UNIVERSIDADE DO ESTADO DO RIO GRANDE DO NORTE – UERN

FACULDADE DE CIÊNCIAS EXATAS E NATURAIS – FANAT

DEPARTAMENTO DE INFORMÁTICA – DI

Valesca Juliane Souza da Silva

ANÁLISE COMPARATIVA ENTRE BANCOS DE DADOS NOSQL POR MEIO DE UM ESTUDO DE CASO DE RECOMENDAÇÃO DE INTERESSES

MOSSORÓ - RN

2017

Valesca Juliane Souza da Silva

ANÁLISE COMPARATIVA ENTRE BANCOS DE DADOS NOSQL POR MEIO DE UM ESTUDO DE CASO DE RECOMENDAÇÃO DE INTERESSES

Monografia apresentada à Universidade do Estado do Rio Grande do Norte como um dos pré-requisitos para obtenção do grau de bacharel em Ciência da Computação, sob orientação da Prof[a]. M. Sc. Ceres Germanna Braga Morais e coorientação do Prof° Alysson Mendes de Oliveira.

MOSSORÓ - RN

2017

Catalogação da Publicação na Fonte.
Universidade do Estado do Rio Grande do Norte.

S586a Silva, Valesca Juliane Souza da
ANÁLISE COMPARATIVA ENTRE BANCOS DE DADOS NOSQL POR MEIO DE UM ESTUDO DE CASO DE RECOMENDAÇÃO DE INTERESSES. / Valesca Juliane Souza da Silva. - Mossoró, 2017.
54p.

Orientador(a): Profa. Mª. Ceres Germanna Braga Morais.
Coorientador(a): Prof. Esp. Alysson Mendes de Oliveira.
Monografia (Graduação em Ciência da Computação). Universidade do Estado do Rio Grande do Norte.

1. Análise Comparativa. 2. NoSQL. 3. ArangoDB. 4. Neo4j. I. Morais, Ceres Germanna Braga. II. Universidade do Estado do Rio Grande do Norte. III. Título.

Valesca Juliane Souza Da Silva

ANÁLISE COMPARATIVA EM BANCOS DE DADOS NOSQL POR MEIO DE UM ESTUDO DE CASO DE RECOMENDAÇÃO DE INTERESSES

Monografia apresentada como pré-requisito para a obtenção do título de Bacharel em Ciência da Computação da Universidade do Estado do Rio Grande do Norte – UERN, submetida à aprovação da banca examinadora composta pelos seguintes membros:

Aprovada em: 26/10/2017

Banca Examinadora

Profa. Ma. CERES GERMANNA BRAGA MORAIS
Universidade do Estado do Rio Grande do Norte - UERN

Prof. ALYSSON MENDES DE OLIVEIRA
Universidade do Estado do Rio Grande do Norte - UERN

Prof. Me. ANTÔNIO OLIVEIRA FILHO
Universidade do Estado do Rio Grande do Norte - UERN

Profa. Dra. CARLA KATARINA MONTEIRO MARQUES
Universidade do Estado do Rio Grande do Norte - UERN

A minha mãe.

AGRADECIMENTOS

Agradeço em primeiro lugar à Deus, pois sem ele nada seria possível. A ele toda honra e glória pelo simples fato de existirmos.

À mainha, Ana Sílvia, por todo cuidado, esforço e amor, por ter me apoiado, acreditado em mim e nunca ter permitido que as dificuldades me fizessem desistir. À toda minha família maravilhosa, em especial às minhas irmãs Vanessa, Beatriz e Bianca, e minha vó Marlene.

Ao meu namorado, e também colega de curso, Hitalo Vinicius, por todo amor, paciência, incentivo e força. Aos ingressantes da turma 2013.1 e a minha grande amiga Bruna Maria por sempre estar ao meu lado e ter me incentivado cursar Ciência da Computação.

À todos os professores e técnicos do Departamento de Informática que de algum modo contribuíram para minha formação durante esses quase cinco anos. Em especial ao professor Dario Aloise pela participação no projeto de Iniciação Científica no Laboratório de Otimização e Inteligência Artificial (LOIA) nos meus primeiros anos de graduação; aos professores Marcelino Pereira e Rommel Wladimir pela oportunidade de participar do Programa de Educação Tutorial em Ciência da Computação (PETCC) e vivenciar experiências maravilhosas, e aos professores Ceres Germanna e Alysson Oliveira por me orientarem durante este trabalho de conclusão.

-São as perguntas que não sabemos responder que mais nos ensinam. Elas nos ensinam a pensar. Se você dá uma resposta a um homem, tudo o que ele ganha é um fato qualquer. Mas, se você lhe der uma pergunta, ele procurará suas próprias respostas.

(…)

-Assim, quando ele encontrar as respostas – continuei -, elas lhe serão preciosas. Quanto mais difícil a pergunta, com mais empenho procuramos a resposta. Quanto mais a procuramos, mais aprendemos.

Kavothe – O Temor do Sábio

RESUMO

Com o crescente volume de dados associados ao termo *Big Data* e imensos conjuntos de dados variados que auxiliam as tomadas de decisões críticas, surge uma nova alternativa ao modelo de dados relacional: a abordagem NoSQL. Nesse contexto, este trabalho apresenta uma análise comparativa entre Bancos de Dados NoSQL através de um estudo de caso. O trabalho apresenta um referencial teórico a respeito de bancos de dados NoSQL, juntamente com os trabalhos mais relevantes para o escopo deste. Para validação, foi proposta a modelagem de um estudo de caso baseado em sistemas de recomendação de conteúdo, utilizando para isto dois SGBDs NoSQL: Neo4j e ArangoDB, um banco de dados orientado a grafos e o outro híbrido (que também possui o modelo de dados de grafos), respectivamente. Posteriormente, foi realizada uma análise comparativa entre esses bancos de dados, com base em quatro requisitos: Interface Web, Características, Consultas e Desempenho.

Palavras-chave: Análise comparativa; NoSQL; ArangoDB; Neo4j.

ABSTRACT

With the increasing volume of data associated with the term Big Data and huge assorted data sets that help as opinion takers, a new alternative to the relational data model emerges: a NoSQL approach. In this context, this work presents a comparative analysis between NoSQL databases through a case study. The paper presents a theoretical reference regarding NoSQL databases, together with the most relevant works for the scope of this. For validation, we proposed the modeling of a case study based on content recommendation systems, using two NoSQL DBMSs: Neo4j and ArangoDB, one database oriented to graphs and the other hybrid (which also has the graph data model), respectively. Posteriorly, a comparative analysis was carried out between these databases, based on four requirements: Web Interface, Characteristics, Queries and Performance.

Keywords: Comparative analysis, NoSQL; ArangoDB; Neo4j.

LISTA DE SIGLAS

ACID	Atomicidade, Consistência, Isolamento, Durabilidade
AQL	*ArangoDB Query Language*
JSON	*JavaScript Option Notation*
MR	Modelo Relacional
NoSQL	*Not Only SQL*
SGBDs	Sistema de Gerenciamento de Banco de Dados
SGBDG	Sistema Gerenciador de Bancos de Dados em Grafos (SGBDG)
SPARQL	*SPARQL Protocol and RDF Query Language*
SQL	*Structured Query Language*
SSL	*Secure Sockets Layer*
TSL	*Transport Layer Security*
UERN	Universidade do estado do Rio Grande do Norte
XML	*eXtensible Markup Language*

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

A sociedade é inserida em um cenário onde os bancos de dados estão constantemente presentes. Seja em sites de compras, sistemas de cadastro de produtos, sistemas bancários, milhões de compartilhamentos em redes sociais ou registros contendo informações de um aluno em uma universidade.

Esses volumosos e complexos dados que são armazenados, possuem fontes e formatos diferentes e têm rápido crescimento. O processamento desta enorme quantidade de dados em velocidades aceitáveis exige soluções específicas, sendo que uma delas é o *Big Data*. A proposta de *Big Data* não se refere apenas à volume de dados, oferece abordagem ampla no tratamento desses dados que se originam nos variados meios. Nesses meios, os dados se dividem em duas classes: os dados estruturados e os dados não estruturados, sendo estes conhecidos como *Not Only SQL* (NoSQL) (Bernardo e Borlone, 2015).

Atualmente, existe uma grande adoção e difusão de tecnologias NoSQL nos mais diversos domínios de aplicação no contexto de *Big Data*. Nesses domínios os Sistemas Gerenciadores de Banco de Dados (SBGD) tradicionais ainda são fortemente dominantes, como por exemplo, instituições financeiras. Isto pode ser explicado pelo fato que existe uma demanda muito grande para soluções que tenham alta flexibilidade, escalabilidade, performance, e suporte a diferentes modelos de dados complexos (Vieira et al., 2012).

Diante desse contexto, o presente trabalho apresenta um referencial teórico a respeito de bancos de dados NoSQL, e traz sua aplicação através da modelagem de um estudo de caso baseado em sistemas de recomendação de conteúdo, utilizando para isto dois SGBDs: Neo4j e ArangoDB. A proposta deste trabalho é realizar uma análise comparativa entre os bancos de dados Neo4j e ArangoDB, um banco de dados orientado a grafos e o outro híbrido (que também possui o modelo de dados de grafos), respectivamente. A análise comparativa será feita com base em quatro requisitos: Interface Web, Características, Consultas e Desempenho.

Dessa forma, o objetivo geral do trabalho é apresentar um comparativo entre bancos de dados NoSQL orientado a grafos, utilizando os bancos Neo4j e ArangoDB, por meio da implementação de um estudo de caso.

De modo a alcançar o objetivo geral, existem alguns objetivos específicos, como:

1. Descrever os principais conceitos relacionados a SGBDs NoSQL;
2. Apresentar as categorias existentes de bancos de dados NoSQL, com ênfase no modelo orientado a grafos;
3. Descrever os SGBDs NoSQL Neo4j e ArangoDB e suas linguagens de consultas;
4. Realizar análise através da elaboração de estudo de caso e verificação de quesitos que serão utilizados para a comparação de acordo com o estudo de caso proposto;
5. Avaliar o estudo proposto e apresentar os resultados obtidos.

Para melhor apresentação, este trabalho está organizado da seguinte maneira: o Capítulo 2 aborda os conceitos sobre Banco de Dados e *Big Data*, bem como o termo NoSQL, suas características e classificação. Ainda aborda os trabalhos relacionados mais relevantes em consideração ao que este trabalho propõe.

O Capítulo 3 traz um breve conceito sobre sistemas de recomendação de conteúdo, apresenta o estudo de caso proposto os bancos de dados escolhidos (Neo4j e ArangoDB) e sua linguagens de consultas: *Cypher* e AQL.

O Capítulo 4 apresenta o comparativo entre os bancos de dados levando em consideração quatro quesitos: Interface Web, Características, Consultas e Desempenho.

O Capítulo 5 aborda as considerações finais e trabalhos futuros.

2 REFERENCIAL TEÓRICO

2.1 Banco de Dados e *Big Data*

Bancos de Dados estão presentes no nosso cotidiano, seja quando atualizamos nossas informações nas redes sociais, fazendo compras *online*, quando vamos ao banco depositar ou sacar dinheiro.

