ANÁLISE COMPARATIVA DE DESEMPENHO DE CONSULTAS ENTRE UM BANCO DE DADOS RELACIONAL E UM BANCO DE DADOS NÃO RELACIONAL

Gilmar José da Silva
gilmar.silva@outlook.com
Júlio Cesar Oliveira Ferreira
julioengdc@gmail.com
Orientador (a): Antônio Manoel Batista da Silva
antonio.silva@uniube.br
Coorientador (a): André Luís Silva de Paula
andre.paula@uniube.br

RESUMO

Atualmente, a tecnologia abrange diversas áreas de conhecimento e a informação está disponível em maior volume do que outrora, seja considerando aspectos pessoais, profissionais ou empresariais. No caso das empresas, esse aumento de volume de informação é significativo à medida que seus processos internos são automatizados, informatizados e que utilizam de alguma tecnologia para o armazenamento da informação. Os bancos de dados são ferramentas responsáveis pelo armazenamento de informações geradas por determinado *software* e são estruturas importantes neste contexto. Na busca pelo aprimoramento do armazenamento e no retorno da busca pelas informações, foram desenvolvidas diversas tecnologias de armazenamento e, entre elas, o modelo de banco de dados relacional, paradigma utilizado pela maioria dos sistemas há décadas, e um conceito mais recente que é o modelo de banco de dados não relacional, que tem repercutido no mercado por ser mais rápido em base de dados de grande volume. Este trabalho procura comparar a diferença de desempenho entre estes dois tipos de modelo com uma simulação de consulta através de um software desenvolvido em *.net* para este único objetivo. Para tanto, foram escolhidos os bancos SQL *Server*, como representante do modelo relacional, e o *MongoDB*, representando o modelo não relacional. Os testes foram realizados considerando 6 grupos quantitativos crescentes de registros e os resultados foram analisados em termos de tempo médio de resposta.

Palavras-chaves: Desempenho de Banco de dados, Banco de dados relacional, Banco de dados não relacional.

COMPARATIVE SEARCH PERFORMANCE ANALYSIS BETWEEN A RELATIONAL AND A NON-RELATIONAL DATABASE

ABSTRACT

Nowadays, the technology covers the vast areas of knowledge and the information is available in a much higher volume than before, considering a personal, professional or corporate aspect. In the later, there is a significant rise the volume of information considering that internal procedure processes have been automated, computerized that use some type of data storage technology. Database are tools responsible for the storage of information generated by a given software and are structures of utmost importance in the context. In the quest for information storage enhancement, there has been developed different storage technologies. Among them the relational database, which has been the paradigm used by most systems over decades, and a more recent concept, the non-relational database which has had repercussions for being faster in high volume data banks. This study compares the differences between these two models with a simulated search through a software developed using .net for this sole purpose. There are, it was chosen the SQL SERVER database, as the relational model, and the MongoDB for the non-relational. Tests were conducted considering 6 growing quantitative record groups and the results were considered in terms of average response time.

Keywords: Database performance, Relational database, Non- relational database.

1. INTRODUÇÃO

Atualmente, nos encontramos num contexto onde a tecnologia abrange praticamente todos os processos e o volume de informações cresce a cada instante, tanto nas redes sociais, quanto nos sites de multimídia e também nas empresas que utilizam algum tipo de software para gerenciamento. Para Bazzotti e Garcia (2010), “o mundo vive na era da informação, exigindo das organizações uma gestão estratégica eficiente, a qual pode ser facilitada pela utilização de recursos inteligentes oferecidos pela tecnologia de informação e sistemas de informação”.

Segundo Date (2004, p. 32), “um sistema de banco de dados é basicamente apenas um sistema computadorizado de manutenção de registros. O banco de dados, por si só, pode ser considerado como o equivalente eletrônico de um armário de arquivamento”. E essas informações arquivadas sempre tem um alto grau de importância, pois, uma vez armazenada, a qualquer instante é necessário resgatá-la. E a forma e a velocidade com que essa informação será resgatada, estão relacionadas ao paradigma de banco de dados utilizado. Nesse projeto, utiliza-se o paradigma Relacional e o Não Relacional, conhecido também como NoSQL (not only SQL), para comparar, em termos de performance, a recuperação de dados na base.

Além do banco de dados, conhecer o SGBD (Sistema de Gerenciamento de banco de dados) é fundamental no desenvolvimento de sistemas de informação. E tem como seu principal objetivo, segundo Silberschartz (1999, p. 102), “proporcionar um ambiente conveniente e eficiente para recuperar e armazenar informações em um banco de dados, permitindo uma facilitação quanto à manutenção de dados possibilitando seu compartilhamento por diversas aplicações”. Para cada paradigma de Banco de Dados apresentado, utiliza-se um SGBD como forma de tratamento da informação.

A utilização de um banco de dados tornou-se um requisito necessário na maioria das empresas para a gestão da informação. Geralmente, elas utilizam um software de gestão, para administrar todas as suas transações ao longo dos anos e a maioria delas utilizam como paradigma de banco de dados o modelo Relacional, que é muito bem aceito onde a base de dados é menor e requer buscas de informações de maneira mais simples.

Entretanto, quando a base de dados começa a aumentar, considerando o crescimento das organizações e manutenção de um histórico das transações, o modelo relacional de banco de dados começa a apresentar lentidão na consulta de informações. Assim, este trabalho investigou as vantagens e as desvantagens na utilização do paradigma de banco de dados NoSql, frente ao modelo relacional. No projeto, são realizadas simulações para obter o tempo de resposta em

cada consulta, considerando sua realização nos dois paradigmas de banco de dados, por meio de um software. Assim, apresenta-se de forma clara os resultados obtidos e permitindo a identificação do modelo mais rápido, em relação a complexidade da busca da informação.

A linguagem escolhida para o software foi *.net*, pois, além de ser uma linguagem orientada a objetos, gratuita e multiplataforma, é compatível com ambos os modelos de Banco de Dados, além de possuir uma representativa parcela do mercado, conforme estudo da BUSINESS 2 COMMUNITY, 2017. Entre os bancos de dados propostos para este estudo, optou-se pelo SQL *Server*, como Banco de dados Relacional, por ser gratuito e estar entre os 3 bancos de maior destaque no ranking mundial na sua categoria, segundo a DB-ENGINES, 2017. Como Banco de Dados NoSql, optou-se pelo *MongoDB*, também por ser gratuito e por ser bastante utilizado na comunidade, além de ocupar o primeiro lugar no ranking mundial de sua categoria, também segundo a DB-ENGINES, 2017.

