DESEMPENHO DE BANCO DE DADOS NÃO RELACIONAIS COM BIG DATA

Bruna Carvalho Kalles Gomes[1]
Carla de Almeida Martins Basso[2]

RESUMO
O volume de informação gerada pela humanidade vem aumentando de forma exponencial, levando as organizações a buscarem alternativas para conseguir obter dados concretos e específicos em um curto espaço de tempo. A grande maioria desses dados encontram-se nos mais diversos meios, como *Facebook*, *Twitter*, câmeras de monitoramento, entre outros. Na busca pelo aprimoramento no retorno das informações, surge o conceito Big Data o qual refere-se a um conjunto de dados cujo crescimento não pode ser mensurado. O Big Data vem tornando-se a escolha de grandes empresas, pois possibilita englobar vários tamanhos e formatos de dados em uma única analise, esses dados são armazenados em bancos de dados não relacionais, os quais tem por conceito tornar os processos mais eficientes. Para demostrar a utilização do conceito do Big Data e avaliar o desempenho, utilizou-se dois sistemas gerenciadores de banco de dados, o não relacional *MongoDB* juntamente com a ferramenta de gerenciamento do banco o *MongoVUE* e o relacional *MySQL*. Foram realizadas consultas utilizando dados públicos, os quais foram inseridos no *MongoDB*, simulando assim o Big Data pelo seu tamanho e estrutura. A partir desses dados foi comparada a performance entre ambos os bancos, sendo assim possível visualizar como os mesmos se comportam nas mesmas situações de consultas. A análise permitiu concluir que os resultados obtidos no *MongoDB* e no MySQL é visível a diferença no tempo de execução em uma consulta mais complexa. Essa diferença ocorre pois no *MongoDB* todas as informações ficam ao alcance, enquanto que no MySQL é necessário buscar em diferentes tabelas.
Palavras- chaves: Big Data, Performance de banco de dados, Banco de dados não relacionais.

ABSTRACT
The volume of information generated by humanity is increasing exponentially, leading organizations to seek alternatives to achieve concrete and specific data in a short time. The vast majority of these data will be available in various media, such as Facebook, Twitter, monitoring cameras, among others. In the search for improvement in the return of information, there is the Big Date concept which refers to a data set whose growth cannot be measured. The Big Date is becoming the choice of large enterprises, because enables encompass various sizes and data formats in a single analysis, this data is stored in non-relational databases, which is concept make more efficient processes. To demonstrate the use of the concept of Big Data and assess performance, we used two database management systems, relational not MongoDB along with the Bank's management tool MongoVUE and MySQL relational. Consultations using public data, which were inserted into MongoDB, simulating how the Big Data by its size and structure. From these data was compared the

[1]Bacharel em Sistemas de Informação pela Unoesc Chapecó. b.kallesgomes@gmail.com

[2]Professora Orientadora Unoesc Chapecó. carla.basso@unoesc.edu.br - Mestre em Ciência da Computação pela UFRGS e Especialista em Gerenciamento de Projetos pela FGV.

performance between both banks, being thus possible to visualize how they behave in the same situations of queries. The analysis allowed us to conclude that the results obtained in the MongoDB and in MySQL is visible the difference at runtime in a more complex query. This difference occurs because MongoDB all information is within reach, while in MySQL is necessary to get into different tables.
Keywords: Big Data, Performancedatabase, Non-relational database

1 INTRODUÇÃO

Obter informações processadas rapidamente e ao mesmo tempo confiáveis sempre foi algo muito importante e desafiador para as organizações, ainda mais com a disseminação da internet onde as mídias sociais podem ser acessadas de qualquer lugar, tornando as informações quase que momentâneas. A partir do entendimento que as informações devem ser analisadas e processadas momentaneamente ou o mais próximo disso, fica visível a necessidade da utilização do Big Data, o qual refere-se a um agrupamento de dados cujo crescimento é exponencial e cujo o tamanho está além da habilidade das ferramentas de gerenciar e analisar esse montante de dados.

O Big Data pode ser utilizado nas mais diversas áreas para os mais diversos fins, no entanto ainda são necessários muitos estudos para que isso se concretize e seja atingida a sua plenitude. O que impulsionou a realização deste artigo, foi a falta de informação sobre como os dados são analisados e processados em um banco de dados não relacional juntamente com a falta de informação quanto a funcionalidade do Big Data, que como o nome diz trabalha com grandes quantidades de dados dos mais diversos formatos e oriundos das mais diversas fontes. Por conta dessa ausência de informação e pela demanda de gerenciamento desta grande massa de dados é que esse artigo justifica-se.

Este trabalho tem como objetivo avaliar e demostrar através de exemplificações práticas em uma base de dados pública a diferença de performance entre bancos de dados relacionais e não relacionais, exibindo essas informações de forma clara e sucinta.

Além desta seção introdutória, o presente artigo apresenta, na segunda seção, a revisão bibliográfica onde é abordado o Big Data e suas aplicações bem como as características dos bancos de dados não relacionais e do *MongoDB*. A terceira seção descreve os procedimentos metodológicos utilizados no estudo. Na quarta seção são apresentados e discutidos os resultados da pesquisa, e na quinta seção são apresentadas as conclusões e considerações finais do estudo.

2 REVISÃO BIBLIOGRÁFICA

Para atender a proposta de desenvolvimento de uma aplicação utilizando as diretrizes do Big Data, optou-se por utilizar a base de dados pública do Instituto de Pesquisa Econômica Aplicada (IPEA). Entre os dados disponibilizados no IPEA, para a realização deste artigo utilizou-se os índices de desenvolvimento humano (IDH) – educação, os valores da renda per capita e o número de mulheres grávidas.

Para a extração de informações utilizou-se o banco de dados não relacional *MongoDB* o qual é um banco de dados orientado a documentos e que possuiu boa aceitação no mercado. Para visualizar os dados de forma mais fácil e rápida utilizou-se a ferramenta *MongoVUE*, que é um gerenciador do banco de dados *MongoDB*.

2.1 BIGDATA

O termo Big Data refere-se a um conjunto de dados cujo crescimento é exponencial e cuja a dimensão está além da habilidade das ferramentas de gerenciar e analisar dados (TAURION, 2013, p.29).

Para efetivamente usar o conceito de Big Data não basta apenas comprar pacotes de tecnologia, Big Datanecessita de transformações em processos de negócio, fontes de dados, infraestrutura de tecnologia, capacitações e mudanças tanto na empresa quanto na área de Tecnologia da Informação (TI) (TAURION, 2013, p.34).

Para se utilizar o Big Datanecessita-se passar por algumas fases, e segundo Taurion (2013, p.34-35) essas fases seriam:

- Coleta de dados - Cada negócio tem necessidade de coletar dados diferentes podendo ser de sistemas transacionais, comentários que circulam nas mídias sociais, em sensores que medem o fluxo de veículos ou até mesmo em câmeras de vigilância nas ruas.
- Limpeza e formatação - Apenas armazenar dados de forma crua como foi obtida tornará difícil a análise dos mesmos, além do que também é importante validar os dados coletados, erros e dados incompletos ou inconsistentes devem ser eliminados para não prejudicar futuras análises.
- Integração e agregação - Os dados de diferentes tipos e formatos precisam receber tratamentos específicos sendo importante definir categorias de dados e seus critérios de validação e aceitação assim como também critérios de segurança.
- Análise e interpretação dos resultados - Existem *terabytes* de dados armazenados e a grande questão é "que perguntas fazer" para conseguir chegar a padrões e correlações que gerem valor para o negócio? Normalmente usam-se consultas e *queries* em um *Data Warehouse* gerado por transações obtidas por *Enterprise Resource Planning* (ERP) o qual são estruturadas e dentro de um domínio de conhecimentos bem restrito, mas quando há coleta de diversas fontes criar essas *queries* demanda de muito conhecimento por parte dos usuários. É nesse ponto que entra o *datascientist* (cientista de dados), que é um profissional multidisciplinar, que possui habilidades em ciência da computação, matemática, estatística e logicamente conhecimento do negócio onde está inserido, esta fase demanda de investimentos para novas formas de visualização dos resultados.