Segundo Silberschatz et al. (2006), Banco de Dados é uma coleção de dados que contém informações relevantes para uma empresa. Date (2004), conceitua Banco de Dados como sendo uma coleção de dados persistentes, usada pelos sistemas de aplicação de uma determinada empresa, onde o termo empresa é simplesmente um termo genérico para qualquer organização. Para Elmasri e Navathe (2011) é correto afirmar que os Bancos de Dados desempenham um papel crítico em quase todas as áreas em que os computadores são utilizados.

Bancos de Dados relacionais tem sido a principal opção das empresas quando se fala em armazenamento e processamento de dados estruturados, os quais são bem definidos e apresentam o mesmo formato (esquema) e descrição (atributo). O modelo de dados relacional (MR) consiste em uma ou mais tabelas que são utilizadas para representar os dados e as relações. Utilizam a linguagem de consulta *Structured Query Language* (SQL) e são característicos por apresentarem segurança nas suas transações, devido às propriedades estabelecidas pelo paradigma ACID (Atomicidade, Consistência, Isolamento e Durabilidade):

- Atomicidade: garante que as transações irão executar totalmente ou não irão executar.
- Consistência: garante que as transações irão respeitar a integridade dos dados.
- Isolamento: se duas transações estão sendo executadas concorrentemente seus efeitos devem ser isolados uma da outra.
- Durabilidade: os efeitos de uma transação devem persistir no banco de dados mesmo em presença de falhas.

O processamento e manipulação de grandes volumes de dados no contexto *Big Data* tem sido um dos maiores desafios atualmente na área da Computação (Vieira et al., 2012). Definem-se *Big Data*, em termos gerais, como imensos conjuntos de dados variados que, por aproveitamento melhor do processamento dessa massa de dados em busca da geração da informação, auxiliam as tomadas de decisões críticas (Bernardo; Borlone, 2015). Segundo Zanela et al. (2017), *Big Data* é capaz de cruzar inúmeras quantidades de informações desestruturadas para chegar a uma, ou várias, tomadas de decisões, talvez não a melhor, porém a com maior porcentagem de acerto.

Segundo Bernado e Borlone (2015), *Big Data* se caracteriza por cinco aspectos:

- Volume: aumento exponencial na quantidade de dados.
- Velocidade: processamento e obtenção de grandes volumes de dados em tempo hábil.
- Variedade: capacidade de tratar e relacionar vários tipos de dados existentes.
- Veracidade: qualidade ou credibilidade dos dados.
- Valor: é a união de todos os aspectos anteriores, o *Big Data* precisa gerar valor para quem o adota.

Neste ambiente de crescente demanda por volume de dados e desempenho, surge o modelo de SGBD chamado NoSQL (Souza et al., 2014), apresentado na Seção a seguir.

2.2 Banco de Dados NoSQL

O termo NoSQL, de acordo com Sadalage e Fowler (2012), surgiu no final dos anos 90 como o nome de um banco de dados relacional de código aberto que foi dirigido por Carlo Strozzi, denominado Strozzi NoSQL, pois o Banco de Dados não usava a linguagem SQL como linguagem de consulta. O termo reapareceu e foi discutido em 11 de junho de 2009 em um encontro organizado pelo desenvolvedor de software Johan Oskarsson em São Francisco/Califórnia. Foram reunidos neste

evento vários projetos que utilizavam essa nova forma de armazenamento de dados, que foram inspirados em sua maioria no *BigTable* da Google e no *Dynamo* da Amazon que foram desenvolvidos antes de 2009.

Essa nova categoria de Banco de Dados foi proposta com o objetivo de atender aos requisitos de gerenciamento de grandes volumes de dados, semiestruturados ou não estruturados, que necessitam de alta disponibilidade e escalabilidade (Lócio et al., 2011).

NoSQL é definido como Banco de Dados que possui as seguintes características: (NoSQL, 2017)

- Não relacional: não utiliza a linguagem de consulta SQL.
- Distribuído: em vários servidores.
- De código aberto: permite que o usuário modifique o código fonte.
- Escalável horizontalmente: aumento no número de máquinas disponíveis para o armazenamento e processamento dos dados.
- APIs simples: foco na aplicação e não nos dados. O foco está em recuperar os dados de forma simples e rápida.
- Suporte nativo a replicação: esta é outra forma de prover escalabilidade, pois, no momento em que permitimos a replicação de forma nativa o tempo gasto para recuperar informações é reduzido.
- Ausência de esquema ou esquema flexível: é justamente essa ausência parcial ou total de esquemas que definem a estrutura dos dados que facilita a alta escalabilidade e alta disponibilidade.

2.3 Tipos de Bancos de Dados NoSQL

Os Bancos de Dados NoSQL são classificados em quatro grandes categorias: Chave-valor (*key-value*), orientado a documentos (*documents*), orientado a colunas (*column*) e orientado a grafos (*graph-based*), as quais são apresentadas a seguir.

2.3.1 Chave –Valor

Essa é a categoria mais simples entre todas. Os dados consistem em duas partes, uma *string* que representa a chave e os dados reais referenciados como valor, criando assim um par chave-valor (Nayak et al., 2013). Esse modelo se assemelha a uma grande tabela *hash* onde as chaves são usadas como índices para os valores, tornando assim mais rápida a recuperação dos dados. Devido a essa estrutura de dados bem simples, o modelo chave-valor é completamente isento de esquemas. Novos valores de qualquer tipo podem ser adicionados em tempo de execução sem conflitos com outros dados armazenados e sem influenciar a disponibilidade do sistema (Hecht e Jablonski, 2011).

2.3.2 Orientado a Documentos

Os dados são organizados em forma de coleções de documentos. Este modelo de dados se assemelha com a categoria chave-valor, pois cada documento que é inserido na base de dados possui um identificador único que irá representá-lo. Lócio et al. (2011), ressaltam que no modelo chave-valor, apenas uma única tabela *hash* é criada para todo o banco. No modelo orientado a documentos temos um conjunto de documentos e em cada documento temos um conjunto de campos (chaves) e o valor deste campo. Esses documentos estão tipicamente sob o formato de *Extensible Markup Language* (XML) e *JavaScript Option Notation* (JSON).

2.3.3 Orientado a Colunas

Similar ao modelo relacional por apresentar linhas e colunas, e uma evolução do modelo chave-valor, os bancos orientados a colunas armazenam os dados de forma colunar. Sua orientação não é através das linhas (tuplas), e sim através dos atributos (colunas) (Cardoso, 2012). Trata-se de um mapa de dados amplo, persistente, distribuído e multidimensional.

2.3.4 Orientado a Grafos

Um grafo é definido como $G = (V, E)$, onde V é um conjunto de vértices e E é um conjunto de arestas. Os vértices representam entidades do mundo real e as arestas são as conexões que esses vértices possuem entre si. O modelo de dados orientado a grafos possui três elementos principais: os nós (vértices), as arestas

(conexões) e as propriedades (atributos). Em geral, não possuem um esquema rígido. Assim, dois objetos representados por nós de um mesmo grafo podem ter atributos bem distintos. As arestas são usadas para materializar o relacionamento entre nós. As propriedades podem ser usadas tanto para descrever atributos de nós quanto de arestas (Queiroz et al., 2013).

O mapeamento lógico-físico classifica os Sistemas Gerenciadores de Banco de Dados de Grafos (SGBDGs) em não-nativos ou nativos. Os não-nativos modelam logicamente seus dados como grafos, porém armazenam estes por meio de outros modelos. Já os nativos, em geral, usam listas de adjacência. Em uma lista de adjacência, cada vértice mantém referências diretas para seus vértices adjacentes formando uma espécie de micro-índice para vértices próximos (Penteado et al., 2014).

Angles (2012) cita quatro tipos de modelos de grafos usados para modelar os dados:

- Grafos simples: é a definição básica em que um grafo é definido por um conjunto de vértices conectados por arestas par a par.
- Hipergrafos: uma aresta (hiper-aresta) pode se relacionar com número arbitrário de vértices.
- Grafos aninhados: um vértice (hiper-vértice) também pode ser um grafo.
- Grafo de propriedades: vértices e arestas contém atributos para descrever suas propriedades.

O modelo mais utilizado pelos SGBDGs atuais é o de grafos de propriedades. Rodrigez e Neubauer (2010) definem um grafo de propriedades (*Graph Property Model*) como um grafo multi-relacional que contém atributos e arestas direcionadas. O grafo de propriedades tem as seguintes características (Robinson et al., 2015):

- Contém nós e relacionamentos;
- Os Nós contêm propriedades (pares chave-valor);
- Os Nós podem ser rotulados com um ou mais rótulos;

- Os Relacionamentos são nomeados e direcionados, e sempre têm um início e um fim;
- Os Relacionamentos também contêm propriedades.

Dentre as categorias abordadas de Bancos de Dados NoSQL, os modelos escolhidos foram os modelos de dados orientado a grafos, pois este modelo é otimizado para o processamento eficiente de conjuntos de dados interligados, e orientado a documentos. Os bancos orientados a grafos são os únicos dentre as categorias de bancos NoSQL que se preocupam com as relações e representação visual dos dados (Moniruzzaman, Hossain, 2013 apud Barros, 2017).

2.4 Trabalhos Relacionados

Nesta seção serão apresentados os trabalhos mais relevantes para o escopo deste trabalho.

2.4.1 Análise de Consultas em Bancos de Dados NoSQL em Ambiente de Computação nas Nuvens

Moreira, (2013), tem como objetivo principal realizar uma análise comparativa entre dois SGBDs NoSQL orientado a documentos (MongoDB e CouchDB) apresentando suas características e os desempenhos das consultas em um ambiente de computação nas nuvens. Os SGBDs foram configurados no ambiente de computação em nuvem, para o CouchDB foi utilizado o Cloudant da IBM e para o MongoBD foi utilizado o MongoLab, um *startup* do MongoDB, que oferece serviços de bancos de dados em nuvem hospedando bancos de dados MongoDB. Os bancos de dados em questão foram submetidos aos mesmos testes.

Para a realização das consultas, foram criados sete Documentos com quantidades de registros distintos, conforme a Tabela 1:

Tabela 1: Documentos e Registros.

Nome Documento	Qdt de Registros
Qtde005milReg	5mil
Qtde010milReg	10mil
Qtde050milReg	50mil

Qtde100milReg	100mil
Qtde500milReg	500mil
Qtde1milhaoReg	1milhao
Qtde2milhoesReg	2milhoes

Fonte: Moreira (2013).

Os testes foram feitos com Documentos com 5.000, 10.000, 50.000, 100.000, 500.000, 1.000.000 e 2.000.000 de registros tendo as colunas chave, numérico, texto, data, booleano e revisão para os bancos CouchDB e MongoDB. As consultas foram feitas a partir de cada coluna (numérico, texto, booleano, data). O resultado final foi gerado a partir da média dos resultados parciais.

2.4.2 Análise de Bancos de Dados NoSQL e Desenvolvimento de uma Aplicação

Friess (2013) teve como objetivo realizar uma pesquisa selecionando alguns Bancos de Dados NoSQL para estudo e comparação e selecionar um dos bancos estudados para o desenvolvimento de uma aplicação para manipulação dos dados. Foram escolhidos sete Bancos de Dados: Redis, Riak, Cassandra, HBase, CouchDB, MongoBD e Neo4j. A comparação foi feita levando em consideração para quais sistemas operacionais tem-se pacotes de instalação disponível, o tipo de licença, *design* da arquitetura, modelo de dados utilizado, como são armazenados os dados, como é possível escalar ou replicar e como pode ser executado, como apresentado na Tabela 2.

