UNIVERSIDADE FEDERAL DO PARÁ – UFPA
CAMPUS UNIVERSITÁRIO DE TUCURUÍ – CAMTUC
FACULDADE DE ENGENHARIA DE COMPUTAÇÃO – FECOMP

UM ESTUDO TEÓRICO DO CONCEITO DO MONGODB

Luan Ribeiro Siqueira[1]
Pedro Luiz Braga Pinheiro[2]

Resumo

Este artigo teve como objetivo abordar um estudo teórico sobre a utilização do banco de dados MongoDB (NoSQL), além de apresentar diferenças entre um sistema de gerenciamento de banco de dados SQL - MySQL e sistema de gerenciamento de banco de dados orientados a documentos – MongoDB. Os critérios de comparação incluem diferenças teóricas, características, restrições, integridade, consulta, desempenho.

Palavra-chave: MongoDB. MySQL. Modelo de dados.

1. Introdução

A quantidade de informação nas organizações vem crescendo constantemente e os bancos de dados mais utilizados (bancos de dados relacionais) começam a não ser mais eficientes. Isso acontece porque eles normalmente baseiam-se em estruturas computacionais centralizadas, que permitem uma expansão limitada dos recursos computacionais (troca ou adição de processador, adição de memória) (SCHNEIDER, 2014). A partir destes problemas surgiu como solução o movimento NoSQL (Not Only SQL), com um

novo conceito de gerenciamento de banco de dados diferente do convencional para o armazenamento de dados, tendo como exemplo, o armazenamento de documentos sem esquema definido.

O termo NoSQL surgiu em 1998, criado por Carlos Strozzi citado como um banco de dados relacional de código aberto que não possuía interface SQL. segundo Strozzi (2007), O movimento NoSQL é completamente distinto do modelo relacional e, portanto, deveria ser mais apropriadamente chamado NoREL. Os bancos de dados NoSQL é desenvolvido sobre arquiteturas distribuídas, possibilitando processar grandes volumes de dados, com alta disponibilidade e escalabilidade. Posteriormente, outros projetos NoSQL foram sendo criados como MongoDB, Cassandra, entre outros.

Este trabalho de pesquisa é baseado no MongoDB que vem sendo utilizado por grandes corporações atualmente e comparar seu desempenho em relação a outros respectivos bancos de dados. A metodologia adotada para este trabalho é uma coleta de dados através de pesquisa bibliográfica.

2. Referencial Teórico

Nesta seção são apresentados os conceitos e definições essenciais para o entendimento do trabalho realizado.

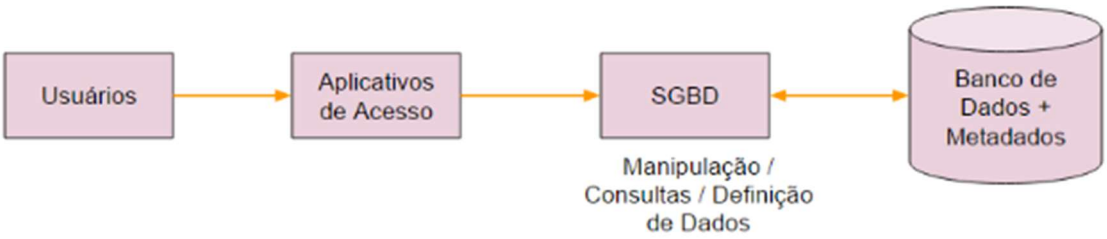
2.1. Banco de dados

Segundo Date (2004), “Um banco de dados é uma coleção de dados persistentes, usada pelos sistemas de aplicação de uma determinada empresa”, isto é, um montante de informações pertinentes a um determinado sistema de aplicação. Em outras palavras, pode se definir que banco dados é um conjunto de informações armazenados em um local com múltiplos níveis de interação que um usuário habilitado poderá acessá-las sempre que necessário para tomar decisões importantes.

No entanto, para uma manipulação desses dados há um sistema de gerenciamento de banco de dados (SGBD) que é um software que possui

recursos capazes de realizar essas manipulações. Exemplos mais comuns de SGBDs são: MySQL, Oracle, SQL Server, entre outros. Logo abaixo, tem-se a representação de um sistema de banco de dados (Figura 1).

