autobase

How the PostgreSQL Stack Works

— and where Autobase fits in

In plain language: what running PostgreSQL in production really costs and what changes with Autobase — for those who are deeply technical and those who have never touched a database.

PART 1 Architecture **PART 2** Economics

Main idea

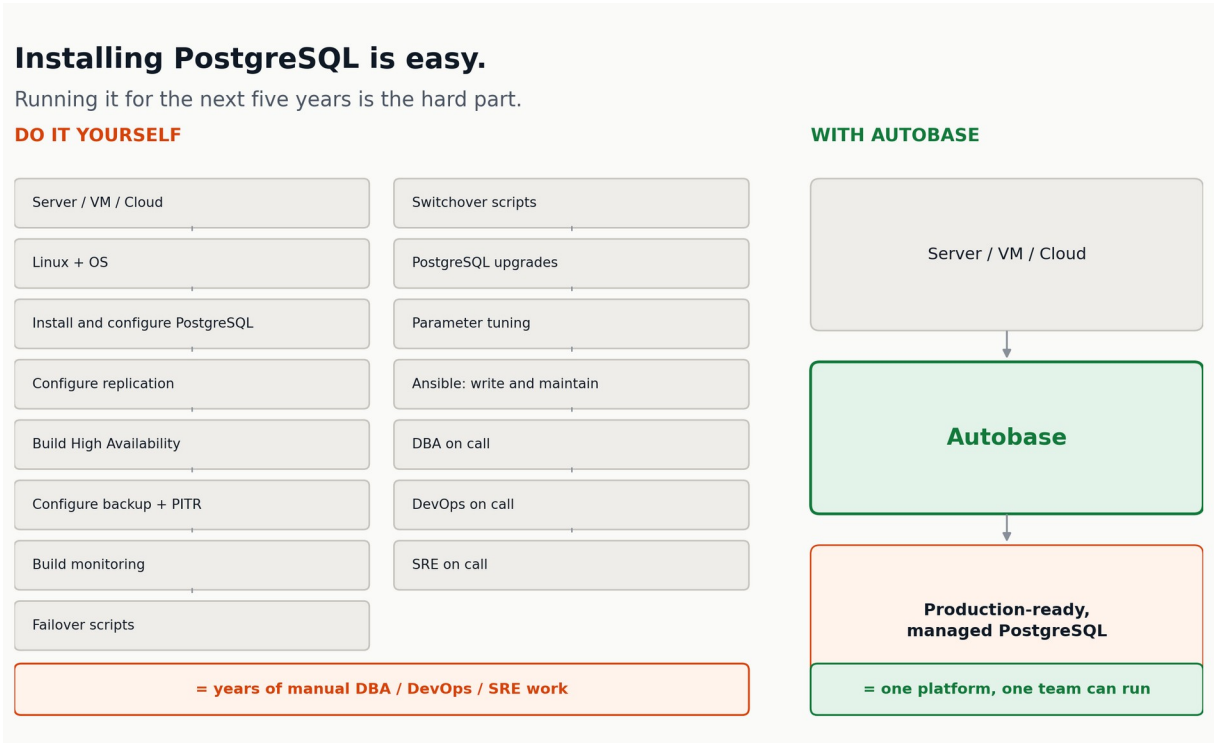
People say “we run PostgreSQL” as if it were one thing you install once and you're done. It isn't. A real production PostgreSQL database — one a bank, SaaS company, or marketplace can rely on — is roughly nine separate layers stacked on top of one another. Weaken one layer and it doesn't matter how good PostgreSQL itself is: everything built above inherits that weakness.

Think of it like a cake. Nobody simply “eats a cake” — someone bakes every layer, stacks them, and only then tastes the frosting on top. The server is the oven, PostgreSQL is one of the middle layers, and the application your users work with is the frosting: what everyone sees, on top of everything nobody thinks about.

This document walks through the whole stack in plain language, shows which layers Autobase already handles today, and then translates it into real numbers.

Installing is easy. Operating it for five years isn't.

