UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

ANÁLISE COMPARATIVA DE BENCHMARKS PARA BANCO DE DADOS NOSQL ORIENTADOS A GRAFOS DE PROPRIEDADES

LAÍS BETHÂNIA BRITO SILVA

ORIENTADOR: PROF. DR. RICARDO RODRIGUES CIFERRI

São Carlos – SP

Julho/2021

UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

ANÁLISE COMPARATIVA DE BENCHMARKS PARA BANCO DE DADOS NOSQL ORIENTADOS A GRAFOS DE PROPRIEDADES

LAÍS BETHÂNIA BRITO SILVA

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação, área de concentração: Engenharia de *Software*, Banco de Dados e Interação Humano-Computador.

Orientador: Prof. Dr. Ricardo Rodrigues Ciferri

São Carlos – SP

Julho/2021

Laís Bethânia Brito Silva
Análise Comparativa de Benchmarks para Banco de Dados NoSQL Orientados a Grafos de Propriedades/ Laís Bethânia Brito Silva. – São Carlos – SP, Julho/2021-
92 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Dr. Ricardo Rodrigues Ciferri.

– Universidade Federal de São Carlos, Julho/2021.

1. *Benchmark*. 2. Análise de Desempenho. 3. Banco de Dados. 4. *NoSQL*. 5. Banco de Dados Orientados a Grafos. 6. Grafo de Propriedade. 7. Análise Comparativa. I. Orientador. II. Universidade Federal de São Carlos. III.Centro de Ciências Exatas e de Tecnologia. IV. Análise Comparativa de Benchmarks para Banco de Dados NoSQL Orientados a Grafos de Propriedades.

UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências Exatas e de Tecnologia
Programa de Pós-Graduação em Ciência da Computação

Folha de Aprovação

Defesa de Dissertação de Mestrado da candidata Laís Bethânia Brito Silva, realizada em 06/07/2021.

Comissão Julgadora:

Prof. Dr. Ricardo Rodrigues Ciferri (UFSCar)

Profa. Dra. Marcela Xavier Ribeiro (UFSCar)

Profa. Dra. Valéria Cesario Times (UFPE)

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

O Relatório de Defesa assinado pelos membros da Comissão Julgadora encontra-se arquivado junto ao Programa de Pós-Graduação em Ciência da Computação.

Laís Bethânia Brito Silva

Análise Comparativa de Benchmarks para Banco de Dados NoSQL Orientados a Grafos de Propriedades

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação, área de concentração: Engenharia de *Software*, Banco de Dados e Interação Humano-Computador.

São Carlos – SP, 06 de julho de 2021:

Orientador
Prof. Dr. Ricardo Rodrigues Ciferri

Professora
Profa. Dr. Marcela Xavier Ribeiro
(UFSCar)

Professora
Profa. Dr. Valéria Cesário Times
(UFPE)

São Carlos – SP
Julho/2021

RESUMO

A nova era da economia de dados, baseada em conjuntos de dados com enorme volume, grande variedade de formatos e rápida velocidade na produção e utilização de dados, trouxe a necessidade de novas estruturas e métodos de gerenciamento de dados. Surgem então os bancos de dados *Not Only SQL (NoSQL)*, que fornecem uma nova forma de armazenamento e recuperação de dados com capacidades que vão além dos sistemas gerenciadores de bancos de dados relacionais. Bancos de dados *NoSQL* são divididos em quatro grandes modelos: chave-valor, orientados à colunas, orientados a documentos e orientados a grafos. São capazes de lidar de forma escalável com o armazenamento e o processamento de gigantescos volumes de dados com formato flexível, onde as manipulações não são exclusivamente realizadas por meio da linguagem SQL. Esses conjuntos de dados, muitos dos quais, modelados como grandes grafos, tornaram-se um desafio para a indústria e a academia, que vem se empenhando cada vez mais em pesquisa e inovação na área. Assim, naturalmente ocorre o aumento da demanda por sistemas de análise de desempenho de Bancos de Dados *NoSQL* Orientados a Grafos. Isso implica na necessidade de novos *benchmarks* capazes de testar essas novas tecnologias e que orientem os usuários a identificar as ferramentas que melhor se adéquem às suas aplicações. Muitos estudos têm abordado a proposta de *benchmarks* para a análise dos sistemas de Bancos de Dados *NoSQL* Orientados a Grafos. Porém, nenhum estudo descreve a comparação detalhada desses *benchmarks*. Dessa forma, este trabalho de pesquisa de Mestrado teve por objetivo realizar uma análise comparativa de *benchmarks* para avaliação de desempenho de Bancos de Dados *NoSQL* Orientados a Grafos, com ênfase especificamente no modelo de grafos de propriedade, e assim destacar as principais diferenças entre esses, além de identificar pontos positivos e limitações de cada *benchmark*. Assim, torna-se possível identificar o *benchmark* mais adequado para analisar o desempenho de sistemas de Bancos de Dados *NoSQL* Orientados a Grafos de Propriedades em função de um conjunto de requisitos de um domínio específico de aplicação. Dentre os diversos *benchmarks* existentes, foi investigada a relevância dos *benchmarks LDBC SNB*, *XGDBench HPC* , *TGDB* e *Cyclone*. Em termos gerais, o benchmark LDBC-SNB se sobressaiu em relação aos outros benchmarks, principalmente quando se trata dos tipos de consultas e medidas de desempenho e por prover suporte a diversos SGBD's NoSQL orientados a grafos.

Palavras-chave: *Benchmark*, Análise de Desempenho, Banco de Dados, *NoSQL*, Banco de Dados Orientados a Grafos, Grafo de Propriedade, Análise Comparativa.