O objetivo deste trabalho é comparar e demonstrar, por meio de exemplificações práticas em uma base de dados disponibilizada gratuitamente pela Microsoft, dois paradigmas de Banco de Dados, o modelo Relacional e o Não Relacional, por um software desenvolvido em linguagem *.net.*

2. LISTA DE ABREVIATURAS E SIGLAS

A seguir, são apresentadas as abreviaturas e siglas utilizadas neste trabalho.

NoSQL	*Not Only SQL*
SQL	*Structure Query Language*
SGBD	Sistema de Gerenciamento de Banco de Dados
JSON	*JavaScript Object Notation*
BSON	*Binary* JSON
API	*Application Programming Interface*

3. REVISÃO BIBLIOGRÁFICA

Neste tópico serão detalhados conceitos sobre os modelos relacionais e NoSQL, objetos deste estudo, e um breve comparativo entre eles, bem como sobre as características de cada banco que será utilizado para a análise.

3.1 Banco de dados relacional

O modelo de Banco de Dados Relacional é o mais utilizado nas últimas décadas, e o estudo da DB-ENGINE, 2017, mostra que no ranking geral de 2017, sete bancos relacionais estão entre os dez mais utilizados. Trata-se de um paradigma muito confiável e organizado e a estrutura fundamental do modelo relacional é a relação.

Para Ferreira, Italiano, Takai (2005, p. 6),

> O Modelo relacional revelou-se ser o mais flexível e adequado ao solucionar os vários problemas que se colocam no nível da concepção e implementação da base de dados. A estrutura fundamental do modelo relacional é a relação (tabela). Uma relação é constituída por um ou mais atributos (campos) que traduzem o tipo de dados a armazenar. Cada instância do esquema (linha) é chamada de tupla (registro).

Desta forma, revela-se que o modelo relacional apresenta uma grande organização dos dados, e, se usado de forma correta, evita-se a repetição de informações na base de dados.

Segundo França, Goya, Puga (2014, p. 55),

> Entre os pontos que merecem destaque no modelo relacional, estão:
>
> - **Construção de níveis múltiplos de integridade:** a integridade dos dados, neste modelo, é construída em nível de campo, para assegurar a acuracidade dos dados; em nível de tabela para assegurar que os registros não estão duplicados e para detectar a falta de valores na chave primária; em nível de relacionamentos, para assegurar que o relacionamento entre duas tabelas é válido e, por fim, em nível do negócio, para assegurar que os dados estão acurados de acordo com o negócio. Dessa forma, a garantia de acuracidade e consistência dos dados ocorre graças aos vários níveis de integridade que podem ser impostos ao banco de dados.
> - **Independência física e lógica da aplicação do banco de dados:** nem as mudanças efetuadas pelo usuário no projeto lógico do banco de dados, tampouco as mudanças físicas de implementação do banco de dados efetuadas pelos fabricantes de software, prejudicarão as aplicações construídas no modelo relacional.
> - **Fácil armazenamento de dados:** ao comando do usuário, os dados podem ser armazenados em uma em várias tabelas do banco de dados.

Neste contexto, tem-se, então, uma visão sobre as principais vantagens do modelo relacional, o que, de fato, explica de forma clara a confiança em se utilizar esse paradigma. A relação entre tabelas torna a estrutura muito prevenida contra possíveis falhas, como a duplicidade de informações, por exemplo, assim como também pode-se ter a certeza de que determinada tabela não ficará vazia, por meios de configurações.

Para manipulação das informações em um banco de dados de forma rápida e simples, utiliza-se o SGBD (Sistema de gerenciamento de banco de dados). Para Medeiros (2013 p. 10), "os SGBD funcionavam como entidades separadas do sistema, permitindo que funções de armazenamento e recuperação de dados pudessem ser feitas em seu próprio domínio de gerenciamento".

3.2 Banco de dados não relacional

Para Brito (2010), "o termo NoSQL surgiu em 1998, a partir de uma solução de banco de dados que não oferecia uma interface SQL, mas esse sistema ainda era baseado na arquitetura relacional". Devido ao alto custo de escalabilidade dos bancos de dados relacionais, propôs-se um novo ambiente conhecido como banco de dados não relacional ou NoSQL. Nele, a proposta era a de ter uma escalabilidade bem mais barata, se comparada com o outro modelo, e trabalhar de forma regular em sistemas distribuídos.

Para Aniceto, Xavier (2014, p. 8)

> Uma definição de NoSQL é que ele é um conjunto de conceitos que permitem o processamento de dados de forma rápida e eficiente com um foco em performance [43]. É um modelo alternativo pensado para se modelar os dados sem ter que seguir os padrões rígidos criados para o modelo relacional. Como medida para se contornar esse problema, ele foi proposto trazendo consigo planos de eliminar ou reduzir essa estruturação [8]. Para evitar a perda de informações, o NoSQL tem uma arquitetura distribuída tolerante a falhas que se baseia na redundância de dados em vários servidores.

Esse paradigma baseia-se muito em sistemas distribuídos e neles tem uma maior escalabilidade horizontal, podendo-se montar vários servidores interligados, fato que, se desenhado no modelo relacional, implicaria num custo maior e um planejamento complexo.

Os bancos de dados NoSQL possuem diversas características semelhantes, mas se diferenciam quanto aos modelos de dados utilizados. Enquanto no relacional tem-se apenas um modelo, o NoSQL possui quatro. Segundo Vieira *et al*. (2012, p.23):

> Atualmente, os principais produtos NoSQL disponíveis, são organizados segundo seu modelo de dados a seguir:
> - Baseado em Chave-Valor
> - Baseado em Documentos
> - Baseado em Coluna
> - Baseado em Grafos.

De acordo com Brito (2010), na primeira categoria, existe uma coleção de chaves únicas e de valores, os quais são associados com a chaves. No segundo grupo, os documentos são as unidades básicas de armazenamento e estes não utilizam necessariamente qualquer tipo de estruturação pré-definida, como é o caso das tabelas do modelo relacional. Na terceira categoria, muda-se o paradigma de orientação a registros (ou tuplas) para orientação a atributos (ou colunas). Na quarta categoria, os dados são armazenados em nós de um grafo cujas arestas representam o tipo de associação entre esses nós.

