TECNOLOGIA NoSQL:
A SOLUÇÃO PARA LIDAR COM O CRESCENTE VOLUME DE DADOS

MARCOS HENRIQUE IZIDIO MONTEIRO

marcoshenrique.monteiro@gmail.com
UFF- Universidade Federal Fluminense

RESUMO

Dados a evolução tecnológica e os constantes lançamentos de novos produtos pela indústria eletrônica, o volume de dados vem crescendo de forma avassaladora. Em vista disso, as organizações precisam se preparar para lidar com essa realidade, de modo a poder transformar seus dados em informação útil, com agilidade, inovação e pioneirismo, para manter-se cada vez mais competitiva em seu mercado de atuação.

Este trabalho visa apresentar a tecnologia NoSQL, seus modelos de dados, assim como o algoritmo criado pela Google para processamento de um grande volume de dados de forma distribuída.

Um ponto forte da tecnologia NoSQL é a fácil distribuição horizontal, sem a necessidade de equipamento poderosos, de alto custo e alta performance. É uma tecnologia mais flexível que a dos bancos de dados relacionais porque é livre de esquemas e é escalável. Além disso, apresenta um melhor desempenho comparado ao dos bancos de dados relacionais.

Uma restrição importante que deve ser mencionada é a falta, no mercado de trabalho, de profissionais qualificados na tecnologia NoSQL.

Palavras-chave: NoSQL, MapReduce e Neo4j

1. INTRODUÇÃO

Atualmente, tanto a população em geral quanto as organizações encontram-se expostas a uma quantidade avassaladora de dados, sem que muitas vezes sequer se deem conta disso. Basta porém dimensionar a quantidade de pessoas que navegam nas redes sociais, por exemplo, para se ter uma ideia, ainda que pálida. Não muito tempo atrás, o processo de revelação fotográfica exigia que o filme fosse enviado ao laboratório, e o fotógrafo só podia ter noção do resultado depois de pronta a revelação. Com a evolução tecnológica dos diversos equipamentos eletrônicos e também das máquinas fotográficas, hoje é possível, tão logo tirada a foto, analisar, avaliar e decidir se vale ou não a pena mantê-la armazenada. Pois é justamente o armazenamento a grande questão a ser abordada neste trabalho.

Para armazenar as fotografias digitalizadas, é preciso utilizar um equipamento apropriado para isso, que pode ser o próprio equipamento de fotografia, um *pendrive*, cartão de memória, um computador pessoal, HD externos. Assim, algum equipamento eletrônico que possua essa finalidade se encarregará do armazenamento. Torna-se possível então, a partir daí, compartilhar os momentos registrados, disponibilizando as fotografias nos sites de redes de relacionamentos através da internet. Uma vez disponibilizado esse material para compartilhamento entre as pessoas nos sites de relacionamento, as fotografias tornam-se dados, que também deverão ser armazenados por esses mesmos sites de relacionamentos. Nessa progressão, pode-se imaginar o volume de dados com que empresas como Google, Twitter, Facebook, LinkedIn, Instagram e muitas outras precisam lidar diariamente. Como fazer para armazenar esse imenso e extraordinário volume de dados?

Essas empresas precisam de bases de dados robustas, ágeis e extremamente consistentes para suportar tamanha demanda e gerar credibilidade. Recorremos ao exemplo simples da fotografia para ilustrar a imensidão de volume de dados apenas com referência às máquinas de fotografia digital, porém é possível o compartilhamento de músicas, vídeos, textos etc. Hoje, qualquer celular é capaz de registrar uma fotografia, gravar um vídeo, armazenar e transferir músicas através do mecanismo de Bluetooth. A vida moderna é cercada de equipamentos como Ipods, Tablets, Smartphones, entre vários outros, todos os quais geram, compartilham e armazenam dados.

Tiware (2011) afirma que a indústria eletrônica vem contribuindo com a alta significativa no volume dos dados devido ao lançamento constante de diversos novos produtos e

equipamentos eletrônicos, criando desafios e oportunidades em torno do armazenamento, da análise e do arquivamento de dados.

Esse crescimento no volume de dados trouxe a necessidade de escalabilidade, e um dos grandes problemas apresentados pelos bancos de dados relacionais é justamente conciliar o modelo de dados relacional com a demanda por escalabilidade. Em uma aplicação web qualquer, executada sobre um sistema de gerenciamento de banco de dados relacional, o crescimento do número de usuários ocasiona uma queda de desempenho no sistema. Para resolver esse problema, opta-se muitas vezes por fazer um upgrade no servidor, porém, se o número de usuários continuarem a crescer, o problema passa a ser o acesso à base de dados. A solução seria então escalar o banco de dados, distribuindo-o em outras máquinas, o que, no caso de um banco de dados relacional, não é tão simples de promover. (BRITO, 2010).