ABSTRACT

The new era of data economics, based on data sets with enormous volume, wide variety of formats, and rapid speed in data production and utilization, has brought the need for new data management structures and methods. Then come Not Only SQL (NoSQL) databases, which provide a new way of storing and retrieving data with features that go beyond relational database management systems. NoSQL databases are divided into four major models: key -value, column-oriented, document-oriented and graph-oriented. They are able to scalably handle the storage and processing of gigantic volumes of data with a flexible format, where manipulations are not exclusively performed through the SQL language. These data sets, many of which, modeled as large graphs, to - a challenge for industry and academia, which has been increasingly committed to research and innovation in the area. Thus, there is a natural increase in demand for performance analysis systems for Graph Oriented NoSQL Databases. This implies the need for new benchmarks capable of testing these new technologies and that guide users to identify tools that best adapt to their applications. Many studies approach the proposal of benchmarks for an analysis of Graph Oriented NoSQL Database systems. However, no study studies the comparison of such benchmarks. Thus, this Master's research work aimed to carry out a comparative analysis of benchmarks for performance evaluation of Graph Oriented NoSQL Databases, with specific emphasis on the property graph model, and thus highlight as main differences between these, in addition to identifying strengths and limitations of each benchmark. Thus, it becomes possible to identify the most appropriate benchmark to analyze the performance of Property Graph-Oriented NoSQL Database systems according to a set of requirements of a specific domain. Application. Among the several existing benchmarks, it was investigated to compare the LDBC SNB, XGDBench, HPC-SGAB, TGDB and Cyclone benchmarks. In general terms, the LDBC-SNB benchmark stood out in relation to the other benchmarks, especially when it comes to query types and performance measures, and because of its tester support for several graph-oriented NoSQL DBMS's.

Keywords: Benchmark, Performance analysis, Databases, NoSQL, Graph Database, Property Graph, Comparative Analysis.

LISTA DE SIGLAS

ACID	Atomicidade, Consistência, Isolamento e Durabilidade
BASE	Basicamente disponível, estado leve e consistência eventual
BD	Banco de Dados
CAP	Consistência, Disponibilidade e Tolerância a Partição
NoSQL	*Not Only SQL*
OLTP	*Online Transaction Processing*
SGBD	Sistema Gerenciador de Bancos de Dados
SGBDR	Sistema Gerenciador de Banco de Dados Relacional
SQL	*Structured Query Language*
TPC	*Transaction Processing Performance Council*

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

Capítulo 1
INTRODUÇÃO

1.1 Considerações Iniciais

Neste capítulo é apresentada a introdução desta dissertação de Mestrado, cuja pesquisa teve por objetivo realizar a análise comparativa dos principais *benchmarks* de sistemas gerenciadores de Bancos de Dados (SGBDs) *NoSQL* Orientados a Grafos, com ênfase especificamente no modelo de grafos de propriedade. Desta maneira, é feita uma contextualização do projeto na Seção 1.2. Na Seção 1.3 é discutida a motivação para a realização desta pesquisa de Mestrado, destacando a lacuna existente na literatura investigada. Os objetivos são detalhados na Seção 1.4, assim como a hipótese que permeia os mesmos. Por fim, a estrutura desta dissertação de Mestrado é detalhada na Seção 1.5.

1.2 Contextualização

A explosão de dados gerados pelas novas aplicações na era do *Big Data* levou a um aumento significativo de novos meios de armazenamento, gerenciamento e recuperação de dados, com diferentes arquiteturas e decisões de projeto para gerenciar a variedade, o gigantesco volume e a alta velocidade na produção desses conjuntos de dados. Assim, surgem os bancos de dados *NoSQL*, que diferentemente dos bancos de dados relacionais, foram projetados para lidar com os principais aspectos do *Big Data*. Por serem mais flexíveis quanto às propriedades ACID, e baseados nas propriedades BASE (Basicamente disponível, estado leve e consistência eventual) e teorema CAP (Consistência, Disponibilidade e Tolerância a Partição), esses novos paradigmas de bancos de dados propiciam melhor suporte a relacionamentos complexos, nos quais as manipulações dos conjuntos de dados não são exclusivamente executadas mediante o uso da linguagem SQL (SASAKI et al., 2019; HARRISON, 2015). Além disso, bancos de dados *NoSQL* são flexíveis com relação ao formato dos dados armazenados e são projetados de forma a garantir escalabilidade e alta disponibilidade dos dados usando ambientes distribuídos.

Em particular, há um grande número de aplicações que requerem que os dados sejam modelados na forma de grafos, para as quais os relacionamentos entre os dados são mais

importantes do que os dados descritivos em si. Isso despertou o interesse de pesquisadores da indústria e academia, pois o armazenamento e análise de dados em forma de grafos expandiram demasiadamente nos últimos anos, e são adotados em diversos domínios como: Redes sociais, Internet das Coisas, Detecção de Fraudes, Mecanismos de Recomendação e Operações de Rede (SASAKI et al., 2019; ROBINSON et al., 2015; HARRISON, 2015).

Desta forma, surgiram diversas iniciativas que implementam e comercializam bancos de dados *NoSQL* orientados a grafos. O gráfico da Figura 1 mostra as principais soluções desenvolvidas, de acordo com o *ranking* da DB-Engines [1] [2], sendo o *Neo4j* o mais conhecido dentre os sistemas, seguido do *Microsoft Azure Cosmos DB*, um banco de dados comercial da *Microsoft*.

