PlanetScale

Database downtime:

why it happens, what it costs you, and how to prevent it

PlanetScale.com

Table of Contents

Database downtime:
why it happens, what it costs you,
and how to prevent it

Database downtime:

why it happens, what it costs you, and how to prevent it

An hour of downtime can cost your company millions of dollars. And yet, 80% of organizations report experiencing an outage in the past 3 years, the majority of which can be caused by software issues. Beyond lost business and lost trust, downtime focuses your most precious resource – your people – on fixing broken tooling instead of innovating. 88% of companies report that common database errors can take more than an hour to resolve, an 8% increase from last year; all lost time for your organization.

This whitepaper will survey how much downtime is costing your company, the common causes of outages, and powerful solutions for avoiding them, so your team can stay focused and productive.

What is downtime costing you?

Downtime is expensive, but it might be more costly than you think:

$9,000

A minute of downtime can cost an organization $9,000, with the average data center outage costing a whopping $740,000.

$100,000

Over 60% of outages end up costing more than $100,000, and the number is increasing quickly.

$1M+

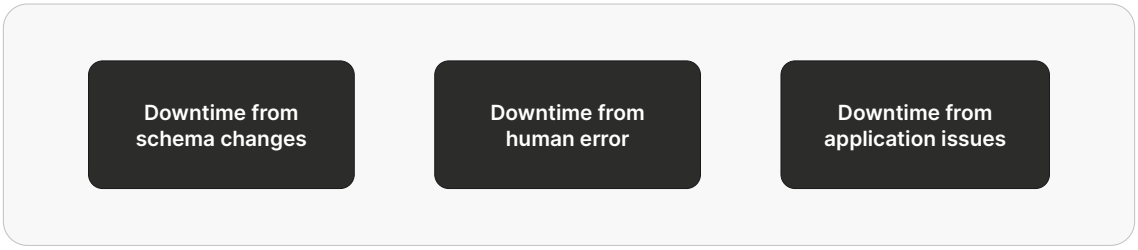
15% of outages cost more than $1M to their organizations.

But the cost of an outage is more than business loss, it's also about time: almost half of outages cause a major loss in productivity, and half of them were also reported to cause employees to work overtime. 40% of downtime led to a degree of reputational damage to the company, and 1 in 4 respondents said that they won't do business with a company following a data breach.

Common causes of downtime and how to avoid them

If we had 100% knowledge coverage of what caused downtime, it would never happen. Having said that, there are a few common culprits, most of which are preventable with thoughtful preparation and the right infrastructure provider.

Database-related downtime can generally be grouped into three categories:



The rest of this ebook will explore each of these causes and how you can prevent them.

[1] Source [2] Source

Downtime from schema changes

A growing business is a changing business; updating your data model is part and parcel of running an application.

> But schema changes are one of the most common causes of downtime: 84% of application stakeholders report serious production issues due to database change errors.

Large or complex relational databases require some degree of downtime during a schema change, most commonly a lock on any tables affected by the change. Some changes, like adding a column in MySQL, can be done live without a lock, but many operations (like changing data types) require a table rebuild. In theory, using online DDL minimizes locks and downtime, but building a robust workflow in an organization where many engineers have access to the database is a challenge.

Beyond causing outages, schema changes themselves – even when executed successfully – can be a major ordeal for mission-critical applications. 91% of developers said that initial database changes need to be reworked several times before making it into production. 92% reported general difficulties with deploying database changes.

Preventing downtime from schema changes

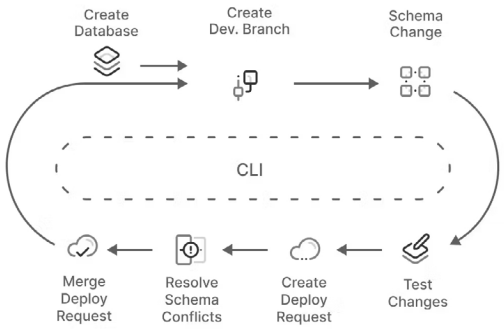
There are several tools designed to help developers apply schema changes. They operate through the "ghost" concept: the basic idea is instead of applying changes in place, they create a "ghost" table elsewhere with the new schema, and incrementally update the table to bring it to parity with production. When the time is right, the two tables are swapped.

A popular solution is gh-ost, an open-source migration solution for MySQL built by the engineering team at GitHub. Facebook open-sourced a similar tool called Online Schema Change for MySQL. They differ slightly in their implementation (log stream vs. triggers) but follow the same fundamental pattern. Similar solutions exist for Ruby (LHM from Soundcloud), and general-purpose data transfer (OAK from Google).

