UNIVERSIDADE ESTADUAL DA PARAÍBA
CAMPUS VII – GOVERNADOR ANTONIO MARIZ
CENTRO DE CIÊNCIAS EXATAS E SOCIAIS APLICADAS - CCEA
CURSO DE BACHAREL EM COMPUTAÇÃO

ELIAS FERREIRA JUNIOR

UM ESTUDO COMPARATIVO ENTRE MYSQL E MONGODB PARA ARMAZENAMENTO DE DADOS PROVENIENTES DE SENSORES.

PATOS – PB
2018

ELIAS FERREIRA JUNIOR

UM ESTUDO COMPARATIVO ENTRE MYSQL E MONGOBD PARA ARMAZENAMENTO DE DADOS PROVENIENTES DE SENSORES.

Projeto de trabalho de conclusão de curso apresentado ao Curso de Ciências da Computação da Universidade Estadual da Paraíba, para obtenção de nota parcial para aprovação no componente curricular TCC II.

Orientador: Pablo Ribeiro Suárez.

PATOS – PB

2018

Elias Ferreira Júnior

UM ESTUDO COMPARATIVO ENTRE MYSQL E MONGODB PARA ARMAZENAMENTO DE DADOS PROVENIENTES DE SENSORES

Trabalho de Conclusão de Curso apresentado ao Curso de Bacharelado em Ciências da Computação da Universidade Estadual da Paraíba, em cumprimento à exigência para obtenção do grau de Bacharel em Ciências da Computação.

Aprovado em 29/11/2018

BANCA EXAMINADORA

Msc. Pablo Ribeiro Suárez
(Coorientador)

Pablo Roberto Fernandes de Oliveira
Msc. Pablo Roberto Fernandes de Oliveira
(Examinador)

Amanda Mayara Sobral Rodrigues
Prof. Amanda Rodrigues Sobral
(Examinador)

F383e Ferreira Junior, Elias.
Um estudo comparativo entre MySQL e MongoDB para armazenamento de dados provenientes de sensores. [manuscrito] / Elias Ferreira Junior. - 2018.
25 p. : il. colorido.
Digitado.
Trabalho de Conclusão de Curso (Graduação em Computação) - Universidade Estadual da Paraíba, Centro de Ciências Exatas e Sociais Aplicadas , 2018.
"Orientação : Prof. Me. Pablo Ribeiro Suárez , Coordenação do Curso de Computação - CCEA."
1. Banco de dados. 2. MySQL. 3. MongoDB. I. Título
21. ed. CDD 005.74

Elaborada por Kênia O. de Araújo - CRB - 15/649 BC/UEPB

“Não faça suposições, só está vendo o fim da guerra garoto, não sabe o que levou a isso.”

Kratos (God of War)

DEDICATÓRIA

Dedico este trabalho a todos os meus familiares que me apoiaram muito em todos os momentos da minha vida, em especial a minha mãe Francisca Caetano da Silva Ferreira.

AGRADECIMENTOS

Agradeço em primeiro lugar a Deus que iluminou o meu caminho durante esta caminhada. Agradeço também a minha amiga e namorada Vanessa dos Santos Lucena por toda paciência e apoio durante essa pequena jornada, aos meus amigos de graduação, aos meus orientadores pelas ótimas ideias propostas, e por fim, aos meus companheiros de trabalho, Regineide Torres e Allan Charlys que me ajudaram muito durante a produção.

RESUMO

O mundo vivencia uma constante evolução tecnológica, como atualmente existe uma grande variedade de dispositivos que estão sempre conectados, aplicações baseadas na Internet das Coisas, as quais muitas delas utilizam sensores. Estes dispositivos geram uma grande quantidade de dados, o que implica em problemas que impactam negativamente tanto o desempenho das redes quanto o armazenamento dos dados. Diante do exposto, este trabalho apresenta uma comparação entre o MySql representando os bancos relacionais e o MongoDB representando os não relacionais, para identificar em qual dos bancos é mais viável armazenar os dados gerados por esses sensores. Para isso, foi desenvolvida uma aplicação em Java para simular os dados, com os dados produzidos foram comparados os espaços necessários para armazena-los, bem como verificado o tempo de recuperação dos mesmos. Diante da temática do trabalho, observou-se que o MongoDB obteve um desempenho mais favorável a aplicações dessa natureza.

Palavras-chave: Bancos de Dados, Sensores, MySQL, MongoDB.

SUMÁRIO

1 – INTRODUÇÃO

O mundo vivencia uma constante revolução tecnológica. Diante disto a informação tornou-se indispensável no cotidiano da sociedade, pois acompanha o homem em toda sua trajetória, mudando a cultura e a humanidade com o desenvolvimento e a disseminação de novas tecnologias da informação e comunicação.

Sabe-se que o dia a dia, a sociedade está constantemente conectada aos novos dispositivos como celulares, smartphones, notebooks, tablets, etc. O mercado atual oferece uma enorme variedades destes dispositivos e quase todos possuem algum ou vários deles. Nota-se neste cenário, que a informação tem sido bastante disseminada através dos dispositivos digitais conectados à internet, o que exige uma maior capacidade de gerenciamento e desempenho dos bancos de dados. (LIMA, 2017).