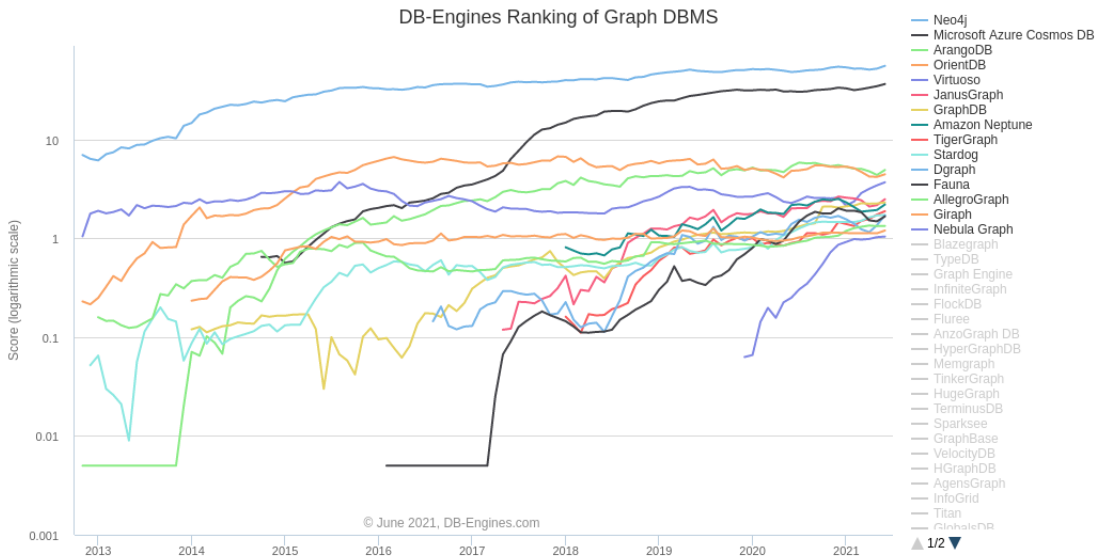


Figura 1 – Ranking dos Sistemas de Bancos de Dados Orientados a Grafos

Fonte: DB-Engines

A ampla mudança no cenário de gerenciamento de dados de grafos exige novos *benchmarks*, capazes de testar as funcionalidades específicas destas novas tecnologias em ascensão, comparando, de maneira justa, as diferentes estruturas. Dessa forma, torna-se possível identificar o *benchmark* mais adequado para analisar o desempenho de sistemas de Bancos de Dados *NoSQL* Orientados a Grafos de Propriedades em função de um conjunto de requisitos da aplicação.

[1] https://db-engines.com/en/

[2] DB-Engines é uma iniciativa para coletar e apresentar informações sobre sistemas de gerenciamento de banco de dados. Além dos bancos de dados relacionais estabelecidos, são enfatizados os sistemas e conceitos da crescente área *NoSQL*

1.3 Motivação

A comunidade de Bancos de Dados *NoSQL* Orientados a Grafos está em constante evolução, disponibilizando diversos sistemas. Entretanto, a escolha de um desses sistemas, para um determinado cenário, pode ser difícil, desde que os produtos apresentam diferentes características, oferecem funcionalidades distintas e possuem diferentes desempenhos no processamento de consultas e operações em grafos. Desta forma, destaca-se a grande necessidade de bons parâmetros de *benchmarks* que contribuam no fornecimento de metas claras e orientadas ao usuário para desempenho e funcionalidade. Além disso, a realização de análises comparativas por meio de *benchmarks* possibilita a compreensão das alternativas de projeto disponíveis (DOMINGUEZ-SAL et al., 2011).

Medidas como tempo de resposta e taxa de transferência eram suficientes antes do advento do *big data*, onde a consistência era uma das principais preocupações dos SGBDs. Porém, atualmente um *benchmark* para SGBDs deve quantificar o desempenho de dados imprevisíveis. Os *benchmarks* para SGBDs *NoSQL* Orientados a Grafos consistem em medidas relevantes e não enganosas, para medir o desempenho do sistema sob certas condições (BARATA et al., 2014; DOMINGUEZ-SAL et al., 2011).

Existem diversos *benchmarks* para SGBDs *NoSQL* Orientados a Grafos, sendo que os principais *benchmarks* são descritos no Capítulo 4. O Conselho de *Benchmark* de Dados Vinculados (LDBC), um projeto que reúne indústria e comunidade acadêmica para o desenvolvimento de *benchmarks* de código aberto, para avaliar tecnologias de gerenciamento de dados de grafos de propriedades e grafos RDF, em nível industrial, é um dos principais exemplos das iniciativas de *benchmarks* para esse fim (ANGLES et al., 2014). O LDBC atualmente possui três *benchmarks*, a saber: 1) O *Social Network Benchmark* (SNB), que objetiva testar tecnologias de gerenciamento de grafos em três cenários diferentes: interativo, inteligência de negócios e análise de grafos; 2) O *Semantic Publishing Benchmark* (SPB), um *benchmark* com base no site *BBC News*, modela uma carga de trabalho mista e atualiza a carga de trabalho com uma quantidade limitada de inferências semânticas; 3) *LDBC Graphalytics*, um *benchamrk* de nível industrial, utilizado para medir operações de análise de grafos.

Entretanto, no melhor do conhecimento da autora desta dissertação de Mestrado, e de acordo com as pesquisas realizadas com base na revisão sistemática (Apêndice A), nenhum estudo existente na literatura detalha a análise comparativa entre as tecnologias de *benchmarks* voltadas à análise de desempenho de SGBDs *NoSQL* Orientados a Grafos, com ênfase especificamente no modelo de grafos de propriedades, levando em consideração as cargas de trabalho, recursos de dados, métricas de desempenho e diversas outras características. Esta lacuna identificada na literatura caracteriza a motivação do desenvolvimento desta pesquisa de Mestrado.

1.4 Objetivo

Esta dissertação de Mestrado teve como objetivo principal realizar uma análise comparativa dos principais *benchmarks* voltados à análise de desempenho de SGBDs *NoSQL* Orientados a Grafos, com ênfase especificamente no modelo de grafos de propriedades. Dentre os diversos *benchmarks* existentes, foi investigada a relevância dos *benchmarks LDBC SNB* (ANGLES et al., 2020), *XGDBench* (DAYARATHNA; SUZUMURA, 2012) *HPC-SGAB* (DOMINGUEZ-SAL et al., 2010), *TGDB* (ABUL-BASHER et al., 2016) e *Cyclone* (TANG, 2016). A escolha dos *benchmarks* foi realizada em função das suas funcionalidades e pela análise de sua grande utilização tanto no meio acadêmico quanto na indústria.

Com base nesse objetivo, define-se a seguinte hipótese:

A análise comparativa de benchmarks para SGBDs NoSQL Orientados a Grafos, com ênfase em grafos de propriedade, pode auxiliar usuários na escolha do benchmark mais apropriado para suas aplicações, assim como direcionar futuros trabalhos de pesquisa relacionados à proposição de novos benchmarks.

Por meio da investigação da hipótese, foi possível obter as seguintes contribuições:

- Análise comparativa de diferentes sistemas de *benchmarks* de SGBDs *NoSQL* Orientados a Grafos, com ênfase em grafos de propriedades.
- Identificação dos *benchmarks* mais completos.
- Identificação de qual *benchmark* melhor se adéqua a determinados domínios de aplicação.

