

To make Medium work, we log user data.  
By using Medium, you agree to our  
Privacy Policy, including cookie policy.



Published in Dev Genius

You have **2** free member-only stories left this month.

[Sign up for Medium and get an extra one](#)



Razvan L

Follow

Jan 8 · 4 min read · ✨ · [Listen](#)



Save



## 2 Database Scaling Patterns Every Developer Should Know



The book [Build Layered Microservices](#) is out! Buy your own copy now at [learnbackend.dev](https://learnbackend.dev).



When an application grows ~~wether in terms of functionalities~~ or users, the load on its database usually inc: To make Medium work, we log user data. ata is stored and retrieved. By using Medium, you agree to our Privacy Policy, including cookie policy.

Past a certain threshold, this increase often causes the database to:

1. Become slow or unresponsive due to a CPU overload.
2. Run out of storage due to a lack of disk space.

First-hand solutions for mitigating these effects usually consist in optimizing the code and storing some of the most frequently computed data into a cache. However, when the system's resource limits are reached, *scaling* often becomes the next best option.

In this article, we'll discuss two database scaling patterns; one for reads called *read replication*, and one for writes called *sharding*.

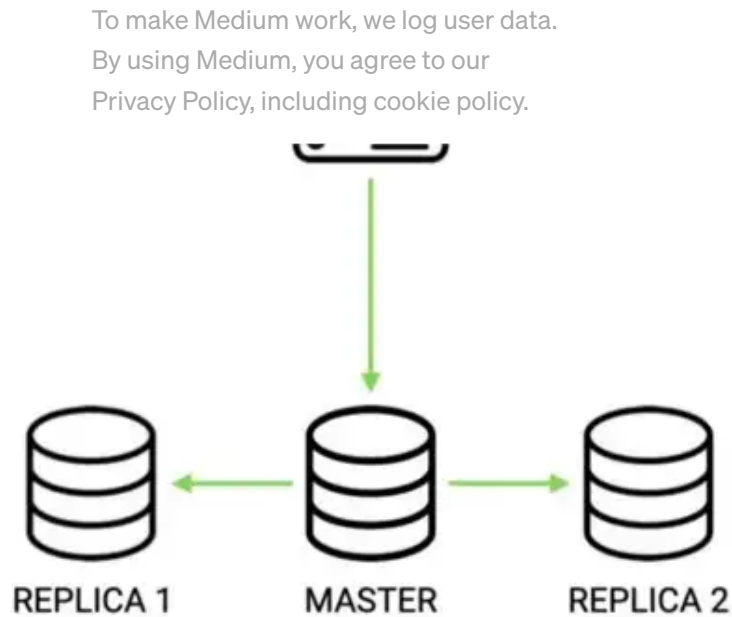
## Scaling for reads: read replication

Let's take the example of a service responsible for managing the list of products of an online shop.

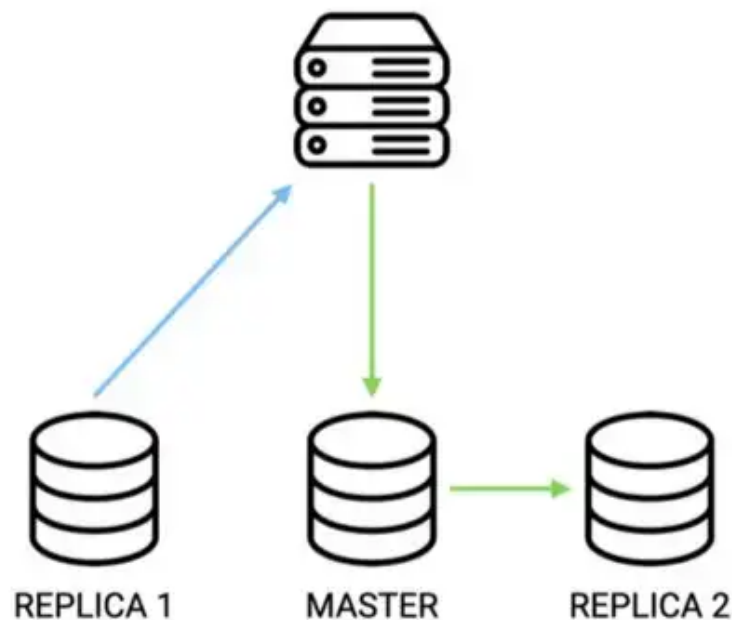
In that context, it is safe to say that the database containing these records is usually under more stress on reads than on writes, as customers will request the products list much more often than developers will update it.