De acordo com Taurion ([2013]) e Queiroz (2013) o Big Data se baseia em 5 'Vs' sendo eles: Velocidade, volume, variedade, veracidade e valor. Baseia-se na velocidade por que no Big Data os dados são transmitidos em grandes velocidades e muitas vezes a resposta precisa ser praticamente em tempo real. É baseado em volume levando em consideração o crescimento exorbitante de dados gerados nos últimos anos, mas tende a crescer ainda mais devido à adoção da computação nas nuvens das grandes companhias. Baseia-se na variedade visto que o Big Data se ocupa de dados não estruturados para contextualizá-los permitindo respostas estratégicas. A veracidade tem a função de validar obrigatoriamente a credibilidade das informações e o valor é a relação de credibilidade que os dados devem produzir.

Mesmo com tantas informações sobre o Big Data, ainda não há um único conceito para esclarecer esse fenômeno. Em uma visão macro pode-se dizer que são grandes quantidades de dados analisadas em um tempo relativamente curto. Segundo Alecrim (2013):

> A princípio, podemos definir o conceito de Big Data como sendo conjuntos de dados extremamente grandes e que, por este motivo, necessitam de ferramentas

> especialmente preparadas para lidar com grandes volumes, de forma que toda e qualquer informação nestes meios possa ser encontrada, analisada e aproveitada em tempo hábil.

2.2 ALGUMAS APLICAÇÕES DE BIG DATA

Neste item é apresentado a descrição de alguns *cases* nacionais e internacionais, mostrando como os mesmos estão utilizando o conceito de Big Data dentro de suas organizações.

Um desses cases é o *MapLink*que é uma empresa brasileira especializada em digitalização de mapas. De acordo com Dalmazo (2012, p.1), alguns anos atrás a empresa pôs em xeque a credibilidade da Companhia de Engenharia de Tráfego (CET) da cidade de São Paulo.

A CET utilizava câmeras espalhadas pela cidade e contava com os fiscais de trânsito para calcular os índices de congestionamento, enquanto que o *software* da *MapLink*, usado por empresas de rastreamento por satélite, cruza, em tempo real, as informações enviadas por mais de 400.000 veículos espalhados pela cidade.

A empresa utiliza a seguinte métrica, se os veículos estão parados, há congestionamento, se estão andando em velocidade baixa é trânsito e se a velocidade for plena a pista está livre. A precisão é tanta que hoje, além dos mapas, a *MapLink*vende informações sobre o trânsito para companhias de logística decidirem os melhores horários e rotas para suas entregas.

Outra empresa que utiliza o Big Data é a americana Walmart que é a maior varejista do mundo. A empresa é considerada referência por colher dados *on-line* para impulsionar as vendas de suas lojas físicas. Os *softwares* desenvolvidos pela empresa conseguem monitorar discussões sobre, por exemplo, futebol, e sabendo dessas informações em questões de horas os gerentes das lojas passam a expor nas vitrines produtos de determinados times (DALMAZO, 2012, p.2).

De acordo com Dalmazo (2012, p.2) “Hoje, o Walmart tem mais de 12 sistemas diferentes que processam, diariamente, cerca de 300 milhões de atualizações de internautas em redes sociais, como o *Facebook* e o *Twitter*”.

A demanda interna por soluções Big Data levou a varejista a criar a WalmartLabs em 2011 o qual surgiu com a compra da Kosmix. A Kosmix ganhou fama em 2010 quando desenvolveu um sistema para gerenciamento de estoques de varejistas durante a *Black Friday* (sexta-feira negra), em novembro o qual é o principal dia de queima de estoque do varejo nos Estados Unidos (DALMAZO, 2012, p.2).

Além das empresas citadas acima, outra empresa que também tem usufruído do Big Data é o Google. Os usuários fornecem aos computadores do Google inúmeras informações todos os dias. De acordo com Petry (2013, p.71), quando alguém faz buscas, é informado ao Google sua localização geográfica, seu tempo de permanência em cada página na *Web*, revelando assim suas predileções.

Cada clique gera uma nova informação, e com isso a Google consegue identificar onde poderá ocorrer determinado tipo de epidemia como, por exemplo, uma gripe, antes mesmo dos órgãos de saúde da cidade saberem. Isso ocorre devido às pesquisas que as pessoas fazem sobre o assunto.

Mas além desses exemplos existem inúmeros outros que podem ser inseridos nessa lista, pois a tecnologia tem sido apontada como uma ferramenta de resolução de problemas e para esse fim o Big Datavem sendo usado no gerenciamento do tráfego de Los Angeles (Estados Unidos), em prevenções de doenças em Nova York (Estados Unido) e no saneamento de South Beach (Estados Unidos) (OLHAR DIGITAL, 2013a).

2.3 BANCO DE DADOS NÃO RELACIONAL

Segundo Gondim (2013) "*NoSQL* é um termo utilizado para definir um tipo de banco de dados que não segue normas de tabelas (*schemas*) determinadas previamente. Seu significado é (*Not only SQL* - Não só SQL) e vem do conceito de que o banco de dados não necessita de normalização e relacionamentos".

Em primeiro momento era um nome de um banco de dados relacional de código aberto que não possuía interface *Structured Query Language* (SQL). O *NoSQL* consiste em sistemas de armazenamento que suprem necessidades que os bancos relacionais são ineficazes (NASCIMENTO, 2010).

Segundo KOKAY([2013?]), as características que tornam os bancos de dados *NoSQL* diferentes do modelo relacional são:

- Escalabilidade Horizontal: Na medida em que os dados crescem aumenta a necessidade de melhorar o desempenho e para isso a escalabilidade horizontal se torna viável, ela não é simples, pois requer *threads* ou que processos de uma tarefa sejam criados e distribuídos. É possível escalar tanto em bancos de dados relacionais como em bancos *NoSQL* , só que nos bancos relacionais essa escalada é mais complexa. Isso por que quando há muitos processos conectados simultaneamente há uma alta concorrência aumentando o tempo de acesso as tabelas. E nesse sentido a grande vantagem dos bancos *NoSQL* é a ausência de bloqueios, permitindo a escalabilidade horizontal com mais facilidade e eficiência, ou seja, ele não é afetado pelo aumento da concorrência.
- Ausência de Esquema (*Schema-free*) ou esquema flexível: Bancos de Dados *NoSQL* possuem ausência parcial ou total de esquema que define a estrutura de dados, facilitando uma alta escalabilidade e alta disponibilidade, com a facilidade de escalabilidade e disponibilidade não há uma garantia de integridade dos dados, fato que não ocorre no modelo Relacional.
- Suporte Nativo a replicação: Essa é outra forma de prover escalabilidade, pois quando se realiza a replicação de forma nativa o tempo gasto para recuperação diminui.
- *Application Programming Interface* (API) simples para acessar o banco de dados. Nos bancos de dados *NoSQL* o foco está em recuperar os dados de forma eficiente e para isso é essencial APIs desenvolvidas para acessar as informações de forma rápida e eficiente.
- Consistência eventual: Nos bancos *NoSQL* nem sempre a consistência é mantida. Essa característica é baseada no teorema de *Consistency, Availability e Partition Tolerance* (CAP) que traduzindo significa (Consistência, Disponibilidade e tolerância à partição), que afirma que só é possível garantir duas dessas propriedades. Por conta disso é necessário que haja um planejamento para que o sistema possa tolerar inconsistências com o objetivo de priorizar a disponibilidade.Além de possuírem características diferentes dos bancos de dados relacionais, de acordo com Kokay ([2013?]), os bancos de dados *NoSQL* possuem quatro categorias que os diferenciam entre si, são elas.
- Chave-valor (*key-value*): Modelo considerado simples o qual permite a sua visualização através de uma tabela de *hash*, no qual há uma chave única e um indicador de determinado dado, podendo ser uma *String* ou um binário.
- Banco de Dados Orientado a Documento:Este modelo armazena coleções de documentos. Um documento, no geral, é um objeto identificador único e um conjunto de campos que podem ser *strings*, listas ou documentos aninhados, neste