Figura 1- Representação de um sistema de banco de dados



Fonte: Reis (2016)

"O principal objetivo de um SGBD é proporcionar um ambiente tanto conveniente quanto eficiente para a recuperação e armazenamento das informações do banco de dados", (SILBERSCHATZ; KORTH; SUDARSHAN, 1999). Desse modo, podemos entender SGBD como um agrupamento de programas poderosos capazes de organizar, gerenciar e recuperar dados a partir de diferentes, mas relacionadas, tabelas.

2.2. Modelo de Dados

Um modelo de dados é uma coleção de conceitos que podem ser usados para descrever a estrutura de um banco de dados (ESLMARI: NAVATHE, 2011). Assim sendo, um modelo de banco de dados apresenta toda a sua estrutura lógica, incluindo relações e restrições que determinam como os dados podem ser inseridos e acessados. Na maioria dos modelos de dados podem ser representados por um diagrama de banco de dados acompanhante.

Cada SGBD segue um modelo de banco de dados, de forma a diferenciar esses dados de acordo com sua representação. Cabe ao usuário optar pelo modelo mais apropriado quanto ao desempenho, redução de custos, usabilidade para o desenvolvimento do banco de dados.

2.2.1. Modelo Relacional

O modelo mais comum, o modelo relacional, classifica os dados em tabelas, também conhecidas como relações. são compostas por uma série de elementos (atributos), que possuem características comuns. Um determinado atributo ou combinação de atributos é escolhido como uma chave primária que pode ser consultada em outras tabelas, quando é chamada de chave estrangeira.

Segundo Elsmari e Navathe (2011), um banco de dados que segue o Modelo Relacional, representa seus dados como um conjunto de relações. Considerando que uma relação é, de certo modo, similar a uma tabela de valores e aplicando a terminologia do Modelo Relacional diz que as linhas se denominam tuplas; as colunas, atributos; e a tabela em si, relação.

2.2.2. Modelo NoSQL

Segundo Suissa (2010), o termo NoSQL foi utilizado pela primeira vez em 1998, por Carlo Strozzi, como nome de seu SGBD, baseado no modelo relacional, sem interface SQL.

Este tipo de banco de dados começou a ser projetado para suprir as necessidades de alto armazenamento e escalabilidade. O NoSQL, está sendo tratado como o futuro do grande armazenamento de informações, já que, grandes corporações investem nesta tecnologia para o tratamento de suas informações e desenvolvendo novas soluções para complementar e incrementar o NoSQL. O banco de dados NoSQL são classificados conforme a maneira de armazenar dados. Existem 4 tipos de categorias: chave/valor, orientado a colunas, orientado a documentos e baseados em grafos.

1. Armazenamento chave-valor (Key/Value): O armazenamento do tipo chave/valor é semelhante ao uso de mapas ou de dicionários, os dados são endereçados por uma única chave, permitindo aos clientes colocar e solicitar valores por chaves. Esses sistemas podem conter dados estruturados ou não estruturados. Esses bancos são úteis para operações simples, baseadas somente em atributos chave. Por

exemplo, sistemas com rápida troca de dados e com frequente escrita. Como a maioria dos armazenamentos chave/valor mantém seu conjunto de dados em memória, eles são bastante usados para cache de consultas SQL (LEAVITT, 2010).