Brito (2010) cita ainda que os bancos de dados relacionais são extremamente utilizados e conhecidos atualmente, porém apresentam determinadas limitações ao lidar com esse excesso de volume de dados e com os dados semiestruturados, quais sejam, email, foto, vídeos etc. Em virtude dessas limitações, surgiu uma nova classe de produtos de banco de dados, que consiste em armazenar os dados orientados a coluna, pares de chave e valor, documentos e grafos. Essa nova tecnologia de banco de dados está sendo denominada NoSQL.

De acordo com Sadalage e Fowler (2013), a tecnologia NoSQL surgiu motivada pela necessidade de lidar com os grandes volumes de dados, uma vez que a tendência é que o volume dos dados continue a crescer. Seus defensores alegam que os sistemas desenvolvidos com essa nova tecnologia são mais flexíveis que os bancos de dados relacionais por serem livres de esquemas, distribuídos, escaláveis, apresentarem um melhor desempenho e não precisarem de servidores potentes para armazenar seus dados.

Considerando o grande sucesso de grandes empresas como Facebook, Amazon, Google, Twitter, LinkedIn, entre outras, que adotaram a tecnologia NoSQL visando, para melhor adequar seus negócios, suprir uma lacuna deixada pelas bases de dados relacionais, este trabalho busca responder como as organizações podem se preparar para lidar com o crescente volume de dados e, assim, transformá-los, com agilidade, em informação útil, para manter-se competitivas em seu mercado de atuação. Em seguida, o objetivo é apresentar a tecnologia NoSQL e seus modelos de dados, assim como o banco de dados Neo4J orientado a grafos. O objetivo final é apontar os pontos fortes da tecnologia NoSQL.

Optou-se pela pesquisa bibliográfica de caráter exploratório, sendo os dados tratados de forma qualitativa. A composição deste trabalho se desenvolverá por meio de introdução, referencial teórico, metodologia e conclusão.

2. METODOLOGIA DE PESQUISA

A metodologia utilizada para a produção deste trabalho foi exploratória, por se tratar de assunto de pouco conhecimento acumulado, tendo em mente apresentar e esclarecer um novo conceito através de uma nova tecnologia de banco de dados. Como meios para pesquisa, foram utilizados a literatura especializada, artigos científicos, internet e publicações que fornecem instrumento analítico para todo tipo de pesquisa. Os dados foram tratados de forma qualitativa, com visão crítica, por se tratar de informações geradas por terceiros, com reflexões, interpretações, análise e conclusões dos respectivos autores.

3. REFERENCIAL TEÓRICO

Segundo Kline (2010), na década de 1970 um pesquisador da IBM chamado E. F. Codd criou o modelo de dados relacional que primeiramente foi intitulado Sequel (Structured English Query Language), ou Linguagem de Consulta em Inglês Estruturado, que algum tempo depois passou a ser chamada de SQL (Structured Query Language), ou Linguagem de Consulta de Dados. Mesmo tendo a IBM criado a teoria de bancos de dados relacionais, foi a Oracle a primeira a comercializar o produto, que a partir daí passou a ser muito utilizado pelos sistemas de gerenciamento de bancos de dados. Sua principal característica é a preocupação com a consistência dos dados, garantindo assim a integridade dos dados e o processo de normalização (BRITO, 2010)

O sistema de gerenciamento de banco de dados relacional é o principal mecanismo de sistemas de informação das aplicações web e sistemas computacionais cliente/servidor. Através dele, os usuários podem consultar, inserir, alterar e apagar dados rápida e simultaneamente sem impactar outros usuários. Também permite aos desenvolvedores de aplicações acesso aos seus recursos, além de disponibilizar aos administradores de dados ferramentas para manter, assegurar e aperfeiçoar os dados organizacionais (KLINE, 2010).

De acordo com Brito (2010), atualmente os sistemas gerenciadores de banco de dados oferecem aos desenvolvedores de aplicações facilidades como validação e verificação dos

dados, controle de concorrência, recuperação de falhas, segurança do modelo e dos dados, controle de transações, otimização das consultas, entre outras. Isso facilita sobremaneira a vida dos desenvolvedores de aplicações, que precisam se preocupar apenas com o desenvolvimento de suas aplicações.

O autor acrescenta ainda que desde sua criação até os dias de hoje os bancos de dados relacionais sofreram muitas evoluções devido a sua grande utilização e aos problemas organizacionais já solucionados. Mesmo assim, devido ao aumento vertiginoso no volume dos dados, os bancos de dados relacionais apresentaram deficiências em lidar com esse excesso de dados. Para suprir essa lacuna, surgiu uma nova tecnologia de bancos de dados não relacional, que recebeu o nome de NoSQL.