Por fim, reforça-se que o banco de dados NoSQL também possui os SGBD, mas não há o relacionamento de informações como no paradigma relacional.

3.3 *MongoDB*

Segundo Boaglio (2015, p. 8), "o *MongoDB* é um *document database* (banco de dados de documentos), mas não são os documentos da família Microsoft, mas documentos com informações no formato JSON". O autor ainda ressalta que o *MongoDB* trabalha com documentos, sem a restrição dos bancos de dados relacionais. Esses documentos são agrupados em coleções e esses conjuntos de coleções formam os bancos de dados. E, para Hows, Membrey e Plugge (2015, p. 16), "o *MongoDB* é um tipo relativamente novo de banco de dados que não tem conceitos de tabelas, esquemas, SQL ou linhas".

Os documentos JSON são bastante conhecidos, pois é muito utilizado em aplicações que utilizam API. Hows, Membrey e Plugge (2015, p. 21) reforçam que "o *MongoDB*, na verdade, não usa JSON para armazenar dados; em vez disso, ele usa um formato aberto de dados chamado BSON, desenvolvido pela equipe do *MongoDB*". O BSON, por sua vez, torna o *MongoDB* bem mais rápido, de forma que o computador entende melhor quando se trata de binários. Também é possível realizar a modificação de apenas um atributo sem a necessidade de alterar o restante da estrutura nos mesmos.

Neste trabalho, foi utilizado um gerenciador de Banco de Dados *MongoDB,* o NoSQL *Manager*, que possui uma interface muito amigável.

O *MongoDB Manager* é um software pago, disponível de forma gratuita por 30 dias, com algumas funções inativas. Na importação dos dados, foi utilizado o MS SQL *Server*.

3.4 SQL *Server*

Dentro do SGBD, a linguagem utilizada para gerenciamento de dados no programa é o SQL *Server*, um mecanismo exclusivo da Microsoft que foi criado em parceria com a Sybase, em torno de 1988. (RAMALHO, 2005). Esse sistema é responsável por traduzir a linguagem do programa, sendo um dos modelos mais usados no mundo atual em concorrência com sistemas de linguagem como *MySQL* e *Oracle.*

Segundo Ramalho (2005, p. 3) " SQL *Server* é um banco de dados relacional destinado a suportar aplicações, com arquitetura cliente/servidor em que o banco de dados fica residente em um computador central chamado servidor e cujas informações são compartilhadas por diversos usuários que executam aplicações em seus computadores locais".

Neste trabalho, foi utilizado o SQL *Server*, que possui uma interface relativamente simples e intuitiva. E para operacionalizar o SQL *Server* de forma simples, optou-se pelo SQL *Server* Management Studio, uma ferramenta de SGBD, e um software muito popular nesta área de conhecimento.

3.5 Distinção entre os Bancos de Dados Relacional e não Relacional *MongoDB*

O modelo de dados relacional, segundo Elmasri, Navathe (2010 p. 38), "... foi introduzido inicialmente por Ted Codd, da IBM *Research*, em 1970, em um artigo clássico, que atraiu atenção imediata devido a sua simplicidade e base matemática". É conhecido por suas relações entre tabelas e a linguagem padrão que utiliza é a SQL.

Elmasri, Navathe (2010 p. 38) ainda enfatiza que "quando uma relação é considerada uma tabela de valores, cada linha na tabela representa uma coleção de valores de dados relacionados. Uma linha representa um fato que normalmente corresponde a uma entidade ou relacionamento do mundo real". Um exemplo de tabela ou entidade pode ser visualizado na tabela 1.

Tabela 1- Tabela/Entidade Pessoa física

Número	Nome	CPF	Identidade	Órgão emissor	Data de nascimento	Sexo	Estado civil
77345	ALICE GUIMARAES	99988877722	2211	SSP	15/01/1982	F	1
78345	BENEDITO SILVA	92981873722	22331	SSP	25/02/1970	M	2
79345	CARMEM DOLORES	19196887722	1122	SSP	01/03/1965	F	3
71345	DALTON CRUZ	98989898982	33221	SSP	09/04/1976	M	4
72345	EUNICE SOUZA	09077832111	19370400	SSP	29/04/1976	F	2
75345	ZULMIRA CREEP	23239874123	18765432	SSP	09/05/1969	F	1

Fonte: França, Goya, Puga (2014).

O ponto forte do modelo relacional é a possibilidade de relacionar várias tabelas de forma a evitar a redundância das informações. Segundo França, Goya, Puga (2014, p. 54), “duas tabelas (A e B) estarão conectadas quando uma delas (A) tiver um campo de compartilhamento, que poderá ser plenamente utilizado em outra (B), diminuindo a redundância de dados nas inserções de registros e facilitando futuras buscas e pesquisas”. Por meio destas relações, o banco de dados torna-se mais confiável.

A utilização dos SGBD’s para manipulação das informações em um banco de dados relacional, por meio da linguagem SQL, necessita de vários comandos impostos pela linguagem. Para Dias Neto (2011), “o uso mais comum de SGBDR’s é para implementar funcionalidades simples do tipo CRUD (do inglês *Create, Read, Update e Delete* - que significa as operações de Inserção, Leitura, Atualização e Exclusão de dados)”.

Há também os operadores relacionais, que podem ser utilizados para estabelecer uma relação de comparação entre valores ou expressões, resultando sempre em um valor lógico, sendo verdadeiro ou falso (FRANÇA, GOYA, PUGA 2014 p. 216). Veja os operadores relacionais na tabela 2.

Tabela 2 – Operadores relacionais

Operador	Descrição	Exemplo
=	Igualdade	vlr_fatura = 15.00
<>	Diferente	vlr_fatura <> 15.00
>	Maior do que	vlr_fatura > 15.00
<	Menor do que	vlr_fatura < 15.00
>=	Maior ou igual a	vlr_fatura >= 15.00
<=	Menor ou igual a	vlr_fatura <=5.00
between valor1 and valor2	Entre valor1 e valor2 (similar a >= valor1 e <= valor2)	vlr_fatura between 5.00 and 25.00
not between valor1 and valor2	Não está entre valor1 e valor2	vlr_fatura not between 5.00 and 25.00
in (conjunto de valores)	Igual a um dos valores do conjunto	vlr_fatura in(5.00, 10.00, 10.25, 15.00, 22.13)
not in (conjunto de valores)	Não é igual a um dos valores do conjunto	vlr_fatura not in(5.00, 10.00, 10.25, 15.00, 22.13)
is null	Nulo	vlr_fatura is null
is not null	Não é nulo	vlr_fatura is not null
LIKE expressão	Contém a expressão	vlr_fatura LIKE %5%
not LIKE expressão	Não contém a expressão	vlr_fatura not LIKE %5%

Fonte: França, Goya, Puga (2014).