modelo temos um agrupamento de documentos sendo que em cada documento há um conjunto de campos e o valor deste campo. Neste modelo há ausência de esquema pré-definido (*schema free*). Isso significa que se for incluído novos campos no documento, os outros documentos existentes não serão afetados. Outra característica é que não é necessário armazenar valores de dados vazios para campos que não possuem um valor.

- Orientado a Coluna (*column family*):Este tipo de banco de dados foi criado para armazenar e processar uma grande quantidade de dados distribuídos em diversas máquinas. Existem chaves, mas elas apontam para atributos ou colunas múltiplas, onde os dados são indexados por uma tripla (coluna, linha e *timestamp*), a coluna e a linha são identificadas por chaves e o *timestamp* permite diferenciar múltiplas versões de um mesmo dado. O conceito associado a este modelo é o de família de colunas, o objetivo é reunir colunas que armazenam o mesmo tipo de informação.
- Orientado a Grafos:Este modelo possui três componentes básicos: nós (vértices dos grafos), os relacionamentos (arestas) e as propriedades (conhecidos também como atributos). Ele é visto como multigrafo rotulado e direcionado, onde cada par de nós pode ser conectado por mais de uma aresta. O modelo orientado a grafos possui uma alta performance, permitindo um bom desempenho nas aplicações, sendo muito útil para fazer consultas complexas.

2.3.1 *MongoDB*

O *MongoDB* é um banco de dados orientado a documentos, escalável, de alta performance e livre de esquemas que foi lançado em 2009 (BATISTA et al., 2013).

Utiliza documentos *JavaScript Object Notation* (JSON) para o modelo de dados e suas consultas são baseadas em documentos. Ele inclui o modulo *sharding* automático (particionamento em múltiplos servidores), suporta internamente MapRaduce, é desenvolvido em C++ e possui *drivers* para várias linguagens como: *Python*, *Java*, *PHP*, *Erlang* entre outras (NOSSAL; MOURA, 2011).

Segundo os autores acima citados os documentos JSON são internamente guardados como documentos *Binary JavaScript Object Notation* (BJSON) (formato binário) isso quando for menor que 16MB, quando os arquivos forem maiores que 16MB utiliza-se a especificação GridFS.

Os GridFS são uma especificação para armazenamento e recuperação de arquivos, que divide os arquivos em pedaços de 256Kb. GridFS é útil não só para armazenar arquivos que excedem 16 MB, mas também para armazenar todos os arquivos que se deseja acesso sem ter que carregar o arquivo inteiro na memória (*MONGODB*, [entre 2011 e 2013]).

2.3.2 *MongoVUE*

Para trabalhar com o *MongoDB* de forma mais fácil e simples, foi realizado o *download* de um gerenciador do banco de dados *MongoDB*, foi escolhido o *MongoVUE*, pois ele possui uma interface relativamente simples e intuitiva.

O *MongoVUE* possui uma opção *Free*, onde é disponibilizado duas conexões ao servidor e pode-se abrir apenas três *views*, e para realizar importação de dados é possível utilizar apenas o MySQL, não é possível gerenciar *GridFS* (*GridFS* é uma especificação para armazenamento e recuperação de arquivos que excedam o BSON -documento limite

de tamanho de 16MB) e também não é possível monitorar servidores *MongoDB*. As demais opções são todas iguais.

Além da limitada opção *free* há também as opções pagas, onde há mais recursos disponíveis. Mas embora limitada a opção *free* foi o suficiente para a realização deste estudo.

3 PROCEDIMENTOS METODOLÓGICOS

Para a realização deste estudo foram utilizados os métodos de abordagem dedutiva e procedimento monográfico, onde parte-se de ideias gerais e, a seguir, desce a uma questão particularizada e o procedimento monográfico foi utilizado através de um aprofundado estudo sobre o tema envolvido.

Quanto às técnicas esse estudo se caracteriza por documentação indireta no qual faz uso de pesquisa documental e bibliográfica, utilizando de arquivos públicos, artigos e monografias para sanar as dúvidas existentes.

O estudo caracterizou-se por consultas de laboratório, através simulações em computador utilizando o banco de dados *MongoDB* e a ferramenta de gerenciamento de banco de dados do *MongoDB* o *MongoVUE*, realizou-se também pesquisa documental e bibliográfica envolvendo documentos públicos e privados, registros e regulamentos com fontes confiáveis e um estudo de caso para aprofundamento e detalhamento do assunto.

Quanto aos fins esse estudo caracteriza-se por ser aplicada, pois é motivada pela necessidade de resolver problemas concretos, tendo, portanto uma finalidade prática. Caracteriza-se também por ser exploratória, pois foi realizada em uma área nova possibilitando assim novas experiências.

A amostragem utilizada é a não probabilística, e a partir da escolha da amostragem não probabilística o estudo foi focado no tipo de amostra intencional onde o pesquisador usa seu julgamento para selecionar os membros da população que são boas fontes de informação precisa.

O presente estudo abordou a análise quantitativa bem como a análise qualitativa. Onde quantitativa é caracterizada pela atuação nos níveis de realidade e apresenta como objetivos a identificação e apresentação de dados, indicadores e tendências observáveis (MIRANDA, 2008).

E análise qualitativa trabalha com representações, hábitos, atitudes e opiniões, e na medida em que encontra-se padrões nos dados vai se desenvolvendo conceitos, ideias e entendimento sobre esses dados (MIRANDA, 2008). Esse método emprega procedimentos interpretativos, não experimentais, com valorização dos pressupostos relativistas e a representação verbal dos dados.

4 APRESENTAÇÃO DOS RESULTADOS

Inicialmente são apresentadas as diferenças entre os banco de dados relacional e o banco não relacional *MongoDB*, bem como os resultados das execuções de consultas no *MongoDB* e no MySQL.

4.1 DIFERENÇAS ENTRE BANCOS DE DADOS RELACIONAIS E O BANCO NÃO RELACIONAL *MONGODB*

Os bancos de dados relacionais foram idealizados por um cientista britânico chamado Edgar Frank “Ted” Codd, que em 1970 surgiu com treze (13) leis, as quais descreveram o que é um banco de dados relacional e o que é um SGBDR (Sistema

Gerenciador de Banco de Dados Relacionais), além de algumas leis de normalização que descrevem as propriedades dos dados relacionais (COSTA, 2010).