1.5 Estrutura da Dissertação

Além deste capítulo introdutório, esta dissertação de Mestrado possui mais 5 capítulos, estruturados de seguinte forma:

- No Capítulo 2 são descritos os principais conceitos relacionados a análise de desempenho e a técnica de *benchmark* de banco de dados.
- No Capítulo 3 primeiramente são abordados conceitos sobre sistemas de bancos de dados *NoSQL* e em seguida são descritos os principais conceitos sobre sistemas de Bancos de Dados *NoSQL* Orientados a Grafos de propriedade.
- No Capítulo 4 são descritos os principais trabalhos relacionados ao tema do projeto de pesquisa.
- No Capítulo 5 são discutidos os resultados obtidos na análise comparativa de diferentes sistemas de *benchmarks* de SGBDs *NoSQL* Orientados a Grafos de Propriedades.

- No Capítulo 6 é apresentada a conclusão desta pesquisa de Mestrado e indicados alguns trabalhos futuros que podem ser realizados como continuidade a esta pesquisa.

Capítulo 2
ANÁLISE DE DESEMPENHO

2.1 Considerações Iniciais

A evolução tecnológica tem disponibilizado soluções de *hardware* e *software* com elevado poder computacional, altamente relacionadas com a expansão de sua complexidade, causando impacto direto no estudo dos aspectos e funcionamento do computador. Dessa forma, tornou-se indispensável recursos que pudessem medir e comparar estas soluções com base em critérios bem definidos e preestabelecidos para domínios específicos de aplicações.

Um dos recursos mais utilizados para tal finalidade é a Análise de Desempenho. Aplicada em aproximadamente todas as áreas da ciência, trata-se de aferir o desempenho de determinadas funcionalidades efetuadas pelos produtos avaliados. O desempenho pode ser expresso por meio de alguma medida escalar tais como tempo, distância e tamanho, sendo produtos similares facilmente comparados usando operadores relacionais básicos por exemplo, operadores maior, menor e igual para classificar os produtos e, assim, destacar os melhores produtos e identificar os piores. (CIFERRI, 1995).

Dentre as sub-áreas da Ciência da Computação que utilizam a Análise de Desempenho, destacam-se: sistemas operacionais, arquitetura de computadores, teoria da computação e banco de dados.

Neste cenário, a análise de desempenho ajuda a resolver questões do tipo:

- Avaliar a capacidade máxima de um produto;
- Comparar diferentes tecnologias;
- Avaliar uma capacidade específica de um produto; e
- Avaliar a relação custo x benefício de um produto.

Para se realizar a aplicação da Análise de Desempenho, utiliza-se um dos seguintes modelos:

- Modelo Analítico: É baseado na obtenção de um conjunto de equações matemáticas e algoritmos, que relacionam medidas de desempenho a parâmetros do sistema que está sendo analisado.
- Modelo de Simulação: Este modelo considera a entrada e a saída de dados como parte do mundo real. Este modelo deve ser capaz de gerar dados de saída a partir de cálculos ou de inferência sobre os dados de entrada.
- Modelo Experimental: Utiliza o próprio sistema que está sendo analisado para obter resultados de desempenho.

Como relação ao modelo experimental, de forma mais detalhada, pode-se afirmar que os resultados são dependentes da maturidade do produto. As principais técnicas que fazem parte deste modelo são: Monitoração e *Benchmark*.

A técnica de monitoração utiliza ferramentas de avaliação ou estatísticas disponíveis nos produtos. Essas ferramentas não possuem uma padronização, tornando os resultados limitados, impossibilitando a comparação de resultados gerados por diferentes produtos. Enquanto isso, a técnica de *benchmark* opera executando um conjunto fixo e padronizado de tarefas, podendo ser aplicadas na comparação de diferentes produtos, visto que os testes são padronizados e bem definidos.

2.2 Técnica de *Benchmark* de Banco de Dados

A técnica de *Benchmark* é vastamente utilizada para a avaliação de desempenho de sistemas computacionais, medindo de forma sistemática as principais funcionalidades dos sistemas. Inicialmente a técnica de *benchmark* foi muito utilizada para avaliação de computadores, contudo, com o passar do tempo, ela começou a ser utilizada para outros tipos de avaliações dentro da área da Ciência da Computação, com ênfase para testes de verificação e de conformidade com determinados modelos de SGBDs. De acordo com Berry (1987), um *benchmark* de Banco de Dados é composto por uma série representativa de testes funcionais e de desempenho, os quais são efetuados em um determinado subconjunto de dados, simulando assim um ambiente de aplicação alvo, gerando resultados altamente confiáveis. O enfoque de um *benchmark* de Banco de Dados é medir o desempenho do SGBD no processamento de consultas e na realização de operações de inserção, remoção e alteração nos dados, por meio de operações básicas ou de operações mais complexas construídas sobre operações básicas. A partir deste ponto da dissertação, usaremos apenas *benchmark* para se referir a *benchmark* de Banco de Dados.

Segundo Ciferri (1995), para ser útil, espera-se que um *benchmark* tenha as seguintes características:

- **Relevância/Representatividade** para a aplicação alvo que representa;

- **Portabilidade** entre diferentes arquiteturas e configurações de sistemas computacionais;
- **Escalabilidade** podendo lidar com diferentes e crescentes valores dos parâmetros sendo analisados, tais como volume de dados, seletividade das consultas, quantidade de operações e números de nós de um *cluster* de computadores em um ambiente distribuído;
- **Simplicidade**, ou seja, um *benchmark* deve ser facilmente compreensível, não gerar ambiguidade na interpretação de suas definições de forma a manter a sua credibilidade e correto uso.

Benchmarks padrão são definidos por algumas organizações para tipos específicos de domínios. Um dos padrões mais conhecidos para a área de banco de dados é o *Transaction Processing Performance Council* (TPC) [1]. Uma corporação sem fins lucrativos, o TPC (TPC, 2010) produz *benchmarks* que medem o desempenho de sistemas no processamento de transações (OLTP) em termos de quantas transações um determinado sistema de banco de dados pode executar por unidade de tempo, por exemplo, transações por segundo ou transações por minuto. Algumas das principais avaliações são apresentadas na Tabela 1.

