The last thing we wanted to pay for was an engineering team to host our own [database], especially when we were trying to get a product in the market.

Karl Matthias
Engineering Director

Community chose PlanetScale for Flexibility, Scale and Speed

Community is a text messaging platform enabling direct and instant communication at scale. The platform enables pop stars to connect in a more meaningful and authentic way with their audience. Community originally chose Cassandra as their database, taking advantage of NoSQL’s un-opinionated data structure to get their business off the ground. When Community began to plan for the scale of sending millions of messages a month they knew they were going to need to find a relational database that gave them the same scale and flexibility of a NoSQL database. By migrating to PlanetScale as their database in the cloud, Community was able to get that scale and flexibility while continuing to scale their business.

Planning For Scale and Finding The Limitations of NoSQL

Community was founded in 2019 as a way to connect brands, celebrities, and enterprise companies directly to their audiences via text messaging. They subvert the typical “marketing” strategies taken by celebrities by using technology to directly connect brands and personalities with their millions of fans and customers. By utilizing text messaging they have seen a greater increase in audience engagement compared to typical communication channels like emails and social media. “We grew pretty rapidly through word of mouth and early success with notable celebrities,” says Karl Matthias, Director of Architecture and Platform Engineering at Community.com. The team had two problems: rapidly building the platform while simultaneously scaling the platform.

Originally the team at Community chose Apache Cassandra, but quickly realized they needed more flexibility in their data layer. The engineering team at Community identified two limitations of their existing NoSQL database:

- Lack of flexibility: they couldn’t rewrite queries easily without changing the way the data was written
- Slow speed of development: often, new features required rewriting some or all of the data

“We knew we were going to have to have a very scalable store for messages, because we were going to send billions of messages,” says Karl. Once they began cataloging their requirements, Community landed on a short but critical list for their database:

- It needed to allow them to easily change schemas and join on new tables
- It needed to allow them to shard indefinitely
- It needed to be hosted so they wouldn’t have to manage infrastructure
Luckily the team had heard of Vitess and PlanetScale and found that it would give them both sharding and a reliable key store that would allow them to change data structure on the fly.

“The last thing we wanted to pay for was an engineering team to host our own [database], especially when we were trying to get a product in the market. You don’t need to hire engineers to manage databases if you can pay somebody else to do it.”

Results: Speed and Control

Once Community had migrated to PlanetScale, they saw an immediate benefit in their engineering workflow and ability to ship quicker: “I used to work for New Relic and it was nice to work on a flexible SQL store that can scale—without building and operating all of the scaling tech ourselves, like we did back then.” says Karl after switching to PlanetScale’s cloud database.

One benefit the team appreciated from the start was not needing to stand-up their own database cluster. “The last thing we wanted to pay for was an engineering team to host our own [databases], especially when we were trying to get a product in the market. You don’t need to hire engineers to manage databases if you can pay somebody else to do it.” Aside from not worrying about infrastructure, the team is also looking forward to the upcoming release of developer features that will make schema changes and migrations even easier.

The second immediate benefit Community saw from switching to PlanetScale was having explicit control of their sharding story. Unlike many NoSQL cloud data providers, sharding with PlanetScale is not a mystery.

With PlanetScale, Karl and his team were able to start with two shards and then re-shard as needed. “The truth is that sharding is an art. You can’t know all of the requirements you will eventually have up front, so you have to think it through carefully. Having the flexibility to change our sharding strategy later is a huge win for us.”

Of course all of this is just icing on the cake since ultimately it came down to speed of feature development for the Community engineering team. Previously feature changes often required rewriting and touching the data. When asked about how much time they now spend on managing their database the response was “honestly, nobody works on it most of the time.” The ability to arbitrarily add fields to their data model, and to join against other tables, meant they no longer had to worry about making their data fit their growing business. “Planning for massive scale while also rapidly iterating on our feature set was a demanding piece to be—especially for a small team at a startup—and PlanetScale was able to deliver world class operations, a scalable platform, and the flexibility the business needed to grow both the member base and the feature set. It just does what we need.”

PlanetScale is a MySQL compatible, serverless database platform that allows developers to get back to the business at hand, and leave the scaling to us. PlanetScale’s cloud database brings new developer-first experiences to the trusty MySQL database. Using PlanetScale, teams can more quickly deploy features alongside their data changes using tools like Data Branching and Auto Migrations which bring the power of github-style workflows to the cloud database. PlanetScale works for every developer allowing engineering teams to quickly scale their data using the languages and frameworks they already use (Python, JavaScript, Rails, React, Vercel, Go, .NET/C#). Finally, teams can build applications meant for scale from day one, and deploy with confidence knowing their database isn’t a blocker. PlanetScale isn’t magic, it’s just the new way of doing databases.