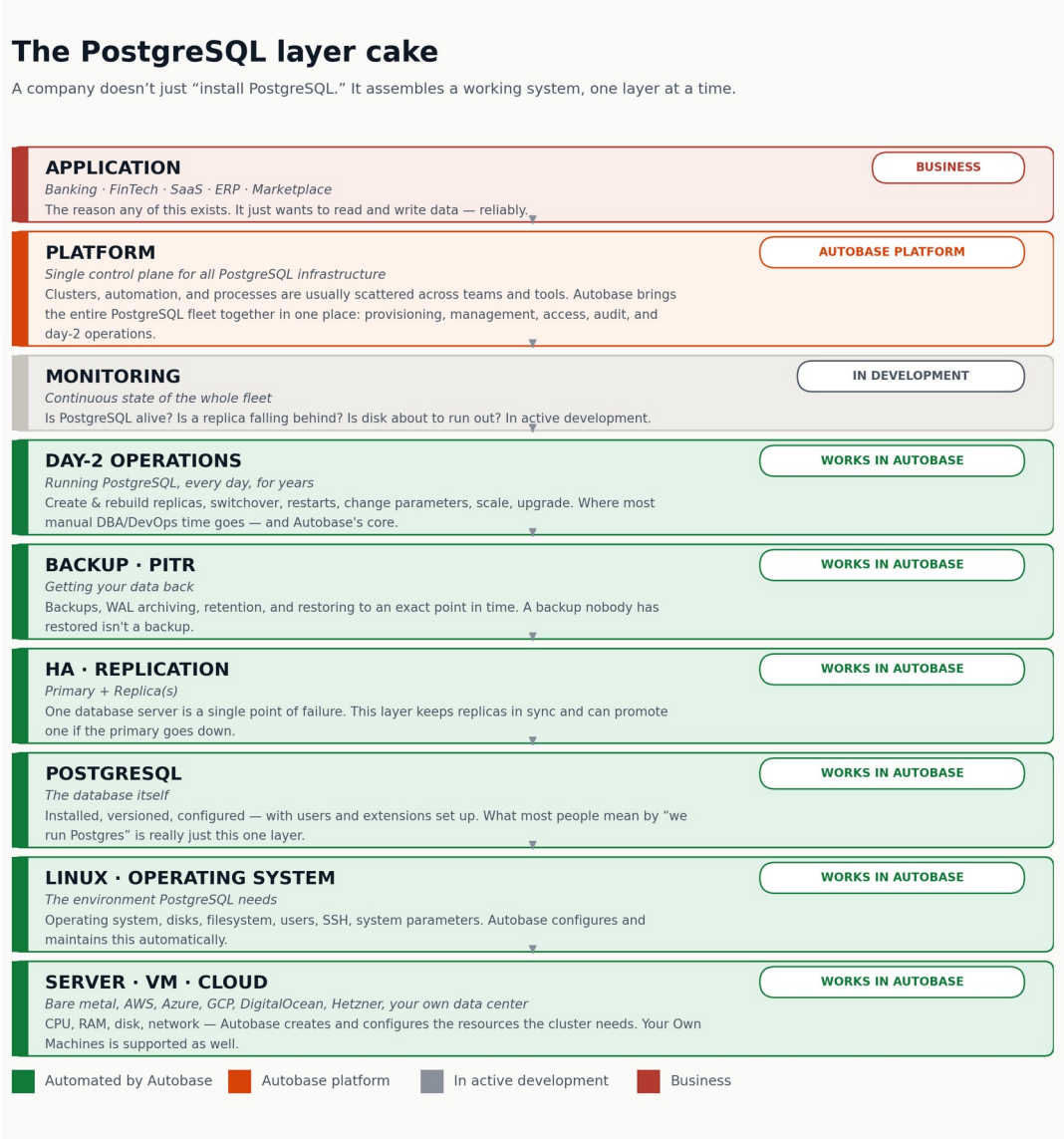
You can install PostgreSQL on a server in ten minutes — that was never the hard part. The hard part is everything after that: keeping the database alive when a server fails, restoring it correctly, spotting problems before users do — and doing it all again every time a cluster is added, an engineer leaves, or PostgreSQL needs an upgrade. For years.



Left — what a team usually has to build and maintain piece by piece to operate PostgreSQL safely. Right — the same outcome, but through one platform.

The entire PostgreSQL stack, layer by layer

Every layer between “the server exists” and “our application works,” in the order it appears. Green — Autobase already automates it today. Orange — the Autobase platform itself. Gray — in active development. Red — the business itself, the reason all of this exists.