Além dos dados produzidos pela interação humana, dispositivos conectados à internet também produzem uma grande quantidade de dados, a exemplo de sensores. De acordo com Thomazini e Alburqueque (2005) sensores são dispositivos sensíveis à alguma forma de energia do ambiente que pode ser luminosa, térmica, cinética, que se relaciona com as informações de através da física que precisa ser medida, como: temperatura, pressão, velocidade, corrente, aceleração, posição, etc.

Diante o exposto este trabalho tem como objetivo realizar uma comparação de desempenho entre um banco de dados relacional e um banco não-relacional ao se armazenar dados proveniente de sensores, comparando os tempos de recuperação desses dados assim como o tamanho necessário que os dados ocupam de acordo com a estrutura do respectivo banco em que se está armazenado.

Muito tem se falado sobre as vantagens de bancos de dados não estruturados, principalmente com relação à escalabilidade. São comuns relatos vindos de empresas que obtiveram sucesso na migração de bancos de dados relacionais para não relacionais, e alegam terem resolvido seus problemas de escalabilidade. Mas ainda falta um melhor entendimento do contexto e dos limites desses bancos de dados. O resultado de um estudo comparativo como esse é importante pois ajuda desenvolvedores a decidirem qual banco de dados é apropriado para o problema que têm em mãos, além de conhecer um pouco das limitações de cada banco a respeito das características do dado armazenado.

Foi executado experimentos que ajudem a identificar a influência que determinados aspectos

dos bancos de dados MySql e MongoDB. Alguns aspectos a serem estudados são modelo de dados, escalabilidade e tamanho que os mesmos dados ocupam em cada BD. Para isso, foi desenvolvido um programa em Java que gerou os dados que sensores de temperatura produzem, para que seja possível realizar os testes e comparar o comportamento dos respectivos bancos com relação a dados dessa natureza.

De acordo com Basili (1996), a experimentação é utilizada para ajudar a analisar melhor, prever, controlar e melhorar o que é produzido como também o processo de desenvolvimento de software. Portanto a relevância da experimentação na engenharia de software e em outras áreas da computação tem crescido.

Os estudos sistematizados ligados a utilização de experimentos pode agilizar o processo e com isso reduzir o tempo de obter os ganhos almejados por inovações cientificas que são produzidas nas universidades. Neste contexto os desenvolvedores de software tem aumentado a demanda por estudos empíricos visando obter evidencia cientifica referente as tecnologias produzidas, através de experimentos controlados. De acordo com o autor supracitado, um experimento controlado é uma técnica que permite cientistas testarem uma hipótese de pesquisa e o relacionamento entre as variáveis.

O devido trabalho se organiza como segue. A seção 2 traz uma introdução aos conceitos que contribuem para a motivação e o desenvolvimento do presente estudo. Na seção 3, são apresentas as características dos bancos de dados utilizados no experimento. A seção 4 apresenta o ambiente de testes e as ferramentas que foram utilizadas, por fim a seção 5 apresenta a conclusão do estudo, mostrando um pouco das limitações durante o desenvolvimento, assim como as contribuições e possíveis pesquisas futuras.

2 – TRABALHOS RELACIONADOS

Um fator que contribuiu para a criação deste trabalho é a pouca quantidade de trabalhos relacionados, além de que o modelo de dados usado nesta pesquisa e uma das características comparadas não foram encontradas em outros trabalhos. Matheus (2018) faz uma comparação de desempenho utilizando uma aplicação que realizava operações nos mesmos bancos abordados nessa pesquisa. Como resultado foi observado que o MongoDB apresentou vantagens em quase todos os testes.

Sharvari (2016) compara SQL e NoSQL para aplicações de Internet das Coisas em pequena escala de em um sistemas aspersão de agua e investiga se o NoSQL funciona melhor do que SQL em diferentes cenários. Foi observado que cada banco possui suas vantagens e desvantagens, assim

ele concluiu que a escolha do banco para o contexto de internet das coisas possui maior dependência do modelo de aplicação.

Gerson realizou um comparativo entre o modelo relacional e não relacional utilizando o MySql e MongoDB no armazenamento de uma aplicação real para o gerenciamento de matriculas de uma instituição, onde os resultados apresentados pelo MongoDB mostrou um desempenho melhor para as condições testadas.

Como citado anteriormente, existem alguns trabalhos relacionados que apresentam comparações entre os modelos abordados neste artigo, assim como os mesmos bancos de dados utilizados, entretanto são poucos e ainda não concluem em uma melhor escolha de modelo. Este trabalho irá agregar novas experiências e contribuir para que novos programadores decidam qual modelo ou banco é mais apropriado para o problema que possuem.

3 – EXPERIMENTO CONTROLADO

3.1 MySql

O MySQL é um banco de dados relacional gratuito, eficiente e otimizado para aplicações Web, é desenvolvido e mantido pela empresa MySQL AB, que também oferece uma versão comercial (paga). Esse SGBD também é multi-plataforma, sendo compatível com o Windows, Linux, BSDs(Berkeley Software Distribution), entre outros sistemas operacionais. As tabelas criadas podem ter tamanho de até 4 GB. Fora isso, o MySQL é compatível com várias linguagens de programação, tais como PHP, C, Java, Visual Basic, entre outros.

