PostgreSQL and Debian

Michael Banck <michael.banck@credativ.de>
DebConf19, Curitiba, 25 July 2019

credativ GmbH
Statistics - pkg-postgresql

Uploader of pkg-postgresql team

- Christoph B: 778
- Martin P: 251
- Adrian V: 55
- Michael B: 51
- Marco N: 16
- Michael M: 11
- Olo S: 7
- Markus W: 5
- Jean Michel: 3
- Denis B: 1

Year: 2004 to 2019
PostgreSQL - Overview

- Extensible, object-relational database system
- Created as a research project at Berkeley, community-based development since the mid-90s
- Vendor-neutral, commercial support available from multiple companies
- “Postgres Global Development Group”, core team (5 members), release team (3 members), security team, around 333 committers
- No copyright assignments, no open-core, no dual licensing
- BSD/MIT-style licence
- Many (also proprietary) forks
PostgreSQL - Release Cycle

- Yearly release-cycle, 5 years community maintenance per release
- Quarterly patch-releases, white-listed by Debian SRM
- Moved from Major.Minor to Major release numbering with 10
- Time-based feature-freeze in spring (Beta1), release in autumn (when it’s ready)
- Release cycle aligns well with Debian

Stefano Zacchioli @zacchiro · Jul 5
Every now and then I still hear comments about alleged @Debian release “unpredictability”. In view of the upcoming #Buster release, here’s a fact: from 2007 (Etch) to now the average #Debian stable release took 2 years and 5 days, with a standard deviation of less than 2 months.

The Debian Project @debian · Jul 5
Upcoming Debian 10 Buster! #Debian10Buster dlvr.it/R7rJcV
PostgreSQL - Release Cycle

- Yearly release-cycle, 5 years community maintenance per release
- Quarterly patch-releases, white-listed by Debian SRM
- Moved from Major.Minor to Major release numbering with 10
- Time-based feature-freeze in spring (Beta1), release in autumn (when it’s ready)
- Release cycle aligns well with Debian
  - Latest Postgres release in stable at time of Debian release
    - squeeze: 8.4
    - wheezy: 9.1
    - jessie: 9.4
    - stretch: 9.6
    - buster: 11
    - bullseye: 13?
PostgreSQL - Main Features

- Rock Solid
- Good and consistent coverage of the SQL-standard, useful modern SQL extensions
- Cost-based query-planner
- Large number of extensions and associated projects
- Transactional changes to database structure (DDL)
- Drivers for many programming languages
- Many different procedural languages
- Foreign-data-wrapper (FDW) for federated access to many other data sources/databases
- Packaged in Debian
New Features Between stretch and buster - 10

- Logical replication
- Improved parallel query including parallel index scans
- Native, declarative partitioning
- Client-based connection-failover
- Quorum-commit for synchronous standbys
- Easier replication configuration
- Salted Challenge Response Authentication Mechanism (SCRAM)
- Improved and production-ready hash indexes
- Multi-column correlated statistics for the query planner
New Features Between stretch and buster - 11

- Improved partitioning
  - Performance improvements
  - Default partitions
  - Hash partitioning
  - Update of tuples between partitions
  - Simple index-creation on all partitions
  - Partition-wise joins

- Improved parallel query, including parallel index creation
- Just-in-Time compilation of expressions using LLVM
- Transaction-controlled SQL procedures (stored procedures)
- `ALTER TABLE ADD COLUMN` with fast non-NULL defaults
- Covering indexes
- Basebackup checksum validation
New Features - Version 12

- Partitioning performance improvements
- Foreign-Key support for partitioned tables
- Indexing performance improvements
- `REINDEX CONCURRENTLY`
- Offline activation/deactivation of data checksums
- `COPY` allows `WHERE` conditions
- JSON path queries per SQL/JSON specification
- Pluggable storage API
- Recovery configuration merged into `postgresql.conf`
- Inlined Common Table Expressions (CTEs), no longer an optimization barrier
New Extensions / Projects Since stretch

▶ More than 40 new packages since stretch, including:
▶ bgw-replstatus - Extension reporting whether node is primary or standby
▶ jsquery - JSON query language with GIN indexing support extension
▶ londiste - Database replication for PostgreSQL based on PgQ
▶ patroni - PostgreSQL High Availability with ZooKeeper, etcd, Consul, or Kubernetes
▶ pg-checksums - Activate/deactivate/verify PostgreSQL data checksums
▶ pg-cron - Periodic jobs extension
▶ pg-dirtyread - Extension to read dead but unvacuumed tuples from a relation
▶ pg-snakeoil - ClamAV-based virus scanner extension
New Extensions / Projects Since stretch

