

GUIDES AND REPORTS

Updating a 50 terabyte PostgreSQL database

By Reinier Haasjes, Senior System Administrator, Adyen

March 14th, 2018 - 4 minutes



Guides and reportsDive into tech



Our database setup at Adyen is unique for a few reasons. We currently process 5,000+ PostgreSQL transactions per second across multiple clusters. In addition, as a payments service provider dealing with sensitive financial data, redundancy is even more critical than normal, and downtime is unacceptable.

On top of that, we're scaling our infrastructure at a rapid rate. In 2015, our database was under 10 terabytes. But our latest upgrade was 50 terabytes, and still growing fast. With global digital payments volumes projected to reach over 700 billion annual transactions by 2020, we need to think in terms of much bigger volumes than we currently have even now.

In this post I'll look at Adyen's evolving approach to updating our PostgreSQL database, against the requirements and challenges posed by redundancy, uptime, and scalability.

Choosing our technology

My colleague Michiel has written previously on how we have made a conscious decision to be "ruthless" when choosing a solution. We make careful decisions that give significant weight to the reliability of a solution and the maturity of the ecosystem (support, community, documentation and so on) that surrounds it, as well as the functionality. We built our database stack with PostgreSQL, as it provides consistency, isolation, and reliability we need. In addition, it is open source, and has an active ecosystem including multiple support and consultancy companies. Finally, as we work with sensitive financial data, having a transactional database was key as it ensures we don't lose track of records.

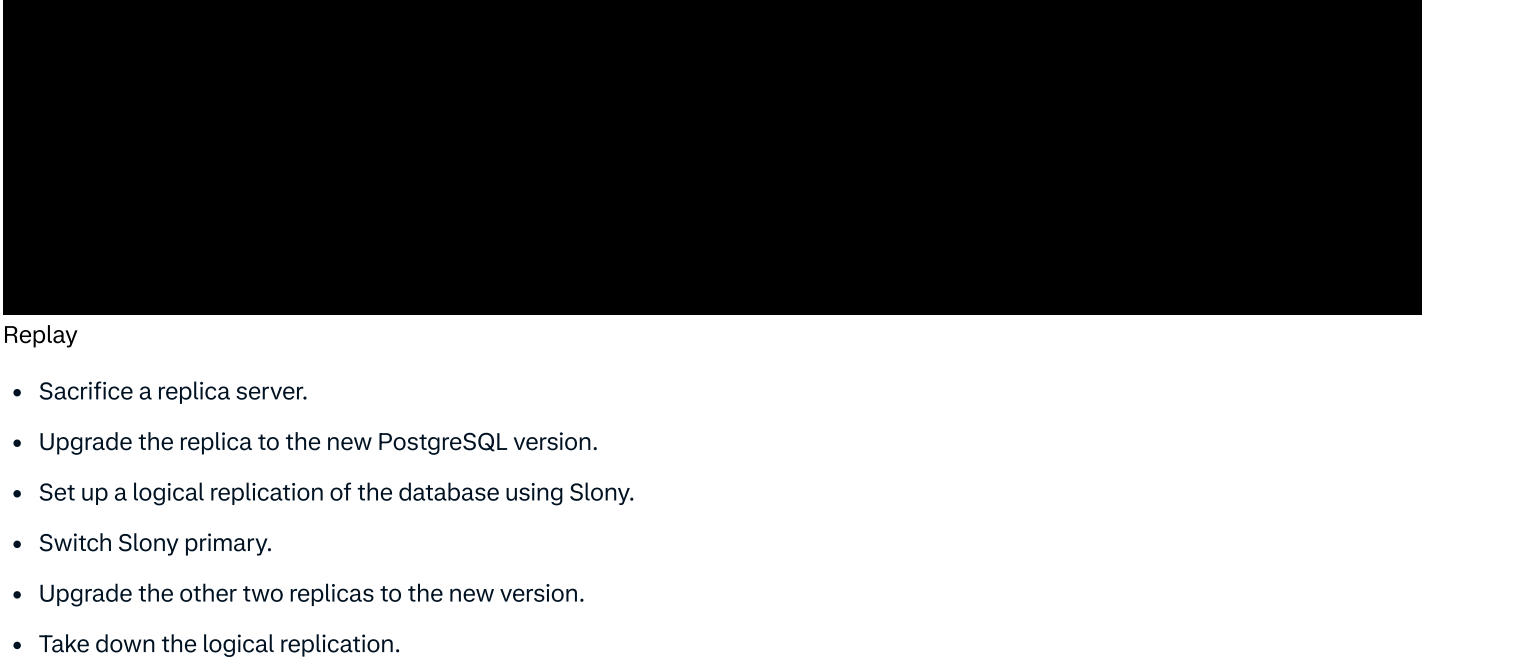
A look at our database architecture

Our PostgreSQL database clusters consist of one primary and at least three replica servers, spread over multiple data centers. The database servers are dual socket machines with 768 GB of RAM. Each server connects to its own shared storage device with fiber channel or iSCSI, and has a raw capacity of 150+ terabytes of SSDs and average compression ratio of 1:6.