O MySQL é um banco de dados relacional, que armazena dados através de tabelas separadas, em vez de colocar os dados em um só local, proporcionando velocidade e flexibilidade. O servidor MySQL foi criado para lidar com bancos enormes e com mais rapidez, é usado em ambiente de produção de alta escala e de maneira bem sucedida, mesmo estando em desenvolvimento constante, além disso oferece um grande leque de funcionalidades. A conectividade, velocidade, e segurança faz com que esse banco de dados seja adaptável para acessar bancos via internet.

3.2 MongoDB

O MongoDB é um banco de dados NoSql orientado a documentos, escrito na linguagem C++. É um projeto OpenSource e impulsionado pela empresa 10gen Inc, que também oferece serviços licenciados para o MongoDB. Segundo os próprios desenvolvedores a principal finalidade do seu

Banco de Dados é encerrar o espaço entre rapidez e escalabilidade (STRAUCH, 2010, p. 76).

O projeto do MongoDB iniciou-se por volta de 2007 com a startup 10gen, no começo eles trabalhavam com uma plataforma composta de um banco de dados que quando necessário seria escalado e um servidor de aplicativos. A plataforma 10gen foi desenvolvida para lidar com a gestão de infraestrutura de hardware e software automaticamente e principalmente para resolver problemas com o dimensionamento, desta maneira possibilitou que os desenvolvedores voltassem sua atenção apenas nos códigos dos aplicativos a serem desenvolvidos. Em uma analise a 10gen descobriu-se o interesse de diversos usuários no banco de dados, isso fez com que os empenhos fosse exclusivamente para o banco de dados, que veio a se tornar o MongoDB (BANKER, 1012, p. 5).

As características que apresentam maior importância para o comparativo a ser realizado neste trabalho são duas, escalabilidade e recuperação dos dados, as mesmas são mostradas no quadro abaixo.

Tabela1: Quadro comparativo de características de cada banco.

Características	MySQL	MongoDB
Escalabilidade	Vertical, refere-se à capacidade de aumentar o desempenho de nó único com a adição de recursos, como memória ou processadores para o mesmo nó.	Horizontal, o número de nós (servidores) é aumentado para compartilhar a carga do sistema.
Recuperação de dados	Para pesquisar dados de diferentes tabelas, o usuário deve usar instruções JOIN onde será necessário um maior tempo para combinar as tabelas e exibir os dados.	Os dados são armazenados em forma de objetos que conterão todos os dados. Isso elimina o processo de combinar e depois exibi-los, economizando tempo de resposta.

3.3 Ambiente de Testes

Com a finalidade de se obter resultados relevantes, precisos e evitar problemas durante os testes o possível foi tomada a decisão de não virtualizar a máquina para o teste, isso devido à necessidade de deixar o ambiente de cada BD o mais homogêneo possível, os testes serão realizados no mesmo equipamento.

A máquina utilizada para os teste foi um notebook SAMSUNG ATIVBOOK6, com processador de 3ª geração Intel(R) Core(TM) i5-3230M de quatro núcleos e oito threads e clock de 2.60GHz, 8 GB de memória RAM DDR3 1600 MHz, placa de vídeo AMD Radeon HD 7730M com 2 GB de memória RAM dedicada, HD 120gb SSD e Sistema Operacional Windows 10.

Para o desenvolvimento dos testes foi utilizada a versão 5.5.17 do MySQL, com suas configurações padrões. Já a versão MongoDB que foi utilizada para os testes, foi a versão 4.0.2 com suas configurações padrões.

Os dados que foram gravados simulam os dados coletados por um sensor de temperatura, tanto no MongoDB quanto no MySql, foram gravadas a chave primaria da tabela/documento, a data e a Hora da coleta do dado, a temperatura registrada pelo sensor e o número do sensor em que foi coletado o registro de temperatura.

A figura a seguir representa a abstração destes dados.

Figura 1: MetaDados.

Fonte: Autoria própria.

A principal finalidade é avaliar se existe uma diferença com relação ao espaço no Hard Disk(HD) necessária para armazenar uma mesma quantidade de informações nos dois bancos de dados propostos. Outro aspecto a ser avaliado, é o custo temporal necessário para se recuperar os dados salvos, avaliando o tempo de recuperação de todos os registros gravados, como também os registros de um sensor especifico.

Os dados gerados usaram o modelo mostrado anteriormente, a partir dele foi implemento o

código em Java, onde foram desenvolvidas uma classe de comunicação para cada banco, a classe objeto com os dados que foram salvos, e a classe com o laço principal que simula os valores coletados por um sensor de temperatura.

Foi escolhida a linguagem Java devido a fácil comunicação com os bancos utilizados no trabalho. Cada classe de comunicação necessita apenas do driver respectivo de cada banco, para o MySql foi usado o MySql Connector versão 5.1.39 e para o MongoDB foi usado o MongoDriver 3.0.4.