Tabela 2: Tabela comparativa NoSQL *Databases.*

	Ano	Desenvolvedor	Sistema Operacional	Licença	Arquitetura	Modelo de Dados	Aramazenamento	Execução
Redis	2009	Salvatore Sanfilippo	Linux, Mac OS e Solaris	BSD	Não distribuída. É um servidor de estrutura de dados. Dados podem ser replicados em *Master/Slave*	Chave / valor	Em memória RAM. *Snapshots* em disco para durabilidade	Gerenciamento através de programas clientes em C, C++, Java, entre outros
Riak	2010	Basho Tecnologies	Linux, Mac OS e Solaris	Apache 2.0	Distribuída. Nós físicos compostos de	Cave / valor	Em disco. *Buckets* com pares	API HTTP / REST. Cliente

					nós virtuais, os “vnodes”. Dados distribuídos automaticamente no *cluster*. Fácil adição de máquinas.		chave / valor	Erlang nativo ou clientes em diversas linguagens de programação, como Java, php e C
Cassandra	2008	Desenvolvido pelo Facebook. Mantido por ASF	Linux, Windows	Apache 2.0	Arquitetura distribuída. Possível adicionar nós ao *cluster*. Replicação de dados entre os nós.	Orientado a colunas	Configurável armazenamento em disco ou memória	Apache Thrift para acesso em muitas linguagens
HBase	2009	ASF	Linux	Apache 2.0	Roda sobre o HDFS. Suporta tanto arquitetura totalmente distribuída, como independente	Orientado a colunas	Configurável armazenamento em disco ou memória	
CoutchDB	2005	Damien Katz (ex IBM). Mantido por ASF	Linux, Mac OS e Windows	Apache 2.0	Arquitetura distribuída com replicação bidirecional (sincronização) dos dados	Orientado a documentos	Documentos JSON em disco, árvores B para indexar. Mantém versões	Oferece API REST/HTTP. Também um gerenciador web chamado Futon
MongoDB	2007	10gen	Windows, Mac Os, Linux e Solaris	AGPL	Cliente / servidor. Serviço “mongos” define para qual servidor enviar requisição. Dados distribuídos automaticamente entre nós.	Orientado a documentos	Coleção de documentos armazenados no disco, organizados em forma de Árvore B	*Drivers* oficiais disponíveis para C, C++, Haskell, Java, JavaScript, Perl, PHP, entre outros
Neo4j	2007	Neo Technology	Linux, Mac OS e Windows	GPLv, AGPL e comercial	Replicação dos dados *Master / slave*	Orientado a grafos		API REST e gerenciamento *web*

Fonte: Adaptado de Friess (2013).

Após a realização dos estudos o Banco de Dado MongoDB foi escolhido, levando em consideração as informações obtidas na análise individual de cada um dos Bancos de Dados, para o desenvolvimento de uma aplicação. Foi proposto o desenvolvimento de uma aplicação que consiste em um Repositório de Monografias.

2.4.3 Um Estudo sobre Bancos de Dados em Grafos Nativos

Penteado et al., 2014, em sua pesquisa “Um Estudo sobre Banco de Dados em Grafos Nativos”, teve como objetivo analisar, com base em características previamente definidas, cinco Bancos de Dados em Grafos nativos: InfiniteGraph, Neo4j, OrientDB, Titan e Trinity. Na pesquisa foram definidas nove características que podem influenciar na escolha de um SGBDG, que são: tipo de licença, suporte ao framework *TinkerPop* (o que permite utilizar qualquer ferramenta de manipulação de grafos desenvolvida pelo projeto *TinkerPop*, como por exemplo a linguagem de consulta *Gremlin*), modelagem lógica, linguagem de consulta, modelagem física, armazenamento físico, suporte à transação, tipo de arquitetura e suporte a replicação, como apresentado na Tabela 3.

Com base nessa comparação pode-se notar que o suporte ao *TinkerPop* é uma tendência entre eles devido a facilidade de manipulação de dados que as ferramentas do projeto oferecem aos usuários. O modelo de grafo de propriedades e a linguagem de consulta *Gremlin* estão presentes na maioria dos sistemas analisados, apesar de ainda não existir um padrão de linguagem de consulta para banco de dados em grafos. Outro destaque é o suporte à distribuição de dados. Todos procuram atender às necessidades das aplicações atuais que demandam grandes volumes de requisições e de dados. O suporte à distribuição e à replicação oferece uma maior escalabilidade e disponibilidade às aplicações.

Tabela 3: SGBDGs e características.

SGBDG	Aberto / Proprietário	*Tinkerpop*	Modelo Lógico	Linguagem de consulta	Modelo físico	Memória / Disco	Centralizada / Distribuída	Transação / Replicação
InfiniteGraph	proprietário	Sim	propriedades	Gremlin	objetos	disco	ambas	ambas
Neo4j	ambos	Sim	propriedades	Gremlin/ Cypher/ SPARQL	chave-valor	ambos	ambas	ambas
OrientDB	aberto	Sim	propriedades	Gremlin/ SPARQL	documentos	ambos	ambas	ambas
Titan	aberto	Sim	propriedades	Gremlin	chave-valor	ambos	ambas	ambas
Trinity	proprietário	Sim	hipergrafos	SPARQL	chave-valor	memória	distribuída	ambas

Fonte: Adaptado de Penteado et al. (2014).

2.4.4 Proveniência de Dados de Workflows de Bioinformática usando o Banco de Dados NoSQL ArangoDB

Sousa, (2015), realizou um estudo sobre a utilização do ArangoDB como um SGBD para armazenar dados brutos e de proveniência gerados a partir de *workflows* de Bioinformática. Tomando o ArangoDB, dados brutos e de proveniência foram armazenados em um mesmo banco. O estudo preliminar analisou as características necessárias para o armazenamento de diferentes modelos de dados em um mesmo banco.

Verificou-se a necessidade de um mesmo banco que pudesse armazenar e relacionar os dois tipos de dados: gerados pelo *workflow* e os dados de proveniência. Buscou-se então por bancos de dados híbridos.

Os resultados obtidos possibilitaram otimizar o processo de inserção e extração de dados no ArangoDB através da metodologia e estruturas criadas para utilização do banco. Como proposto, foi possível avaliar o desempenho do banco para inserção e extração através do armazenamento de dados das fases de filtragem e de mapeamento de um *workflow* de Bioinformática. Além disso, verificou-se que o ArangoDB possui um desempenho melhor para extração de dados em relação à inserção dos mesmos.

O modelo de proveniência PROV-DM utilizado em outros trabalhos também foi adaptado para que os dados de proveniência fossem armazenados de forma que se pudesse visualizar tanto a proveniência quanto os dados que a ela pertencem no mesmo banco de dados.

2.4.5 Modelagem de Dados NoSQL: Uma Modelagem de Banco de Dados de Grafos para Persistência de Dados de Pesquisa – Um Estudo de Caso do Projeto “Oficina Mímeses”

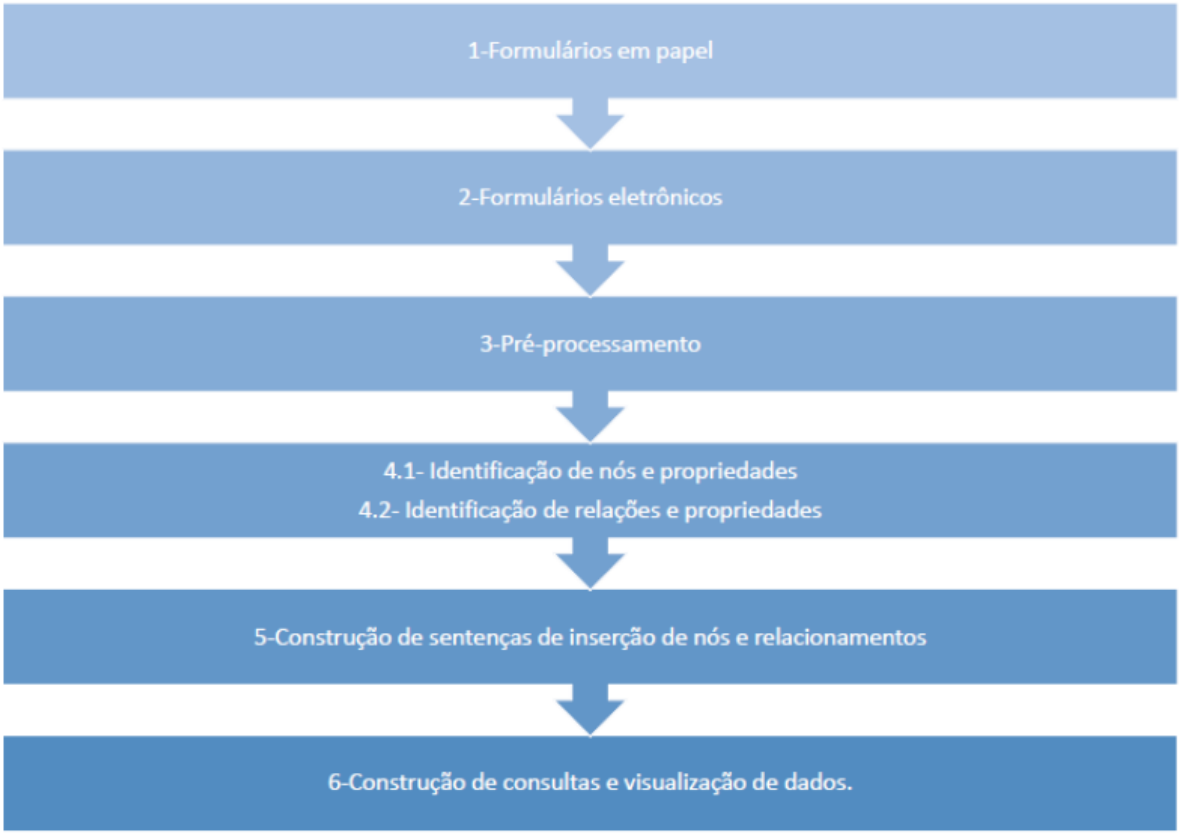
Barros, 2017, em sua pesquisa teve como objetivo modelar dados, os quais são provenientes do projeto Oficina Mímeses do departamento de Filosofia da UERN (Universidade do Estado do Rio Grande do Norte), utilizando o Banco de Dados de Grafos Neo4j. A proposta central do projeto Oficina Mímeses era levantar alguns questionamentos que poderiam avaliar a prática pedagógica do curso de Filosofia, como por exemplo, saber quantos diplomados existem na cidade de

Mossoró, quantos ensinam Filosofia e como os mesmos trabalham os conteúdos filosóficos com seus alunos.

O trabalho se concentrou em apenas um dos dados do projeto, questionários (em papel e eletrônicos) que foram aplicados a alunos, professores e gestores das escolas. Para desenvolver a modelagem organizou-se um fluxo de trabalho (workflow), como apresenta a Figura 1.