Segundo Dias Neto (2011) “Um banco de dados relacional armazena dados em tabelas. Tabelas são organizadas em colunas, e cada coluna armazena um tipo de dado (inteiro, números reais, *strings*, data, entre outros.)”.

A linguagem padrão dos bancos de dados relacionais é a *Structured Query Language* (SQL) como é conhecida. As tabelas ou entidades são uma estrutura que contém linhas e colunas, onde em cada tabela as linhas possuem os mesmos números de colunas. Um banco de dados é composto por diversas tabelas (ALBANI, 2011). Um exemplo de tabela pode ser visualizado na figura 1.

Figura 1 - Tabela/Entidade Cliente.

ID	Nome	Telefone	Cidade
1	Loja São Francisco	3220-8967	Salto do Lontra
2	Loja 10	3456-9078	Palmira
3	Loja Travel	3456-2312	Palmas
4	Loja Nono Micoud	2344-1234	Pato Branco

Fonte: Albani (2011).

Para conseguir manipular mais facilmente as informações de um banco de dados, utiliza-se um sistema de gerenciamento de banco de dados relacionais (SGBDR). O SGBDR é um software que controla o armazenamento, recuperação, exclusão, segurança e integridade das informações em um banco de dados (DIAS NETO, 2011).

Segundo Dias Neto (2011) “O uso mais comum de SGBDRs 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)”.

Além do CRUD, na manipulação dos dados existem as cláusulas, que são as condições de modificação que permitem definir os dados que se deseja selecionar ou modificar, podendo ser utilizado com todas as instruções de manipulação de dados(ALBANI, 2011).

Há também os operadores relacionais, os quais permitem realizar comparação de valores, sendo a verificação feita de acordo com a expressão, possibilitando assim a execução de uma instrução em vários dados (ALBANI, 2011).

Figura 2 - Operadores relacionais.

Operador	Descrição
<	Menor
>	Maior
<=	Menor ou igual
>=	Maior ou igual
=	Igual
!=	Diferente

Fonte: Albani (2011).

No SGBDR há também os operadores lógicos, os quais permitem a verificação de expressões lógicas como mostra a figura 3. Eles são utilizados juntamente com as instruções de manipulação de dados, auxiliando no processo de escolha de dados a serem manipulados (ALBANI, 2011).

Figura 3 - Operadores Lógicos.

Operador	Descrição
and	Avalia as condições e devolve um valor verdadeiro caso ambos sejam corretos.
or	Avalia as condições e devolve um valor verdadeiro se algum for correto.
not	Devolve o valor contrário da expressão.

Fonte: Albani (2011).

Em contrapartida, o *MongoDB* é um banco de dados que foi desenvolvido pela empresa Norte Americana 10gen, sobre o conceito de documentos e coleções, onde cada coleção armazena vários documentos (SOUZA, 2013).

Ainda segundo o autor acima, no *MongoDB* não há relacionamento entre as coleções, como existe entre tabelas em um banco relacional, os documentos são considerados dinâmicos.

SegundoSouza(2013), "Os documentos no *MongoDB* seguem o padrão JSON (*JavaScript Object Notation*), e após serem persistidos são transformados em BSON, um tipo de dado binário e serializado próprio do *MongoDB*."

Como os documentos possuem formato JSON, a manipulação dos mesmos é realizada com base na chave (*key*) e no valor (*value*), tornando assim possível o dinamismo entre os documentos. Por conta desse dinamismo nos documentos, há diferentes métodos de consultas que podem ser utilizadas no *MongoDB*, sendo elas: consultas simples (*Simple Queries*), consultas avançadas (*Advanced Queries*) e o *framework* de agregação (*Aggregation Framework*)(SOUZA, 2013).

Uma consulta é caracterizada como *Simple Query*, quando utiliza apenas chaves e valores para retornar uma informação, ela pode ser executada utilizando o método find().

No padrão SQL existem alguns operadores ou funções, que são bastante utilizadas em consultas como soma, média, menor ou igual, entre outas. Esses operadores também existem e são utilizados no *MongoDB*. Quando utilizados nas consultas são chamadas de *Advanced Queries* (SOUZA, 2013).

Figura 4 - Operadores.

Operador	Função
$sum	Utilizado para a soma de valores.
$avg	Utilizado para encontrar a média entre os valores.
$gte	Utilizado como critério de "maior ou igual".
$lte	Utilizado como critério de "menor ou igual".
$min	Utilizado para recuperar o "menor valor".
$max	Utilizado para recuperar o "maior valor".
$in	Utilizado para localizar dentro de um array qualquer um dos parâmetros.
$all	Utilizado para localizar dentro de um array todos os elementos do parâmetro.
$regex	Utilizado para o uso de expressões regulares.
$and	Idêntico ao operador "and" do SQL.
$or	Idêntico ao operador "or" do SQL.

Fonte: (Souza, 2013).

De acordo com Souza (2013), o *MongoDB* assim como os bancos relacionais possui índices. O índice permite localizar documentos com base em valores armazenados em campos específicos. Os índices no *MongoDB* são muito semelhantes aos disponíveis em bancos de dados relacionais.

No *MongoDB*, não é interessante criar índices para chaves que não serão utilizadas em pesquisas, pois criar muitos índices pode ocasionar um processo mais lento, no

momento de salvar os dados no banco, pois para cada documento o Mongo criará vários indexadores (SOUZA, 2013).

Segundo Souza (2013) "o *framework* de agregação do *MongoDB* fornece funcionalidade semelhantes à *GroupBy*, *OrderBy*, *Where* e operadores afins do SQL, assim como formas simples de '*join*'. O objetivo do *framework* é juntar informações de diversos documentos, retornando um resultado com as informações agrupadas.

Utilizando o conceito de agregação, pode-se adicionar campos temporários, que podem guardar informações como o resultado de uma média ou de uma soma. Os resultados de uma agregação, serão valores definidos por chaves que encontram-se na própria consulta (SOUZA, 2013).

Na figura5, exibe-se uma comparação, para mostrar como são os operadores no SQL dos bancos relacionais e como eles apresentam-se no *MongoDB* (SOUZA, 2013).

Figura 5 - Operadores SQL X *MongoDB*.

Operador SQL	Framework de Agregação MongoDB
WHERE	$match
GROUP BY	$group
HAVING	$match
SELECT	$project
ORDER BY	$sort
LIMIT	$limit
SUM()	$sum
COUNT()	$sum
JOIN	$unwind

Fonte: Souza(2013).

4.2 SELEÇÃO DA BASE DE DADOS

Para que esse trabalho fosse realizado, houve a necessidade de realizar buscas por dados que estivessem a disposição de forma gratuita, entre tantos segmentos a serem abordados, o escolhido foi o relacionado a parte social.

[B1] Comentário: verificar

Neste contexto, escolheu-se a base do IPEA, onde os dados encontram-se divididos em três assuntos: macroeconômico, regional e social. O macroeconômico aborda dados financeiros do Brasil, o Regional aborda assuntos econômicos, demográficos e geográficos por estados e municípios e o social aborda diversos indicadores como, por exemplo, o de distribuição de renda, de pobreza, educação e saúde.

Realizou-se o trabalho levando em consideração os municípios brasileiros, e os índices de desenvolvimento humano (IDH) – educação, o qual é obtido a partir da taxa de alfabetização e da taxa bruta de frequência à escola, convertidas em índices por: (valor observado - limite inferior) / (limite superior - limite inferior), o IDH vai de 0 (nenhum desenvolvimento humano) a 1 (desenvolvimento humano total). Quanto mais próximo de 1, mais desenvolvido é o município.