Figura2: Código principal.

```
public static void main(String[] args) throws Exception {
        //Inicia Conexão com o MySql
        dao.DaoSensoresTemperatura DaoSensor = new DaoSensoresTemperatura();
        //Inicia Conexão com MongoDB
        MongoConection mc = new MongoConection();
        //Instancia o Objeto
        SensoresTemperatura dados = new SensoresTemperatura();
        Date dataInicial = new Date();
        int numSensor = 1;          double LowerLimit = 25.0;        double UpperLimit = 45.0;        double var1;
        //24horas*60minutos*6registros * 10 sensores * 30dias  por minuto
        for (int i = 1; i <= (((24 * 60 * 6) * 10) * 30) * 12; i++) {
        //Gera a temperatura dentro do intervalo
            var1 = LowerLimit + new Random().nextDouble() * (UpperLimit - LowerLimit);
        //Prechence o Numero do sensor no Objeto
            dados.setNumSensor(numSensor);
        //Preenche a data no objeto
            dados.setDataHora(StringUtil.formataData(dataInicial));
        //Preenche a tempereratura gerada anteriormente
            dados.setTemperatura(truncar(var1, 2));
        //Preenche o Codigo do dado
            dados.setCodigo(i);
        //Salva o objetoos no MySql
            DaoSensor.salvar(dados);
        //Salva o objeto no MongoDB
            mc.inserir(dados);
        //Apaga a referencia do objeto
            dados = null;
        //Instancia um Novo objeto
            dados = new SensoresTemperatura();
            if (numSensor < 10) {
                numSensor++;
            } else {
        //Incrementa 10segundos no data, e seta o numero do sensor de novo para 1
                dataInicial.setTime((dataInicial.getTime() + 10000));
                numSensor = 1;
            }
        }
    }
```

Fonte: Autoria Própria

Estes dados foram gerados randomicamente através de funções do próprio Java, usando um limite superior e inferior para que o "random" (função do java) não gerasse um valor que fugisse da realidade. O laço principal gera as informações, preenche o objeto e o grava nos dois bancos usando o método das respectivas classes de comunicação, o tempo de execução para a geração desses registros foi pouco mais de 18 horas. Como o foco do trabalho é o comportamento dos bancos em relação ao espaço de armazenamento e o desempenho na recuperação das informações, não será testado o tempo de inserção, remoção, ou atualização desses dados devido a maneira que foi escolhida para a geração dos registros.

3.4 Métricas avaliadas

- **Tamanho necessário no HD**

  Com relação a quantidade de espaço, foi utilizada a maneira respectiva de cada banco para resgatar o tamanho ocupado no HD após cada etapa de testes, fazendo assim, uma comparação do comportamento das ferramentas estudas em relação ao armazenamento das mesmas informações. Para o MySQL foi utilizada a consulta SQL mostrada na figura abaixo:

Figura3: SQL para recuperação do tamanho do BD no MySql

```
mysql> SELECT table_schema 'sensores', ROUND(SUM( data_length + index_length ) / 1024 / 1024 / 1024, 3)
    -> "Size in GB" FROM information_schema.TABLES GROUP BY table_schema LIKE 'sensores';
+--------------------+------------+
| sensores           | Size in GB |
+--------------------+------------+
| information_schema |      0.477 |
| sensores           |      0.000 |
+--------------------+------------+
2 rows in set (0.45 sec)
```

Fonte: Autoria Própria.

Para o MongoDB foi usado o comando “show dbs” que mostra todos os bancos existentes e o tamanho respectivo de cada banco, como mostra a figura a seguir:

Figura 4: Função para recuperação do tamanho do BD no MongoDB

```
> show dbs
Sensores  0.003GB
admin     0.000GB
config    0.000GB
local     0.000GB
teste     0.000GB
>
```

Fonte: Autoria Própria

- **Extração do tempo das instruções**

Para cada banco de dados foi utilizada a maneira respectiva para se resgatar o tempo de execução de cada busca realizada. No MySQL foi utilizado o prompt de comando, que retorna o tempo em segundos ao final da operação realizada, já para o MongoDB a função utilizada foi a função “explain”, que ao final da busca ele mostra o tempo que foi levado para executa-la.

4 - TESTES E RESULTADOS

Algumas nomenclaturas podem mudar do modelo relacional usado no Mysql, para o modelo não relacional, como por exemplo as tabelas, que no MongoDB são chamadas de collections.