Com a modelagem dos dados em contexto de grafos, uma das vantagens notadas foi uma menor complexidade na estruturação dos dados, já que há uma liberdade maior em adaptar o contexto já modelado, com adição ou remoção de determinados elementos. E os dados que conseguiram se manter interconectados desde a modelagem até o contexto final, quando os dados são inseridos no Neo4j.

Figura 1: Fluxo de trabalho (workflow).



Fonte: Barros (2017).

Com base no referencial teórico abordado neste Capítulo e os cinco trabalhos relacionados que foram apresentados, o Capítulo 3 a seguir traz o estudo de caso

proposto e as análises desenvolvidas para a obtenção dos resultados deste documento.

3 ESTUDO DE CASO

Para validação da pesquisa, foi modelada uma rede de recomendação com dados fictícios. A mesma modelagem foi utilizada nos bancos de dados Neo4j e ArangoDB para que a haja um grau de comparação justo.

3.1 Contextualização

No contexto de Internet, um *website* personalizado é aquele capaz de reconhecer o internauta e apresentar páginas com conteúdo, produtos e ofertas selecionados especialmente para ele. Os sistemas responsáveis por estas funcionalidades são conhecidos como sistemas de recomendação (Reategui; Lorenzatti, 2005). Esta técnica prevê as preferências de um usuário com base nas preferências de outros com comportamento semelhante.

3.2 Modelagem do Estudo de Caso

Para o estudo de caso foi modelado um sistema de recomendação de Livros e Autores. O modelo possui três conjuntos de vértices: Pessoa (com a propriedade nome), Livro (com as propriedades título e gênero), e Editora (com a propriedade nome); e três conjuntos de arestas: Leu (com propriedade gostou, que pode ser sim ou não), Autor e Publicou, como mostrado na Figura 2.

Figura 2: Modelagem do Estudo de Caso.

Fonte: Autoria própria (2017).

3.3 Tecnologias Utilizadas

Nesta seção serão apresentadas as tecnologias utilizadas neste trabalho.

3.3.1 Neo4j

O Neo4j é um Banco de dados orientado a grafos, *open-source*, sua implementação foi desenvolvida em Java e sua primeira versão foi lançada em fevereiro de 2010 pela empresa *Neo Technology*. Possui duas versões de licenciamento: a *Community Edition*, com código aberto, e a *Enterprise Edition* com código fechado que oferece suporte aos seus usuários e possui mais recursos para garantir o desempenho de grandes aplicações. Além disso, o Neo4j é compatível com as propriedades ACID (Penteado et al., 2014).

Esse Banco de Dados oferece dois tipos de arquitetura, centralizada e distribuída, utiliza o modelo de grafos de propriedades e possui uma linguagem de consulta própria chamada *Cypher*, suportando também outras linguagens como a SPARQL (SPARQL Protocol and RDF Query Language), a *Gremlin* e possui a opção de realização de consultas pela API Java.

No Neo4j, para a composição de um grafo, é necessário realizar a adição de nós e de relacionamentos, onde os nós representam os vértices e os relacionamentos representam as arestas do conjunto, suas propriedades e seus rótulos (Lopes; 2014).

- Nós: São usados para representar entidades, podem ser rotulados e agrupados. Um nó pode ter relações e propriedades associados a ele e podem ser rotulados com *labels*;

- Relações: As relações organizam os nós ligando-os, conectando um nó início e um nó final e da mesma forma que o nó, pode conter propriedades;

- Propriedades: As propriedades são valores nomeados através de uma *string*, que são associados tanto a nós quanto as relações, associando metadados necessários ao problema (Robinson et al., 2015).

- Rótulos: No Neo4j quando é preciso categorizar os nós usamos o conceito de *label*, que significa rótulo. Um nó pode ter zero ou mais rótulos associados a

ele, além disso, qualquer sequência de caracteres Unicode não vazio pode ser usada como um rótulo (Barros, 2017).

3.3.1.1 Cypher

A linguagem de consulta *Cypher* foi projetada para ser facilmente lida e compreendida por desenvolvedores, profissionais de banco de dados e qualquer outro participante do projeto. Sua facilidade de uso vem do fato de que está de acordo com a maneira de como descrevemos os grafos (Robinson et al., 2015).

A seguir são descritas algumas cláusulas da linguagem *Cypher*:

- *Create*: Crie nós e relacionamentos.
- *Delete*: Exclua elementos do grafo (nós, relacionamentos ou caminhos). Qualquer nó a ser excluído também deve ter todos os relacionamentos associados explicitamente excluídos.
- *Return*: Define o que incluir no conjunto de resultados da consulta.
- *Match*: Especifique os padrões a serem pesquisados no banco de dados.
- *Where*: Adiciona restrições aos padrões de pesquisa de uma consulta.
- *Merge*: Garante que existe um padrão no grafo. O padrão já existe, ou ele precisa ser criado.

Na Figura 3, é demonstrado um exemplo de uma consulta no Neo4j utilizando a linguagem *Cypher*:

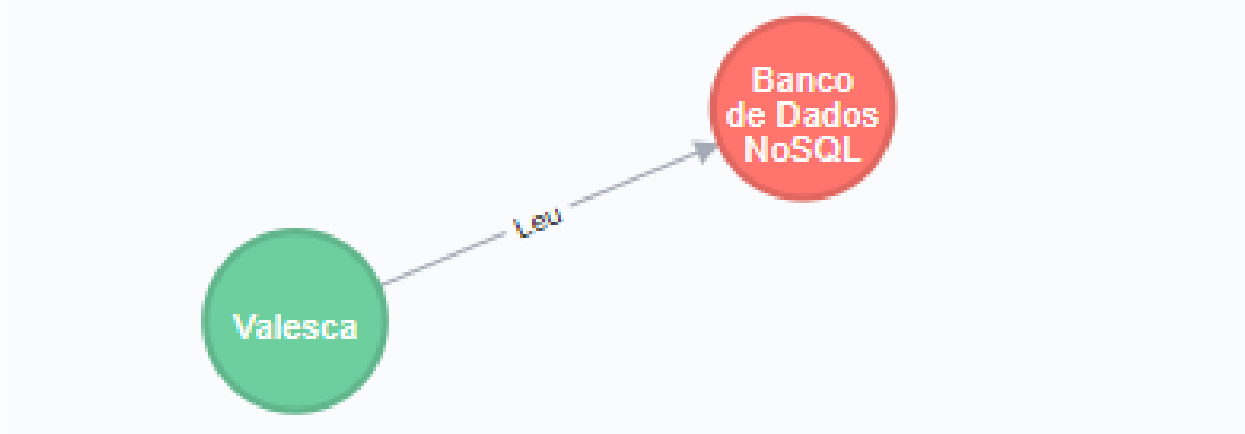
Figura 3: Criação de Nós e Relações em *Cypher.*

```
create(p:Pessoa{nome:"Valesca"}),
(l:Livro{titulo:"Banco de Dados NoSQL"})
create(p)-[:Leu]->(l)
return p, l
```

Fonte: Autoria própria (2017).

Na Figura 4 é apresentado o resultado da cláusula *return* do exemplo da Figura 3:

Figura 4: Resultado da cláusula *return*.



Fonte: Autoria própria (2017).

3.3.2 ArangoDB

Ao contrário de muitos bancos de dados NoSQL, o ArangoDB é um Banco de Dados NoSQL nativo e híbrido que combina os modelos chave-valor, orientado a documentos e a grafo. De código aberto, sua implementação foi desenvolvida em C++ e sua primeira versão foi lançada em 2011 pela *ArangoDB GmbH*. Possui duas versões de licenciamento: a *Community Edition* e a *Enterprise Edition* que oferece suporte aos seus usuários e possui mais recursos para garantir o desempenho de grandes aplicações (ArangoDB. 2017).

É compatível com as propriedades ACID, possui uma linguagem de consulta própria chamada *ArangoDB Query Language* (AQL). Os dados também podem ser acessados usando extensões do JavaScript (Sousa, 2015).

É possível acessar qualquer ou todos os seus dados usando uma única linguagem de consulta. Também é possível combinar diferentes modelos em uma única consulta. Devido ser um Banco de Dados nativo é possível criar aplicações de alta performance e escalabilidade com os três modelos de dados (Silva, 2017).

O ArangoDB possui cinco elementos que compõem um banco de dados: índices, *Databases*, Coleções, Documentos e Grafos.

- Índices: Permitem o acesso aos documentos, desde que o atributo indexado seja utilizado em uma consulta. O próprio banco já indexa alguns atributos automaticamente, mas o usuário também tem a liberdade de criar índices adicionais em atributos fora do sistema de documentos.

- *Databases*: O ArangoDB é capaz de trabalhar com vários *databases* na mesma instância do servidor. Cada *database* pode ser utilizado para agrupar dados logicamente relacionados e separá-los de outros grupos de dados.

- Coleções: O ArangoDB é constituído por um conjunto de coleções, sendo que uma coleção é constituída por documentos. Os documentos contidos nas coleções podem ser de dois tipos: documentos (padrão) ou *edges*. O tipo de documento a ser armazenado deve ser especificado na criação da coleção e não poderá ser modificado posteriormente (Sousa, 2015). No contexto de grafos uma coleção de documentos representa uma coleção de vértices. Uma coleção de *edge* (aresta) também é uma coleção de documentos, mas incluem dois atributos especiais: *_from* (origem) e *_to* (destino), que são usados para criar relações entre os documentos.

- Documentos: Os documentos no ArangoDB podem conter um ou mais atributos, cada atributo tem um valor. Um valor pode assumir os seguintes tipos: número, *string, boolean, null, array* e objeto. Os *arrays* e objetos podem conter todos esses tipos, o que significa que estruturas de dados arbitrariamente aninhadas podem ser representadas em um único documento.

- Grafos: A utilização de grafos é completamente gerenciada pelo ArangoDB, sendo inclusive visível pela interface web.

3.3.2.1 AQL

A Linguagem de Consulta AQL pode ser utilizada para obter e modificar dados que estão armazenados no ArangoDB. Com propósito similar a Linguagem de Consulta Estruturada SQL, também utiliza palavras-chave em inglês e tem como objetivo ser legível para humanos. Porém, a sintaxe é diferente do SQL tradicional,

apesar de algumas palavra-chave serem as mesmas em ambos. Outra característica do AQL é que os clientes utilizam a mesma linguagem e sintaxe para suas consultas, independentemente, daquela em que a aplicação do cliente foi construída.

A seguir são descritas algumas cláusulas da linguagem AQL:

- *Insert*: Insere novos documentos em uma coleção.
- *Remove*: Remove documentos de uma coleção.
- *Return*: Define o que incluir no conjunto de resultados da consulta.
- *For*: Interação sobre todos os elementos de um *array*.
- *Filter*: Restringir os resultados aos elementos que combinam condições lógicas arbitrárias.
- *Upsert*: Atualize/substitua um documento existente, ou crie-o no caso de não existir.

Na Figura 5, é demonstrado um exemplo de como inserir um documento numa determinada coleção no ArangoDB utilizando a linguagem AQL.

A Figura 6 apresenta a inserção do documento contendo a informação de origem e destino da aresta na coleção Leu.