No SGBD, também temos os operadores lógicos, que permitem a verificação de expressões lógicas como mostra na tabela 3. Segundo França, Goya, Puga (2014 p. 221), “os operadores lógicos são utilizados para concatenar ou associar expressões que estabelecem uma relação de comparação entre valores, resultando sempre em um valor lógico (booleano), sendo verdadeiro ou falso”.

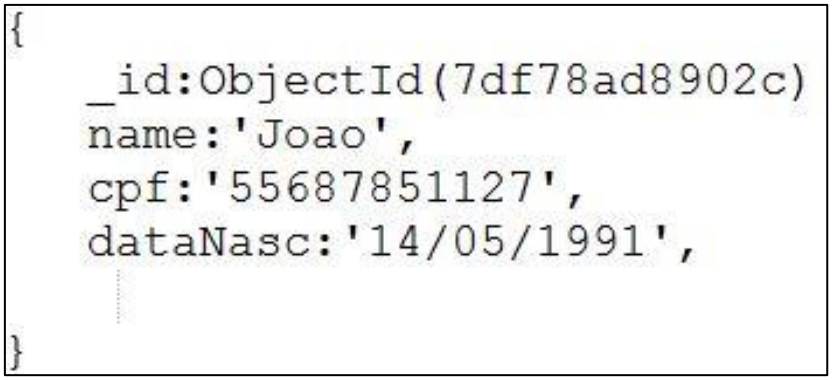
Tabela 3 – Operadores lógicos

Operador	Descrição
AND	O valor retorna verdadeiro, caso todas as expressões envolvidas na operação sejam verdadeiras.
OR	O valor retorna verdadeiro, caso ao menos uma expressão envolvida na operação seja verdadeira.
NOT	Negação do resultado da expressão. Caso o resultado seja verdadeiro será considerado falso e vice-versa. Este operador pode ser combinado com outros operadores.

Fonte: França, Goya, Puga (2014).

Em contrapartida, o *MongoDB* é um banco de dados totalmente diferente do modelo relacional. Segundo Hows, Membrey, Plugge (2015 p. 16), “o *MongoDB* é um tipo relativamente novo de banco de dados que não tem conceitos de tabelas, esquemas, SQL ou linhas. Não há transações, conformidade com *joins* (junções) e chaves estrangeiras”. Esse paradigma não segue o padrão de tabelas, mas, sim, o de documentos que seguem o padrão do JSON e, após serem persistidos, são transformados em BSON, como no exemplo da figura 1.

Figura 1 – Exemplo de documento BSON no *MongoDB*.



Fonte: O Autor (2017).

Reforçando, o *MongoDB* usa o formato aberto de dados denominado BSON, que retém os dados usando pares de chave/valor, desenvolvido pela própria equipe. O BSON torna o *MongoDB* mais rápido ainda ao fazer com que seja muito mais fácil para o computador processar e efetuar pesquisas nos documentos (HOWS, MEMBREY, PLUGGE, 2015 p. 21).

Um recurso muito utilizado no *MongoDB* é o *framework* de agregação, que fornece funcionalidades semelhante do SQL. Esse *framework* é baseado em pipelines e permite encadear partes individuais de uma *query* para obter o resultado que estiver procurando, por exemplo: o resultado de uma média ou de uma soma (HOWS, MEMBREY, PLUGGE, 2015 p. 43). Na tabela 4, demonstra-se uma comparação apresentando os operadores no SQL e como eles são representados no *framework* de agregação do *MongoDB.*

Tabela 4 – Operadores SQL Vs. *MongoDB*.

SQL Terms, Function, Concepts	MongoDB Aggregation Operators
WHERE	$match
GROUP BY	$group
HAVING	$match
SELECT	$project
ORDER BY	$sort
LIMIT	$limit
SUM	$sum
COUNT	$count
JOIN	No direct support

Fonte: Band (2014).

4. MATERIAIS E MÉTODOS

Para a realização deste trabalho, foi feito um levantamento bibliográfico sobre banco de dados e os paradigmas Relacional e NoSQL. Em seguida, foram escolhidos os bancos de dados que representariam cada modelo neste estudo e desenvolveu-se um software para simulação das pesquisas nas duas bases de dados. Para isto, dois modelos de dados foram construídos, sendo que um teve por base o modelo Relacional, constituído de tabelas, e o outro baseado no NoSQL e constituído de documentos, com chave e valor.

O software criado para a simulação e para realização dos testes utilizou então o banco de dados *MongoDB*, para o modelo NoSql, acompanhado da ferramenta de gerenciamento de banco de dados *MongoDB* Manager. Já o SQL *Server* foi utilizado para representar o modelo Relacional, acompanhado do SQL *Server* Management Studio como ferramenta de gerenciamento de banco de dados.

O software realizou as mesmas consultas nas duas bases de dados, com o mesmo equipamento e com a mesma quantidade de registros para cada teste e armazenou o tempo de resposta das pesquisas realizadas. A Microsoft, fornecedora do SQL *Server*, disponibiliza em seu site algumas bases de dados gratuitamente para fins de testes e, entre elas, a *AdventureWorks*. Esta base foi utilizada para o modelo relacional e convertida para o modelo NoSql, através de ferramenta de gerenciamento de banco de dados *MongoDB* Manager. Assim,

o estudo considerou o uso de duas bases de dados com o mesmo conteúdo, mas com modelos distintos de estrutura.

O desenvolvimento do software foi realizado em *.net* na plataforma Microsoft Visual Studio 2015, através da versão gratuita chamada Community. A linguagem é muito bem-conceituada e está no ranking das mais utilizadas, segundo a BUSINESS 2 COMMUNITY, 2017. Ela trabalha com o paradigma de orientação a objetos, tornando o processo mais flexível.

O software proposto tem uma interface bem amigável e bastante objetiva. O usuário pode escolher o tipo de busca que será realizada em ambas bases de dados e, após a execução, o tempo de resposta consumido em cada Banco de Dados é devolvido.