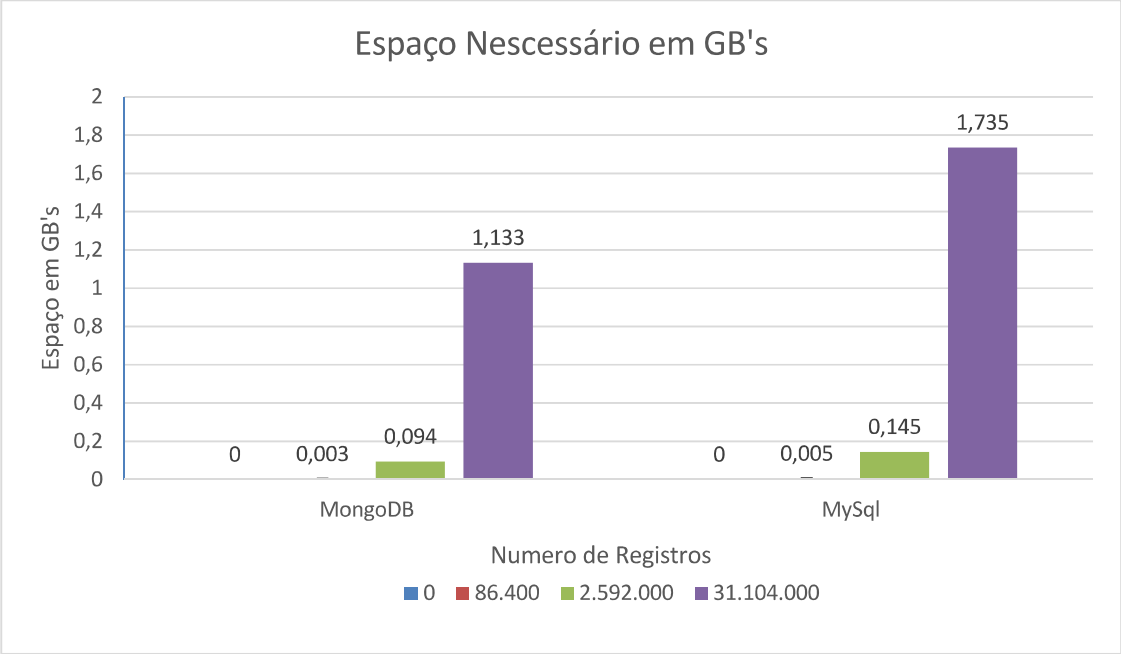
O primeiro teste realizado foi o do tamanho, foi a notado o tamanho antes dos dados serem inseridos e logo após sua inserção. Dentro dessa perspectiva no primeiro caso foram 86.400 registros salvos que equivalem a dados de 10 sensores de temperatura coletando informações com intervalo de captura de 6s durante 24horas.

No segundo caso 2.592.000 registros foram salvos, que representam os dados dos mesmos 10 sensores com o mesmo intervalo de captura (6s), entretanto a duração de 1 mês. Por fim no último caso foram salvos registros que equivalem aos dados de 1 ano dos 10 sensores capturando no intervalo de 6 segundos, que resultou em 31.104.000 registro.

4.1 Tamanho necessário no HD

Ao final de cada teste foi coletado a quantidade de espaço que estava sendo utilizada para o armazenamento dos dados, o resultado dos testes está representado no gráfico abaixo. Quanto menor o espaço necessário melhor é o desempenho do banco nesse quesito.

Gráfico 1: Resultado de teste de espaço ocupado.



Ao analisar o gráfico notamos que o MongoDB tem um desempenho melhor que o Mysql devido a quantidade do espaço necessária para armazenar a mesma quantidade de informações é menor desde o início dos testes, e a medida em que os dados crescem a diferença continua a crescer tornando-se assim a melhor opção quando se preocupa com o a quantidade de espaço utilizado para o armazenamento dos dados.

4.2 Velocidade de Recuperação dos Dados

A segunda métrica de comparação foi avaliada através da análise do tempo necessário de cada banco de dados para a recuperação dos registros. Os testes de busca foram divididos em duas partes, já que podemos fazer diferentes tipos de consulta. Foram feitas buscas por todos os registros e buscas por registros de um sensor especifico, ou seja, pelo atributo número do sensor. A cada vez que o teste é executado, a máquina era reiniciada para que os resultados não fossem afetados pelo buffer de memória.

4.2.1 Recuperação de Todos os Registros

Nestes testes era realizada uma busca de todos os registros na tabela utilizando o "select * from". No MongoDB foi utilizada as funções "find" e "explain". Os resultados das baterias de testes foram registrados nas tabelas 2 e 3.

Tabela 2: Resultados obtidos da busca no MongoDB usando find() e explain().

MongoDB			
Busca de Todos os Registros			
	86.400 Registros	**2.592.000 Registros**	**31.104.000 Registros**
1	0,035s	0,785s	10,513s
2	0,032s	0,776s	10,758s
3	0,031s	0,781s	10,657s
4	0,037s	0,775s	10,825s
5	0,030s	0,776s	10,705s
6	0,038s	0,779s	10,615s
7	0,034s	0,776s	10,585s

8	0,033s	0,772s	10,597s
9	0,031s	0,773s	10,815s
10	0,032s	0,783s	10,706s
Média	0,033s	0,777s	10,677s

Tabela 3: Resultados obtidos na busca no Mysql usando "Select * from".

MySql			
Busca de Todos os Registros			
	86.400 Registros	**2.592.000 Registros**	**31.104.000 Registros**
1	0,110s	3,26s	44,05s
2	0,120s	3,21s	47,23s
3	0,115s	3,31s	45,38s
4	0,117s	3,20s	46,75s
5	0,118s	3,22s	44,88s
6	0,113s	3,22s	45,54s
7	0,114s	3,23s	48,33s
8	0,111s	3,24s	45,63s
9	0,112s	3,25s	44,94s
10	0,110s	3,21s	45,78s
Média	0,114s	3,23s	45,851s

A tabela 2 apresenta o resultado da bateria de testes de busca completa na collection do MongoDB, a tabela 3 apresenta os resultados do MySQL. O tempo de execução estão em segundos e o último campo da tabela é a média utilizada no gráfico de resultados.