Segundo Strauch (2010), o termo NoSQL surgiu em 1998 com a proposta de um banco de dados relacional sem a interface SQL.

Tiware (2011) afirma que o termo NoSQL seria uma combinação de duas palavras: No [Não] e SQL. Provavelmente a intenção era não expor o banco de dados ou modelo relacional. Tanto que em dado momento, como uma alternativa a denominação NoSQL, foi proposta a denominação NonRel, significando um banco de dados não relacional. Porém, os defensores do termo alegaram que este na verdade seria apenas uma sigla de “não apenas SQL”, e o mantiveram. Qualquer que seja o significado do termo NoSQL, porém, ele é usado atualmente para abranger todo armazenamento de dados que não segue a linha dos sistemas de gerenciamento de banco de dados relacionais. Assim, o NoSQL não é um produto, e sim uma tecnologia que representa uma classe de produtos e uma coleção de diversos conceitos sobre armazenamento e manipulação de dados.

Em 2009, em uma conferência em São Francisco, nos Estados Unidos, levada a efeito para debater os bancos de dados *open source* distribuídos, o termo e o mundo NoSQL ganharam nova vida entre os defensores dos bancos de dados distribuídos e não relacionais (STRAUCH, 2009)

Segundo Nascimento (2010), o NoSQL tem como característica a fácil distribuição horizontal; assim, quanto maior o volume dos dados, é suficiente disponibilizar mais servidores, sem a necessidade de equipamentos poderosos, de alta performance e alto custo. A Google é uma empresa adepta dos bancos de dados distribuídos, e seus computadores são de pequeno e médio portes, o que representa redução de custos. A redução nos custos por si só já

é um ponto positivo para as organizações, que poderiam lidar com o aumento no volume de dados sem sofrer um forte impacto financeiro em seus orçamentos.

O mesmo autor afirma ainda que, com o crescente desenvolvimento tecnológico que ocorre atualmente, o volume dos dados também cresce na mesma proporção, a ponto de o Google atingir a marca de pentabytes, ou seja, um quatrilhão de bytes, que pode ser representado pela potência de 10^{15} bytes. Esse aumento vertiginoso no volume de dados acabou gerando um grande problema de desempenho e escalabilidade aos sistemas de gerenciamento de banco de dados relacionais. Como já afirmamos anteriormente, escalar um banco de dados relacional é possível, porém não é uma atividade muito simples para se desempenhar em função do seu modelo de dados (NASCIMENTO, 2010).

Existem hoje quatro categorias de modelagem de dados para as diversas classes de bancos de dados que compõem a tecnologia NoSQL (PARTNER, VUKOTIC & WATT, 2012):

Pares de chave-valor	Possui coleção de chaves única e valores, os quais são associados com as chaves; Essa é a categoria mais simples, porém muito poderosa; Projetado para lidar com grande volume de acesso simultâneo aos dados.
A documentos	Não possuem qualquer tipo de estruturação predefinida, ou seja, são totalmente livres de esquemas; Os documentos dessa categoria são coleções de atributos e valores, em que um atributo pode ser multivalorado.
A família de colunas	Orientado a atributos e não mais a registros, ou seja, os registros são armazenados em colunas, e não linhas; Alto desempenho e alta disponibilidade quando se trabalha com grande volume de dados.

A grafos	Armazena os dados em nós, com as arestas representando os relacionamentos entre os nós; Excelente desempenho em relação ao banco de dados relacional.

Vivemos um momento especial em relação às bases de dados e seu armazenamento porque até bem pouco tempo não existia outra opção de armazenamento de dados que não fossem os bancos de dados relacionais, e era através deles que eram solucionados os diversos problemas, fossem eles grandes ou pequenos. As bases de dados relacionais não deverão desaparecer, porém há quem esteja buscando alternativas viáveis, como modelos livres de esquemas, com estruturas alternativas de dados, replicação mais simples, alta disponibilidade, métodos de consultas aos dados, bom dimensionamento horizontal e novas formas (REDMOND & WILSON, 2012).

Brito (2010) salienta que a opção pela tecnologia NoSQL em lugar dos bancos de dados tradicionais é influenciada por três pontos importantes, que precisam ser avaliados: escalonamento, consistência e disponibilidade:

- ✓ Escalonamento: É uma das principais vantagens da tecnologia NoSQL.

  O escalonamento vertical ocorre quando existe apenas um servidor de banco de dados e ele precisa ser reforçado, ou seja, receber um upgrade, com a inserção de mais memória, discos mais espaçosos e aumento do número de processadores, além de torná-lo mais potente. Esse é um cenário muito comum nos bancos de dados relacionais.