Utilizou-se valores da renda per capita, o qual é a razão entre o somatório da renda familiar per capita de todos os domicílios e o número total de domicílios no município, esses valores são expressões em reais. Também utilizou-se o número de mulheres grávidas, o qual expressa o número de gestantes por município.

4.3 IMPORTAÇÃO DOS DADOS

Após possuir os dados realiza-se a seleção deles, onde dados inconsistentes são excluídos, pois se os dados não forem tratados nesse momento, em análises futuras as informações geradas serão apresentadas de forma incorreta, tornando assim todo o

processo de extração de conhecimento inválido. Após a etapa de seleção, os dados estarão “limpos”, prontos para as etapas seguintes.

Após possuir todos os dados necessários, remove-se todas as linhas que possuíam em ambos os campos (1991 e 2000) os valores 0 ou *null*, como mostra a figura 6. Retirando essas linhas, é possível garantir que em um dos anos, houve a captação de dados nos municípios, sendo possível assim realizar análises com eles. Após realizar a limpeza nos dados, importa-se os mesmo para o banco MySQL.

Figura 6 - Linhas selecionadas para serem excluídas.

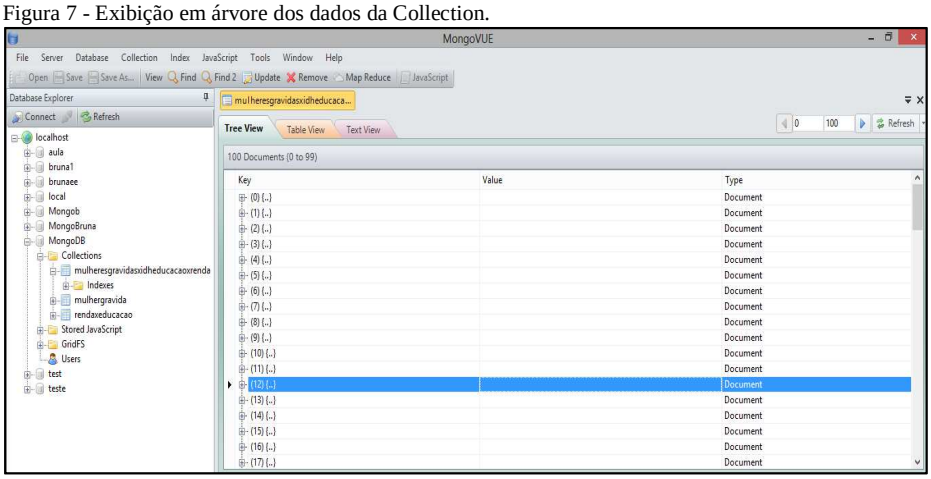
	Sigla	Regiao	Código	Município	1991	2000
9	AC	Norte	1200336	Mâncio Lir	0	1190
10	AC	Norte	1200344	Manoel Ur	0	774
11	AC	Norte	1200351	Marechal	0	0
12	AC	Norte	1200385	Plácido de	0	
13	AC	Norte	1200807	Porto Acre	0	741
14	AC	Norte	1200393	Porto Wal	0	0
15	AC	Norte	1200401	Rio Branco	0	10488

Fonte: O Autor.

Após importado os dados para o MySQL, possui-se agora tabelas com essas informações. Após esse processorealiza-se a importação para o banco *MongoDB* utilizando o *MongoVUE*. Para realizar a importação dos dados, faz-se necessário acessar o RDBMS Import,

A figura 7demostra como ficou a estrutura do banco *MongoDB*, após o processo de importaçãodas tabelas do MySQL.

Figura 7 - Exibição em árvore dos dados da Collection.



Fonte: O Autor.

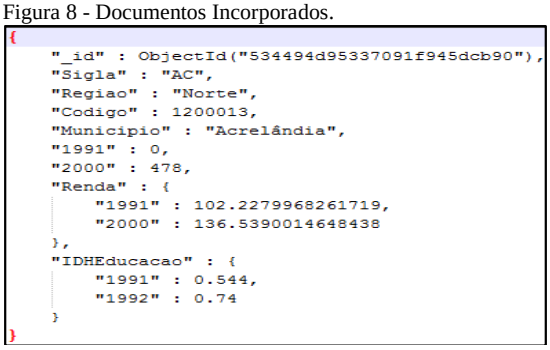
4.4 ANÁLISE DOS DADOS E COMPARAÇÃO *MONGODB* X MYSQL

Como no *MongoDB* não é utilizado o *join*, para poder ter acesso às informações de outras tabelas, pode-se utilizar relações de referências ou documentos incorporados. As relações de referênciasarmazenam as informações entre os dados incluindo ligações e referências de um documento em outro. Essas ligações de referências são o ("_id": ObjectId()), de um documento, que é inserindo em outros documentos. De modo geral a relação de referência é um modelo de dados normalizados.

As relações de documentos incorporados armazenam dados relacionados em uma única estrutura de documento. Em documentos *MongoDB* é possível incorporar estruturas de documentos como sub documentos em um campo ou uma matriz dentro de outro documento. As relações de documentos incorporados são modelos de dados desnormalizados, o que permite manipular dados relacionais em uma única operação de banco de dados.

Neste trabalho optou-se pela utilização de documentos incorporados, o que pode ser observado figura 8, através de um exemplo.

Figura 8 - Documentos Incorporados.



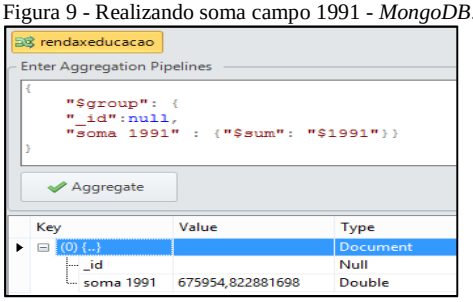
Fonte: O Autor.

Após possuir os dados incorporados a documentos, realizou-se consultas no *MongoDB* e no MySQL, com a execução das consultas em ambos os bancos averiguou-sequais apresentaram os melhores resultados. Para demostrar os resultados, verificou-se o tempo de processamento de ambos os bancos.

A primeira consulta busca saber o resultado da soma do campo '1991', realizou-se esse mesmo cálculo nos respectivos bancos, como mencionado anteriormente. A figura 9 contém o resultado da soma no *MongoDB* enquanto que a figura 10 contém o resultado da soma do MySQL.

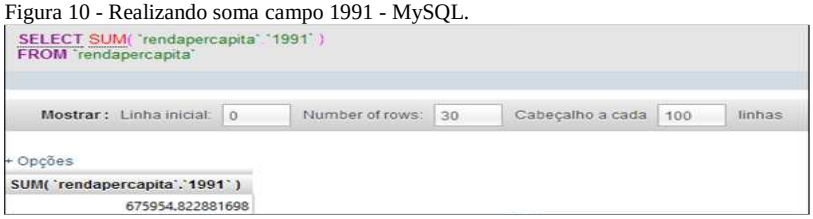
A figura 11 contém as informações referente ao tempo de processamento das consultas nos respectivos bancos de dados.

Figura 9 - Realizando soma campo 1991 - *MongoDB*.



Fonte: O Autor.

Figura 10 - Realizando soma campo 1991 - MySQL.



Fonte: O Autor.

Figura 11 - Tempo de execução da soma campo 1991.