2. Armazenamento em colunas: Um SGBD orientado a colunas armazena seu conteúdo de forma inversa aos bancos de dados orientados a linhas. Este formato de armazenar as informações torna-se vantajoso para Data Warehouses, onde agregações são processadas sobre uma quantidade de dados de características similares.
3. Armazenamento baseado em Grafos: O armazenamento baseado em grafos fundamenta-se na teoria dos grafos. Em geral, grafos consistes de nós, propriedades e arestas. Os nós representam as entidades, as propriedades representam os atributos e as arestas representam as relações (LEAVITT, 2010).
4. Armazenamento orientados a documentos: Segundo Lennon (2013), o MongoDB armazena dados dentro de documentos semelhantes a JSON (usando BSON — uma versão binária de JSON), que retém os dados usando pares de chave/valor. Um banco de dados orientado a documentos armazena, recupera e gerencia dados semiestruturados. O elemento dos dados é chamado documento. Os documentos são endereçados no banco de dados via uma chave única que representa o documento. Uma das características de um banco de dados orientado a documentos é que, além da simples chave/valor de pesquisa, é possível recuperar um documento através de uma API, ou linguagem de consulta disponibilizada pelo banco. Todos os bancos de dados correntes fornecem suporte a documentos no formato JSON (LEAVITT, 2010).

A diferença mais notável entre o banco de dados relacionais e NoSQL é o escalonamento, é nesse ponto onde o NoSQL contém suas principais vantagens com a de seu concorrente. Os atuais bancos relacionais são muitos limitados quanto à escalabilidade, pois usam escalonamento vertical em seus servidores. Já o NoSQL, utiliza escalonamento horizontal e tem grande facilidade

na distribuição de dados, dividindo seus dados em vários servidores, diminuindo assim, o tempo de processamento, armazenamento e gerenciamento dos dados.

No entanto, Schneider (2014) diz que, o NoSQL não pode ser usado em todas as situações, pois alguns sistemas, como exemplo, sistemas de automação industrial e comercial, necessitam ter integridade entre os dados, e como a grande parte dos bancos de dados NoSQL não trabalham com ACID (Atomicidade, Consistência, Isolamento e Durabilidade) o que torna inviável sua implementação em alguns casos.

2.3. MongoDB

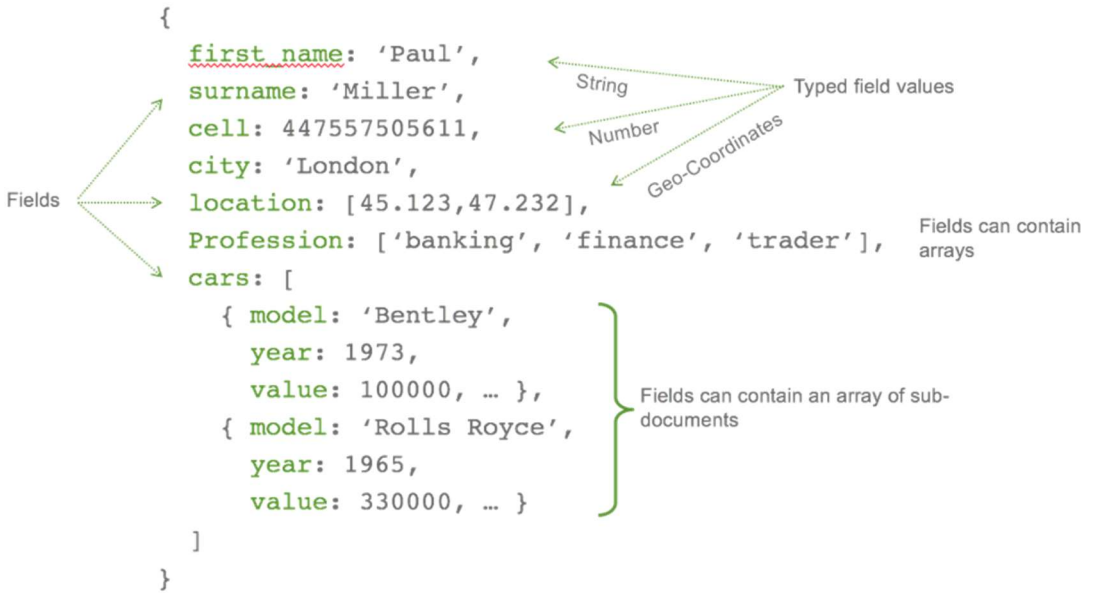
O MongoDB inicialmente foi desenvolvido em 2007 pela empresa 10gen e passou a ser um software open source em 2009. MongoDB é derivado da palavra ‘Humongous’ que significa ‘Gigantesco’, pois o banco de dados possui uma característica excepcional de capacidade de expansão. É uma aplicação de alta performance, sem esquemas e orientado a documentos. Foi projetado na linguagem de programação C++.