Dentro do contexto desta pesquisa de Mestrado, similar à organização TPC, têm-se a LDBC, um projeto que reune indústria e comunidade, para o desenvolvimento de *benchmarks* de código aberto, para avaliar tecnologias de gerenciamento de dados de grafos de propriedades e RDF, em nível industrial, é um dos principais exemplos das iniciativas de *benchmarks* para esse fim. Atuando com dois *benchmarks*, sendo estes O *Social Network Benchmark* (SNB), que objetiva testar tecnologias de gerenciamento de grafos em três cenários diferentes: interativo, inteligência de negócios e análise de grafos e o *Semantic Publishing Benchmark* (SPB), um *benchmark* com base no *site BBC News*, modela uma carga de trabalho mista e atualiza a carga de trabalho com uma quantidade limitada de inferências semânticas

Neste trabalho, *benchmark* será tratado exclusivamente para avaliação de sistemas computacionais, mais propriamente, para avaliação de desempenho de SGBDs. As principais etapas do processo de um *benchmark* de banco de dados são: definição do esquema, geração de dados, definição e execução da carga de trabalho, coleta de medidas de desempenho, ajuste de parâmetros e execução (ou validação) (SCABORA, 2016).

2.2.1 Esquema

O esquema do banco de dado define a estrutura dos dados usados no *benchmark* em termos de entidades e relacionamentos. Cada modelo de banco de dados implementa um esquema sob sua ótica de domínio, como exemplo, nos bancos de dados relacionais os dados são representados em tabelas e os relacionamentos por meio de chaves estrangeiras. Enquanto nos Bancos de Dados *NoSQL* Orientados a Grafos os dados são representados m estruturas de grafos.

[1] http://www.tpc.org/tpcx-v/default.asp

Tabela 1 – Tabela de classificação dos TPCs

TPC-C	é um *benchmark* para *Online Transaction Processing* (OLTP) que tem como objetivo avaliar as transações realizadas pelo sistema em um ambiente de entrada de pedidos.
TPC-DI:	é um *benchmark* para integração de dados que avalia a análise, combinação e transformação de dados de uma variedade de fontes e formatos em uma representação do modelo de dados unificado.
TPC-DS	é um *benchmark* para medir o desempenho de sistemas de suporte à decisão. Essa avaliação inclui sistemas de Big Data, porém não se limita a eles.
TPC-E	é um *benchmark* que simula um ambiente OLTP de um ambiente empresarial de uma corretora de clientes que geram transações relacionadas as negociações, pesquisa de conta e pesquisas de mercado.
TPC-H	é um *benchmark* de suporte à decisão estratégica com ênfase em consulta OLAP (*Online Analytical Processing*).
TPC-VMS	é um *benchmark* que avalia o desempenho de bancos de dados virtualizados. Esse método tem por objetivo representar um ambiente de virtualização no qual três cargas de trabalho de bancos de dados são consolidadas em um servidor.
TPCx-BB	é um *benchmark* para medir o desempenho dos sistemas de Big Data baseados em Hadoop, o qual é projetado para rodar em *clusters* de computadores em um ambiente de processamento paralelo e distribuído.
TPCx-HS	é um *benchmark* desenvolvido para fornecer uma medida objetiva de *hardware*, sistema operacional e distribuições comerciais de *software* compatíveis com a API do *Apache Hadoop File System*, além de fornecer à indústria métricas verificáveis de desempenho, preço-desempenho e disponibilidade.
TPCx-V	foi desenvolvido para medir o desempenho de servidores executando cargas de trabalho de banco de dados em máquinas virtuais

Fonte: Autoria Própria

Um dos principais exemplos de esquema de dados para Bancos de Dados *NoSQL* Orientados a Grafos é a baseada em rede social, utilizada por grande parte dos *benchmarks* para esse tipo de sistema.

2.2.2 Tipos de Dados

Um benchmark pode usar dois tipos de dados: dados reais e dados sintéticos. Dados reais são obtidos diretamente de aplicações e são muito importantes por representarem os dados em seu formato tipicamente usado por aplicações de um certo domínio. Assim, dados reais são usados para aferir o desempenho em certos cenários reais específicos. São ditos dados reais conjuntos de dados existentes, e fazem parte do ambiente ou de aplicações que se deseja avaliar. Porém, dados reais não conseguem ter as suas propriedades alteradas de forma a medir o impacto de certas características dos dados no desempenho do sistemas, tais como volume de dados,

distribuição dos dados e formato dos dados. Desta forma, o uso de dados sintéticos é um requisito mandatório para *benchmarks*.

Dados sintéticos são dados gerados artificialmente e portanto não presentes de fato nos ambientes e aplicações-alvo, o que possibilita a inserção de características especiais de modo a explorar e estressar possíveis deficiências nos sistemas a serem analisados. A geração de dados sintéticos deve ser feita sem ambiguidade, de forma a inserir uma constante de degradação no desempenho de todos os sistemas a serem analisados.

2.2.3 Carga de Trabalho

Para Scabora (2016), Gatti et al. (1997), Ciferri (1995), a Carga de Trabalho é a etapa de um *benchmark* na qual é inserida uma sobrecarga de execução específica, composta por operações sobre os dados (inserção, remoção, e alteração dos dados) e por consultas que irão recuperar parte dos dados armazenados, em geral gerados pelo próprio *benchmark* no caso do uso de dados sintéticos ou fornecidos por aplicações específicas no caso do uso de dados reais.