Este trabalho considerou a análise quantitativa, para efeito de volume de dados a serem pesquisados, e a análise qualitativa, que tratou do resultado dos dados conquistados com a sua aplicação. Lembrando que, para Miranda (2008, p. 1), "a investigação quantitativa caracteriza-se pela atuação nos níveis de realidade e apresenta como objetivos a identificação e apresentação de dados, indicadores e tendências observáveis". Além disso, Miranda (2008, p. 1) reforça que "a investigação qualitativa, ao inverso da investigação quantitativa trabalha com valores, crenças, representações, hábitos, atitudes e opiniões". Esse método emprega procedimentos de representação verbal dos resultados obtidos.

5. RESULTADOS

Neste tópico são abordadas as características da tabela de dados escolhida para a realização dos testes, bem como são apresentados os resultados obtidos no uso de cada modelo de banco. Por fim, é apresentada uma análise dos resultados.

5.1 Configuração de hardware

Os testes foram realizados num notebook da marca Dell com as seguintes configurações:

- Sistema Operacional: Windows 10 Professional.
- Processador: Intel core i5-4210U 1.70GHz.
- Memória RAM: 8GB.
- Disco rígido: SSD 480GB.

5.2 Escolha da Base de Dados

Para este estudo, optou-se pela base de dados da Microsoft *AdventureWorks* e nela os dados encontram-se divididos em cinco assuntos: funcionários de uma organização, departamento pessoal, catálogo de produtos, compras e vendas, cada um bem detalhado, por exemplo, o de funcionários é constituído por todos os dados pessoais. Como essa base de dados possui várias tabelas e o foco deste trabalho é analisar uma tabela, então, optou-se por analisar apenas a tabela relacionadas vendas.

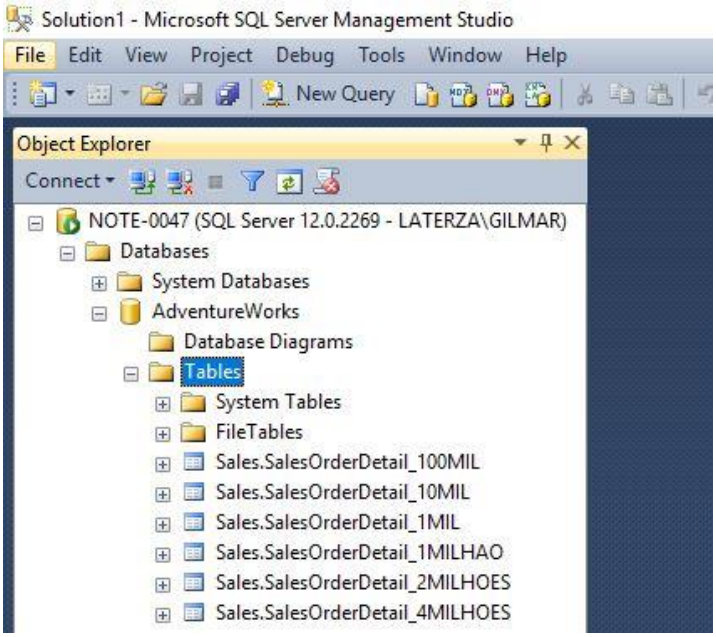
O foco da ação foi a de realizar consultas na tabela *SalesOrderDetail,* que é a tabela de detalhes do pedido de venda, e com as simulações considerando quantitativos progressivos de registros, sendo aplicadas para grupos de 1.000, 10.000, 100.000, 1.000.000. 2.000.000 e 4.000.000 de registros.

5.3 Dados utilizados

Os dados utilizados, para o banco relacional, foram importados utilizando-se o SGBD *SQL Server management studio*. Já para o paradigma NoSQL, a importação foi realizada através do SGBD *MongoDB Manager*.

A figura 2, a seguir, demonstra a estrutura do banco relacional no *SQL Server management studio,* após o processo de importação.

Figura 2 – Tabelas no SQL *Server Management Studio*

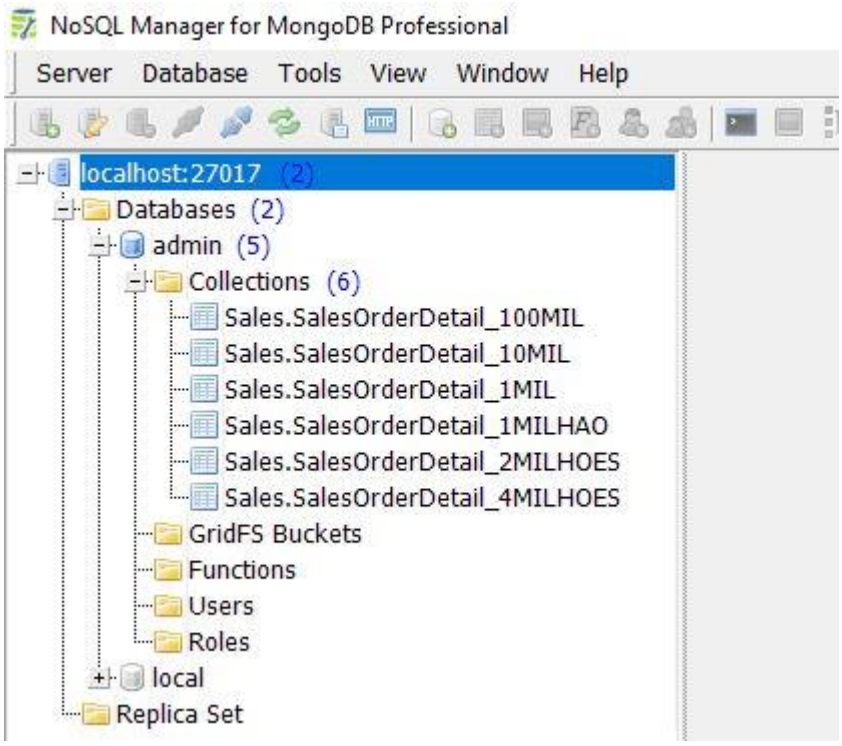


Fonte: O Autor (2017).

Para realizar a importação dos dados é utilizada a função "*import data from external database* " no *MongoDB Manager*.

Já a figura 3, a seguir, exibe a estrutura do banco de dados no *MongoDB*, após o processo de importação das tabelas pelo SQL *Server*.

Figura 3 – Exibição dos dados da *Collection* no *MongoDB*.