Each layer in plain language

1 · Server / VM / Cloud WORKS IN AUTOBASE

A physical or virtual machine everything else runs on — bare metal, AWS, Azure, GCP, DigitalOcean, Hetzner, or the company's own data center. Your Own Machines is also supported. Autobase creates and configures the resources the cluster needs — CPU, memory, disk, network — so this layer does not need to be prepared manually.

2 · Linux / OS WORKS IN AUTOBASE

The operating system PostgreSQL needs: disks and filesystem, users and SSH, system and kernel parameters. Autobase configures and maintains this layer automatically — this is where engineers used to lose the most time.

3 · PostgreSQL WORKS IN AUTOBASE

What most people actually mean by “we run Postgres”: choosing a version, installing packages, creating the data directory, setting parameters, creating users, enabling extensions. The layer that gets the most attention — and is actually just one of nine.

4 · HA / Replication WORKS IN AUTOBASE

A single PostgreSQL server is a single point of failure. This layer adds replicas and manages their role and state. It also answers the hard questions: if the primary fails, which replica becomes the new primary, how does the application find it, and how does the old one safely rejoin the cluster?

5 · Backup / PITR WORKS IN AUTOBASE

Backups, WAL archiving, retention and — most importantly — point-in-time recovery. A backup nobody has tried to restore is not a backup. The real question: “if the database fails right now, can we restore it to exactly one minute ago?”

6 · Day-2 Operations WORKS IN AUTOBASE — CORE OF THE PRODUCT

The bulk of the ongoing work, and where Autobase puts most of its effort. Once PostgreSQL is running, someone has to keep managing it: create and rebuild replicas, perform switchovers, restarts, change parameters, scale, upgrade versions, and manage extensions. This is the largest source of manual DBA/DevOps work in most companies — and the center of what Autobase already automates today.

7 · Monitoring IN DEVELOPMENT

Is PostgreSQL definitely alive? Is the replica keeping up? How much disk space is left? Without this layer, the company usually learns about problems from users instead of its own systems. In active development.

8 · Platform AUTOBASE PLATFORM

In most companies PostgreSQL is a collection of separate clusters, scripts, and procedures. Autobase adds what's usually missing: a single control plane for the entire PostgreSQL infrastructure — one place to create, manage, and operate the whole database fleet.

9 · Application BUSINESS

A bank, SaaS product, marketplace, ERP system. It sends a query and expects a reliable answer — and should not need to know or care about the eight layers below it. That is the entire point of the other eight.

Where things stand today

Today Autobase automates six operational layers and combines them into a single management platform. Monitoring is in active development. The top layer, the application, always remains the responsibility of the business itself.

> **PostgreSQL is not a product you install once and forget. Operating it requires many interconnected components, processes, and continuous management. Autobase's job is to bring these layers together into a single system and gradually turn the most labor-intensive manual processes into a managed platform.**

The next section puts numbers on all of it: what running PostgreSQL costs today, and what changes with Autobase.

PART 2

Economics

PostgreSQL itself is free — it is open source. The money goes to everything in Part 1: the layers of work someone has to build and maintain around it. This section translates that work into real numbers — and shows what changes when Autobase takes it on.

Where the money really goes

Operating PostgreSQL always has three cost layers: infrastructure, the software/services that support it, and engineering time for all of it. Autobase does not eliminate infrastructure — but it significantly reduces and standardizes the other two layers.