Os bancos de dados orientados a documentos são bastante diferentes dos tradicionais bancos de dados relacionais. Em vez de armazenar dados em estruturas rígidas, como tabelas, eles os armazenam em documentos vagamente definidos (CUNHA, 2011). Sendo assim, o MongoDB vem crescendo no mercado graças ao aumento de aplicações que não são supridas por banco de dados relacionais ou são bastante complexas para implementá-las.

Segundo Introduction (2014), o MongoDB é um banco de dados que fornece um excelente desempenho, alta disponibilidade, escalabilidade. Por ser orientado a documentos é fácil o manuseio dos objetos via linguagem de programação. Além de um excelente serviço de replicação de dados. Quando ser fala em fácil escalabilidade no MongoDB o termo usado por ele é Sharding, que é a divisão de coleções entre instancias dele.

Sadalage (2013) defini que o conceito principal de banco de dados de documentos são Documentos. Os bancos de dados armazenam documentos que podem ser JSON, BSON entre outros tipos. E são essencialmente o próximo nível do chave-valor, permitindo valores aninhados associados a cada chave. Abaixo, temos na Figura 2 uma representação de um documento MongoDB.

Figura 2 – Documento MongoDB

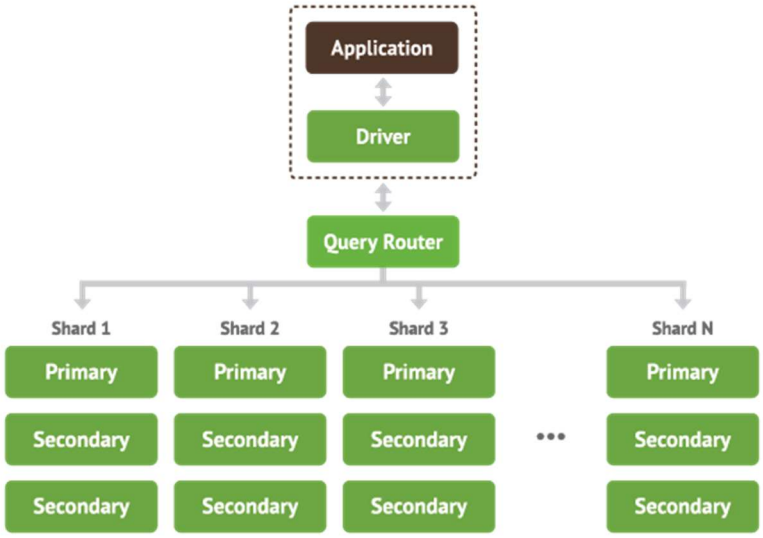


Fonte: Walters (2017)

Segundo Sharding (2014), quando um banco de dados começa a ficar muito grande e a quantidade de acessos aumentar, com uma única CPU pode ocasionar na interrupção do serviço. E consequentemente mais caro se torna isso ao adicionar mais processadores e memória. Com o MongoDB esse trabalho se torna mais fácil e menos custoso.

Por conta disso Sharding (2014) destaca que para tratar desses problemas temos o escalonamento vertical e o Sharding. O escalonamento vertical adiciona mais CPUs e quantidade de armazenamento para com isso conseguir mais processamento. Mas isso pode se tornar mais caro. O Sharding por sua vez, divide o conjunto de dados e distribui os dados em múltiplos servidores. Cada fragmento é um banco de dados independente, e juntos, os fragmentos compõem um único banco de dados lógico. Em alguns casos, quando alguns dos clusters falhar, haverá outro assumindo automaticamente o processo. Tem-se abaixo, a Figura 3 representando em diagrama, os dados distribuídos.