Paralelamente à carga de trabalho do *benchmark*, é possível a existência de processos independentes, gerando uma sobrecarga de execução particular no sistema computacional, chamada de carga de trabalho externa ao *benchmark*. Estes processos, ao utilizarem recursos computacionais, degradam também o desempenho do sistema como um todo. Pode-se, desta forma, dividir os processos ativos em um sistema computacional em dois grupos: os que pertencem à carga de trabalho do *benchmark* e os processos independentes. A sobrecarga de execução gerada por estes dois grupos caracteriza o que chamamos de carga de trabalho total do sistema computacional. Por isso, em geral, executa-se a carga de trabalho de um *benchmark* em um ambiente *stand-alone*, ou seja, em um ambiente controlado que somente executa o *benchmark* e *software* necessário para isso, tais como sistema operacional e o próprio sistema gerenciador de banco de dados. Outros processos não são executados em um ambiente *stand-alone*, de forma a impedir que aconteça variação na sobrecarga de execução dos processos no sistema computacional. Hoje, porém, com o uso cada vez mais intenso de sistemas baseados remotamente em nuvem, tal abordagem de uso de sistemas *stand-alone* tem migrado para o uso de máquinas virtuais alocadas na nuvem. Isso gera uma dificuldade adicional para blindar as interferências de outros processos na análise de desempenho. Para superar essa limitação, roda-se várias vezes o mesmo *benchmark* em nuvem para verificar se não houve grande oscilação dos resultados de desempenho.

Uma questão importante a se identificar é o sistema de análise. Por definição, o sistema de análise é formado por um subconjunto de carga de trabalho, presente no sistema computacional durante a realização dos testes de desempenho. Pode-se englobar o sistema computacional inteiro, ou apenas parte deste, tais como um nível do sistema computacional, alguns processos pertencentes a um nível específico ou alguns processos pertencentes a vários níveis, entre outros. Esse panorama pode ser visto na Figura 2, ou seja, um sistema computacional multinível, onde podemos observar a carga de trabalho do *benchmark* (acionada direta ou indiretamente), os

processos independentes e o sistema de análise.

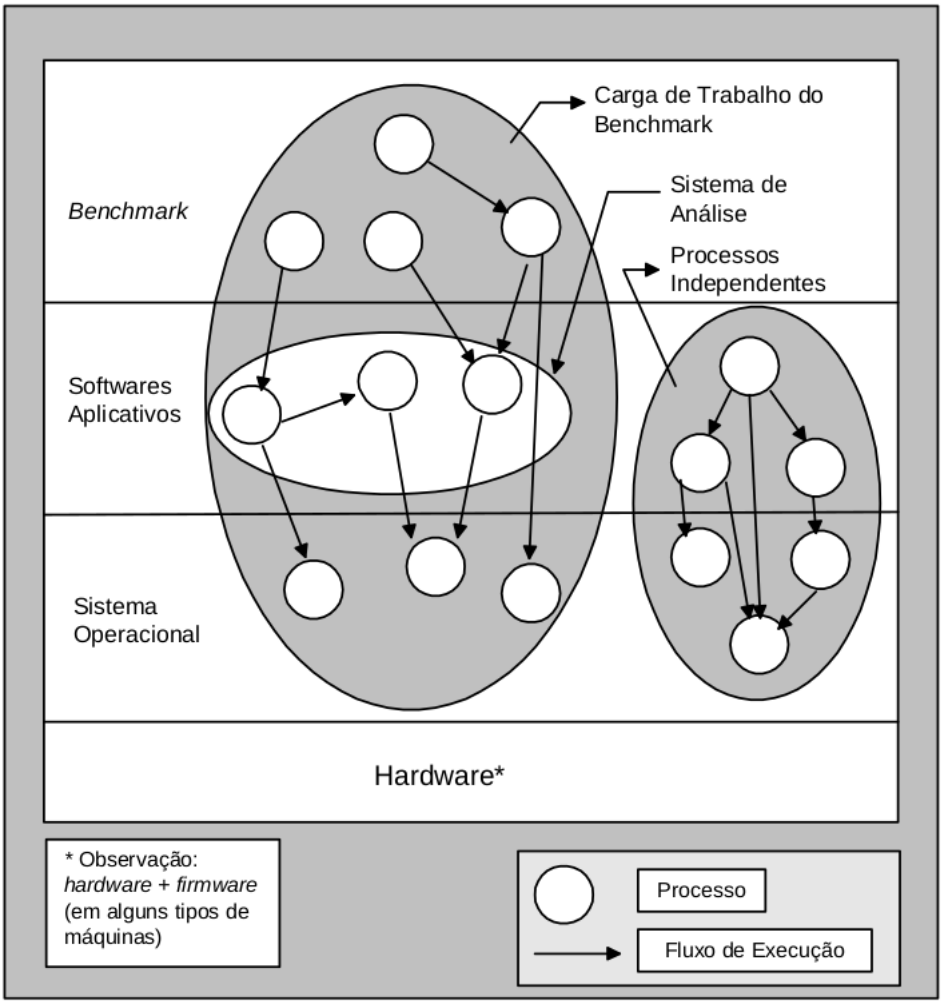


Figura 2 – Carga de Trabalho e Sistema de Análise.

Fonte: (GATTI et al., 1997; CIFERRI, 1995)

2.2.4 Medidas de Desempenho

Outro fator importante são as medidas de desempenho utilizadas pelo *benchmark*. Três classes de medidas são citadas por Gatti et al. (1997): medidas de produtividade, medidas de utilização e medidas de tempo de resposta. Porém, algumas medidas são customizadas, tais como latência, que indica o tempo que uma requisição está aguardando para ser atendida antes de ser efetivamente processada, e taxa de transferência de dados em ambientes distribuídos. Todas essas medidas anteriores são chamadas de medidas quantitativas pois reportam o desempenho por meio de coleta experimental (por exemplo, tempo decorrido para executar uma transação). Há também a possibilidade do *benchmark* usar medidas qualitativas para comparar sistemas, tais como precisão, revocação e acurácia em técnicas de aprendizado de máquina que usam SGBDs para armazenar os seus dados. Outro tipo de importante medida indireta é o e o custo do sistema

e de suas partes, que pode ser decomposto em custos de *hardware*, de *software* e de infraestrutura (serviços de nuvem).