  O escalonamento horizontal ocorre quando distribuímos o banco de dados em mais de um servidor. Esse processo de particionar horizontalmente os servidores é conhecido como *sharding*, e com ele os dados são paralelizados nesses vários servidores conectados uns aos outros. Esse processo é característico na tecnologia NoSQL.
- ✓ Disponibilidade: O fato de os dados serem distribuídos em vários servidores garante que uma requisição sempre seja atendida. Isso porque, caso haja falha em um servidor por qualquer motivo, o sistema não vai parar ou sair do ar, garantindo, assim, que toda requisição ao banco de dados seja sempre respondida. Pode inclusive ocorrer de o que foi solicitado ao banco de dados não retornar na íntegra em virtude de ter ocorrido alguma falha.

- ✓ Consistência: Os bancos de dados que fazem parte da tecnologia NoSQL deixam a desejar no tocante à consistência dos dados em comparação aos bancos de dados relacionais, que fazem isso muito bem. Em sistemas distribuídos, é preciso optar pelo que se deseja obter: forte consistência, alta disponibilidade e uma tolerância ao particionamento, em virtude do teorema CAP (**Consistency, Availability e Partition tolerance**), segundo o qual só é possível garantir duas dentre essas três propriedades em sistemas distribuídos.

  Em decorrência desse teorema, as propriedades transacionais ACID (*Atomicity, Consistency, Isolation* e *Durability*), que são fundamentais para garantir a consistência do banco de dados, não poderão ser estabelecidas de forma simultânea. Existe porém um conceito ligado a outras propriedades, conhecida como BASE (*Basically Available, Soft state* e *Eventually Consistent*), no qual os sistemas deverão ser planejados para eventuais inconsistências a fim de garantir a disponibilidade ou o particionamento.

4. MapReduce

Quando se fala em sistemas distribuídos, não se pode deixar de citar a função MapReduce, um algoritmo criado pelo Google que permite o processamento de grandes quantidades de dados divididos em pedaços entre um conjunto de computadores (ÉVORA, 2013).

O modelo de programação tem o objetivo de proporcionar, através da paralelização, a geração, o processamento e a análise de grandes quantidades de dados, distribuídos em um sistema computacional de grande escala.

Utiliza-se o paralelismo para dividir a carga de dados, em vez de dividir as etapas de processamento. Assim, cada máquina é responsável por processar completamente um pequeno grupo de dados, ao invés de processar todos os dados em uma determinada etapa da computação.

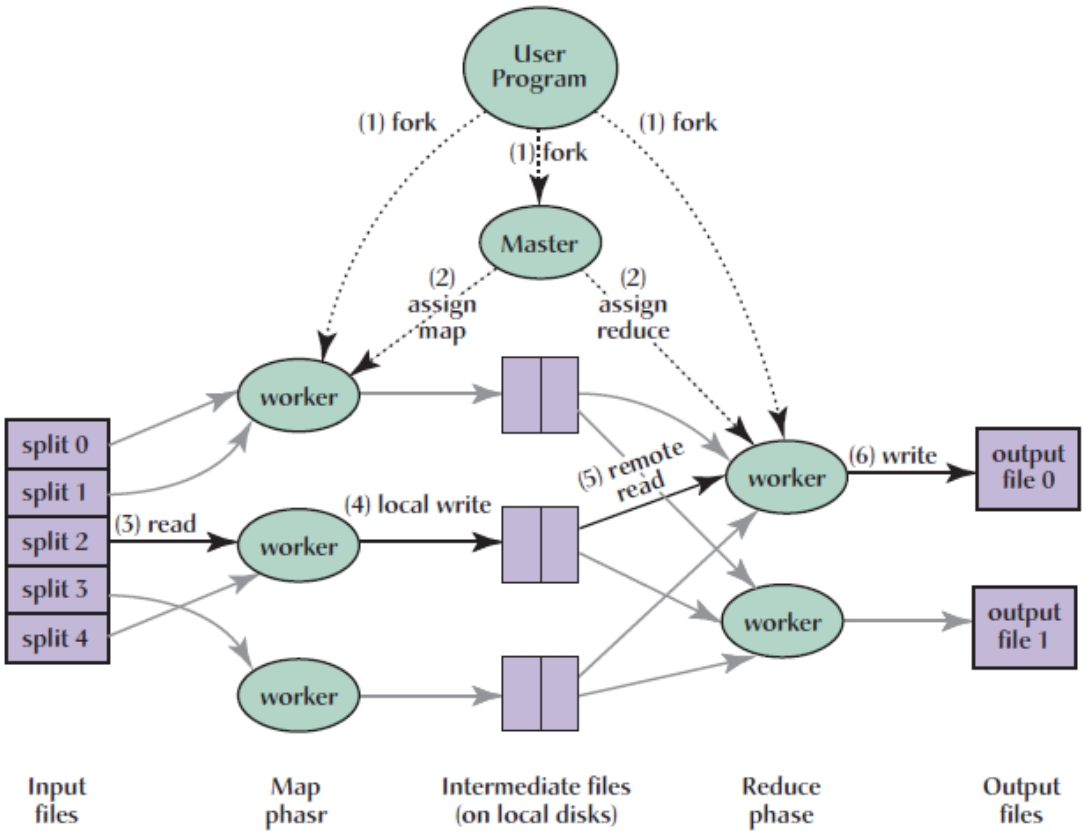
O modelo de programação utilizado no MapReduce são pares de chave e valor, em que a chave é o elemento identificador do par e o valor, o seu conteúdo. Tanto na entrada quanto na saída se utiliza o conjunto de pares de chave e valor. Nesse modelo, o programador só precisa se preocupar com a composição de duas funções:

- ✓ *Map()*: função que recebe como entrada um conjunto de pares, analisa cada um e gera um novo conjunto intermediário de pares com chave e valor. Os pares desse conjunto

de saída da função serão agrupados de acordo com suas chaves e, em seguida, enviados como conjunto de entrada para a função *reduce*.

- ✓ *Reduce()*: esta função recebe como entrada o conjunto de pares intermediários gerado como saída da função *map*, organizados de acordo com suas chaves, realiza uma determinada computação sobre os valores desses pares e gera um novo conjunto de pares, que será considerado a saída da aplicação MapReduce.

A função MapReduce, ao ser chamada pelo programa do usuário, atua da forma ilustrada na figura que segue:



Fonte: MapReduce: Simplified Data Processing on Large Clusters **by Jeffrey Dean and Sanjay Ghemawat**

Os dados de entrada são fragmentados em pedaços de até 64 MB. Essa fragmentação pode ocorrer de forma automática ou através da programação do usuário. O programa do usuário é copiado e inicializado em todas as máquinas. Uma das máquinas é eleita a máquina *master* e as outras são máquinas *worker*. A *master* escolhe uma *worker* livre para enviar as tarefas.

As máquinas *worker* que executam tarefas de *map* guardam na memória os valores intermediários e periodicamente as escrevem no disco local, informando à máquina *master* onde os pares chave e valor intermediários estão armazenados. Máquinas *worker* que executam tarefas de *reduce* recebem da máquina *master* autorização para iniciar as tarefas, e também recebem os locais onde estão armazenados os valores relacionados àquela redução. A máquina *worker* de redução que recebeu os parâmetros de entrada armazena seus dados de saída no final do arquivo de saída dessa redução. Ao final, quando todas as tarefas de *map* e *reduce* são concluídas, a máquina *master* desperta o programa do usuário (DEAN & GHEMAWAT, 2004).

5. Banco de dados orientados a grafos

A teoria dos grafos nasceu com o histórico problema das sete pontes de Konigsberg, solucionado pelo matemático suíço Leonhard Euller (PARTNER, VUKOTIC & WATT, 2012).

Netto (2012) menciona que o que veio a se tornar a teoria dos grafos passou um século perdido junto a um amontoado de produções científicas do matemático Leonhard Euler, assinalando que o famoso problema relacionado às sete pontes de Konigsberg não passou de uma brincadeira sem grandes interesses ou qualquer pretensão para a época. Porém a teoria de Euler hoje é fortemente utilizada por diversas áreas, entre as quais administração, comunicação e transporte.

No século XIX surgiram as primeiras aplicações de grafos, nas áreas de eletricidade e química. Atualmente vemos sua utilização ainda maior nos campos da química e da bioquímica molecular para responder a estrutura do DNA e sua composição química.

Campos (2012) diz que os habitantes da cidade de Konigsberg tinham como hábito caminhar aos domingos pela cidade; começaram então a tentar passar por todas as pontes apenas uma única vez. Como não encontraram uma solução, submeteram o problema a um matemático, que chegou ao resultado e explicou por que não seria possível.

Ao caminhar por um vértice, é preciso entrar e sair dele; logo, precisamos de um número par de arestas cada vez que passarmos por um vértice; como nesse caso o grafo possui um

número ímpar de arestas, não seria possível caminhar de um ponto qualquer passando por cada ponte apenas uma vez (GUIMARÃES, 2001)

É interessante assinalar que o pensamento referente à teoria dos grafos não é nova: ela foi iniciada no século XVIII e tem sido ativamente pesquisada e aperfeiçoada por matemáticos, sociólogos, antropólogos, químicos, biólogos, entre outros. Apenas nos últimos anos porém é que essa teoria vem sendo aplicada à gestão da informação. Nesse curto período de tempo, os bancos de dados orientados a grafos contribuíram muito, ajudando a resolver problemas importantes nas áreas de redes sociais.