Figura 3 – Diagrama de dados distribuídos (Sharding)



Fonte: Sharding in MongoDB (2013)

2.4. MySQL

O MySQL é um SGBD relacional open source utilizado em boa parte de aplicações gratuitas para gerir suas bases de dados. Atualmente pertence a Oracle Corporation. O MySQL utiliza a linguagem SQL (Structure Query Language – Linguagem de Consulta Estruturada), linguagem mais comum para inserir, gerenciar, acessar o conteúdo armazenado em um banco de dados. Ela armazena os dados em tabelas e comandos como ‘SELECT’, ‘UPDATE’, ‘INSERT’, e ‘DELETE’ para gerenciá-los. As informações relacionadas podem ser inseridas em tabelas diferentes, mas com o uso da operação JOIN é permitido correlaciona-las, executar consultas em várias tabelas e diminuir a chance de duplicar dados. Esse SGBD é compatível com quase todos os sistemas operacionais como Linux, Windows, Apple, entre outros.

3. Metodologia

O presente trabalho e classificado conforme Silva e Menezes (2001) como pesquisa bibliográfica pois a mesma é elaborada a partir de material já publicado,

constituído principalmente de livros, artigos de periódicos e atualmente com material disponibilizado na Internet.

Pesquisa Qualitativa, considera que há uma relação dinâmica entre o mundo real e o sujeito. E também, avalia-se o trabalho como pesquisa explicativa, segundo Silva e Menezes (2001) visa identificar os fatores que determinam ou contribuem para a ocorrência dos fenômenos. Aprofunda o conhecimento da realidade porque explica a razão, o “porquê” das coisas.

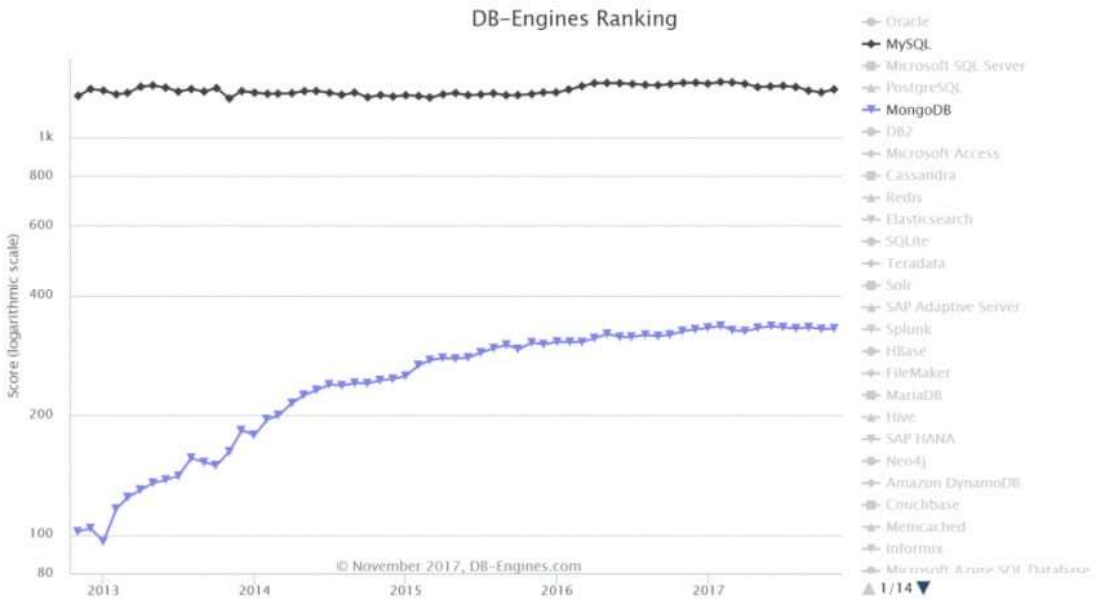
4. Resultados e discussões

Nesta seção são apresentados os resultados obtidos a partir de pesquisa explanatória.

4.1. MongoDB x MySQL

Os bancos de dados relacionais mantiveram sua liderança por bastante tempo. Embora, com o passar do tempo com o aumento da demanda de informações, diversidades e escalabilidade, foram surgindo alternativas no mercado para atender essas devidas necessidades. A Figura 4 mostra a popularidade entre as duas SGBDs no ano de 2017.