While convenient, all of these tools require bespoke setup and a larger team to maintain in the long run. Migrating properly is a process that touches a bunch of different systems and requires several steps (scheduling, discovery, cleanup, etc.). Small errors on whichever server you're running your migration utility on can completely compromise the process.

Schema changes in PlanetScale

PlanetScale is built to handle schema changes natively with PlanetScale workflow. It allows developers to build branches of their databases as they would with application code in git; you can then merge your changes (a new column name, changed data type, etc.) automatically without blocking the database or locking tables. You can set up separate branches for development and production (even automatically as part of CI) and open a deploy request when your changes are ready.



When your database is built to handle changes natively, it keeps power and momentum in the hands of the engineers building your application, instead of routing it to an entire other team to schedule migrations. Empowering engineers to make schema changes quickly andconfidently helps avoid the next most common cause of downtime: human error.

When it comes to migrations, PlanetScale has you covered: it's built on Vitess, which has migration management built in. You can control scheduling, handle schema conflicts, review history, and easily revert schema changes all natively.

Downtime from human error

Human error plays a role in most downtime: up to 70% of it according to one source.

> **If you're running your database on vanilla MySQL or Postgres, seemingly small changes from a single engineer can take down your entire application instantly: even things like minor version updates, or shipping something minor that's incompatible can lead to downtime.**

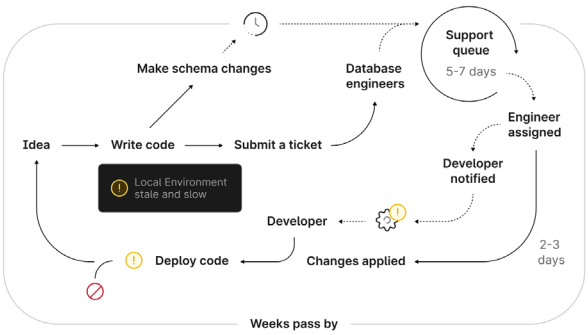
43% of organizations are now deploying their applications daily or weekly, and increased velocity means that many more chances for human error.

Preventing downtime from human error

The first step to preventing human error-related downtime is making sure that only qualified developers have the kinds of access to your infrastructure that could lead to downtime in the first place. But access control for databases – particularly when viewed from the perspective of the database engine itself – is easier said than done. Traditional RBAC can be difficult to set up in databases like MySQL, and even downright obscure in providers like AWS.

Many teams default to relying on a database admin – either an individual or a team – to manage and ship all database-related changes. While this does limit the degree to which an individual engineer's accidental misstep could cause downtime, it creates a significant bottleneck for the rest of the team and can slow down progress to a crawl.

> **49% of IT managers said slow database change and review cycles are a major point of difficulty for them.**



And perhaps most subtly, many teams flat-out avoid making hard schema changes at all because of how risky they can be. The impact on the business from that kind of reticence cannot be understated:

> 81% of developers agreed that regularly building new, innovative applications and features are crucial to the long-term success of their organizations.

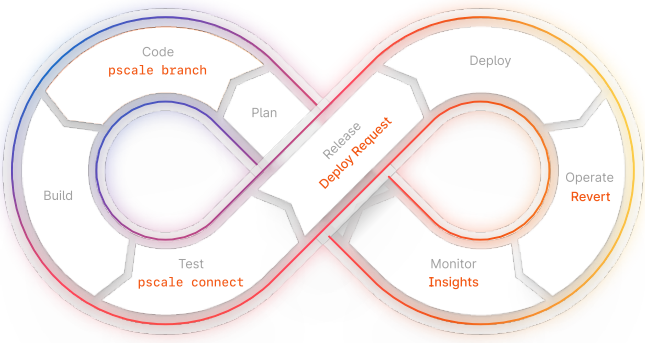
Preventing human error with PlanetScale

PlanetScale helps eliminate human error from the downtime equation. The PlanetScale schema revert feature allows developers to revert a migration with zero downtime and zero data loss. Essentially, human error is something you can never completely avoid: the ability to revert schema changes can at least help you manage it.

Beyond reverting bad changes, PlanetScale Workflow is built to help avoid human error by making the database change process a collaborative, predictable endeavor:

- DDL can be disabled in production, preventing accidental misfires
- Instead of relying on a DBA as a single point of failure, you can send deploy requests to review
- Deploy request workflows allow teams to collaborate on schema changes with schema diffs, team reviews, approvals, and automatic warnings built-in

> "PlanetScale gives us a lot more confidence in schema changes."
>
> - Cameron Aubuchon, CTO at Propfuel



PlanetScale also offers built-in, granular access controls at the database level. Without needing to implement it yourself in code, you can allocate roles, groups, and administrators easily so the wrong person never has access to something they shouldn't.