O grande sucesso de empresas como Facebook, Google, Twitter, Amazon, entre outras, fez aumentar o foco nas bases de dados orientadas a grafos, uma vez que essas empresas são adeptas da tecnologia NoSQL.

Outra forte razão para se optar pelo banco de dados orientados a grafos é o seu bom desempenho ao lidar com grandes volumes de dados, em comparação com o banco de dados relacional.

O desempenho dos *joins* efetuados nos bancos relacionais tende a decair quando o volume de dados aumenta; no banco de dados orientados a grafos, ao contrário, tendem a permanecer constantes, mesmo com o crescimento no volume de dados.

A flexibilidade também é um ponto bem interessante: a modelagem dos dados orientados a grafos expressa a necessidade do negócio de forma que se ajuste à sua velocidade de crescimento; ou seja, os bancos de dados orientados a grafos se adaptam ao negócio, diferentemente dos bancos de dados relacionais, nos quais é o negócio que tem que se adaptar ao modelo.

Os grafos são de natureza aditiva. Isso significa que é possível adicionar novos tipos de relações, nós e subgrafos à estrutura já existente sem alterar as consultas e funcionalidades já existentes. Graças a essa flexibilidade, não precisamos modelar o banco antes do tempo, facilitando qualquer alteração referente a mudanças de requisitos do negócio (ROBINSON, WEBBER & EIFREM, 2013).

Esses mesmos autores argumentam que as organizações optam pelo modelo de dados orientados a grafos porque a performance de suas aplicações passa a ser de milissegundos,

não só pela capacidade de resposta ao negócio mas também porque os dados são importantes para as organizações.

A performance e a pronta resposta às consultas são pontos de grande relevância e preocupação nas organizações no que diz respeito às suas plataformas de dados. As aplicações web são sistemas com transações on-line, logo, devem responder ao usuário final o mais rapidamente possível para que possam ser bem-sucedidas e gerar credibilidade. No mundo relacional, quando há aumento no volume dos dados, as aplicações sofrem com perda de desempenho em virtude da deterioração dos *joins* existentes nos aplicativos. Os bancos orientados a grafos possuem índices, assim como os bancos de dados relacionais. Em consequência, por mais complexa que seja a consulta, o percurso executado para realizá-la torna-se rápido, e uma boa performance é então mantida independentemente do tamanho do conjunto de dados.

Quando são bem-sucedidas, raramente as aplicações ficam estáticas; portanto, mudanças nas condições de negócios, no comportamento dos usuários e infraestruturas técnicas e operacionais conduzem a novas exigências. No mundo relacional, quando ocorrem mudanças nos requisitos do negócio, há uma exigência enorme em relação ao comprometimento das organizações para com a migração de dados. Isso exige sempre uma dose adicional de cuidado, sobretudo quando as modificações são em seu esquema de dados, para atender às características antigas e adaptar-se às novas. Nos bancos orientados a grafos isso não ocorre, porque estes, por natureza, são livres de esquemas, e assim têm a capacidade de evoluir à medida que o negócio evolui, reduzindo o risco e o tempo de lançamento de novos produtos ou serviços em seu mercado de atuação.

Um aplicativo de negócios críticos demanda uma tecnologia de dados que seja robusta, escalável e, acima de tudo, transacional. Alguns bancos de dados orientados a grafos são bem novos e, portanto, não estão totalmente maduros; existem no entanto no mercado bases de dados orientadas a grafos que fornecem todas as propriedades ACID: alta disponibilidade, escalabilidade horizontal de leitura e armazenamento de bilhões de entidades. Esse é o caso do banco de dados orientado a grafos Neo4J, e esse tem sido um fator bastante relevante para sua adoção pelas organizações.

Para Robinson, Webber e Eifrem (2013), os bancos de dados orientados a grafos fornecem uma técnica poderosa de modelagem, mas isso por si só não seria justificativa

suficiente para a substituição de um banco de dados relacional já bem estabelecido, estabilizado e entendido. Segundo esses autores, independentemente do tamanho ou do volume de dados, os bancos de dados orientados a grafos seriam a melhor maneira de se representar e consultar os dados conectados.

O Neo4J é um banco de dados não relacional orientado a grafos que goza de muito respeito dentro da tecnologia NoSQL. Foi desenvolvido em Java pela empresa Neo Technology.