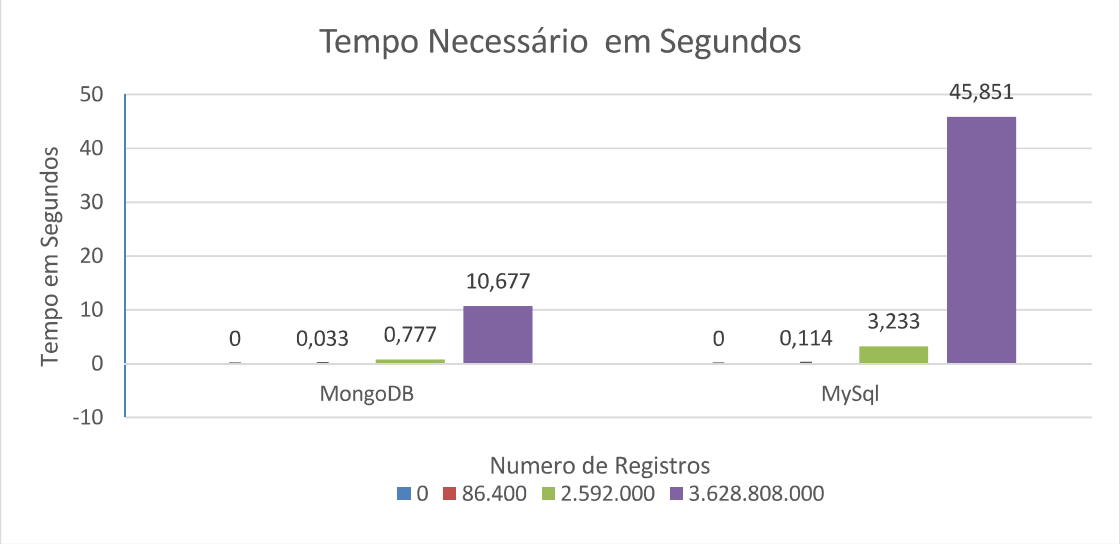
Tabela 4: Comparativo dos resultados da tabela 2 e 3

Comparativo						
Busca de Todos os Registros						
	86.400 Registros		**2.592.000 Registros**		**31.104.000 Registros**	
	MongoDB	MySQL	MongoDB	MySQL	MongoDB	MySQL
1	0,035s	0,110s	0,785s	3,26s	10,513s	44,05s
2	0,032s	0,120s	0,776s	3,21s	10,758s	47,23s
3	0,031s	0,115s	0,781s	3,31s	10,657s	45,38s
4	0,037s	0,117s	0,775s	3,20s	10,825s	46,75s
5	0,030s	0,118s	0,776s	3,22s	10,705s	44,88s
6	0,038s	0,113s	0,779s	3,22s	10,615s	45,54s
7	0,034s	0,114s	0,776s	3,23s	10,585s	48,33s
8	0,033s	0,111s	0,772s	3,24s	10,597s	45,63s
9	0,031s	0,112s	0,773s	3,25s	10,815s	44,94s
10	0,032s	0,110s	0,783s	3,21s	10,706s	45,78s
Média	0,033s	0,114s	0,777s	3,23s	10,677s	45,851s

Legenda: Verde = Ótimo, Amarelo = Aceitável, Vermelho = Ruim.

A comparação dos resultados fica melhor representada no Gráfico 2, o tempo é representado em segundos e quanto menor o valor, melhor é o desempenho do banco de dados.

Gráfico 2: Resultado de teste de Busca de Todos os Registros.



Nota-se que o MongoDB possui vantagem na leitura de todos os dados desde o início dos testes, a medida que o volume dos dados aumenta, a diferença de desempenho entre os BD's cresce da mesma maneira. A partir da busca com 2592000 registros, o tempo necessário para o retorno das informações é 4 vezes menor no MongoDB em relação ao MySQL, já na busca de 31.104.000 registros a diferença permanece.

4.2.2 Recuperação de dados de um sensor especifico

Nestes testes era realizada uma busca de todos os registros de um sensor especifico na tabela utilizando o "select * from where numsensor = X". No MongoDB foi utilizada a função find() e o explain(). Os resultados das baterias de testes foram registrados nas tabelas 5 e 6.

Tabela 5: Resultados obtidos na busca dos dados de um mesmo sensor no MongoDB.

MongoDB			
Busca de Registros de um sensor especifico			
	86.400 Registros	**2.592.000 Registros**	**31.104.000 Registros**
1	0,058	1,209	27,084
2	0,060	1,253	27,764
3	0,059	1,121	25,853
4	0,057	1,175	26,254
5	0,061	1,285	26,785
6	0,060	1,175	27,258
7	0,058	1,242	27,562
8	0,057	1,286	27,050
9	0,059	1,208	26,845
10	0,060	1,230	27,125
Média	0,058	1,218	26,958

Tabela 6: Resultados obtidos na busca dos dados de um mesmo sensor no Mysql.

