Week 9 Paper

This week we continued our quest to understand MongoDB. With this we learned more about the potential problems that could arise with it compared to SQL. Through this paper, I hope to answer those questions as well as go over more issues that MongoDB could have.

The first thing we will cover is how MongoDB uses a schemeless collection. So what is a schemeless collection in MongoDB? This means that MongoDB doesn’t require a rigid, pre-defined schema like a relational database does. The DBMS or database management system, enforces a partial schema as data is written. With this it is explicitly listing collections and indexes.

The next item to cover is how to update data in a MongoDB. With this we can use a single line of code to do so. This code is as follows, db.collection.updateOne(). With this, we can update a single document. To update multiple documents we can use, db.collection.updateMany(). To replace a document we can use, db.collection.replaceOne(). These are the commands we can use to update a database within MongoDB.

There are a few different things that will happen if we change a field or field value in a document. Since the field or field value is most likely connected to other fields or values. Then those would be affected by this change. Luckily, MongoDB has a plan for that. You can either manually change the connected fields or values or you can create a automated change so that you can get those fields or values to change when any field or value changes.

Lastly, we will cover how all of this can have an impact on the technical debt of the company. Having a MongoDB source flowing into the data warehouse, there is a potential for technical debt. What do I mean? Technical debt is a place within the system where there is a potential for any bugs or fixes which in turn could cause performance issues, more downtime, and slow delivery. So, could having a MongoDB system within another system cause this? Yes, it could. With this, as a company we need to be mindful of what we plan to change within the system of the MongoDB. How often are we planning on changing documents, fields, and/or values within the database. The more often you plan on making changes, then the potential for technical debt is higher.

All in all, having a MongoDb working along with the database that you have installed within the company isn’t a bad thing. There are many benefits to having one installed. Like anything, the more changes you plan on making the more opportunities there are for technical debt to arise. So keeping this in mind, this is where the engineers need to come up with safety plans to keep the technical debt at bay.

References

* <https://www.mongodb.com/unstructured-data/schemaless#:~:text=As%20a%20NoSQL%20database%2C%20MongoDB,explicitly%20listing%20collections%20and%20indexes>.
* <https://www.mongodb.com/docs/mongodb-shell/crud/update/>