Banco	Tempo (Segundos)
MySQL	0,0109
MongoDB	**0,029**

Fonte: O Autor.

Realizando a soma em ambos os bancos, constatou-se que o MySQL obteve um resultado melhor, mas os retornos ocorrem de forma tão rápida, que simplesmente olhando a execução das consultas não é possível verificar qual obteve o melhor resultado. Isso demostra que consultas extremamente simples o *MongoDB* e o MySQL se igualam.

A segunda consulta busca encontrar duas cidades especificas entre os dados das mulheres grávidas, para isso realizou-se o filtro por essas cidades. As figuras 12 e 13 contém a SQL para o banco *MySQL* e as figuras 14 e 15 contém as SQL do banco *MongoDB*.

A figura 16 contém as informações referente ao tempo de processamento das consultas nos respectivos bancos de dados.

Figura 12 - Realizando filtro por duas cidades - MySQL

```
SELECT `Municipio`,`Sigla`,`Regiao`
FROM `idheducacao`
WHERE `Municipio` in ('Xaxim', 'Chapecó')
```

Fonte: O Autor

Figura 13 - Filtrando por duas cidades - MySQL

Municipio	Sigla	Regiao
Chapecó	SC	Sul
Xaxim	SC	Sul

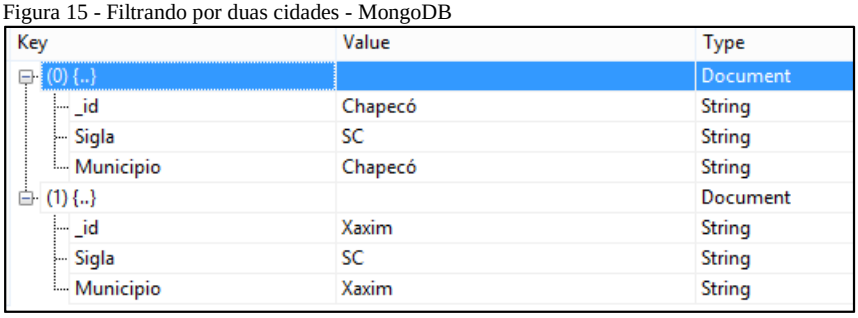
Fonte: O Autor

Figura 14 - Realizando filtro por duas cidades - MongoDB

```
{"$match" : {"Municipio" : {"$in" : ["Xaxim", "Chapecó"]}}},
{"$group" :
    {_id: {"Regiao" : "$Regiao", "Municipio" : "$Municipio", "Sigla" : "$Sigla"}}},
    {"$group" :
            {_id: "$_id.Municipio",
               "Sigla" : {"$last" : "$_id.Sigla"},
                "Municipio" : {"$last" : "$_id.Municipio"}}}
```

Fonte: O Autor

Figura 15 - Filtrando por duas cidades - MongoDB



Key	Value	Type
(0) {..}		Document
_id	Chapecó	String
Sigla	SC	String
Municipio	Chapecó	String
(1) {..}		Document
_id	Xaxim	String
Sigla	SC	String
Municipio	Xaxim	String

Fonte: O Autor

Figura 16 - Tempo de execução do SQL por filtro

Banco	Tempo (Segundos)
MySQL	0,0150
MongoDB	**0,014**

Fonte: O Autor

Realizando a consulta onde é filtrado os dados apenas das duas, verificou-se que o *MongoDB* obteve um resultado melhor, e assim como na primeira consulta os retornos ocorrem de forma rápida, que simplesmente olhando a execução das consultas não é possível verificar qual obteve os melhores resultados. Isso mostra que em consultas extremamente simples o *MongoDB* e o *MySQL* se igualam.

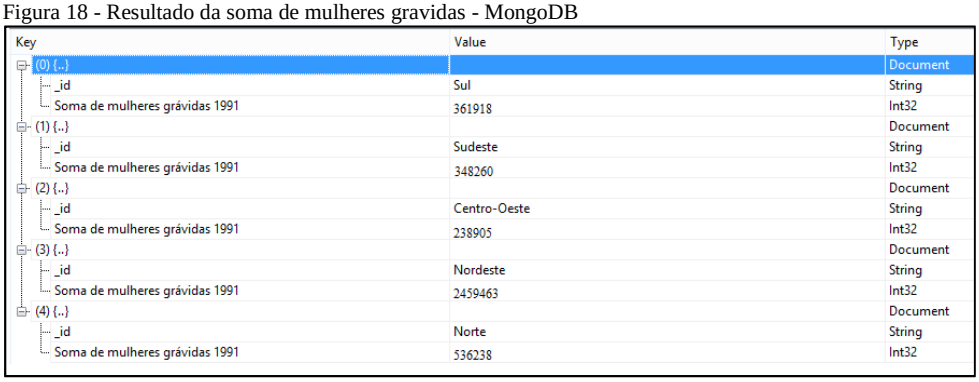
A terceira consulta buscar saber o número de mulheres grávidas no ano1991, para isso realizou-se a soma desse índice. As figuras 17 e 18 contém a SQL e o resultado do banco *MongoDB* e as figuras 19 e 20 contém a SQL e o resultado do banco *MySQL*. A figura 21 contém as informações referente ao tempo de processamento das consultas nos respectivos bancos de dados.

Figura 17 - Realizando soma de mulheres gravidas ano 2000 - MongoDB

```
{"$group":{
    "_id":"$Regiao",
    "Soma de mulheres grávidas 1991": {"$sum": "$1991"}}
}
```

Fonte: O Autor

Figura 18 - Resultado da soma de mulheres gravidas - MongoDB



Key	Value	Type
(0) {..}		Document
_id	Sul	String
Soma de mulheres grávidas 1991	361918	Int32
(1) {..}		Document
_id	Sudeste	String
Soma de mulheres grávidas 1991	348260	Int32
(2) {..}		Document
_id	Centro-Oeste	String
Soma de mulheres grávidas 1991	238905	Int32
(3) {..}		Document
_id	Nordeste	String
Soma de mulheres grávidas 1991	2459463	Int32
(4) {..}		Document
_id	Norte	String
Soma de mulheres grávidas 1991	536238	Int32

Fonte: O Autor

Figura 19 - Realizando soma de mulheres gravidas ano 2000 - MySQL

```
select `Regiao`, SUM(`mulhergravida`.`1991`)
from `mulhergravida`
group by `Regiao`
```

Fonte: O Autor

Figura 20 - Resultado da soma de mulheres gravidas - MySQL

Centro-Oeste	238905
Nordeste	2459463
Norte	536238
Sudeste	348260
Sul	361918

Fonte: O Autor

Figura 21 - Tempo de execução da soma de mulheres grávidas

Banco	Tempo (Segundos)
MySQL	0,0565
MongoDB	**0,06**

Fonte: O Autor

Realizando a consulta onde busca-se encontrar o número de mulheres grávidas no ano de 1991, verificou-se que o MySQL obteve um resultado superior ao *MongoDB*. Mas em ambos os bancos os retornos foram muito rápidos que somente olhando nos logs da execução conseguiu-se verificar a diferença.

A quarta consulta realizada, busca encontrar a soma e a média dos campos '1991' e '2000' para as regiões sul, norte e nordeste. As figuras 22 e 23 contém a SQL e a soma e média dos campos '1991' e '2000' para o banco MySQL enquanto que as figuras 24 e 25 contém a SQL e o resultado da soma e da média dos campos '1991' e '2000' para o banco*MongoDB*.

A figura 26 contém as informações referente ao tempo de processamento das consultas nos respectivos bancos de dados.