Fonte: O Autor (2017).

5.4 Resultados obtidos, comparação e análise dos dados

Com a bases de dados preparadas nos respectivos bancos, realizou-se uma série de consultas por meio do software no *MongoDB* e SQL *Server*. A execução dos testes considerou a simulação de consultas em ambos os bancos visando a análise dos resultados, demonstrada por comparação do tempo de resposta no processamento de ambos os bancos de dados.

A consulta proposta agrupa a venda por produtos dos campos "ProductID", "ProductName", "YEAR" e "VL_UNITARIO". Caso o valor unitário do produto sofreu alguma alteração de preço durante o ano é considerado uma nova linha de informações. O campo de "QT_REGISTROS" significa a quantidade de registros de vendas daquele produto,

o campo “QT_PRODUTOS_TOTAL” soma a quantidade de produtos vendidos por registros e o campo “VL_TOTAL” representa a soma do valor total dos produtos vendidos, subtraindo-se o desconto aplicado em alguns pedidos de venda.

A figura 4 apresenta as informações referentes à consulta que foi executada em ambos os bancos. Conta a *query* e a média do tempo de resposta da consulta. Já nesta figura, tem-se o primeiro teste realizado para tabelas de mil registros. Nas figuras 5 e 6, demonstra-se o resultado da consulta realizada nos dois paradigmas, demonstrando as informações e a garantida de confiabilidade da consulta.

Figura 4 – Comparação base de mil registros.

Fonte: O Autor (2017).

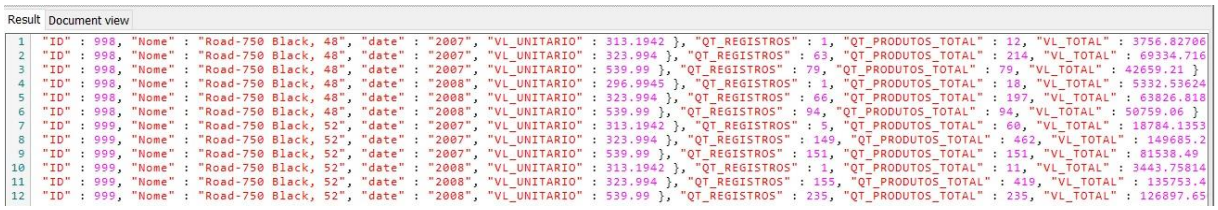
Figura 5 – Resultado da consulta no BD relacional.

Results | Messages

	ProductID	ProductName	YEAR	VL_UNITARIO	QT_REGISTROS	QT_PRODUTOS_TOTAL	VL_TOTAL
1	998	Road-750 Black, 48	2007	313,1942	1	12	3756.827068
2	998	Road-750 Black, 48	2007	323,994	63	214	69334.716000
3	998	Road-750 Black, 48	2007	539,99	79	79	42659.210000
4	998	Road-750 Black, 48	2008	296,9945	1	18	5332.536248
5	998	Road-750 Black, 48	2008	323,994	66	197	63826.818000
6	998	Road-750 Black, 48	2008	539,99	94	94	50759.060000
7	999	Road-750 Black, 52	2007	313,1942	5	60	18784.135340
8	999	Road-750 Black, 52	2007	323,994	149	462	149685.228000
9	999	Road-750 Black, 52	2007	539,99	151	151	81538.490000
10	999	Road-750 Black, 52	2008	313,1942	1	11	3443.758146
11	999	Road-750 Black, 52	2008	323,994	155	419	135753.486000
12	999	Road-750 Black, 52	2008	539,99	235	235	126897.650000

Fonte: O Autor (2017).

Figura 6 – Resultado da consulta no BD não relacional.



Result | Document view


```
1  "ID" : 998, "Nome" : "Road-750 Black, 48", "date" : "2007", "VL_UNITARIO" : 313.1942 }, "QT_REGISTROS" : 1, "QT_PRODUTOS_TOTAL" : 12, "VL_TOTAL" : 3756.82706
2  "ID" : 998, "Nome" : "Road-750 Black, 48", "date" : "2007", "VL_UNITARIO" : 323.994 }, "QT_REGISTROS" : 63, "QT_PRODUTOS_TOTAL" : 214, "VL_TOTAL" : 69334.716
3  "ID" : 998, "Nome" : "Road-750 Black, 48", "date" : "2007", "VL_UNITARIO" : 539.99 }, "QT_REGISTROS" : 79, "QT_PRODUTOS_TOTAL" : 79, "VL_TOTAL" : 42659.21 }
4  "ID" : 998, "Nome" : "Road-750 Black, 48", "date" : "2008", "VL_UNITARIO" : 296.9945 }, "QT_REGISTROS" : 1, "QT_PRODUTOS_TOTAL" : 18, "VL_TOTAL" : 5332.53624
5  "ID" : 998, "Nome" : "Road-750 Black, 48", "date" : "2008", "VL_UNITARIO" : 323.994 }, "QT_REGISTROS" : 66, "QT_PRODUTOS_TOTAL" : 197, "VL_TOTAL" : 63826.818
6  "ID" : 998, "Nome" : "Road-750 Black, 48", "date" : "2008", "VL_UNITARIO" : 539.99 }, "QT_REGISTROS" : 94, "QT_PRODUTOS_TOTAL" : 94, "VL_TOTAL" : 50759.06 }
7  "ID" : 999, "Nome" : "Road-750 Black, 52", "date" : "2007", "VL_UNITARIO" : 313.1942 }, "QT_REGISTROS" : 5, "QT_PRODUTOS_TOTAL" : 60, "VL_TOTAL" : 18784.1353
8  "ID" : 999, "Nome" : "Road-750 Black, 52", "date" : "2007", "VL_UNITARIO" : 323.994 }, "QT_REGISTROS" : 149, "QT_PRODUTOS_TOTAL" : 462, "VL_TOTAL" : 149685.2
9  "ID" : 999, "Nome" : "Road-750 Black, 52", "date" : "2007", "VL_UNITARIO" : 539.99 }, "QT_REGISTROS" : 151, "QT_PRODUTOS_TOTAL" : 151, "VL_TOTAL" : 81538.49
10 "ID" : 999, "Nome" : "Road-750 Black, 52", "date" : "2008", "VL_UNITARIO" : 313.1942 }, "QT_REGISTROS" : 1, "QT_PRODUTOS_TOTAL" : 11, "VL_TOTAL" : 3443.75814
11 "ID" : 999, "Nome" : "Road-750 Black, 52", "date" : "2008", "VL_UNITARIO" : 323.994 }, "QT_REGISTROS" : 155, "QT_PRODUTOS_TOTAL" : 419, "VL_TOTAL" : 135753.4
12 "ID" : 999, "Nome" : "Road-750 Black, 52", "date" : "2008", "VL_UNITARIO" : 539.99 }, "QT_REGISTROS" : 235, "QT_PRODUTOS_TOTAL" : 235, "VL_TOTAL" : 126897.65
```


Fonte: O Autor (2017).