A medida de produtividade, também conhecida como *throughput*, consiste no número de transações efetuadas em um determinado período de tempo. Mostram um panorama do desempenho global, dessa forma são chamadas de medidas macroscópicas. Assim, *throughput* é recomendada para análise de desempenho de sistemas de análise composto pelo sistema computacional total. As unidades utilizadas para representar essa medida são: transações por segundo (TPS) e transações por minuto (TPM), encontradas por exemplo nos *benchmarks* TPC-A/B e TPC-C (GATTI et al., 1997; CIFERRI, 1995). Outros exemplos de unidades de medidas de produtividade são milhões de instruções por segundo (*Mips*) e milhões de operações de pontos flutuantes por segundo (*Mflops*). Para grafos há a medida *Travessed Edges Per Second* (TEPS) e também o uso de taxa de transferência, que mede o número de consultas concluídas em um intervalo de tempo (similar a medida de produtividade). Mesmo sendo muito populares, devido a simplicidade de suas definições, não podem ser utilizadas em sistemas de análise mais abrangentes por serem medidas ditas de baixo nível, ou seja, mais especificamente para análise de desempenho de processadores.

As medidas de utilização tem como finalidade detectar possíveis fontes de gargalos resultantes de determinados recursos computacionais. Podem ser descritas por meio da quantificação absoluta, definida pela capacidade ou uso de memória em bytes (ou múltiplos), e por meio da quantificação relativa (também conhecida como taxa de utilização), representada em porcentagem. Para grafos é comum medir o tamanho do grafo em bytes.

A terceira medida apresentada é a medida de tempo de resposta, que é a quantificação unitária da degradação de desempenho, proporcionada pelas transações individuais. O tempo de resposta da consulta contabiliza o tempo decorrido entre a emissão da consulta até a saída dos resultados. Esta medida é representada por meio de medidas de tempo, tal como hora, minuto e segundo e seus submúltiplos (milessegundos, microssegundos e nanossegundos). Para grafos, alguns trabalhos propõe o uso da medida tempo de carregamento, que mede o tempo decorrido para carregar e pré-processar o conjunto de dados de um grafo. Nas medidas de tempo de resposta há a necessidade da definição do espaço de medição, podendo ser interno ou externo:

- Externo: tempo decorrido do processamento e da entrada e saída de dados.
- Interno: reporta apenas o tempo de processamento dos dados, sem considerar tempos de entrada e de saída de dados.

O custo não é reportado em nenhuma das medidas apresentadas até então. Porém existe a necessidade de relacionar o custo à questão de desempenho, o que constitui uma importante restrição existente no processo de escolha de sistemas. Entretanto, o custo dificilmente é utilizado como item de comparação. Dessa maneira, o custo é disponibilizado apenas na forma de uma

razão entre grandezas, tais como custo/desempenho ou desempenho/custo, as chamadas medidas de custo x benefício (CIFERRI, 1995). O custo do ambiente de computação deve incluir custos de *hardware*, licença e manutenção, se aplicável.

Outra medida inovadora presente em algumas propostas de *benchmark* é o consumo de energia, que mede os requisitos de energia do equipamento de computação e fornece uma medida indireta de seus requisitos de refrigeração.

2.2.5 Parâmetros

Ao se analisar experimentalmente SGBDs ou mesmo componentes específicos de SGBD (tais como índices), pode-se escolher e customizar parâmetros específicos, tais como tamanho de bloco usado pelo SGBD e tipo de particionamento de nó usado por um índice. Mais especificamente, quanto aos parâmetros, Scabora (2016) define-os como: valores configuráveis do sistema ou da aplicação que podem ser variados. A escolha dos valores dos parâmetros afeta diretamente o desempenho aferido e, portanto, devem ser escolhidos os melhores valores possíveis de forma a maximizar o desempenho e ao mesmo tempo permitir interpretar os resultados produzidos, assim como permitir a correta reprodução dos testes experimentais.

2.2.6 Execução

Quanto à última fase do processo de *benchmark*, a execução do *benchmark*, também conhecida por validação, constitui-se da execução da carga de trabalho sob o banco de dados do *benchmark* (com dados reais ou sintéticos), que envolve a execução de operações de dados, assim como consultas sob os dados, explorando os parâmetros, com intuito de coletar medidas quantitativas e qualitativas, que se fazem necessárias para a comparação dos diferentes sistemas analisados.

2.3 *Microbenchmark* e *Macrobenchmark*

Os *benchmarks* são divididos em duas classes *Microbenchmark* e *Macrobenchmark*.

Um *microbenchmark* mede o desempenho de operações primitivas. Toda consulta complexa pode ser especificamente decomposta em uma combinação de operações primitivas, consequentemente, seu desempenho pode ser explicado pelo desempenho dos componentes que as implementam.

Já um *macrobenchmark* é usado para avaliar o desempenho geral de casos de uso complexos derivados de aplicações reais. Porém, são limitados no reconhecimento de operadores com desempenho insatisfatório em uma granulação fina (LISSANDRINI et al., 2018; SZÁRNYAS et al., 2018).

2.4 Considerações Finais

A análise de desempenho de SGBD's *NoSQL* orientados a grafos tem sido alvo de diversas pesquisas na indústria e academia, como poderá ser melhor verificado no Capítulo 4, pois o modelo de dados baseados em grafos vem abrindo novas lacunas pelas quais pesquisadores tem se interessado cada vez mais. Desta forma, o surgimento de novos *benchmarks* para esses SGBD's pode ser explicado pela extensa procura e interess em usar SGBDs *NoSQL* orientados a grafos.

Esta dissertação de Mestrado restringiu-se à investigação de *benchmarks* para gerenciamento de SGBDs *NoSQL* orientados a grafos, mais especificamente, apenas de grafos de propriedades. Não sendo, assim, considerados outros tipos de *bechmarks*, tais com *benchmarks* para SGBDs relacionais ou qualquer outra estrutura de processamento de dados que não aborda específica e explicitamente o armazenamento e o processamento de grafos. Também não foram explorados *benchmarks* para outros modelos de grafos, tais como grafos RDF, desde que o escopo de possíveis tipos de modelos de grafos é muito amplo e optou-se por restringir o escopo a ao modelo de grafos de propriedades que é muito importante e amplamente usado por um grande número de aplicações em domínios bem distintos.