Como o seu modelo de dados é orientado a grafos, o banco de dados Neo4J armazena seus dados em nós (vértice) e arestas (relacionamentos), em que o nó possui o conceito de uma instância de um objeto com um ID único e as arestas fornecem a ligação entre os nós.

Diferentemente dos bancos de dados relacionais, o Neo4j não tem tabelas e nem colunas, por isso não utiliza a interface SQL e nem *joins* para fazer consultas. O Neo4j tem um forte conceito matemático dentro da teoria dos grafos, o que o torna bastante poderoso. Sua consulta percorre todos os nós que se relacionam com o nó inicial de partida, também chamado raiz, e se encerra apenas quando não encontra mais relacionamentos referentes ao nó inicial de partida, ou seja, quando todos os nós ligados ao vértice inicial foram visitados.

Um grande diferencial do banco de dados Neo4J é o suporte às propriedades ACID, propriedades estas transacionais muito fortes e bem-definidas entre os SGBDs relacionais.

Com a popularidade da tecnologia NoSQL, o gerenciamento das propriedades de transação tem sido objeto de muita discussão. A compatibilidade com as propriedades ACID possibilita a migração dos dados de um modelo relacional para um não relacional, oferecendo segurança e comodidade para trabalhar com o modelo orientado a grafos. (PARTNER, VUKOTIC & WATT, 2012).

Os autores mencionados salientam que o Neo4j está otimizado para fazer buscas rápidas no grafo porque sabe por onde tem que iniciar o percurso de busca, reduzindo, assim, a quantidade de nós que precisa percorrer, o que se torna essencial à medida que o volume de dados aumenta.

O Neo4j suporta indexação, e ocorre o mesmo que nos bancos de dados relacionais, fornecendo capacidade de acesso rápida e fácil a um registro de determinada tabela. A

indexação do Neo4j torna mais fácil encontrar nós com certos valores de propriedades ou relações com determinados valores de propriedade.

Assim como os SGBD relacionais, o Neo4j possui um ambiente para efetuar consultas e atualizações diretamente no banco de dados, independentemente de qualquer linguagem de programação. Essa ferramenta é conhecida como Cypher. Trata-se de uma linguagem de consulta declarativa que o Neo4j utiliza para consultas e atualizações expressivas e eficientes. É no entanto uma ferramenta que ainda está em amadurecimento, portanto muito provavelmente haverá alterações futuras em sua sintaxe.

O Neo4j tem um par de ferramentas que suporta a execução do Cypher; são elas o Neo4j-shell e o Web Console Admin.

O shell Neo4j é uma ferramenta de linha de comando que faz parte da distribuição do servidor Neo4j; através dele é possível conectar-se a um banco de dados Neo4j local, apontando-o para o diretório onde os dados Neo4j são armazenados ou para um servidor remoto através do Neo4j RMI, fornecendo o nome do host e a porta para conectar ao script de inicialização do Neo4j Shell.

A Web Console Admin é uma interface web rica baseada em navegador para instanciar o Neo4j. Essa ferramenta tem uma abundância de recursos, que lhe permitem consultar, manipular e visualizar dados do grafo Neo4j.

6. CONCLUSÃO

Este trabalho evidenciou que vivemos atualmente um momento em que a exposição ao crescimento do volume dos dados é visível e perceptível, fato que deve ser um ponto de atenção para as organizações como um todo, independentemente do seu porte econômico, e por dois motivos principais: primeiro, para mantê-las competitivas em seu mercado, e segundo, para que não sejam surpreendidas por uma avalanche de dados, para cujo tratamento seus servidores não estejam preparados, afetando, por consequência, o bom andamento de seu negócio.

Conforme Tiware (2011), com a evolução tecnológica a indústria eletrônica vem contribuindo muito para o aumento no volume de dados, por meio de diversos lançamentos de

produtos, gerando assim grandes desafios e oportunidades em torno do armazenamento e da análise de dados.

Em virtude desse crescimento no volume de dados, os bancos de dados relacionais começaram a apresentar algumas limitações. Segundo Sadalage e Fowler (2013), para responder a essa lacuna surgiu a tecnologia NoSQL, voltada especificamente para a necessidade de lidar com grandes volumes de dados.

Os bancos de dados classificados na tecnologia NoSQL apresentam-se como a solução para as organizações se prepararem para lidar com o excesso de volume dos dados que estamos vivendo atualmente. A tecnologia NoSQL deve ser bem levantada e analisada pela organização, já que, como há vários bancos de dados classificados nessa tecnologia, a organização deverá avaliar o que melhor se enquadre a sua estratégia de negócio.