- **pgaudit** - Audit extension
- **pgformatter** - SQL syntax beautifier
- **pglogical** - Logical replication extension
- **pgmodeler** - PostgreSQL database modeler GUI interface
- **pgq** - Generic queue extension
- **pgsphere** - Spherical data types extension
- **pgstat** - vmstat-like statistics collector
- **pldebugger** - PL/PgSQL Debugger API
- **postgresql-rum** - RUM index extension
- **prometheus-sql-exporter** - Flexible SQL Exporter for Prometheus
- **pspg** - PostgreSQL pager
- **wal2json** - Logical decoding JSON output plugin
New Extensions / Projects Since stretch

- `glom` - Database designer and user interface
- `resource-agents-paf` - PostgreSQL resource agent for Pacemaker
- `hypopg` - Hypothetical indexes extension
- `pgl-ddl-deploy` - Transparent DDL replication
- `pgsql-ogr-fdw` - Foreign Data Wrapper for OGR vector data
- `node-pg` - PostgreSQL client library for Node

In NEW and/or being worked on:

- `omnidb` - Web tool for database management
- `pgadmin4` - Web administration tool for PostgreSQL
- `pgcluu` - PostgreSQL performance monitoring and auditing tool
- `vip-manager` - Patroni virtual IP manager
Debian packages one Postgres release per version

Problem:

- There is Debian sid, buster, stretch, jessie
- There is PostgreSQL 11, 10, 9.6, 9.5, 9.4
- PostgreSQL major releases have incompatible on-disk format
- Upgrading needs both versions installed in parallel (or lots of disk space and a complicated plan)
PostgreSQL in Debian

Solution: make packages for 9.6, 10, 11 etc. co-installable

- `postgresql-common` takes care of creating database clusters in the correct locations
- Each cluster is listening on a different port
- Still, Debian **buster** has 11 only (**stretch** has 9.6, **jessie** had 9.4)
  - Users want to try new PostgreSQL versions
  - Users might run Ubuntu and not Debian
  - Users want to upgrade Debian without upgrading PostgreSQL
Apt repository hosting packages of all upstream supported PostgreSQL versions built for all Debian (and Ubuntu) releases

Somewhat a superset of what backports would do

7 PostgreSQL releases:
9.4 9.5 9.6 10 11 (12) (13)

8 Debian and Ubuntu releases:
jessie stretch buster sid xenial bionic cosmic disco

3 architectures: amd64 i386 ppc64el

7x8x3 = 168 targets

Automatic builds via Jenkins and jenkins-debian-glue

Hosted at https://pgdgbuild.dus.dg-i.net/ (Thanks to DG-i)

Running autopkgtests as well
PostgreSQL Package Layout

- Allow multiple major versions to be co-installed
- Instances ("Clusters") are identified by version and cluster name
- Every new instance gets an incremented port, starting from 5432
- Server package is `postgresql-<version>`
- Client package is `postgresql-client-<version>`
- Database directories are in `/var/lib/postgresql/<version>/<name>`
- Logile is `/var/log/postgresql/postgresql-<version>-<name>.log`
- Configuration files are in `/etc/postgresql/<version>/<name>`
  - `postgresql.conf`, `pg_hba.conf`, `pg_ident.conf`
  - `pg_ctl.conf`, `start.conf` (Debian-specific)
postgresql-common Commands

- **pg_wrapper**: Wrapper for PostgreSQL client binaries allowing to select instance
  - `psql --cluster 9.4/main`
  - `PGCLUSTER=9.4/main psql`
  - `~/.postgresqlrc`
  - `/etc/postgresql-common/user_clusters`

- **pg_conftool**: Read and edit parameters in configuration files

- **pg_*cluster**: Debian-specific cluster administration commands
  - `pg_createcluster <version> <name>`

- **pg_createcluster** behaviour can be configured in
  `/etc/postgresql-common/createcluster.conf`
  - `create_main_cluster, start_conf, data_directory, xlogdir, initdb_options`
  - Additional config parameters are copied into `postgresql.conf`
Some Recent External Projects