One other detail to note is that we built our software architecture in such a way that we can stop traffic to our PostgreSQL databases, queue the transactions, and run a PostgreSQL update without affecting payments acceptance.

Previous approach: Updating 10 terabytes

As recently as 2015, our database needs were much simpler. Our approach to upgrading to new PostgreSQL versions followed these steps:



Replay

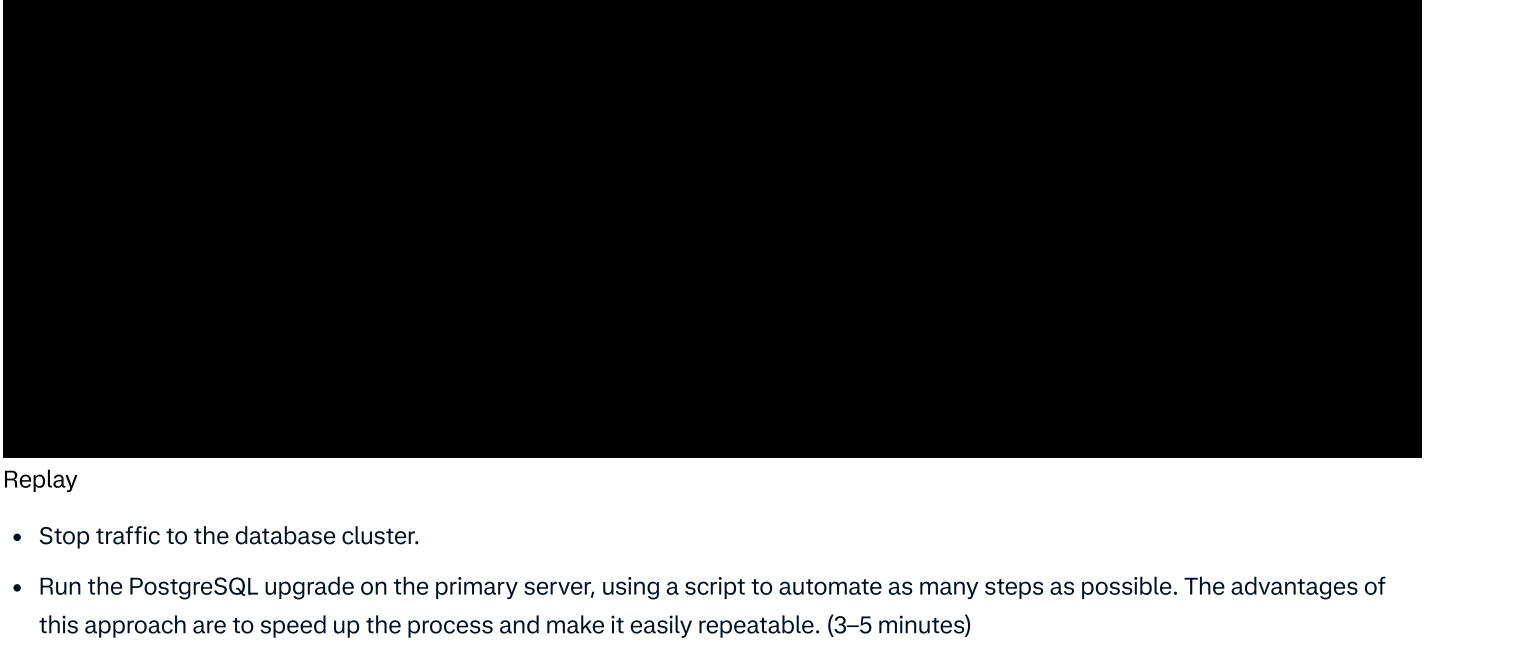
- Sacrifice a replica server.
- Upgrade the replica to the new PostgreSQL version.
- Set up a logical replication of the database using Slony.
- Switch Slony primary.
- Upgrade the other two replicas to the new version.
- Take down the logical replication.
- Upgrade the last replica to the new version.

At 10 terabytes it took a few days before we got the new server up and running, and this process was becoming progressively more time-consuming. Furthermore, we could foresee a potential impact on redundancy in the long term, as replicas were sacrificed at certain points in the process. With a 50-terabyte database on the horizon, the length of time required and the potential risks to redundancy started to become unacceptable.