A Figura 7 apresenta a consulta que irá retornar o documento da coleção Leu que possui como origem e destino os documentos das coleções Pessoa e Livro respectivamente conforme as figuras 5 e 16. A Figura 8 apresenta o resultado da consulta da Figura 7 na visualização em forma de grafo.

Figura 5: Inserção de documentos nas coleções Pessoa e Livro.

```
insert {nome: "Valesca"} in Pessoa
insert {titulo: "Banco de Dados NoSQL"} in Livro
```

Fonte: Autoria própria (2017).

Figura 6: Inserção da aresta de Pessoa para Livro.**Fonte:**

```
for p in Pessoa
for l in Livro
filter p.nome == "Valesca" && l.titulo == "Banco de Dados NoSQL"
insert {_from: p._id, _to: l._id} in Leu
```

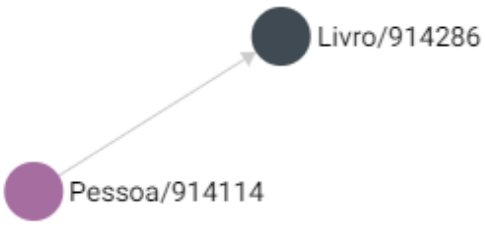
Autoria própria (2017).

Figura 7: Consulta utilizando a cláusula *return*.

```
for p in Pessoa
for l in Livro
for leu in Leu
filter leu._from == p._id && leu._to == l._id &&
p.nome == "Valesca" && l.titulo == "Banco de Dados NoSQL"
return leu
```

Fonte: Autoria própria (2017).

Figura 8: Visualização da consulta da Figura 7.



Fonte: Autoria própria (2017).

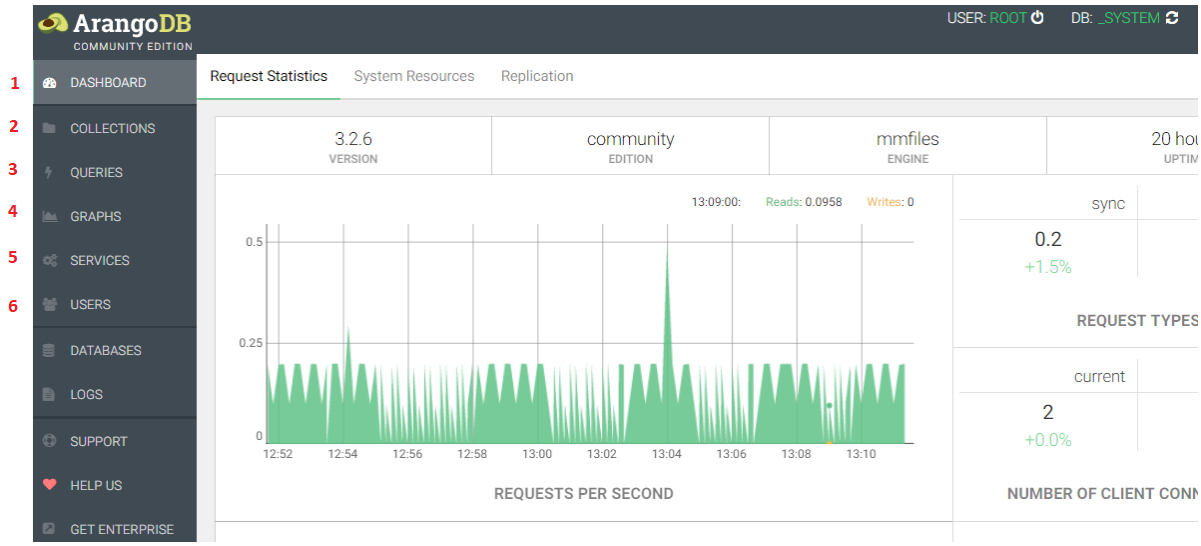
4 COMPARATIVO ENTRE OS BANCOS DE DADOS NEO4J E ARANGODB

Neste capítulo é apresentado o comparativo entre os bancos de dados Neo4j e ArangoDB. O comparativo será feito com base em quatro requisitos: Interface Web, Características, Consultas e Desempenho.

4.1 Interface

Os dois bancos de dados disponibilizam um cliente cujo acesso é feito via interface web. A Figura 9 apresenta a interface web do ArangoDB. O *Dashboard* (1) exibe os recursos utilizados e as estatísticas de uso do banco. Em *Collections* (2) é possível criar e visualizar as coleções. *Queries* (3) é onde são realizadas as consultas utilizando a linguagem AQL. A criação e visualização dos grafos é feita em *Graphs* (4). É possível desenvolver desde pontos de extremidade REST otimizados, com acesso de dados complexos a aplicativos autônomos inteiramente executados dentro do banco de dados utilizando o *framework* Foxx em *Services* (5). E em *Users* (6) pode-se visualizar e adicionar novos usuários.

Figura 9: Interface Web do ArangoDB.

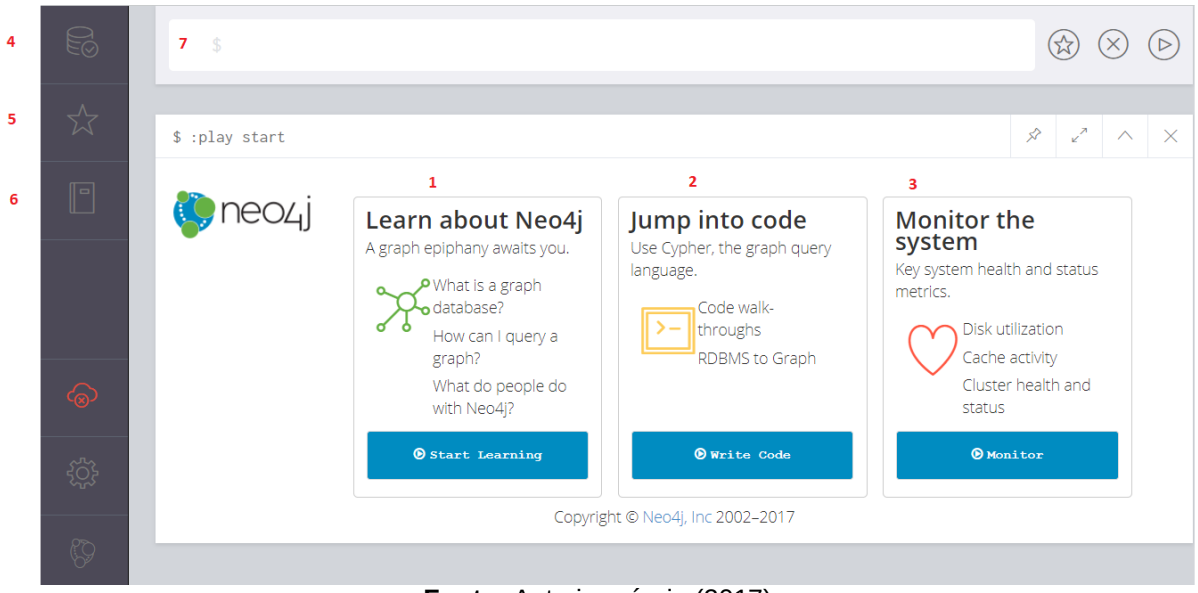


Fonte: Autoria própria (2017).

A Figura 10 apresenta a interface web do Neo4j. Assim que é acessada a interface *web* aparece uma janela com um manual sobre o Neo4j (1), um exemplo de um banco e algumas consultas em *Cypher* (2) e a opção de monitorar o sistema (3), onde são informadas as transações, páginas em cache, o armazenamento e quantidade de ID's cadastrados dos nós, relações e propriedades. Em (4) contém as

informações do banco. Abaixo em (5) é onde ficam salvas as consultas marcadas como favoritas. Em (6), logo abaixo de (5) são os manuais da interface e da linguagem *Cypher*. O retângulo em branco contendo um cifrão é onde as consultas em são realizadas (7).

Figura 10: Interface Web do Neo4j.



Fonte: Autoria própria (2017).

4.2 Características

Foram escolhidas dez características para realizar a comparação entre os bancos de dados em questão: licença, em que linguagem foi desenvolvido, o modelo dos dados, formato dos dados, como esses dados são armazenados, linguagem de consulta utilizada, suporte ao *TinkerPop*, replicação, modelo de transação, extensibilidade e segurança.

Tanto o Neo4j, quanto o ArangoDB usam uma licença para *software* livre. O Neo4j utiliza a licença GNU GLP v3 se baseia em 4 liberdades: a liberdade de executar o programa, para qualquer propósito; a liberdade de estudar como o programa funciona e adaptá-lo às suas necessidades; a liberdade de redistribuir cópias de modo que você possa ajudar ao seu próximo; e a liberdade de aperfeiçoar o programa, e liberar suas modificações, de modo que toda a comunidade se beneficie delas. O acesso ao código-fonte é um pré-requisito para seus objetivos serem atingidos. Com a garantia destas liberdades, a GPL permite que os

programas sejam distribuídos e reaproveitados, mantendo, porém, os direitos do autor (GNU, 2017).

O ArangoDB utiliza a licença Apache 2.0 permite ser usada em qualquer projeto, desde que sejam obedecidos os termos e condições contidos em seu texto. Ela permite o uso e distribuição do código-fonte tanto no software livre, quanto no proprietário. Entretanto, exige a inclusão do aviso de *copyright* (direito autoral) e do termo de responsabilidade (que informa os direitos do leitor e as responsabilidades assumidas e não assumidas pelo autor) no produto. (Apache, 2017).

Como já falado anteriormente, o Neo4j foi implementado em Java e possui o modelo de dados de Grafos. Já o ArangoDB foi implementado em C++ e é um banco de dados híbrido, possuindo três modelos de dados: chave-valor, grafos e documentos. Nos dois bancos utilizam o formato de dados JSON.

O armazenamento dos dados é feito principalmente na memória. A vantagem de manter os dados na memória ao invés do disco está relacionada à latência de acesso a esses dados, pois remove-se a necessidade de acessar a camada mais lenta da hierarquia de memória (Rockenbach et al., 2017).

Os dois bancos de dados possuem linguagens de consultas próprias, *Cypher* (Neo4j) e AQL (ArangoDB). Ambos possuem suporte ao *framework TinkerPop*, o que permite utilizar qualquer ferramenta de manipulação de grafos desenvolvida pelo projeto *TinkerPop*, como por exemplo a linguagem de consulta *Gremlin*.

A replicação dos dados no ArangoDB é feita tanto de forma síncrona (replicação instantânea) como de forma assíncrona (replicação não imediata). Já no Neo4j a replicação é feita somente de forma assíncrona. Ambos os bancos são compatíveis com as propriedades ACID.

A extensibilidade (capacidade que o *software* tende a crescer pela adição de novos componentes) no Neo4j é feita através de *plugins* em Java do lado do servidor, no ArangoDB é feita através do *framework* Foxx.

Ambos possuem autenticação, mas somente o ArangoDB utiliza os protocolos TSL (*Transport Layer Security*) e SSL (*Secure Sockets Layer*) de criptografia que permitem a comunicação segura entre cliente e servidor.