- `pg_checksums` - Data checksumming tool
- `patroni` - PostgreSQL high availability
- `elephant-shed` - PostgreSQL appliance
- `pgapi` - REST API for PostgreSQL instances
pg_checksums

- Data checksums available in all supported version, not on by default
- PostgreSQL 11 ships `pg_verify_checksums`
- PostgreSQL 12 renamed it to `pg_checksums`, offline (de-)activation
- `https://github.com/credativ/pg_checksums/`
- Fork/Merge of `pg_verify_checksums/pg_checksums`
- Additional features:
  - Online verification of checksums
  - Compatibility with all supported Postgres versions
  - Progress reporting and I/O-throttling
Patroni

- https://github.com/zalando/patroni/
- Configures and bootstraps Postgres instances and replication
  - Leverages `etcd` (or `consul`, or `zookeeper`) as Distributed Configuration Store (DCS)
    - Raft-algorithm for leader election
    - Leader time-to-live
    - Split-Brain avoidance
    - Postgres cluster-status
    - Shared configuration state
- REST-API allows for monitoring and administration
- `patronictl` end-user program allows for status display, switchover, configuration changes
- `vip-manager` package (in NEW) configures a virtual IP on the primary, based on DCS state

Michael Banck <michael.banck@credativ.de>
By default, Patroni is oblivious to Debian’s `postgresql-common`.

Debian package ships `pg_createconfig.patroni` utility:

- `'pg_createconfig.patroni 11 main'` creates a Patroni configuration for instance `11/main`.
- `'systemctl start patroni@11-main'` then starts Patroni which bootstraps/clones instance.

By default, things are where Debian expects them:

- Data directory is `/var/lib/postgresql/<version>/_<name>/`
- Configuration is in `/etc/postgresql/<version>/_<name>/`
- Logfile is `/var/log/postgresql/postgresql-<version>-<name>/`

Configuration interaction with Patroni is a bit hairy; `postgresql.conf` is renamed to `postgresql.base.conf` and included from (Patroni-managed) `postgresql.conf`.

Integration with `vip-manager` forthcoming once it hits the archive.

Here’s a [proof-of-concept Ansible playbook](https://github.com/credativ/ansible-playbook-patroni-debian) for a 3-node Debian Patroni cluster.
Elephant Shed - PostgreSQL Appliance

- https://elephant-shed.io/
- https://github.com/credativ/elephant-shed/
- Integrates and bundles proven projects for easy maintenance of PostgreSQL
  - pgAdmin4 - Web-Based Administration of PostgreSQL
  - Grafana - Monitoring Dashboards
  - Prometheus - Monitoring
  - pgBadger - Logfile Analysis
  - pgBackRest - Backups
  - Cockpit - System and Services Administration
  - Shell In A Box - Web-Based Terminal Emulator
- Implemented via `postgresql-common` and `systemd` glue
Elephant Shed - Dashboard

PostgreSQL Appliance Dashboard

PostgreSQL Cluster

Cluster | Port | Data directory | Archiving | Full Backup | Incr Backup
---|---|---|---|---|---
9.6/main | 5432 | /mnt/pgdata/9.6/main | | |
10/main | 5433 | /mnt/pgdata/10/main | | |
10/test | 5434 | /mnt/pgdata/10/test | | |
Elephant Shed - Grafana Dashboard
pgapi

- Technology preview, work-in-progress
- [https://github.com/credativ/pgapi](https://github.com/credativ/pgapi)
- Python-based REST API for PostgreSQL instances
- Allows to manage local or remote PostgreSQL instances
- Get a list of all clusters:
  - curl -X GET http://127.0.0.1:15432/cluster/
- Change port of the 11/main instance:
  - curl -i -X PATCH -H "Content-Type: application/json" -d '{"config": {"port": 6432}}' http://127.0.0.1:15432/cluster/11/main
- Restart the 11/main instance:
  - curl -i -X PATCH -H "Content-Type: application/json" -d '{"state": "restart"}' http://127.0.0.1:15432/cluster/11/main
- Integration into Elephant-Shed ongoing
Contact

- Michael Banck <michael.banck@credativ.de>
- https://www.credativ.de
- https://www.credativ.de/postgresql
- https://www.credativ.de/jobs
- https://elephant-shed.io/
- https://github.com/credativ/