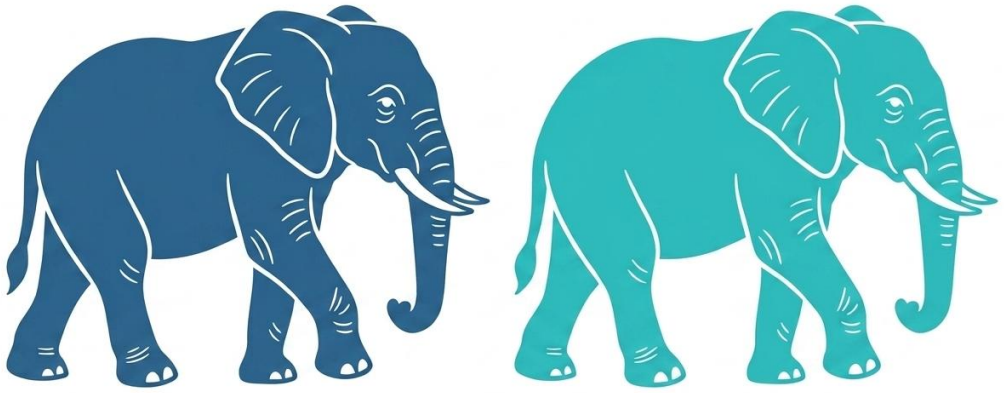

Delegated Database Cloning for PostgreSQL

Self-service database clones for CI/CD - without superuser

Why pgclonedb?

Built to simplify CI/CD testing: load the test data **once** into a master, then clone it on demand for every pipeline run.

Testing without pgclonedb

- Reload the full dataset for every test run
- Slow setup - pipelines serialise on the data
- DBA / superuser needed to create databases
- Parallel runs collide on shared data
- Leftover state drifts between runs

Testing with pgclonedb

- Data loaded once into a master template
- Clone in seconds, drop when finished
- Non-privileged users self-serve - no superuser
- Many isolated clones run in parallel
- Every clone starts from the same clean state

Getting Started

Requirements

- PostgreSQL 13 or newer
- dblink (installed automatically via CASCADE)
- Superuser can loopback-connect over the Unix socket
- Pure SQL extension - no external binaries

Install & onboard a team

```
-- as superuser
CREATE EXTENSION pgclonedb CASCADE;

-- onboard an application admin
CREATE ROLE blue_db_admin LOGIN PASSWORD 'SecurePass!';
GRANT pgclonedb_user TO blue_db_admin;

-- that's it - blue_db_admin can now clone databases
-- starting with blue / blue_* on its own
```

Multi-Tenancy & Cloning Workflow

Users connect to the **postgres** database and call functions as members of **pgclonedb_user**.

The role name sets the scope: **<app>_db_admin** manages only its own databases.

Role	May manage
blue_db_admin	blue, blue_*
red_db_admin	red, red_*

Tenants are isolated at the database level - **blue_db_admin** cannot touch **red**. One cluster, many teams.

Typical workflow

```
-- 1) prepare the master once (load test data into 'blue')
SELECT pgclonedb.close_database('blue');

-- 2) clone on demand, in parallel
SELECT pgclonedb.create_database('blue_clone1', 'blue');
SELECT pgclonedb.create_database('blue_clone2', 'blue');

-- 3) discard when the run is done
SELECT pgclonedb.drop_database('blue_clone1');

-- master stays clean & ready to re-clone
```

More Than Cloning

A small, safe API - role functions ship **disabled by default** and are enabled per cluster by the superuser.

Area	Functions	Default
Databases	create_database, drop_database, open_database, close_database	enabled
Schemas	create_schema, drop_schema	disabled
Roles	create_role, drop_role, reset_password, grant_role, revoke_role	disabled
Audit & info	get_audit_log, help, show_config	enabled
Superuser admin	configure, enable_function, disable_function, add_post_hook	n/a

Every call is recorded in an audit log; `pgclonedb.help()` is an in-session cheat sheet showing what is currently enabled.

Post-Operation Hooks

Run SQL automatically after a clone or drop.

- Fires on post_create_database / post_drop_database
- Connects to a target database via dblink
- Placeholders substituted at run time:
  - **{dbname}, {template}, {owner}**
- Values restricted to [A-Za-z0-9_] (injection-safe)
- Execution is recorded in the audit log
- Superuser-only registration

Example: keep a DB registry in sync

```
SELECT pgclonedb.add_post_hook(
  event        => 'post_create_database',
  target_db    => 'ops',
  target_schema=> 'catalog',
  sql_template => $$
    INSERT INTO db_registry
      (dbname, app, team, contact)
    SELECT '{dbname}', app, team, contact
      FROM db_registry
     WHERE dbname = '{template}'
  $$);

-- new clone inherits the master's
-- registry entry automatically
```

Security by Design

Layer	Mechanism
Privilege escalation	SECURITY DEFINER functions owned by the installing superuser
Group membership	Caller must belong to pgclonedb_user
Scope isolation	App prefix enforced on database / template targets
SQL injection	All identifiers via quote_ident / format(%I), literals via %L
Role guards	No superuser, CREATEROLE/DB, pg_* or other apps' *_db_admin roles
Internal tables	audit_log / config / hooks reachable only through the API
Audit trail	Every call logged - error paths survive rollback via dblink

Disk-space check Before cloning, create_database compares the template size against free space and blocks the clone if too little would remain.

Configurable margin: `default 10 GB - 0 = must just fit - -1 = skip check`

Thank you!

pgclonedb - delegated database cloning for PostgreSQL

https://github.com/raphideb/pgclonedb

My email: raphi@crashdump.ch