O NoSQL apresenta muita flexibilidade em relação aos bancos de dados relacionais. A mera não exigência de um esquema definido, como é o caso dos bancos de dados relacionais, já representa uma vantagem que contribui para que o banco de dados NoSQL se adapte ao negócio, e não o contrário. Também apresenta um melhor desempenho e não requer servidores potentes ou de grande porte para o armazenamento dos dados, como ocorre com os bancos de dados relacionais. Tudo isso resulta, em última análise, em redução de custos para a organização.

Vale salientar que as bases de dados relacionais não irão desaparecer com o surgimento da tecnologia NoSQL, portanto, ao se optar pela tecnologia NoSQL, é preciso avaliar pontos como escalonamento, consistência e disponibilidade.

Uma restrição que as organizações devem levar em consideração com bastante sensibilidade ao optar pelo NoSQL é a falta de profissionais qualificados e disponíveis no mercado com conhecimento e experiência nessa tecnologia. Aqueles que dispõem de qualificação e experiência profissional nessa tecnologia serão de alto custo para a organização, que tem que avaliar com sensatez a relação custo-benefício dessa opção.

REFERÊNCIAS BIBLIOGRÁFICAS

BRITO, Ricardo W. **Bancos de Dados NoSQL x SGBDs Relacionais: Análise Comparativa**. Disponível em: <http://www.infobrasil.inf.br/userfiles/27-05-S4-1-68840-Bancos%20de%20Dados%20NoSQL.pdf> Acesso em 17 de abril de 2013.

CAMPO, Edmilson. **Introdução à teoria dos grafos**. Disponível em: <http://edmilsoncampos.files.wordpress.com/2012/01/introduc3a7c3a3o-c3a0-teoria-dos-grafos.pdf> Acesso em 03 de julho de 2013.

DEAN, Jeffrey; GHEMAWAT, Sanjay. **MapReduce: Simplified Data Processing on Large Clusters**. Disponível em <http://research.google.com/archive/mapreduce.html> Acesso em 20 de julho de 2013.

ÉVORA, Luiz. **MapReduce: Solução para o Big Data?** Disponível em <http://twistsystems.com/blog/2013/01/27/mapreduce-solucao-para-o-big-data/#.Ueqf3o3OsZg> Acesso em 20 de julho de 2013.

GIL, Antonio Carlos. **Como elaborar projetos de pesquisa**. 4. ed. 12ª reimpr. São Paulo: Atlas, 2009.

GUIMARÃES, José de Oliveira. **Teoria dos grafos.** 2001**.** Disponível em < http://www.ebah.com.br/content/ABAAAenhcAE/teoria-dos-grafos> Acesso em 09 de Junho de 2013.

KLINE, Kevin E. **SQL - O guia essencial.** Rio de Janeiro: Alta Books, 2010.

NETTO, Paulo Oswaldo Boaventura. **Grafos: teorias, modelos, algoritmos**. 5ª edição revista e ampliada.São Paulo: Editora Edgar Blücher, 2012. Disponível em <http://issuu.com/editorablucher/docs/issu_grafos/2?mode=embed&layout=http%3A//skin.issuu.com/v/light/layout.xml> Acesso em 01 de maio de 2013.

NASCIMENTO, Jean. **NOSQL – Você realmente sabe do que estamos falando?** Disponível em <http://imasters.com.br/artigo/17043/banco-de-dados/nosql-voce-realmente-sabe-do-que-estamos-falando/>Acesso em 07 de abril de 2013.

PAN, Henrique; CRUZ, Pedro e VASCONCELLOS, Pedro de. **MapReduce (Big Data)**. Disponível em < http://www.gta.ufrj.br/grad/12_1/mapreduce/index.htm> Acesso em 20 de julho de 2013.

PARTNER, Jonas; VUKOTIC, Aleksa; WATT, Nicki. **Neo4j in Action**. MEAP Began, 2012.

REDMOND, Eric; WILSON, Jim R. **Seven Databases in Seven Weeks.** Pragmatic Programmers, 2012.

ROBINSON, Ian; WEBBER, Jim; EIFREM, Emil. **Graph Databases**. O'Reilly Media, 2013.

SADALAGE, Pramod J., FOWLER, Martin. **NoSQL distilled: a brief guide to the emerging world of polyglot persistence**. Pearson Education, Inc., 2013.

STRAUCH, Christof. **NOSQL databases.** Disponível em <http://www.christof-strauch.de/nosqldbs.pdf>. Acesso em 08 de junho de 2013.

TIWARE, Shashank. **Professional NoSQL.** John Wiley & Sons, Inc**.** , 2011.

VERGARA, Sylvia Constant. **Projetos e relatórios de pesquisa em Administração.** 14 ed. São Paulo: Atlas, 2013.