4.3 Consultas

Foram feitas algumas consultas nos dois bancos de dados para mostrar como se dá a visualização dos dados modelados. Para isso foram feitas três perguntas no contexto do estudo de caso deste trabalho.

1. Pessoas que leram o Livro 1 e gostaram, leram quais outros livros do mesmo gênero do livro 1 e também gostaram?

2. Uma determinada pessoa leu e não gostou do Livro 6. Quais outros livros essa pessoa leu e gostou?

3. Indicar autores que escreveram livros do mesmo gênero que o Livro 25?

A Figura 11 apresenta a consulta referente a pergunta 1 no Neo4j.

Figura 11: Consulta referente a pergunta 1 no Neo4j.

```
match(liv:Livro)<-[le:Leu{gostou:"Sim"}]-(p:Pessoa)
-[l:Leu{gostou:"Sim"} ]->(li:Livro{titulo:"Livro 1"})
 where liv.genero = li.genero
 return distinct liv
```

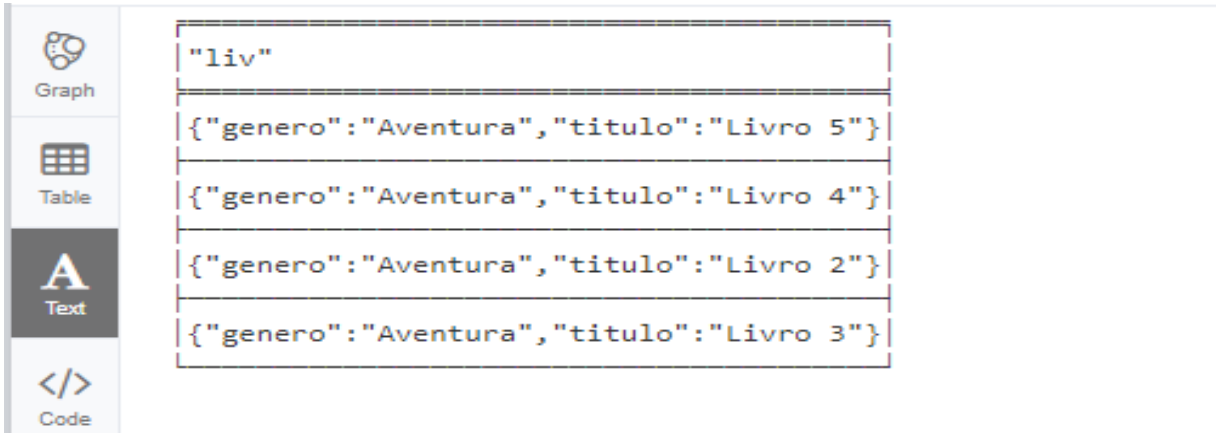
Fonte: Autoria própria (2017).

A Figura 12 apresenta o resultado da consulta da Figura 11 em forma de grafo. Podemos visualizar a consulta de três formas diferentes, em grafos, em tabela ou em texto. A Figura 13 apresenta outra forma de visualizar o resultado da consulta: em texto. Basta clicar no ícone destacado em cinza. A Figura 14 apresenta a consulta referente a pergunta 1 no ArangoDB. Na Figura 15, é apresentado o resultado da consulta da Figura 14 em forma de tabela. Podemos visualizar a consulta de duas formas diferentes, em tabela ou em JSON. A Figura 16 apresenta outra forma de visualizar o resultado da consulta: em JSON. Basta clicar no ícone com o nome JSON em verde.

Figura 12: Resultado em grafo da consulta da Figura 11.

Fonte: Autoria própria (2017).

Figura 13: Resultado em texto da consulta da Figura 12.



Fonte: Autoria própria (2017).

Figura 14: Consulta referente a pergunta 1 no ArangoDB.

```
for p in Pessoa
for li1 in Livro
for li2 in Livro
for l in Leu
for le in Leu
filter p._id == l._from && li1._id == l._to
&& l.gostou == "Sim" && li1.titulo == "Livro 1"
&& li2.genero == li1.genero && l._from == le._from
&& le.gostou == "Sim" && le._to == li2._id && li1 != li2
return distinct li2
```

Fonte: Autoria própria (2017).

Figura 15: Resultado em tabela da consulta da Figura 14.

Query 4 elements 109.148 ms JSON Table

_key	_id	_rev	titulo	genero
19177	Livro/19177	_Vsvdyyu--A	Livro 3	Aventura
19175	Livro/19175	_Vsvdyyu--_	Livro 2	Aventura
19181	Livro/19181	_Vsvdyyu--C	Livro 5	Aventura
19179	Livro/19179	_Vsvdyyu--B	Livro 4	Aventura

Fonte: Autoria própria (2017).

Figura 16: Resultado em JSON da consulta da Figura 14.

Query 4 elements 109.148 ms JSON Table


```
[
  {
    "_key": "19177",
    "_id": "Livro/19177",
    "_rev": "_Vsvdyyu--A",
    "titulo": "Livro 3",
    "genero": "Aventura"
  },
  {
    "_key": "19175",
    "_id": "Livro/19175",
    "_rev": "_Vsvdyyu--_",
    "titulo": "Livro 2",
    "genero": "Aventura"
  },
  {
    "_key": "19181",
    "_id": "Livro/19181",
    "_rev": "_Vsvdyyu--C",
    "titulo": "Livro 5",
    "genero": "Aventura"
  },
  {
    "_key": "19179",
    "_id": "Livro/19179",
    "_rev": "_Vsvdyyu--B",
    "titulo": "Livro 4",
    "genero": "Aventura"
  }
]
```


Fonte: Autoria própria (2017).

Figura 17: Consulta referente a pergunta 2 no Neo4j.

```
match(liv:Livro)<-[le:Leu{gostou:"Sim"}]-(p:Pessoa)
-[l:Leu{gostou:"Não"}]->(li:Livro{titulo:"Livro 6"})
return liv
```

Fonte: Autoria própria (2017).

A Figura 17 apresenta a consulta e o resultado da mesma referente a pergunta 2 no Neo4j. A Figura 18 mostra a consulta referente a pergunta 2 no ArangoDB. A Figura 19 apresenta o resultado da consulta referente a Figura 18 no

ArangoDB. A Figura 20 apresenta a consulta e o resultado da mesma referente a pergunta 3 no Neo4j. A Figura 21 apresenta a consulta referente a pergunta 3 no ArangoDB e a Figura 22 apresenta o resultado da consulta referente a Figura 21 no ArangoDB.

Figura 18: Consulta referente a pergunta 2 no ArangoDB.

```
for p in Pessoa
for l in Leu
for li in Livro
for le in Leu
for liv in Livro
filter l._from == p._id && l._to == li._id && l.gostou == "Não"
&& le._from == l._from && le._to == liv._id && le.gostou == "Sim"
&& li.titulo == "Livro 6"
return liv
```

Fonte: Autoria própria (2017).

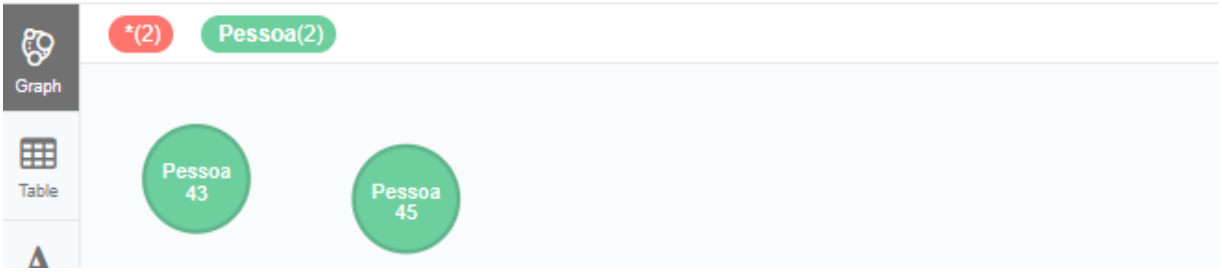
Figura 19: Resultado da consulta da pergunta 2 no ArangoDB.

_key	_id	_rev	titulo	genero
19171	Livro/19171	_Vsvdyyu---	Livro 1	Aventura
19343	Livro/19343	_Vsvf-Vy--A	Livro 28	Terror
19337	Livro/19337	_Vsvf-Vy---	Livro 26	Terror
19217	Livro/19217	_VsveDhK--C	Livro 10	Ficção Científica
19215	Livro/19215	_VsveDhK--B	Livro 9	Ficção Científica
19213	Livro/19213	_VsveDhK--A	Livro 8	Ficção Científica
19177	Livro/19177	_Vsvdyyu--A	Livro 3	Aventura
19175	Livro/19175	_Vsvdyyu--_	Livro 2	Aventura

Fonte: Autoria própria (2017).

Figura 20: Consulta referente a pergunta 3 no Neo4j.

```
1 match (p:Pessoa)<-[a:Autor]-(l:Livro{titulo:"Livro 25"})
2 match (pe:Pessoa)<-[au:Autor]-(le:Livro)
3 where l.genero = le.genero and pe <>  p
4 return pe
```



Fonte: Autoria própria (2017).

Figura 21: Consulta referente a pergunta 3 no ArangoDB.

```
for p in Pessoa
for a in Autor
for l in Livro
for pe in Pessoa
for au in Autor
for li in Livro
filter a._from == l._id && a._to == p._id && l.titulo == "Livro 25" &&
au._from == li._id && au._to == pe._id &&
pe._id != p._id && l.genero == li.genero
return distinct pe
```

Fonte: Autoria própria (2017).

Figura 22: Resultado da consulta da Figura 21 no ArangoDB.

_key	_id	_rev	nome
19028	Pessoa/19028	_VsvcVkO--o	Pessoa 43
19032	Pessoa/19032	_VsvcVkO--q	Pessoa 45

Fonte: Autoria própria (2017).

4.4 Desempenho

A comparação entre o desempenho dos bancos de dados deu-se da seguinte forma: foram utilizadas as consultas descritas na seção 4.3 deste capítulo, onde cada consulta foi executada 40 vezes em três bases de dados diferentes, descritas na Tabela 4. Foram calculados os intervalos de confiança para essas execuções e foram gerados os gráficos para melhor visualização dos resultados. As tabelas de intervalos de confiança possuem as seguintes informações: a quantidade de vezes que as consultas foram as executadas, a média entre os tempos de respostas das execuções, o desvio padrão, valores mínimos e máximos e quantis.

Foram utilizadas as versões 3.2.4 *Community Edition* do ArangoDB e 3.2.6 *Community Edition* do Neo4j. O ambiente de testes usado para comparar o desempenho dos bancos de dados em questão, contou com um equipamento equipado com as configurações descritas na Tabela 5.

Tabela 4: Bases de Dados.

Base 1	85 Nós e 133 Arestas
Base 2	170 Nós e 532 Arestas

Base 3	340 Nós e 2.128 Arestas

Fonte: Autoria própria (2017).

Tabela 5: Descrição do equipamento utilizado para os testes.

Dispositivo	**Descrição**
Processador	Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz 2.71GHz
Memória Cache	3 MB
Memória RAM	8 GB
HD	1 TB
Sistema Operacional	Windows 10 64 bits

Fonte: Autoria própria (2017).