Downtime from application issues

Beyond your database, application issues like security breaches, scalability problems, and hardware failures can all lead to downtime.

Security breaches are a major topic of discussion for engineering leaders when it comes to outages: 76% of businesses reported security and data breaches as a top cause of downtime in 2022. Many high-profile breaches, like the one that plagued Deloitte in 2017, are the result of simple IAM misconfigurations.

> **The harder your database and application are to lock down and implement access controls, the more vulnerable you are to security-driven downtime.**

Scalability is on every engineering manager's mind already; performance issues and timeouts from traffic spikes, or even accelerating growth, can cause downtime for growing organizations. Scaling vertically can be straightforward enough, but scaling out relational databases is easier said than done. During the process of scaling out and migrating to a database system built on shards, it's common to require at least a bit of downtime.

Finally, hardware issues can be a major cause of outages for organizations running their own data centers. A simple power failure took down US-East-1 for AWS in 2021! 43% of IT professionals reported that infrastructure and networking outages were responsible for outages at their companies.

Preventing downtime from application issues

Many organizations try to implement scaling on their own. The initial step is usually to simply upsize your servers to scale vertically, but that becomes very costly very quickly. Scaling horizontally with relational databases is a difficult undertaking: it requires dedicated resources for creating and implementing a sharding strategy.

The best – and most straightforward – way to do your part to prevent security breaches is to lock down your infrastructure. But implementing a modern RBAC setup natively in databases like MySQL or Postgres can be difficult. If you're using a managed service like RDS, you'll need to wade your way through AWS IAM.

To avoid hardware issue-related downtime, teams will usually distribute their infrastructure to whatever degree they can across regions and availability zones.

In all of these instances, the solutions simply come down to adding more resources. More DBAs to implement sharding and access control. More budget to scale up hardware.

And, ultimately, more engineering time focused on your infrastructure that could have been spent adding to what really sets you apart - your product.

Preventing application issues with PlanetScale

With PlanetScale, you don't have to worry about handling application issues caused by sharding, security, or hardware on your own.

PlanetScale is built on Vitess, the database engine that powers some of the largest web-scale companies in the world like Slack, YouTube, and GitHub. Vitess allows MySQL to be sharded infinitely across sets of servers across the globe, so you get an infinitely scalable database without any of the extensive configuration normally required.

PlanetScale supports unlimited connection pooling and allows you to increase resource limits serverlessly in a few clicks.

Hardware issues won't cause downtime when you use PlanetScale, either: databases can be distributed across multiple availability zones easily. PlanetScale also supports read-only regions, and every database deploys with at least one additional replica.

> "We used to check the AWS dashboard practically nightly. Honestly, we never think about PlanetScale. That's the way it should be. The reality is that our team is called the product team — we build products. We don't want to be DevOps experts, and frankly we don't need DevOps database work. We want to always focus on making our product better."
>
> - Andrew Barba, CTO Barstool Sports

PlanetScale's security-first approach helps make sure that your database won't be the reason for a breach:

- SSL is required by default
- Data is encrypted at rest and in transit
- Robust and easy-to-use access controls
- Protection against DDOS attacks with query

PlanetScale is also a GitHub Secret Scanning Partner.

We can help

If you're currently dealing with or worried about downtime issues, now is the time to act. While it may feel daunting to try to solve a multi-layered problem, most companies can't afford to make this a line item for "next year's roadmap". Every minute of downtime is costing you now.

PlanetScale can help.

Nearly every solution we've discussed here is available on our free or self-serve plans – making downtime prevention accessible to nearly everyone. As your database requirements expand, we also offer plans that can grow with you. Our Enterprise solution includes several additional options:

Enterprise support

No more stressful 3am wakeup calls; we monitor your database for you.

Single-tenancy

Get PlanetScale in your own AWS or GCP account. Great option for those requiring additional security and compliance measures or who need a signed BAA for HIPAA.

Sharding

Horizontally shard using our elegant sharding solution. We work with you to identify the best sharding scheme that requires minimal application changes.

Customizable options

Configurable regions and availability zones, SSO, customizable number of branches, backups, and more.

> "Databases are hard. We would rather PlanetScale manage them. We wanted the support PlanetScale offers because they are the experts in the field. We've seen this come to fruition in our relationship."
>
> - Chris Karper, Engineering manager at MyFitnessPal

Get started today with PlanetScale, the most reliable way to scale your MySQL database in the cloud.

Call us at 1-408-214-1997
or send an email to sales@planetscale.com.