NoSQL Databases - Introduction

In this lesson, we will get an insight into NoSQL databases and how they are different from Relational databases.

WE'LL COVER THE FOLLOWING

What Is A NoSQL Database?

In this lesson, we will get an insight into NoSQL databases and how they are different from Relational databases. As the name implies *NoSQL* databases have no *SQL*, they are more like *JSON-based* databases built for Web 2.0

They are built for high frequency read & writes, typically required in social applications like Twitter, LIVE real-time sports apps, online massive multi-player games etc.

How Is A NoSQL Database Different From A Relational Database?

Now, one obvious question that would pop-up in our minds is:

Why the need for NoSQL databases when relational databases were doing fine, were battle-tested, well adopted by the industry & had no major persistence issues?

Scalability

Well, one big limitation with *SQL* based relational databases is *Scalability*. Scaling relational databases is something which is not trivial. They have to be

Sharded or *Replicated* to make them run smoothly on a cluster. In short, this requires careful thought and human intervention.

On the contrary, *NoSQL* databases have the ability to add new server nodes on the fly & continue the work, without any human intervention, just with a snap of the fingers.

Today's websites need fast read writes. There are millions, if not billions of users connected with each other on social networks.

Massive amount of data is generated every micro-second, we needed an infrastructure designed to manage this exponential growth.

Clustering

NoSQL databases are designed to run intelligently on clusters. And when I say intelligently, I mean with minimal human intervention.

Today, the server nodes even have *self-healing* capabilities. That's pretty smooth. The infrastructure is intelligent enough to self-recover from faults.

Though all this innovation does not mean old school relational databases aren't good enough & we don't need them anymore.

Relational databases still work like a charm & are still in demand. They have a specific use-case. We have already gone through this in *When to pick a relational database lesson*. Remember? 😊

Also, *NoSQL* databases had to sacrifice *Strong consistency*, *ACID Transactions* & much more to scale horizontally over a cluster & across the data centres.

The data with *NoSQL* databases is more *Eventually Consistent* as opposed to being *Strongly Consistent*.

So, this obviously means *NoSQL* databases aren't the silver bullet. And it's completely alright, we don't need silver bullets. We aren't hunting werewolves, we are upto a much harder task connecting the world online.

I'll talk about the underlying design of *NoSQL* databases in much detail and why they have to sacrifice *Strong consistency* and *Transactions* in the

upcoming lessons.

For now, let's focus on some of the features of *NoSQL* databases.