Na Tabela 6, são apresentados os resultados referentes às consultas da questão 1. A Figura 23 apresenta o gráfico dos intervalos de confiança da Tabela 6. O Neo4j (modelo de dados de grafos) apresentou melhores resultados que o banco de dados ArangoDB (modelo de dados de documentos). É possível visualizar os valores que tiveram maior discrepância.

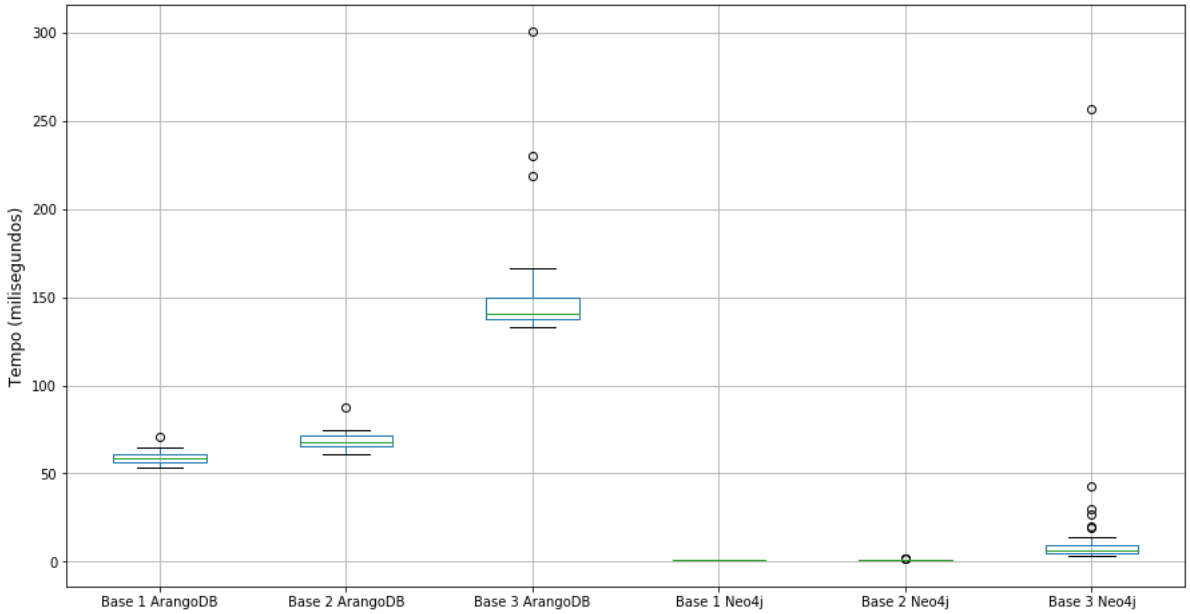
Tabela 6: Intervalo de confiança para a consulta 1.

	Base 1 ArangoDB	**Base 2 ArangoDB**	**Base 3 ArangoDB**	**Base 1 Neo4j**	**Base 2 Neo4j**	**Base 3 Neo4j**
count	40.000000	40.000000	40.000000	40.000000	40.000000	40.000000
mean	59.029250	68.846175	151.105350	1.000000	1.075000	15.225000
std	3.436981	4.828797	31.314286	0.000000	0.266747	40.065643

min	53.145000	60.868000	133.354000	1.000000	1.000000	3.000000
25%	56.678500	65.624500	137.865750	1.000000	1.000000	4.750000
50%	58.935500	67.761500	140.372500	1.000000	1.000000	6.000000
75%	61.167000	71.857250	149.395500	1.000000	1.000000	9.500000
max	70.770000	87.734000	300.848000	1.000000	2.000000	257.000000

Fonte: Autoria própria (2017).

Figura 23: Gráfico de Intervalo de confiança para a consulta 1.



Fonte: Autoria própria (2017).

Na Tabela 7, são apresentados os resultados referentes às consultas da questão 2. A Figura 24 apresenta o gráfico dos intervalos de confiança da Tabela 7. O banco de dados Neo4j teve desempenho melhor que o ArangoDB, os resultados foram similares aos da consulta 1.

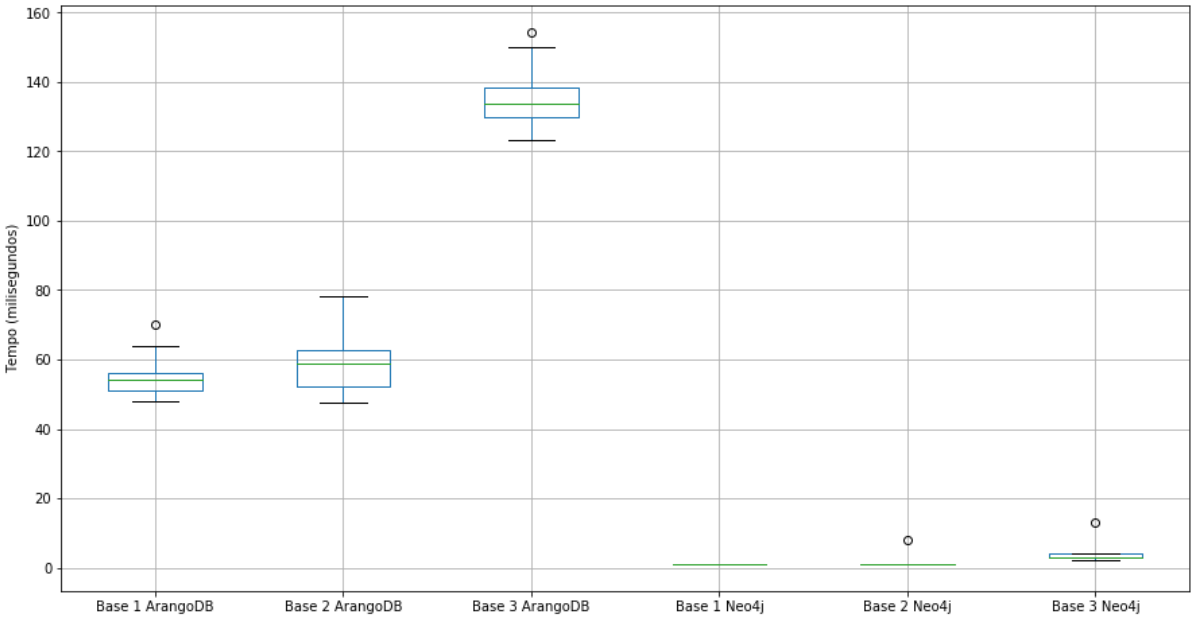
Tabela 7: Intervalo de confiança para a consulta 2.

	Base 1 ArangoDB	**Base 2 ArangoDB**	**Base 3 ArangoDB**	**Base 1 Neo4j**	**Base 2 Neo4j**	**Base 3 Neo4j**
count	40.000000	40.000000	40.000000	40.000000	40.000000	40.000000
mean	54.725000	58.430650	134.689275	1.000000	1.175000	3.625000

std	4.679401	7.561964	7.147356	0.000000	1.106797	1.628079
min	48.000000	47.625000	123.328000	1.000000	1.000000	2.000000
25%	51.000000	52.143250	129.735000	1.000000	1.000000	3.000000
50%	54.000000	58.658500	133.884000	1.000000	1.000000	3.000000
75%	56.250000	62.670750	138.382000	1.000000	1.000000	4.000000
max	70.000000	78.246000	154.411000	1.000000	8.000000	13.000000

Fonte: Autoria própria (2017).

Figura 24: Gráfico de Intervalo de confiança para a consulta 2.



Fonte: Autoria própria (2017).

Na Tabela 8, são apresentados os resultados referentes às consultas da questão 3. A Figura 25 apresenta o gráfico dos intervalos de confiança da Tabela 8. O banco de dados Neo4j teve desempenho melhor que o ArangoDB, os resultados foram similares aos da consulta 1 e 2, exceto os resultados da base 3 do ArangoDB que foram muito elevados pois passaram de 1 segundo.

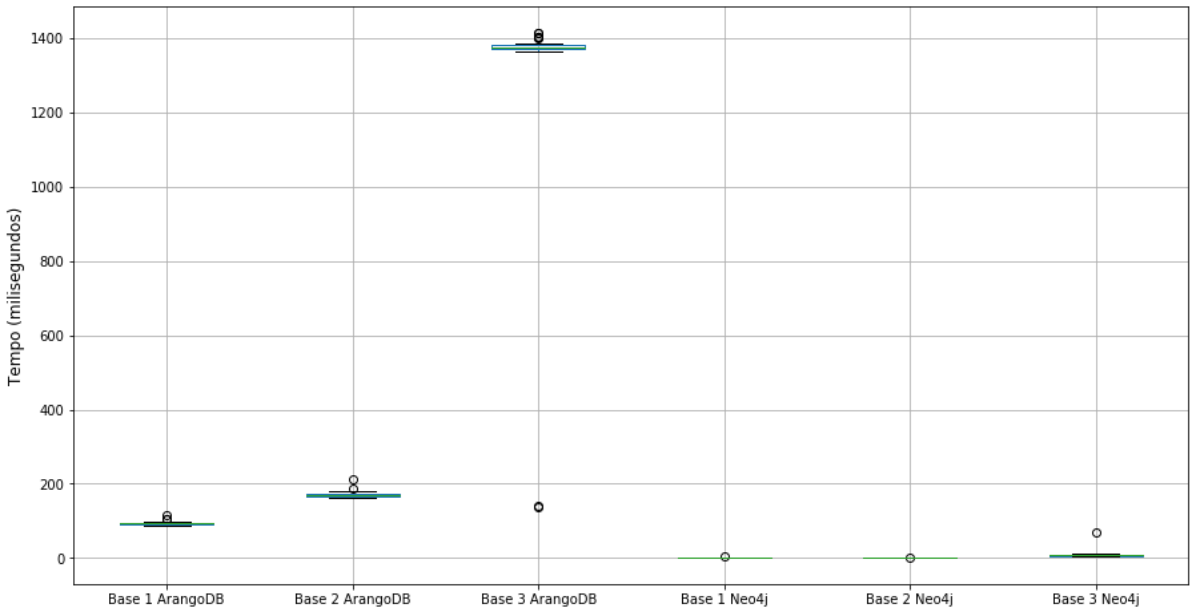
Tabela 8: Intervalo de confiança para a consulta 3.

	Base 1 ArangoDB	**Base 2 ArangoDB**	**Base 3 ArangoDB**	**Base 1 Neo4j**	**Base 2 Neo4j**	**Base 3 Neo4j**

count	40.000000	40.000000	40.000000	40.000000	40.000000	40.000000
mean	93.995175	170.819975	1318.750000	1.050000	1.025000	8.800000
std	4.985861	7.991431	274.523153	0.316228	0.158114	9.842868
min	87.853000	162.083000	138.000000	1.000000	1.000000	6.000000
25%	91.146750	167.311750	1371.750000	1.000000	1.000000	6.000000
50%	92.852000	169.527000	1376.500000	1.000000	1.000000	7.000000
75%	95.546750	172.728250	1382.750000	1.000000	1.000000	8.000000
max	115.378000	210.577000	1416.000000	3.000000	2.000000	69.000000

Fonte: Autoria própria (2017).

Figura 25: Gráfico de Intervalo de confiança para a consulta 3.