One way of scaling this service for reads is to use a pattern called *read replication*.

In combination with a Relational Database Management System (i.e. RDBMS), this pattern consists in directing all write operations to the primary database node, that will in turn copy the data to one or more replicas in order to allow for future reads to be distributed.



Although great to ensure that data is kept safe, and that the main node doesn't have to cope with so much querying stress, it may sometimes happen that the service responds with stale data until the replication has completed.



Such configuration is then designated as *eventually consistent* (c.f. [The CAP Theorem Trade-Off](#)).

Despite the fact that it is a fairly simple and common way to help scale a system, I'd suggest you look into caching first (c.f. [An Introduction to Caching Patterns](#)), as it

often delivers much more significant improvements in terms of performance with less implementation effort.

To make Medium work, we log user data.  
By using Medium, you agree to our  
Privacy Policy, including cookie policy.

Scaling for writes: sharding

Sharding is the process of splitting up either horizontally or vertically the primary database into multiple database nodes called *shards*.

To illustrate this, let's consider the following database table.

id	first_name	last_name	birthdate	phone_number
1	John	Snow	NULL	631-555-3625
2	John	McClane	1955-03-19	965-333-259
3	Robert	Balboa	1947-07-06	451-987-0001
4	Selina	Kyle	NULL	856-658-2324

Horizontal sharding

In *horizontal sharding*, the tables of the primary database are duplicated and the data is broken up and spread across the different copies; which means that tables will have the same attributes but distinct records.

id	first_name	last_name	birthdate	phone_number
1	To make Medium work, we log user data. By using Medium, you agree to our Privacy Policy, including cookie policy.			631-555-3625
2	John	McClane	1955-03-19	965-333-259

id	first_name	last_name	birthdate	phone_number
3	Robert	Balboa	1947-07-06	451-987-0001
4	Selina	Kyle	NULL	856-658-2324

### Vertical sharding

In *vertical sharding*, the attributes of the tables are divided into smaller tables, where the data is broken up by attribute; which means that tables will have both different attributes and records.

id	first_name	last_name
1	John	Snow
2	John	McClane
3	Robert	Balboa
4	Selina	Kyle

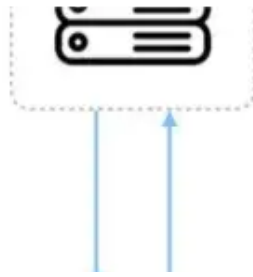
id	birthdate	phone_number
1	NULL	631-555-3625
2	1955-03-19	965-333-259
3	1947-07-06	451-987-0001
4	NULL	856-658-2324

### Sharding benefits

The benefit of this approach is that it greatly decreases the response delay, as well as providing an increased resiliency to failures.

Executing a simple request on sharded data will be more efficient as the search will only be performed on a subset of records, and since shards are more likely to be located on different machines, the impact on the system of a server becoming unavailable is mitigated.

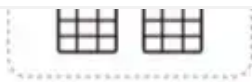
To make Medium work, we log user data.  
By using Medium, you agree to our  
Privacy Policy, including cookie policy.



Open in app ↗

Sign up

Sign In



## Sharding challenges

On the flip side, the complexity automatically increases when handling queries that require data-joins.

Moreover, the addition of extra shards can prove challenging without the right database management system, as there might be a need to take the entire database down in order to rebalance the data.

Finally, resiliency might not actually be improved if one or more shards are unavailable because of a lack of data replication.

Scaling for writes is therefore quite tricky and really requires to explore the different capabilities of the various databases (i.e. Relational, NoSQL) available on the market.



## What's next?

👉 You like this kind of content? Check out the book **Build Layered Microservices** on how to build a production-ready layered authentication microservice using the Express framework, that lives up to the industry standards in terms of development practices and software architecture from the first line of code to the last line of documentation.

Scaling

Sharding

To make Medium work, we log user data.  
By using Medium, you agree to our  
Privacy Policy, including cookie policy.


 50 | 

## Sign up for DevGenius Updates

By Dev Genius

Get the latest news and update from DevGenius publication [Take a look.](#)

By signing up, you will create a Medium account if you don't already have one. Review our [Privacy Policy](#) for more information about our privacy practices.

 Get this newsletter

[About](#) [Help](#) [Terms](#) [Privacy](#)

Get the Medium app

 Download on the  
App Store

 GET IT ON  
Google Play