Após a execução do teste de consulta para 1.000 registros, constatou-se que o *MongoDB* obteve melhor desempenho, entretanto, os retornos ocorrem de uma forma tão rápida que simplesmente olhando a execução das consultas não é possível verificar claramente o resultado. Isso demonstra que numa consulta relativamente simples, mas utilizando agrupamento de dados em uma tabela ou coleção com poucos registros o *MongoDB* supera o SQL *Server*. O SQL *Server* consumiu 30% a mais do tempo médio consumido pelo *MongoDB*.

O segundo teste simula a mesma consulta para uma tabela de 10.000 registros. A figura 7 apresenta os resultados obtidos.

Figura 7 – Comparação base de 10 mil registros.

Fonte: O Autor (2017).

Percebe-se, pelos dados obtidos para o segundo teste, que o SQL *Server* consumiu 59% a mais do tempo médio registrado pelo *MongoDB*.

Na sequência, o terceiro teste simula a consulta numa base com 100.000 registros e a Figura 8 apresenta os resultados obtidos.

Figura 8 – Comparação base de 100 mil registros.

Fonte: O Autor (2017).

Também neste teste, verifica-se que o *MongoDB* obteve um desempenho melhor. O SQL *Server* consumiu 70% a mais do tempo médio entre os resultados.

O quarto teste considerou uma base com 1.000.000 de registros e, mais uma vez, o desempenho do *MongoDB* foi surpreendente: o SQL *Server* consumiu 141% a mais, em termos de tempo médio. Veja os resultados na Figura 9, a seguir.

Figura 9 – Comparação base de 1 milhão de registros.

Fonte: O Autor (2017).

Já o quinto teste considerou a simulação da consulta numa base com 2.000.000 de registros, o dobro do teste anterior. A figura 10 apresenta os resultados obtidos.

Figura 10 – Comparação base de 2 milhões de registros.

Fonte: O Autor (2017).

No quarto teste, o SQL *Server* consumiu um tempo médio 158% maior que o *MongoDB.*

E o quinto teste considerou 4.000.000 de registros, com resultados apresentados na Figura 11, a seguir.

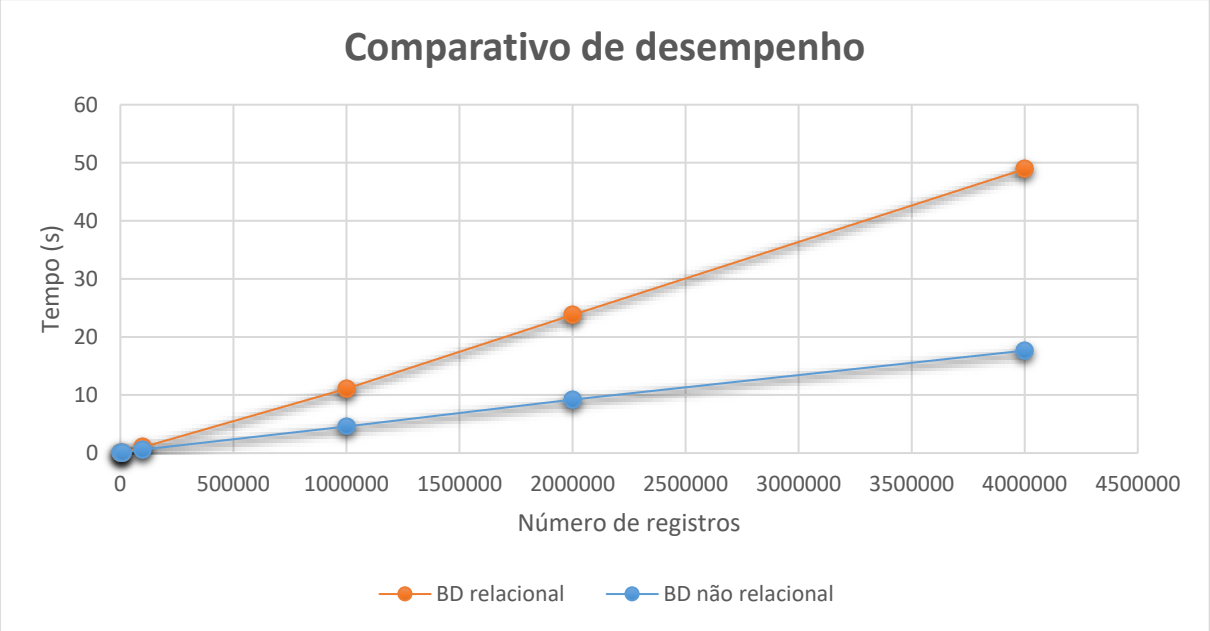
Figura 11 – Comparação base de 4 milhões de registros.

Fonte: O Autor (2017).

De forma nítida, o quinto teste traz um consumo de tempo médio do SQL *Server* na base de 177% a mais que o *MongoDB*.

Por fim, a figura 12 apresenta o gráfico do tempo em relação à quantidade de registros utilizadas em cada teste.

Figura 12 – Gráfico comparativo de desempenho.



Fonte: O Autor (2017).

A Figura 12 demonstra, na forma de gráfico, a evolução entre o tempo de resposta do processamento dos dados no modelo relacional em relação ao modelo não relacional.

Pelos resultados obtidos, e observados no gráfico da Figura 12, fica nítido que o desempenho do *MongoDB* foi extremamente superior ao do SQL *Server*, principalmente considerando a escalabilidade quantitativa de registros ao longo dos seis testes realizados. Em média, o SQL *Server* foi 164% mais lento que o *MongoDB*. Veja que, à medida que os registros nas bases aumentam, o tempo médio das respostas do SQL *Server* cresce de maneira muito significativa, enquanto que o tempo médio das respostas do *MongoDB* tem um crescimento muito inferior, em termos de comparação.