Figura 4 - Ranking de uso das SGBDs



Fonte: Solanki (2017)

Comparar o desempenho do MongoDB e do MySQL é difícil, pois uma vez que ambos os sistemas de gerenciamentos são extremamente importantes e suas principais diferenças estão por trás de suas operações básicas. Embora que o MongoDB tem uma certa flexibilidade em seu esquema por não haver restrições. Pode-se simplesmente inserir alguns documentos sem ser necessário a relação entre eles. Porém, devido à ausência de junções e transações, é preciso otimizar com frequência o esquema com base em como o aplicativo acessará os dados.

No MySQL, antes de inserir alguns dados, o usuário primeiramente precisa definir as tabelas e colunas, e cada linha na tabela deve ter a mesma coluna. Devido a isso, não há muita flexibilidade na maneira de armazenar os dados seguindo a normalização.

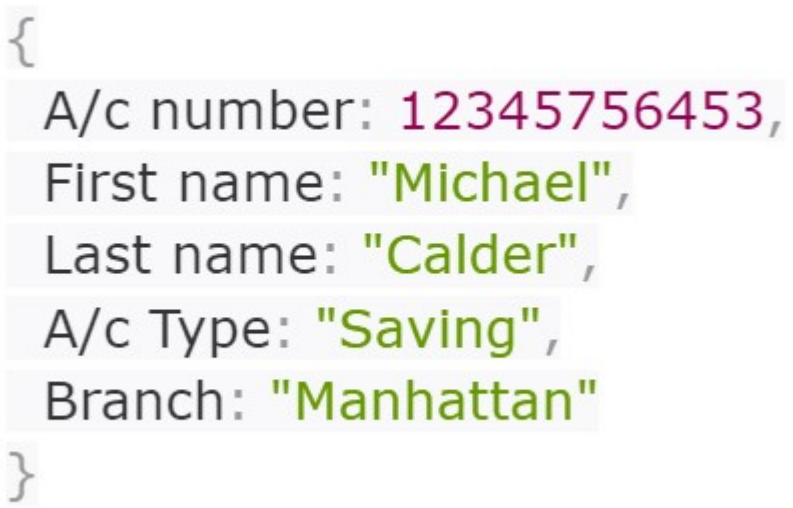
Figura 5 – Tabela Mysql

A/c number	First name	Last name	A/c Type	Branch
12345756453	Michael	Calder	Saving	Manhattan
12345978675	Nick	Brown	Current	Brooklyn

Fonte: Solanki (2017)

A Figura 5 apresenta a forma como o MySQL armazena seus dados. O design da tabela é bem rígido e de difícil alteração. O mongoDB armazena seus dados do tipo JSON, conforme a Figura 6 abaixo:

Figura 6 – Armazenamento MongoDB



Fonte: Solanki (2017)

O MongoDB cria documentos sem esquema que podem armazenar qualquer informação desejada, embora possa causar problemas com a consistência dos dados. O MySQL cria um modelo de esquema estrito e, portanto, está fadado a cometer erros (SOLANKI, 2017).

A linguagem de consulta utilizada pelo MongoDB não é estruturada. Para a criação de uma consulta JSON, precisa-se especificar um documento com propriedades que o usuário deseja que os resultados correspondam e geralmente é executado usando um conjunto de operadores vinculados entre si usando o JSON. O MongoDB trata cada propriedade como tendo um AND booleano implícito. Ele suporta nativamente consultas orais booleanas, mas deve-se usar um operador especial ($ ou) para alcançá-lo. O MySQL usa a linguagem de consulta estruturada SQL para se comunicar com o banco de dados. Linguagem é simples e muito poderosa, que consiste principalmente em duas partes: linguagem de definição de dados (DDL) e linguagem de manipulação de dados (DML). Logo abaixo na Figura 7, há uma comparação rápida entre as duas SGBDs.