MySql			
Busca de Registros de um sensor especifico			
	86.400 Registros	**2.592.000 Registros**	**31.104.000 Registros**
1	0,051	1,38	20,22
2	0,052	1,35	21,57
3	0,053	1,35	20,87
4	0,057	1,37	20,34
5	0,052	1,32	21,05
6	0,056	1,37	22,09
7	0,055	1,38	23,15
8	0,053	1,38	22,67
9	0,052	1,32	21,95
10	0,051	1,32	20,79
Média	0,053	1,35	21,47

A tabela 5 apresenta o resultado da bateria de testes de busca dos registros de um respectivo sensor na collection do MongoDB, a tabela 6 apresenta os resultados da busca dos mesmos dados no MySQL usando a cláusula “Where”. O tempo de execução estão em segundos e o último campo da tabela é a média utilizada no gráfico de resultados.

Tabela 7: Comparativo dos resultados da tabela 6 e 7

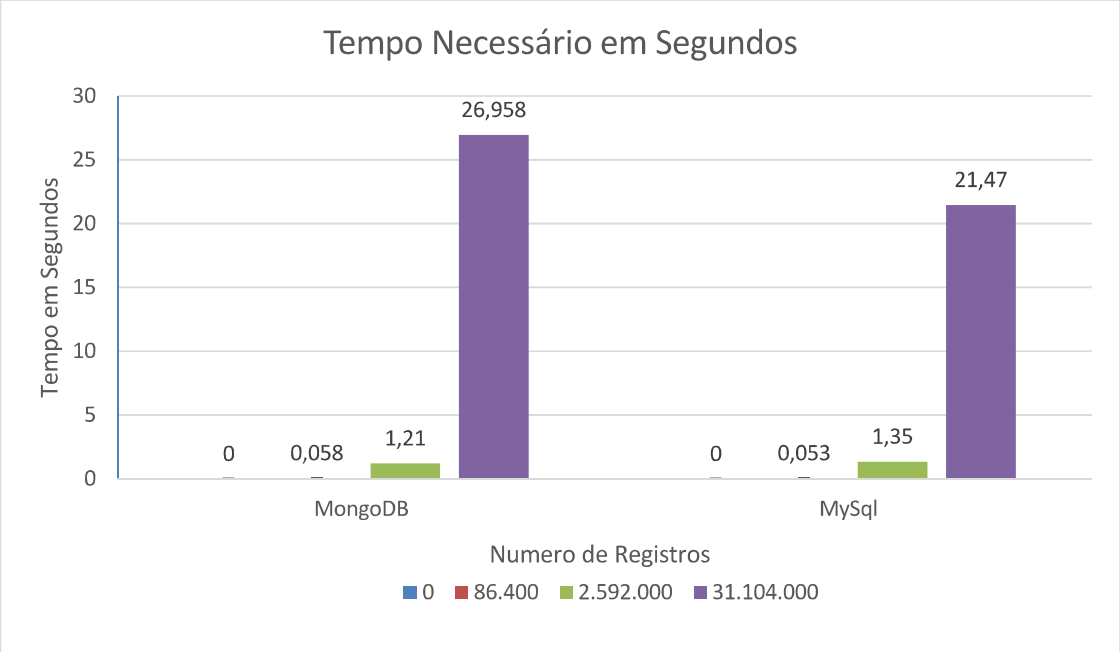
Comparativo						
Busca de Todos os Registros						
	86.400 Registros		**2.592.000 Registros**		**31.104.000 Registros**	
	MongoDB	MySQL	MongoDB	MySQL	MongoDB	MySQL
1	0,058	0,051	1,209	1,38	27,084	20,22
2	0,060	0,052	1,253	1,35	27,764	21,57
3	0,059	0,053	1,121	1,35	25,853	20,87
4	0,057	0,057	1,175	1,37	26,254	20,34
5	0,061	0,052	1,285	1,32	26,785	21,05

6	0,060	0,056	1,175	1,37	27,258	22,09
7	0,058	0,055	1,242	1,38	27,562	23,15
8	0,057	0,053	1,286	1,38	27,050	22,67
9	0,059	0,052	1,208	1,32	26,845	21,95
10	0,060	0,051	1,230	1,32	27,125	20,79
Média	0,058	0,053	1,218	1,35	26,958	21,47

Legenda: Verde = Ótimo, Amarelo = Aceitável, Laranja = Desfavorável.

A comparação dos resultados apresenta uma melhor compreensão no Gráfico 3, o tempo é representado em segundos e quanto menor o valor, melhor é o desempenho do banco de dados.

Gráfico 3: Resultado de teste de Busca de Todos os Registros de um sensor especifico.



Percebe-se que com uma quantidade de registros menor o MongoDB apresenta um desempenho semelhante ao do MySql, entretanto com o aumento no volume dos dados, o desempenho do MySql em relação ao Mongo é melhor. Essa diferença é observada com maior clareza na última busca, a que possui 31.104.000 registros, no MySql a busca foi feita com 5 segundos a menos que no MongoDB.

CAPITULO 5 – CONCLUSÃO

A utilização de bancos de dados não relacionais veem crescendo devido seu modelo não estruturado que faz com que uma quantidade absurda de dados possa ser armazenada sem muitos problemas, e com um espaço necessário para o armazenamento de dados menor que no MySql como foi comprovado na pesquisa realizada. Embora os bancos de dados não relacionais apresentem um crescimento no mercado, o modelo relacional ainda se mostra muito eficiente em seu desempenho, quando foi necessário fazer comparações o modelo relacional mostrou uma leve vantagem.