Figura 22 - Realizando soma e média por três regiões - MySQL.

```
SELECT `Regiao`, sum(`1991`) as soma_indEducacao1991, avg(`1991`) as
media_indeducacao1991, sum(`2000`) as soma_indEducacao2000, avg(`2000`) as
media_indEducacao2000,
sum(`19911`) as soma_renda1991, avg(`19911`) as media_renda1991, sum(`20001`)
as soma_renda2000, avg(`20001`) as media_renda2000
FROM `idheducacao`
join `rendapercapita` on (`idheducacao`.`Codigo` = `rendapercapita`.`Codigo`)
WHERE `Regiao` in ('Sul', 'Norte','Nordeste')
group by `Regiao`, `Regiao1`
```

Fonte: O Autor.

Figura 23 - Soma e média por três regiões - MySQL.

Regiao	soma_indEducacao1991	media_indeducacao1991	soma_indEducacao2000	media_indEducacao2000	soma_renda1991	media_renda1991	soma_renda2000	media_renda2000
Nordeste	896.167	0.501492445438284	1234.04736230469	0.690669312985276	110429.137037277	61.7958237477768	152098.358844721	85.1614551202244
Norte	44262.1550216675	0.6008285077951	338.854	0.754685968819599	44262.1550216675	98.5794098478118	54088.901966095	120.465260503552
Sul	868.656	0.756010443864423	988.484000000001	0.860299390774587	180514.971820831	157.106154761385	268542.117172241	233.718117643378

Fonte: O Autor.

Figura 24 - Realizando soma e média por três regiões - *MongoDB*.

rendaxeducacao

Enter Aggregation Pipelines

```
{ "$match" : { "Regiao" : { "$in" : [ "Sul" , "Nordeste" , "Norte"]}}} ,
{"$group":
    {
        "_id":"$Regiao",
            "sum_1991" : { "$sum" : "$1991"},
            "sum_2000" : { "$sum" : "$2000"},
            "avg_1991" : { "$avg" : "$1991" },
            "avg_2000" : { "$avg" : "$2000" },
                "IDHEducao_sum_1991" : {"$sum" : "$IDHEducacao.1991"},
                "IDHEducao_sum_2000" : {"$sum" : "$IDHEducacao.2000"},
                "IDHEducao_avg_1991" : {"$avg" : "$IDHEducacao.1991"},
                "IDHEducao_avg_2000" : {"$avg" : "$IDHEducacao.2000"},
        "count" : { "$sum" : 1}
    }
}
```

Fonte: O Autor.

Figura 25 - Soma e média por três regiões - *MongoDB*.

Key	Value	Type
(0) {..}		Document
_id	Sul	String
sum_1991	180514,971820831	Double
sum_2000	268542,117172241	Double
avg_1991	157,106154761385	Double
avg_2000	233,718117643378	Double
IDHEducao_sum_1991	868,656	Double
IDHEducao_sum_2000	988,484000000001	Double
IDHEducao_avg_1991	0,75601044386423	Double
IDHEducao_avg_2000	0,860299390774587	Double
count	1149	Int32
(1) {..}		Document
_id	Norte	String
sum_1991	44262,1550216675	Double
sum_2000	54088,901966095	Double
avg_1991	98,5794098478118	Double
avg_2000	120,465260503552	Double
IDHEducao_sum_1991	269,772	Double
IDHEducao_sum_2000	338,854	Double
IDHEducao_avg_1991	0,6008285077951	Double
IDHEducao_avg_2000	0,754685968819599	Double
count	449	Int32
(2) {..}		Document
_id	Nordeste	String
sum_1991	110429,137037277	Double
sum_2000	152098,358844721	Double
avg_1991	61,7958237477768	Double

Fonte: O Autor

Figura 26 - Tempo de execução da soma e média por três regiões.

Banco	Tempo (Segundos)
MySQL	0,0587
MongoDB	**0,057**

Fonte: O Autor.

Realizando a execução de algumas consultas, percebeu-se que utilizado o*join* no *MySQL* as consultas tornam-se mais lentas que no *MongoDB*, mas a diferença ainda é pequena que apenas visualizando a execução da consulta não é possível notar a diferença.

Para conseguir visualizar essa diferença é necessário observar os logs de execução das consultas. E foi exatamente isso que ocorreu na quarta consulta.

Naquinta consulta realizada, buscou-se encontrar o maior valor referente ao índice de educação do ano 2000. As figuras 27 e 28 contém a SQL e o resultado da busca para o banco *MySQL* enquanto que as figuras 29 e 30 contém a SQLe o resultado da busca para o banco *MongoDB*.

A figura 31 contém as informações referente ao tempo de processamento das consultas nos respectivos bancos de dados.

Figura 27 - Realizando buscas pelo valor max do IDHEducacao do ano 2000 - MySQL.

```
select T.*
  from `idheducacao` T
 where T.`2000` = (select max(TT.`2000`)
                     from `idheducacao` TT
                    where T.Regiao = TT.Regiao
                 group by TT.Regiao)
```

Fonte: O Autor.

Figura 28 - Resultado do valor max do IDHEducacao do ano 2000 - MySQL.

	Sigla	Regiao	Codigo	Municipio	1991	2000
Editar Copiar Remover	TO	Norte	1721000	Palmas	0.755	0.934
Editar Copiar Remover	BA	Nordeste	2927408	Salvador	0.856	0.924
Editar Copiar Remover	SP	Sudeste	3548807	São Caetano do Sul	0.913	0.975
Editar Copiar Remover	SC	Sul	4215703	Santo Amaro da Imperatriz	0.791	0.978
Editar Copiar Remover	GO	Centro-Oeste	5201207	Anhanguera	0.82	0.952

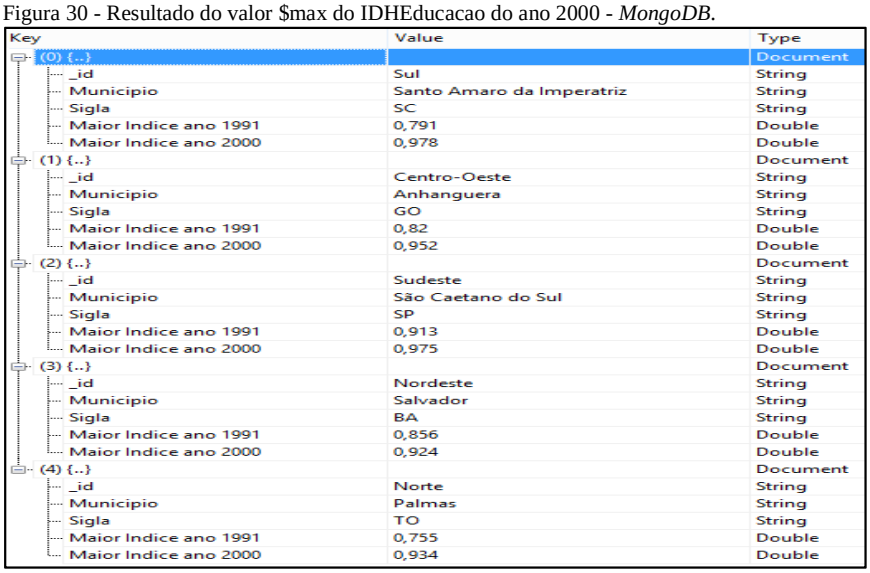
Fonte: O Autor.