Figura 7 – Linguagem de consulta

MySQL	MongoDB
INSERT	
INSERT INTO account (`A/c number`, `first name`, `last name`) VALUES ('12345746352', 'Mark', 'Jacobs');	db.account.insert({ A/c number: "12345746352", first name: "Mark", last name: "Jacobs" });
UPDATE	
UPDATE account SET contact number = 9426227364 WHERE A/c number = '12345746352'	db.account.update({ A/c number: '12345746352' }, { $set: {contact number: 9426227364 } });
DELETE	
DELETE FROM account WHERE e-mail address = 'jv1994@gmail.com';	db.account.remove({ "E-mail address": "jv1994@gmail.com" });

Fonte: Solanki (2017)

4.1.1. Relacionamento entre entidades

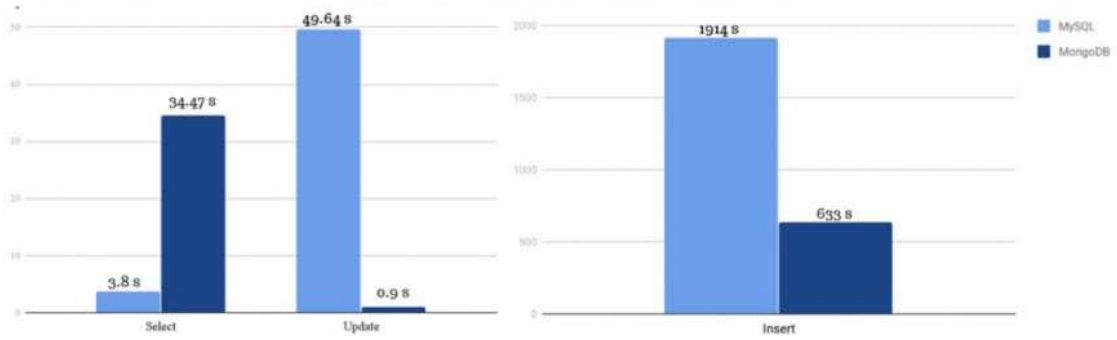
O MongoDB não suporta tecnologia JOIN que permite o usuário vincular seus dados de duas ou mais tabelas em uma única consulta. No entanto, ele suporta tipos de dados multidimensionais, como arrays e outros documentos. Uma das vantagens do MySQL é a operação JOIN pois, a partir dela o banco de dados se torna relacional.

4.1.2. Desempenho e velocidade

Comparando a velocidade do MongoDB com o MySQL, os desenvolvedores observam que este último não tem velocidade e dificuldades de experiência com grandes volumes de dados, então será uma escolha melhor para empresas com bancos de dados menores e procurando uma solução mais geral. Embora essa seja uma das vantagens do MongoDB sobre o MySQL: a capacidade de lidar com grandes quantidades de dados não estruturados (DA 14, 2017).

No entanto, não existe uma referência padrão para definir qual o melhor banco de dados a se utilizar. Somente com sua necessidade, infraestrutura que poderá definir qual o mais aceito. A Figura 8 mostra um exemplo para um melhor entendimento perante a velocidade do MySQL e do MongoDB de acordo com algumas funções.

Figura 8 – Desempenho entre os bancos de dados



Fonte: Solanki (2017)

4.1.3. Segurança

Os recursos de segurança do MongoDB incluem autenticação, auditoria e autorização, além de utilizar TLS (Transport Layer Security) e SSL (Secure Sockets Layer) para fins de criptografia. Isso garante que seja acessível e legível

apenas pelo cliente pretendido. Já o MySQL usufrui de um modelo de segurança baseado em privilégios. Isso significa que ele autentica um usuário e o facilita com privilégios de usuário em um banco de dados específico, como CREATE, SELECT, INSERT, UPDATE e assim sucessivamente.

5. Conclusão

O presente trabalho apresentou definições gerais do banco de dados MongoDB e mostrar uma comparação de desempenho e tempo de execução de instruções, além de, mostrar as linguagens de consultas e funções de armazenamento com o MySQL.

Além disso, após estudos, foram analisados que o MongoDB tem mais propriedades BASE do que propriedades ACID deixando a desejar sobre sua verdadeira capacidade de segurança dos dados, mas em compensação a escalabilidade vertical e horizontal proporcionada pelo MongoDB, consegue totalmente se sobrepor à dificuldade quanto à segurança. Vale ressaltar que os sistemas NoSQL não são inseguros o tempo inteiro, e sim, tem picos de momentos “desprotegidos”.