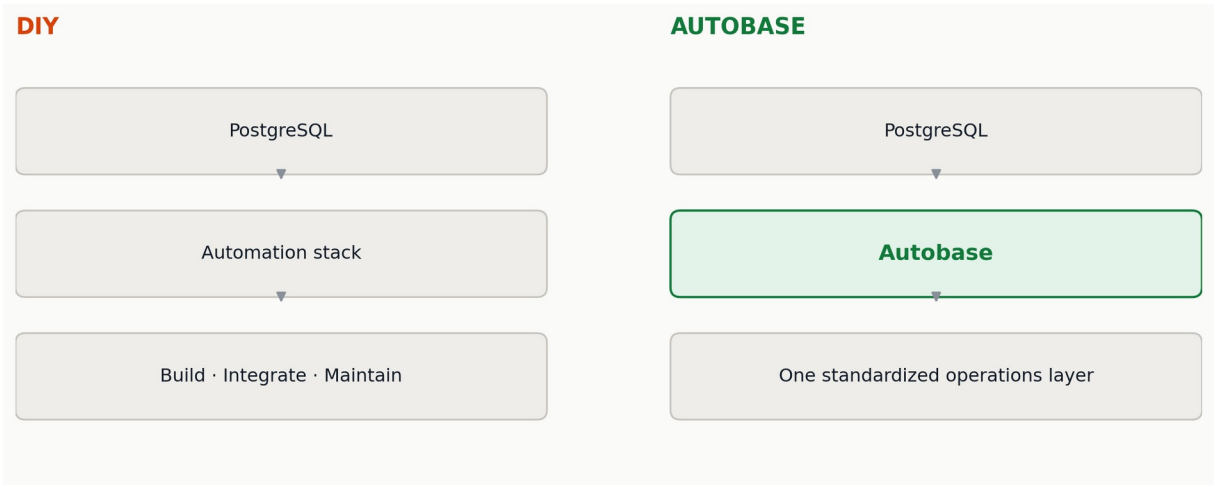
Cost layer	In-house (DIY)	With Autobase
Infrastructure	Compute, storage, backups, network	Same self-hosted infrastructure
Software / Services	Terraform / Ansible + Patroni + pgBackRest + monitoring + CI/CD + scripts + runbooks	Autobase + existing observability
Engineering	Build, integrate, test, and maintain the operations layer	Use a ready-made operations layer
Scaling	More PostgreSQL → more platform work	More PostgreSQL → same platform

For the calculations below, we use a model fully-loaded cost of $8,500 per month for an infrastructure/data engineer. Actual cost depends on region, company, and compensation structure.

~$8.5K model cost of one engineer per month

Where Autobase actually saves time

Both paths start with PostgreSQL and end with PostgreSQL in production. The difference is everything in between: an entire automation stack that must be built and maintained forever, versus one standardized operations layer that is already ready.



Example of one real deployment

Deploying one production installation — VM, backup storage, OS setup, and PostgreSQL configuration.

DIY	Autobase	Result
~4–5 engineering hours	~10–15 minutes	**≈90%+ less manual work**

Approvals, access, and merge requests can stretch the timeline to days or weeks. Autobase's self-service approach reduces manual hours and the waiting between these steps.

How much Autobase Premium costs — and when it pays for itself

Premium costs $4,096 per month for an unlimited number of clusters — a fixed price. At a model engineer cost of ~$8,500/month, break-even requires:

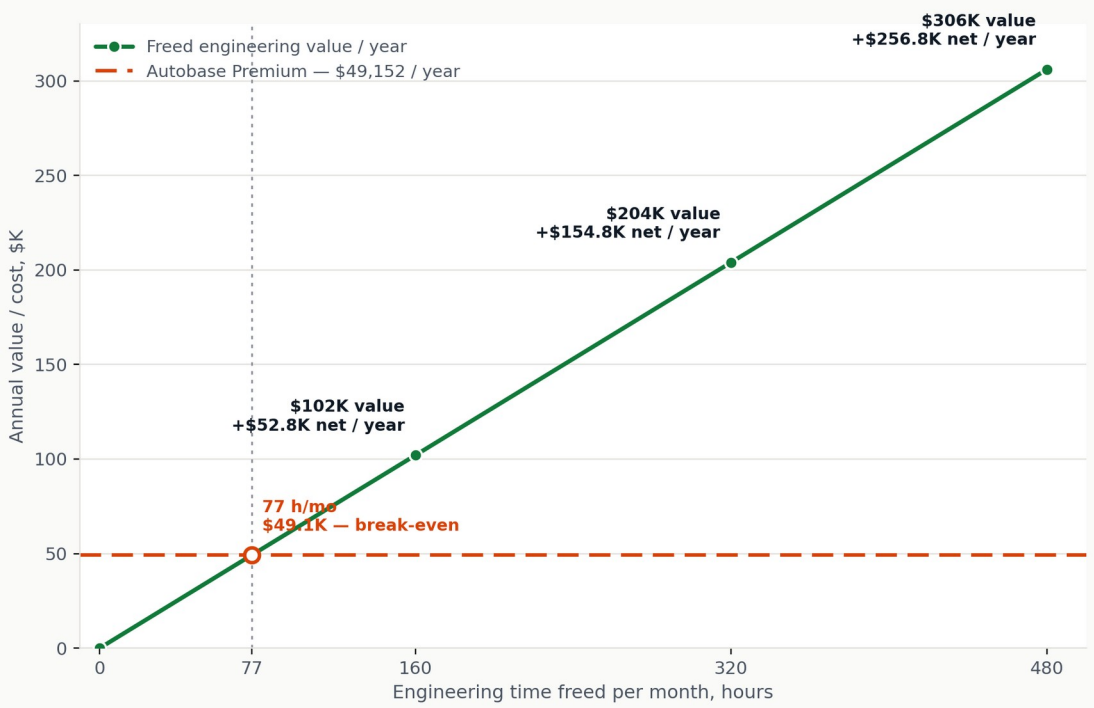
~48% of one engineer's time per month

≈77 h engineering hours per month to break even

Everything above these 77 hours per month already works in the company's favor:

Time freed	Value / mo	Premium cost	Net benefit / year
77 h/mo	~$4.1K	$4.1K	**Break-even**
160 h/mo	~$8.5K	$4.1K	**~$53K / year**
320 h/mo	~$17K	$4.1K	**~$155K / year**
480 h/mo	~$25.5K	$4.1K	**~$257K / year**

The freed time can be directed to tasks where engineering expertise is actually needed.

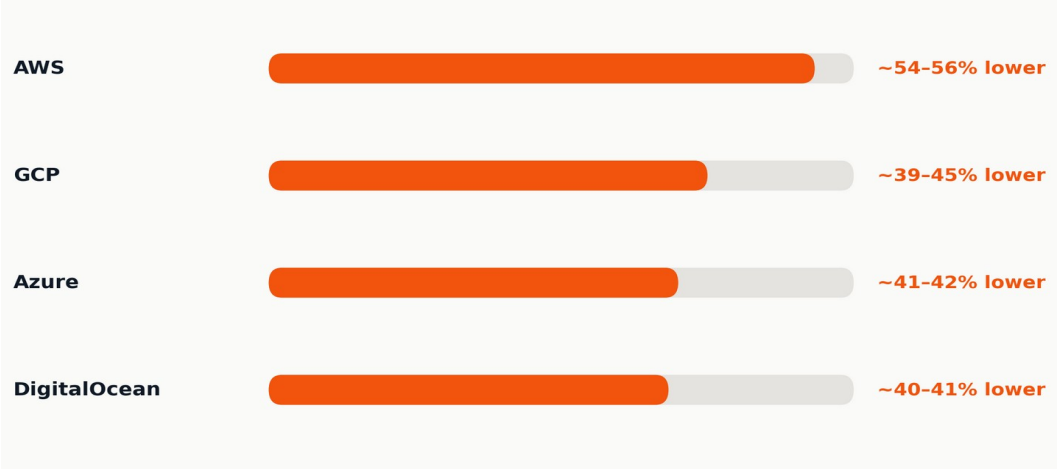


Less engineering time. Faster launch. Predictable platform cost.

Compared with managed PostgreSQL: infrastructure savings

With self-hosted PostgreSQL, compute and storage can be cheaper than a managed service for comparable resources:

~39–56% lower infrastructure cost vs. managed PostgreSQL



Infrastructure only; this is the effect of the self-hosted model. Actual figures depend on region, HA topology, storage, IOPS, backups, traffic, and provider discounts.

Summary

WITHOUT AUTOBASE

The team builds and maintains the PostgreSQL operations layer itself for years — infrastructure, software/services, and internal automation, spending engineering time on it.

WITH AUTOBASE

Same self-hosted infrastructure, but a standardized operations layer instead of a custom set of automation tools — and noticeably less engineering time on routine work.

> **More PostgreSQL no longer means proportionally more engineering work.**

Main takeaway

Autobase turns PostgreSQL operations from a collection of fragmented tools and manual procedures into a single platform. This makes it possible to launch clusters in minutes, standardize day-to-day operations, and manage hundreds and thousands of clusters without proportional team growth, saving significant engineering hours and reducing costs compared with managed database services.