Figura 29 - Realizando buscas pelo valor $max do IDHEducacao do ano 2000 - *MongoDB*.

```
{"$group" :
    {_id: {"Regiao" : "$Regiao", "Municipio" : "$Municipio", "Sigla":"$Sigla"},
     "2000" : {"$max" : "$IDHEducacao.2000"},
     "1991":{"$max" : "$IDHEducacao.1991"}}},
    {"$sort":{"2000":1}},
    {"$group":
          {_id:"$_id.Regiao",
        "Municipio":{"$last" : "$_id.Municipio"},
        "Sigla" : {"$last" : "$_id.Sigla"},
        "Maior Indice ano 1991": {"$last" : "$1991"},
        "Maior Indice ano 2000": {"$last" : "$2000"}}}
```

Fonte: O Autor.

Figura 30 - Resultado do valor $max do IDHEducacao do ano 2000 - *MongoDB*.



Key	Value	Type
(0) {..}		Document
_id	Sul	String
Municipio	Santo Amaro da Imperatriz	String
Sigla	SC	String
Maior Indice ano 1991	0,791	Double
Maior Indice ano 2000	0,978	Double
(1) {..}		Document
_id	Centro-Oeste	String
Municipio	Anhanguera	String
Sigla	GO	String
Maior Indice ano 1991	0,82	Double
Maior Indice ano 2000	0,952	Double
(2) {..}		Document
_id	Sudeste	String
Municipio	São Caetano do Sul	String
Sigla	SP	String
Maior Indice ano 1991	0,913	Double
Maior Indice ano 2000	0,975	Double
(3) {..}		Document
_id	Nordeste	String
Municipio	Salvador	String
Sigla	BA	String
Maior Indice ano 1991	0,856	Double
Maior Indice ano 2000	0,924	Double
(4) {..}		Document
_id	Norte	String
Municipio	Palmas	String
Sigla	TO	String
Maior Indice ano 1991	0,755	Double
Maior Indice ano 2000	0,934	Double

Fonte: O Autor.

Figura 31 - Tempo de execução do maior IDHEducacao do ano 2000.

Banco	Tempo (Segundos)
MySQL	72,5371
MongoDB	**0,134**

Fonte: O Autor.

A quinta consulta, demora tempos distintos para retornar o mesmo conjunto de informações. Como visualizado o *MongoDB* retornou as mesmas informações que o MySQL mas em um tempo muito menor. Conclui-se que quando utilizada subconsultas o *MySQL* demora mais para retornar as informações solicitadas, acredita-se que o *MongoDB* seja mais rápido pois, todos os dados que ele necessitada para retornar as informações solicitadas encontram-se na mesma *collection*, ou seja, não há necessidade de ir em outras *collections* 'pegar' essas informações para então retorna-las, como ocorre no MySQL.

5 CONSIDERAÇÕES FINAIS

Este trabalho demostrou o quão vantajoso é a utilização do Big Data, juntamente com um banco de dados não relacional. A escolha pelo *MongoDB* foi baseada em sua robustez e aceitação do mercado. Analisando-se os resultados obtidos no *MongoDB* e no *MySQL* é visível a diferença no tempo de execução em uma consulta mais complexa. Essa diferença ocorre pois no *MongoDB*todas as informações encontram-seao alcance, enquanto que no MySQL é necessário busca-las em diferentes tabelas.

Ainda há a necessidade de estudar-se mais sobre como o Big Data se comportaria em uma grande corporação ou como o mesmo seria implantando em um sistema que já possui raízes em um mundo relacional. Ainda necessita-se estudar os reais impactos dessas mudanças na organização como um todo, indo da parte financeira até a parte de adaptação da equipe que estaria trabalhando com esse novo conceito.

Mas pelos resultados aqui apresentados, certamente a empresa que decidir investir nessa mudança, terá muitos ganhos, assim como obtiveram outras grandes organizações.

REFERÊNCIAS

ALBANI, Rafael. **Banco de Dados Aula1 - Introdução a Banco de Dados.** 2011. Disponível em: <http://pt.slideshare.net/RafaelAlbani/aula1-apresentao-de-banco-de-dados>. Acesso em: 02 mar. 2014.

ALECRIM, Emerson. **O que é Big Data?** Disponível em: <http://www.infowester.com/big-data.php>. Acesso em: 29 ago.2013.

BATISTA, Fábio et al. ***MongoDB*****:** Banco de dados orientado a documento. 2012. Disponível em: <http://*MongoDB*.machinario.com/>. Acesso em: 05 out. 2013.

COSTA, Rafael. **Entendendo os Bancos de Dados Relacionais.** 2010. Disponível em: <http://infodrama.blogspot.com.br/2010/01/entendendo-os-bancos-de-dados_15.html>. Acesso em: 03 abr. 2014.

DIAS NETO, Arilo Cláudio. **Bancos de Dados Relacionais.** 2011. Disponível em: <http://www.devmedia.com.br/websys.5/webreader.asp?cat=2&artigo=3465&revista=sqlmagazine_86#a-3465>. Acesso em: 17 mar. 2014.

GONDIM, Gustavo Simões Pinto. **Laboratório *NoSQL* (Parte 1): Conceitos de *NoSQL* .** São Paulo, 2013. Disponível em: <http://www.buildando.com.br/p/equipe.html>. Acesso em: 02 out. 2013.

KOKAY, Marília Cavalcante. Banco de dados *NoSQL* : Um novo paradigma. **Banco de Dados *NoSQL*** : Conheça esse novo Paradigma, [S.l], n. 102, p., 01 out. 2012. Disponível em: <http://www.devmedia.com.br/websys.5/webreader.asp?cat=2&artigo=4773&revista=sqlmagazine_102#a-4773>. Acesso em: 01 out. 2013.

MIRANDA, Bruno. **Método Quantitativo versus Método Qualitativo.**2008. Disponível em: <http://adrodomus.blogspot.com.br/2008/06/mtodo-quantitativo-versus-mtodo.html>. Acesso em: 30 out. 2013.

NASCIMENTO, Jean. ***NoSQL* – você realmente sabe do que estamos falando?** São Paulo, 2010. Disponível em: <http://imasters.com.br/artigo/17043/banco-de-dados/*NoSQL* -voce-realmente-sabe-do-que-estamos-falando/>. Acesso em: 03 out. 2013.

NOSSAL, Rodrigo; MOURA, Rudá. ***MongoDB*****.** 2011. Disponível em: <http://www.slideshare.net/terra_neo/*MongoDB*-apresentao>. Acesso em: 05 out. 2013.

OLHAR DIGITAL. **Big Data ajuda cidade a resolver problema de saneamento.** 2013.Disponível em: <http://olhardigital.uol.com.br/pro/noticia/big-data-ajuda-cidade-a-resolver-problema-de-saneamento/28853>. Acesso em: 23 set. 2013a.

PETRY, André. O Berço do Big Data. **Veja**, Nova York, n. , p.71-76, 15 maio 2013a.

QUEIROZ, Rodrigo Sampaio de. **O que é Big Data?** Disponível em: <http://www.webinterativa.com.br/blog/novidades/o-que-e-big-data/>. Acesso em: 30 ago. 2013.

SOUZA, Marcio Ballem de. ***MongoDB*: Realizando Consultas - Revista Java Magazine 115.** 2013. Disponível em: <http://www.devmedia.com.br/*MongoDB*-realizando-consultas-revista-java-magazine-115/27791>. Acesso em: 02 mar. 2014.

TAURION, Cezar. **Big Data.** Rio de Janeiro: Brasport, 2013. Cap. 1-3, p. 27-42.