Por fim, O mongoDB vem atraindo cada vez mais usuários com sua filosofia aberta e simples trazendo conceitos de banco de dados orientados a documentos. Todavia, o usuário precisa levantar uma série de requisitos perante ao desenvolvimento de seu banco. Com a indexação dos dados adequada, o uso do MySQL poderá resolver os problemas com desempenho, garantindo uma interação fácil e robustez. Mas, se seus dados não forem estruturados e complexos, ou se você não puder predefinir seu esquema, é melhor optar pelo MongoDB. E, além disso, se você precisar manipular um grande volume de dados e armazená-los como documentos, o MongoDB o ajudará a atender suas necessidades.

Referências

CUNHA, T. M. de A. **Escalabilidade de Sistemas com Banco de Dados NoSQL: um Estudo de Caso Comparativo com MongoDB e MySQL.** 2011. 85 f. Centro Universitário da Bahia – Estácio, Salvador.

DATE, C. J. **Introdução a Sistemas de Bancos de Dados.** 8. ed. Rio de Janeiro: Elsevier, 2004.

ELMASRI, R; NAVATHE, S, B. **Sistemas de Bancos de dados.** 6. ed. São Paulo: Addison Wesley, 2011.

INTRODUCTION to MongoDB. Disponível em: <http://www.mongodb.org/about/introduction/>. Acesso em: 17 de junho 2019.
LEAVITT, N. **Will NoSQL Databases Live Up to Their Promise?** 2010. Computer, 12. ISSN: 0018-9162, p. 12 – 14.

LENNON, J. (2013). **MongoDB**. Disponível em: http://alfavillecode.blogspot.com.br/2013/01/MongoDB.html. Acesso em: 20 junho 2019.

REIS, F. **Introdução a banco de dados – 01.** 2016. Disponível em: <http://www.bosontreinamentos.com.br/bancos-de-dados/o-que-sao-bancos-de-dados/>. Acesso em: 16 junho 2019.

SCHNEIDER, O. **Comparativo de performance do sistema gerenciador de banco de dados mongodb sobre clusteres heterogêneos.** Universidade tecnológica federal do paraná, 2014.

Sharding in MongoDB. 2013. Disponível em: < http://blog.optimal.io/sharding-in-mongodb/ >. Acesso em: 20 junho 2019.

SHARDING in MongoDB. Disponível em: <http//docs.mongodb.org/manual/core/sharding-introduction/>. Acesso em 20 junho 2019.

SILBERSCHATZ, A; KORTH, H F.; SUDARSHAN, S. **Sistema de Bancos de Dados.** 3. ed. São Paulo: Makron Books, 1999.

SILVA, E. L. MENEZES, E. M. Metodologia da pesquisa e elaboração de dissertação. 3. ed. Florianópolis: Laboratório de Ensino a Distância da UFSC, 2001.

SOLANKI, J. 2017. **Comparing MongoDB & MySQL.** Disponível em: < https://dev.to/jignesh_simform/comparing-mongodb--mysql-bfa >. Acesso em: 21 junho 2019.

STROZZI, C. (2007). [on-line]. Disponível em <http://www.strozzi.it/cgibin/CSA/tw7/I/en_US/nosql/home%20Page>. Acesso em: 16 junho 2019.

SUISSA, J. **Introdução ao NoSQL.** NoSQL Br. 2010. Disponível em: <http://www.nosqlbr.com.br/introducao-ao-nosql.html.>. Acesso em: 16 junho 2019.

WALTERS, R. **Getting started with Phyton and MongoDB.** 2017. Disponível em: <https://www.mongodb.com/blog/post/getting-started-with-python-and-mongodb>. Acesso em: 17 junho 2019.

Zhu Wei-Ping, Li Ming-Xin, Chen Huan, **Using MongoDB to Implement Textbook Management System instead of MySQL.** IEEE, 2011.