O objetivo deste trabalho foi comparar o desempenho do MongoDB e do MySql utilizando operações de busca, e comparando o espaço necessário que cada banco necessita para gravar os dados produzidos por sensores, o escolhido foi o sensor de temperatura. Com sua conclusão, fica evidente que embora o MongoDB tenha sido criado para diminuir alguns problemas de performance, o modelo relacional ainda assim apresenta um desempenho satisfatório, e em algumas ocasiões até melhor levando em consideração os resultados observados nos testes de busca, onde quando foi recuperado os registros de um sensor especifico utilizando a clausula “Where”, o tempo de execução da consulta apresentado pelo MySql foi menor que o tempo apresentado pela função similar do MongoDB.

No teste de armazenamento, o MongoDB apresentou um desempenho melhor comparado com o MySQL. O espaço necessário para se armazenar a mesma quantidade de informações sempre foi menor no banco de dados não relacional, chegando a utilizar cerca de 1/3 de espaço a menos que no modelo relacional do MySql.

Desta forma destaca-se que o MongoDB apresenta um melhor desempenho em relação ao espaço de armazenamento e na busca de todos os dados. O que diante do tema proposto e do outro banco comparado, o MongoDB se torna a melhor opção para o armazenamento deste tipo de informação.

Vale ressaltar que em nenhum dos dois bancos de dados analisados foi feito uma otimização durante os testes o que pode ter influenciado no desempenho dos mesmos. Outro ponto que necessita ser observado é que foram testados apenas 2 bancos de dados, existem outras opções tanto de bancos relacionais como não relacionais, o que pode ajudar a encontrar um BD que se comporte melhor que os utilizados na pesquisa. Além disso devido ao tempo necessário para se coletar os dados reais, foi avaliado e descido que a melhor alternativa era simula-los, além disso o tempo de

inserção em um ambiente normal depende das tecnologias utilizadas além do banco escolhido e por isso não foi comparado o tempo de inserção desses dados.

Como trabalhos para o futuro, ficam a possibilidade de se desenvolver uma aplicação onde os dados coletados sejam verídicos, outro ponto que também pode ser observado é em um banco de dados que possuam outros tipos de dados e que seja necessário uma junção de tabelas. Além disso pode ser realizado uma comparação entre outros Bancos de dados, devido à grande quantidade de opções existentes no mercado.

A COMPARATIVE STUDY BETWEEN MYSQL AND MONGODB FOR STORING DATA FROM SENSORS.

ABSTRACT

The world of living is a technological reality, as the Indian there is a wide variety of products that are always connected, the Internet applications of Things, such as the often used sensors. These devices generate a large amount of data, which can, in some cases, negatively impact the performance of networks for data storage. In view of the above, this work is a comparison between MySql and related banks and MongoDB, making the databases, to identify the most flexible banks, the data generated by these sensors. For this, a Java application was applied to the data calculation, with the data compared to the spaces needed to store, as well as the recovery time of the same. In view of the thematic of the work, it was observed that the MongoDB obtained a more favorable performance to an application of this nature.

Key Words: Databases, Sensors, MySQL, MongoDB.

REFERÊNCIAS

BANKER, Kyle. **MongoDB in action**. Manning Publications Co., 2011.

BASILI, Victor R. The role of experimentation in software engineering: past, current, and future. In: **Proceedings of the 18th international conference on Software engineering**. IEEE Computer Society, 1996. p. 442-449.

GYŐRÖDI, Cornelia et al. A comparative study: MongoDB vs. MySQL. In: **Engineering of Modern Electric Systems (EMES), 2015 13th International Conference on**. IEEE, 2015. p. 1-6.

LAUDON, K.C.; LAUDON, J. P. **Sistemas de informação: organizando as informações**: arquivos e bancos de dados. 4. ed. Rio de Janeiro: J.C. Editora, 1999.

LIMA, Welton Dias. A Internet das Coisas. **TECNOLOGIAS EM PROJEÇÃO**, v. 8, n. 2, p. 67-78, 2017.

MUNIZ, Matheus Henrique et al. Comparação de Performance de Processamento entre Bases de Dados Relacionais e Bases de Dados NoSql. **Seminários de Trabalho de Conclusão de Curso do Bacharelado em Sistemas de Informação**, v. 3, n. 1, 2018.

RAUTMARE, Sharvari; BHALERAO, D. M. MySQL and NoSQL database comparison for IoT application. In: **Advances in Computer Applications (ICACA), IEEE International Conference on**. IEEE, 2016. p. 235-238.

STRAUCH, Christof; SITES, Ultra-Large Scale; KRIHA, Walter. NoSQL databases. **Lecture Notes, Stuttgart Media University**, v. 20, 2011.

THOMAZINI, Daniel; ALBUQUERQUE, Pedro Urbano Braga de. Sensores industriais: fundamentos e aplicações. **São Paulo**, v. 3, p. 32, 2005.

UFERSA, Gerson Viana Marques; MEDEIROS, Claudio R.; PEREIRA, Jeferson Queiroga. Análise comparativa de desempenho de aplicação Java com persistência em Banco de Dados MySQL e MongoDB.