluster.
- Copy custom full-text search file
  - Copy any custom full text search files (dictionary, synonym, thesaurus, stop words) from the old to the new cluster.
- Adjust authentication
  - pg_upgrade will connect to the old and new servers several times, so you might want to set authentication to peer in pg_hba.conf

PG_UPGRADE STEPS

- Stop both servers
- Run pg_upgrade
  - Always run the pg_upgrade binary of the new server, not the old one.
  - pg_upgrade requires the specification of the old and new cluster's data and executable (bin) directories.
  - You can also specify user and port values, and whether you want the data files linked or cloned instead of the default copy behavior.
- Restore pg_hba.conf
  - If you modified pg_hba.conf, restore its original settings.
  - It might also be necessary to adjust other configuration files in the new cluster to match the old cluster, e.g., postgresql.conf (and any files included by it), postgresql.auto.conf.

PG_UPGRADE STEPS

- Start the new server
  - If any post-upgrade processing is required, pg_upgrade will issue warnings as it completes.
  - It will also generate script files that must be run by the administrator.
  - The script files will connect to each database that needs post-upgrade processing.
- Delete old cluster

End Module