New approach: Updating 50 terabytes to 9.6

By the time 9.6 was to be released, we needed a smarter, more scalable solution. In reality, our options were limited. Continuing with logical replication was impossible for the reasons I outlined above. The only other option from a PostgreSQL perspective was to run a pg_upgrade on the primary. However, this would have meant that you would have only one primary, which is also an unacceptable situation from a redundancy perspective.

As PostgreSQL options were not suitable for the next upgrade, in parallel we considered other possibilities. Our storage devices were able to make instant snapshots and also make them available on remote storage devices over the network, within a much smaller timeframe. We started combining the options, and ultimately, our solution involved the following steps:



Replay

- Stop traffic to the database cluster.
- Run the PostgreSQL upgrade on the primary server, using a script to automate as many steps as possible. The advantages of this approach are to speed up the process and make it easily repeatable. (3–5 minutes)
- Stop PostgreSQL on the upgraded primary and create a snapshot of the volumes. (Up to 10 minutes)
- Copy the snapshot of the upgraded primary to the storage device of the replicas. (2–5 minutes)
- Restart the primary.
- Import the snapshot to the replica server, mount the (already upgraded) primary volume on the replica server.
- Reconnect the replica server to the primary, by simply putting in the correct recovery.conf file.
- Only when there are at least two replicas online, connected, and up-to-date, we allow access to the cluster again. So there is never a moment when access to the cluster is open with less than two replicas online.
- Start the software again.

This process then needs to be repeated across all clusters. Altogether, the process takes around a day including preparation work.

This approach has a number of advantages:

- It's relatively easy once it is set up, as many of the steps can be automated, following some test runs and writing the script.
- It's very fast, as subsequent snapshots are only a diff from the original. In fact, to copy the snapshot takes only a few minutes.
- It meets our requirements for redundancy. Before we start the upgrades we make a snapshot of all the servers. If the upgrade would fail at some point, we can simply revert to the old snapshots. This means we may lose an hour of time, but we would not lose any data or compromise redundancy.

Looking toward PostgreSQL 10.0+

The beauty of our new approach is that it can continue to scale indefinitely –horizontally, by adding more clusters, and vertically, by using storage devices with increased capacity.

In fact, we have doubled the number of live clusters in the past year, and our biggest cluster has already grown to 74 terabytes. And later this year, we are looking forward to using this approach to update to PostgreSQL 10.3+, without any of the headaches associated with our previous approach.

Technical careers at Adyen

We are on the lookout for talented engineers and technical people to help us build the infrastructure of global commerce!

Check out developer vacancies

Developer newsletter

Get updated on new blog posts and other developer news.

Subscribe now

Blogs

B2B embedded finance: How this \$185 billion opportunity is transforming SaaS platforms...

November 13th, 2025 - 9 minutes

Blogs

Why card presence is no longer a key security metric

November 12th, 2025 - 6 minutes

Tech stories

Dynamic Identification: building the next layer of trust in global commerce

November 11th, 2025 - 8 minutes

See All

Fresh insights, straight to your inbox

Make the right decisions with the latest insights and advice on business growth and payments innovation.

First name *

Last name *

Business Email *

Subscribe

1

By submitting your information you confirm that you have read Adyen's [Privacy Policy](#) and agree to the use of your data in all Adyen communications.

First name *

Last name *

Business Email *

Subscribe

1

By submitting your information you confirm that you have read Adyen's [Privacy Policy](#) and agree to the use of your data in all Adyen communications.

First name *

Last name *

Business Email *

Subscribe

1

By submitting your information you confirm that you have read Adyen's [Privacy Policy](#) and agree to the use of your data in all Adyen communications.

First name *

Last name *

Business Email *

Subscribe

1

By submitting your information you confirm that you have read Adyen's [Privacy Policy](#) and agree to the use of your data in all Adyen communications.

First name *

Last name *

Business Email *

Subscribe

1

By submitting your information you confirm that you have read Adyen's [Privacy Policy](#) and agree to the use of your data in all Adyen communications.

First name *

Last name *

Business Email *

Subscribe

1

By submitting your information you confirm that you have read Adyen's [Privacy Policy](#) and agree to the use of your data in all Adyen communications.

First name *

Last name *

Business Email *

Subscribe

1

By submitting your information you confirm that you have read Adyen's [Privacy Policy](#) and agree to the use of your data in all Adyen communications.

First name *

Last name *

Business Email *

Subscribe

1

By submitting your information you confirm that you have read Adyen's [Privacy Policy](#) and agree to the use of your data in all Adyen communications.

First name *

Last name *

Business Email *

Subscribe

1

By submitting your information you confirm that you have read Adyen's [Privacy Policy](#) and agree to the use of your data in all Adyen communications.