Fonte: Autoria própria (2017).

4.5 Análise dos Resultados

Conforme apresentado na Seção 4.1 deste Capítulo, a interface do ArangoDB é mais robusta que a do Neo4j, pois além de realizar consultas, inserção e extração de dados, também é possível gerenciar os bancos de dados criados, verificar os recursos utilizados como requisições por segundo, tipo de requisição, número de conexões de clientes, uso de memória, uso de CPU entre outras funcionalidades.

A Tabela 3 mostra as o comparativo das características entre o Neo4j e o ArangoDB que foi feito na Seção 4.2 deste Capítulo:

Tabela 9: Neo4j X ArangoDB.

Característica	Neo4j	ArangoDB
Licença	GNU GPL v3	Apache 2.0
Desenvolvido Em	Java	C++
Modelo de Dados	Grafos	Chave-valor, Grafos e Documentos
Formato dos Dados	JSON	JSON
Armazenamento dos Dados	Principalmente na memória	Principalmente na memória
Linguagem de Consulta	Cypher	AQL
Suporte ao TinkerPop	Sim	Sim
Replicação	Assíncrona	Síncrona e Assíncrona
Modelo de Transação	ACID	ACID
Extensibilidade (Modo /Funções do lado do servidor)	Plugins Java do Lado do Servidor	Estrutura de Microservice Foxx com Base no Google V8
	Sim	Sim
Segurança	Não/Sim	TLS - SSL/Sim

(Criptografia/Autenticação)		

Fonte: Autoria própria (2017).

Nas características apresentadas, o ArangoDB se sobressai em relação a algumas delas, como por exemplo a segurança, pois o Neo4j só possui um sistema de autenticação, enquanto o Arango utiliza os protocolos de criptografia TSL e SSL.

Em relação às linguagens de consultas, conforme visto na Seção 4.3 deste Capítulo, pode-se notar que a linguagem de consulta *Cypher* é de mais fácil entendimento que a linguagem AQL. Em *Cypher* consegue-se visualizar claramente o caminho que percorremos no grafo. O que pode tornar a linguagem AQL um pouco confusa são as várias comparações que tem que ser feitas na cláusula *Filter*.

Nos testes de desempenho, apresentados na Seção 4.4 deste Capítulo, o banco de dados Neo4j teve um desempenho melhor nos três testes. O Neo4j permaneceu com tempo de resposta inferior em todos os testes.

Foi feita uma breve análise do que pode ter acarretado o péssimo desempenho do SGBD ArangoDB, um dos possíveis motivos encontrados, é que a versão utilizada (3.2.4 *Community Edition*) possui como mecanismo de armazenamento o RocksDB que possui um déficit de performance.

Outro motivo é que nas consultas do ArangoDB os documentos estão sendo manipulados como grafos. Está sendo feito um plana cartesiano grande para poder cruzar as informações.

5 CONSIDERAÇÕES FINAIS

Este trabalho apresentou uma análise comparativa entre dois Bancos de Dados NoSQL, um orientado a grafos e outro híbrido, ArangoDB e Neo4j, respectivamente. A análise se deu com base em quatro quesitos: Interface, Características, Consultas e Desempenho.

Primeiramente, foi feito um levantamento bibliográfico acerca dos temas abordados neste trabalho, com foco em Banco de Dados NoSQL, suas características os tipos de modelos de dados. Posteriormente foi feito um estudo sobre os Bancos de Dados escolhidos e suas linguagens de consultas. Foi feita uma modelagem do estudo de caso proposto: um sistema de recomendação de Livros e Autores.

O comparativo se deu da seguinte maneira:

1. Foram analisados os recursos oferecidos pelas interfaces web de cada Banco;
2. Um conjunto de características foram escolhidas e analisadas;
3. Três perguntas foram elaboradas de acordo com o contexto do estudo de caso e foram feitas consultas nos dois bancos para responder essas perguntas;
4. Foram desenvolvidas três bases de dados com quantidades de dados diferentes e foram realizadas as consultas referentes às perguntas elaboradas.

Na análise dos resultados obtidos pode-se perceber que a interface web do ArangoDB é mais robusta que a do Neo4j, pois oferece mais recursos. De acordo com o conjunto de características analisadas, nota-se, que ambos possuem características bem distintas, como por exemplo, o tipo de licença, em que linguagens foram desenvolvidos e o modelo de dados (um é orientado a grafos e o outro é híbrido).

Nas Tabelas de intervalos de confiança dos Bancos de Dados mostra que os tempos de resposta do Neo4j foram inferiores aos do ArangoDB. Mas existem fatores que contribuíram para o péssimo desempenho do ArangoDB.

Como trabalhos futuros pretende-se criar nossas bases de dados aumentando o número de vértices e arestas a fim de realizar os mesmos testes de

desempenho para verificar se o comportamento desses bancos, em relação ao tempo de resposta, continuará o mesmo.

Pretende-se reformular as consultas do ArangoDB utilizando as cláusulas para manipulação de grafos e refazer os testes utilizando outra versão de licenciamento para verificar se o tempo de resposta diminui.

Pretende-se também fazer a mesma análise com outros bancos de dados orientados a grafos para comparar os mesmos em relação ao Neo4j e ArangoDB.

REFERÊNCIAS

ANGLES, Renzo. **A comparison of current graph database models.** In: Data Engineering Workshops (ICDEW), 2012 IEEE 28th International Conference on. IEEE. p. 171-177.2012.

Apache. Apache License. 2017. Disponível em: http://www.apache.org/licenses/LICENSE-2.0. Acessado em 14/10/2017.

ArangoDB. ArangoDB. 2017. Disponível em: https://www.arangodb.com. Acessado em 13/10/2017.

BARROS, Bruna Maria Nunes. **Modelagem de Dados NoSQL: Uma Modelagem de Banco de Dados de Grafos para Persistência de dados de Pesquisa - Um Estudo de Caso do Projeto "Oficina Mímesis".** Monografia. Universidade do Estado do Rio Grande do Norte, Mossoró - RN, 2017.

BERNARDO D. S.; BORLONE D. R. **Análise comportamental por intermédio das tecnologias de Big Data.** REFAS-Revista Fatec Zona Sul, v. 2, n. 2, p. 20-42, 2016.

CARDOSO, Ricardo Manuel Fonseca. **Bases de Dados NoSQL.** Tese de Doutorado. Instituto Superior de Engenharia do Porto, Porto, 2012.

DATE, C. J. Bancos de dados: introdução aos sistemas de bancos de dados. **Tradução da 8º ed. Rio de Janeiro: Campus**, 2004.

ELMASRI, Ramez e NAVATHE, Shamkant B. **Sistema de banco de dados.** Tradução Daniel Vieira, 6ª ed. São Paulo, Pearson Education, selo Addison Wesley. 2011.

FRIESS. Ivan Isaías. **Análise de Bancos de Dados NoSQL e Desenvolvimento de uma Aplicação.** Monografia. Universidade Federal de Santa Maria. Santa Maria - RS. 2013.

GNU. GNU Operating System. 2017. Disponível em: https://www.gnu.org/licenses/quick-guide-gplv3.html. Acessado em 15/10/2017.

HECHT, Robin; JABLONSKI, Stefan. **NoSQL evaluation: A use case oriented survey.** In: Cloud and Service Computing (CSC), 2011 International Conference on. IEEE, 2011. p. 336-341.

LÓSCIO, Bernadette Farias; OLIVEIRA, Hélio Rodrigues de; PONTES, Jonas César de Sousa. **NoSQL no desenvolvimento de aplicações Web colaborativas.** VIII Simpósio Brasileiro de Sistemas Colaborativos, v. 10, p. 11, 2011.

MOREIRA, Diogo Cezar Cunha. **Análise de Consultas em Bancos de Dados NoSQL em Ambiente de Computação nas Nuvens.** Monografia. Faculdade Farias Brito. Fortaleza - CE. 2013.

NAYAK, Ameya; PORIYA, Anil; POOJARY, Dikshay. **Type of NOSQL databases and its comparison with relational databases.** International Journal of Applied Information Systems, v. 5, n. 4, p. 16-19, 2013.

Neo4j. Neo4j. 2017. Disponível em: https://neo4j.com. Acessado em 14/10/2017.

NOSQL. NOSQL Databases. 2017. Disponível em: http://nosql-database.org acessado em 08/08/2017.

PENTEADO, Raqueline RM et al. **Um estudo sobre bancos de dados em grafos nativos.** X ERBD-Escola Regional de Banco de Dados, 2014.

QUEIROZ, Gilberto Ribeiro de; MONTEIRO, Antônio Miguel Vieira; CÂMARA, Gilberto. **Bancos de Dados Geográficos e Sistemas NoSQL: onde estamos e para onde vamos.** Revista Brasileira de Cartografia, n. 65/3, 2013.

REATEGUI, Eliseo; LORENZATTI, Alexandre. **Um assistente virtual para resolução de dúvidas e recomendação de conteúdo.** V Encontro Nacional de Inteligência Artificial (ENIA 2005). São Leopoldo, RS, Julho, 2005.

ROBINSON, Ian; WEBBER, Jim; EIFREM, Emil. **Graph databases: new opportunities for connected data.** " O'Reilly Media, Inc.", 2015.

ROCKENBACH. Dinei A., et al. **Estudo Comparativo de Banco de Dados Chave-Valor com Armazenamento em Memória.** XIII ERBD-Escola Regional de Banco de Dados (2017).

RODRIGUEZ, Marko A.; NEUBAUER, Peter. **Constructions from dots and lines.** Bulletin of the Association for Information Science and Technology, v. 36, n. 6, p. 35-41, 2010.

SADALAGE, Pramod J.; FOWLER, Martin. **NoSQL distilled: a brief guide to the emerging world of polyglot persistence.** Pearson Education, 2012.

SILBERSCHATZ, Abraham; KORTH, Henry F. e SUDARSHAN, S. **Sistema de banco de dados.** Tradução de Daniel Vieira, 5ª ed. Rio de Janeiro, Elsevier Editora Ltda. 2006.

SILVA. Claudivan Barreto da. **Um Modelo Computacional para Integração de Problemas de Otimização Utilizando Banco de Dados Orientado a Grafos.** Monografia. Universidade do Estado do Rio Grande do Norte. Mossoró - RN. 2017.

SOUSA, Bruno Aires de. **Proveniência de dados de workflows de bioinformática usando o banco de dados no SQL ArangoDB**. Monografia. Universidade de Brasília. Brasília - DF. 2016.

SOUZA, Alexandre Morais de et al. **Critérios para Seleção de SGBD NoSQL: o Ponto de Vista de Especialistas com base na Literatura.** Anais do X Simpósio Brasileiro de Sistemas de Informação (Londrina–PR, Brasil. 27 a 30/05/2014, 2014).

VIEIRA, Marcos Rodrigues et al. **Bancos de Dados NoSQL: conceitos, ferramentas, linguagens e estudos de casos no contexto de Big Data.** Simpósio Brasileiro de Bancos de Dados, 2012.

ZANELA, Eduardo Henrique R. et al. Big Data: **A nova fórmula para a inovação, a concorrência e produtividade.** In: Anais do Congresso Acadêmico de Tecnologia e Informática (CATI). 2016. p. 55-62.