6. DISCUSSÃO

Este trabalho analisou o desempenho, em termos de tempo médios, de consulta a uma base de dados idêntica, para quantidade de registros crescentes, num banco de dados relacional e num banco de dados não relacional.

Os resultados obtidos, nos seis testes realizados, demonstraram que a performance do banco de dados não relacional, no caso o *MongoDB*, é muito superior em relação ao desempenho apresentado pela base de dados relacional, no caso o SQL *Server*.

A diferença no tempo de resposta da execução das consultas em relação ao aumento da quantidade de registros na base de dados trouxe um resultado que demonstraram essa superioridade. Enquanto que a base de dados para ambos os modelos cresceu 399.900%, o tempo consumido pelo SQL *Server* aumentou 752.429%, enquanto que no *MongoDB* o aumento foi de 352.954%, ou seja, o *MongoDB* apresentou um desempenho significativo em termos de consumo médio de tempo.

Por fim, é necessário desenvolver trabalhos futuros que aumentem o escopo desta proposta e possibilitem uma perspectiva considerando também coleções de dados inter e multi relacionadas.

6. CONCLUSÃO

1. O banco de dados não relacional foi superior ao banco de dados relacional, pois na medida em que a base de dados cresceu, o desempenho do mesmo foi aproximadamente três vezes mais rápido.
2. Embora utilizando apenas um modelo de consulta juntamente com algumas funções de agregação, agrupamento e ordenação, o banco de dados não relacional apresenta melhor otimização em relação ao banco de dados relacional.

REFERÊNCIAS

ANICETO, Rodrigo; XAVIER, Renê. **Um estudo sobre a utilização do banco de dados NoSQL Cassandra em Dados Biológicos.** 2014. Viii, 50 f., il. Monografia (Bacharelado em Ciência da Computação) – Universidade de Brasília, Brasília, 2014. Disponível em: < http://bdm.unb.br/handle/10483/7927 >. Acesso em: 05 Out. 2016.

BAND, Tyler. **How to integrate MongoDB and SQL data with a single REST API.** 2014. Disponível em: <https://www.slideshare.net/TylerBand/joing-sq-landnosql>. Acesso em: 23 Mai. 2017.

BAZZOTTI, Cristiane; GARCIA, Elias. **A importância do sistema de informação gerencial para tomada de decisões.** Cascavel PR, 2010. Disponível em: <http://www.waltenomartins.com.br/sig_texto02.pdf >. Acesso em: 13 Out. 2016.

BOAGLIO, Fernando. **MongoDB.** 1 Ed.: Casa do código, 2015. 209 p. Disponível em: < https://books.google.com.br/books?hl=pt-PT&lr=lang_pt&id=bWmCCwAAQBAJ&oi=fnd&pg=PA6&dq=Mongodb&ots=waU9QBGobO&sig=ynobyhx4Aof5UiE2SbWWYhYicp4#v=onepage&q=Mongodb&f=false >. Acesso em: 22 Out. 2016.

BRITO, Ricardo W. **Bancos de Dados NoSQL x SGBDs Relacionais:** Análise Comparativa. Faculdade Farias Brito e Universidade de Fortaleza, 2010. Disponível em: < http://www.infobrasil.inf.br/pagina/anais-2010 >. Acesso em: 30 out. 2016.

BUSINNESS 2 COMMUNITY. Top 20 most popular programming languages in 2017. 2017. Disponível em <http://www.business2community.com/tech-gadgets/top-20-popular-programming-languages-2017-01791470#6p607ZxofgG6dzte.97>. Acessado em: 20/06/2017.

DATE, J. Christopher. **Introdução a Sistemas de Banco de Dados.** 8 Ed.: Elsevier Brasil, 2004. 865 p. Disponível em: <https://books.google.com.br/books/about/Introdu%C3%A7%C3%A3o_a_sistemas_de_bancos_de_dad.html?hl=pt-BR&id=xBeO9LSlK7UC >. Acesso em: 10 Out. 2016.

DB-ENGINES. DB-Engines Ranking. 2017. Disponível em <https://db-engines.com/en/ranking>. Acesso em 20/06/2017.

NETO, Arilo Cláudio Dias. **Bancos de Dados Relacionais** - Artigo Revista SQL Magazine 86. Disponível em: <http://www.devmedia.com.br/bancos-de-dados-relacionais-artigo-revista-sql-magazine-86/20401>. Acesso em: 21 Mai. 2017.

ELMASRI, Ramez; NAVATHE, Shamkant B.. **Sistemas de banco de dados.** 6 Ed.: Pearson – São Paulo, 2010. 766p.

FERREIRA, João; ITALIANO, Isabel; TAKAI, Osvaldo. **Introdução a Banco de Dados.** 2005. Disponível em: < https://www.ime.usp.br/~jef/apostila.pdf >. Acesso em: 17 Out. 2016.

FRANÇA, Edson; GOYA, Milton; PUGA, Sandra. **Banco de dados:** Implementação em SQL, PL/SQL e Oracle 11g**.** 1 Ed.: Pearson, 2014. 332 p.

HOWS, David; MEMBREY, Peter; PLUGGE, Eelco. **Introdução ao MongoDB.** 1 Ed.: Novatec, 2015. 167p.

KORTH, Henry F.; SILBERSCHATZ, Abraham; SURDACHAN, S. **Sistema de Banco de dados.** 5 Ed.: Elsevier - Campus, 2006. 805 p.

MEDEIROS, Luciano. **Banco de dados:** Princípios e prática**.** 1 Ed.: Intersaberes - Curitiba, 2013. 185 p.

MIRANDA, Bruno. **Método Quantitativo versus Método Qualitativo.** 2008. Disponível em: < http://adrodomus.blogspot.com.br/2008/06/mtodo-quantitativo-versus-mtodo.html >. Acesso em: 01 Nov. 2016.

RAMALHO, José. **Microsoft SQL Server 2005 Guia prático.** 1 Ed.: Elsevier – Rio de janeiro, 2005. 277 p.

VIEIRA, Marcos et al. **Banco de Dados NoSQL:** Conceitos, Ferramentas, Linguagens e Estudos de Casos no Contexto de Big Data. Universidade Federal do Mato Grosso, 2012. Disponível em: < http://data.ime.usp.br/sbbd2012/artigos/pdfs/sbbd_min_01.pdf>. Acesso em: 10 Out. 2016.