Capítulo 3

SISTEMAS DE BANCO DE DADOS *NoSQL* ORIENTADOS A GRAFOS DE PROPRIEDADES

3.1 Considerações Iniciais

Os Sistemas Gerenciadores de Bancos de Dados Relacionais (SGBDRs), embora eficientes para lidar com o gerenciamento de dados para aplicações tradicionais, não são robustos o suficiente para lidar com enormes volumes de dados, variedade e alta velocidade na produção dos dados que são características típicas de *big data*. Com a necessidade de se trabalhar com essas características surgem os sistemas *Not Only SQL* (NoSQL), que significa "não apenas SQL". Este termo faz referência a sistemas e SGBDs que não adotam o modelo Relacional como padrão e são mais flexíveis quanto às propriedades de atomicidade, consistência, isolamento e durabilidade (ACID) que definem uma consistência forte para os dados. Em geral, esses sistemas sacrificam ou flexibilizam a consistência dos dados de forma a atender a demanda de disponibilidade dos dados. Ademais, tais sistema são capazes de lidar com armazenamento e processamento de gigantescos volumes de dados, onde as manipulações não são exclusivamente realizadas por meio da linguagem SQL.

Algumas características específicas apresentadas pelos SGBDs *NoSQL* são fundamentais para diferenciá-los dos tradicionais SGBDRs, dentre as quais pode-se citar (CELKO, 2014; SASAKI et al., 2019):

- **Escalabilidade horizontal:** A ausência de controle de travas é uma característica dos SGBDs *NoSQL* que torna esta tecnologia adequada para solucionar problemas de gerenciamento de volumes de dados que crescem exponencialmente. Com isso, a sua arquitetura é projetada para usar de forma apropriada a expansão da quantidade de nós alocados para rodar o SGBD NoSQL em ambientes paralelos e distribuídos.
- **Ausência de esquema ou esquema flexível:** Uma característica evidente é a ausência completa ou quase total do esquema que define a estrutura dos dados. Dessa forma,

a estrutura dos dados não é rígida como no modelo Relacional e os dados podem ser armazenados em diferentes formatos de forma conjunta.

- **Suporte nativo a replicação:** Outra forma de prover escalabilidade é por meio da replicação. Permitir a replicação de forma nativa diminui o tempo gasto para recuperar dados.

- **Consistência Eventual:** É uma característica dos SGBDs *NoSQL* relacionada ao fato de que a consistência nem sempre é mantida entre os diversos pontos de distribuição de dados. Esta característica tem como princípio o teorema CAP *(Consistency, Availability e Partition Tolerance)* e o teorema PACELC (INDRAWAN-SANTIAGO, 2012; ABADI, 2012), que diz que em momentos de falha só é possível garantir duas de três propriedades entre consistência, disponibilidade e tolerância a partição e que em momentos de normalidade ainda deve-se escolher entre latência e consistência.

3.1.1 Propriedades ACID x BASE

Quando se trata dos SGBDs *NoSQL* é comum que os modelos de consistência de dados usados sejam bem diferentes dos empregados em SGBDRs. Ou seja, enquanto nos SGBDRs a consistência é forte, em SGBDs NoSQL flexibiliza-se a consistência. Isso se deve à necessidade de permitir diferenças e volumes crescentes.

As transações ACID são comumente conhecidas no contexto de SGBDRs, e fornecem um ambiente seguro para operar com dados. O acrônimo ACID tem os seguintes significados:

- **A - atomicidade:** Todas as operações em uma transação são bem-sucedidas ou todas as operações são integralmente revertidas;

- **C - consistência:** Na conclusão de uma transação, o banco de dados é estruturalmente sólido;

- **I - isolamento:** As transações não afetam a corretude de outras transações, ou seja, não causam problemas na consistência dos dados. O acesso contencioso ao estado é moderado pelo SGBD, para que as transações pareçam executar sequencialmente;

- **D - durabilidade:** Os resultados da aplicação de uma transação são permanentes, mesmo na presença de falhas.

Para alguns domínios as transações ACID acabam provocando uma sobrecarga desnecessária, visto que alguns dos SGBDs diminuem os requisitos, tais como consistência imediata, atualização de dados e precisão, para obter outros benefícios como escalabilidade e resiliência (KLEPPMANN, 2017; CELKO, 2014). Dessa maneira, surge o termo BASE, como uma

Tabela 2 – Comparação ACID x BASE

ACID	**BASE**
Consistência Forte	Consistência Fraca
Isolamento	Foco em disponibilidade
Concentra-se em commit	Melhor esforço
Transações aninhadas	Respostas aproximadas
Disponibilidade	Mais simples e mais rápido
Difícil evolução	Fácil Evolução

Fonte: Autoria Própria.

alternativa mais adequada para descrever as propriedades de uma estratégia alternativa de armazenamento e processamento de transações.

O acrônimo BASE se divide em:

- ***Basically Available:*** O banco de dados parece funcionar a maior parte do tempo. Nesse sentido, operações básicas de leitura e gravação estão disponíveis tanto quanto possível (usando todos os nós de um *cluster* de banco de dados), mas sem qualquer tipo de garantia de consistência. Ou seja, a gravação pode não persistir após os conflitos serem reconciliados, a leitura pode não obter a gravação mais recente.
- ***Soft-State:*** Os banco de dados não precisam ser consistentes com gravação, nem réplicas diferentes precisam ser mutuamente consistentes o tempo todo. Desta forma, não há garantias de consistência dos dados e depois de algum tempo só tem-se alguma probabilidade de conhecer o estado, uma vez que o estado do banco de dados pode ainda não ter convergido.
- ***Eventually Consistent:*** Os bancos de dados exibem consistência em algum momento posterior (por exemplo, futuramente no momento da leitura). Nesse sentido, os dados podem ficar inconsistentes em alguns períodos de tempo entre a atualização das suas diversas cópias distribuídas, mas em algum momento haverá convergência para a consistência dos dados.

As propriedades BASE são muito mais fracas do que as propriedades ACID. Porém, não há um mapeamento direto entre esses modelos de consistência. Um armazenamento seguindo o modelo BASE, no geral, fornece uma garantia menos rigorosa quanto a oferecida pelo modelo ACID. Os dados serão consistentes no futuro, ocasionalmente no momento da leitura, ou sempre serão consistentes, porém, apenas para determinados instantâneos passados já processados (KLEPPMANN, 2017).

Uma breve comparação entre propriedades ACID e BASE pode ser vista na Tabela 2.

3.1.2 Teorema CAP x PACELC

As propriedades ACID são usadas pelos SGBDRs para garantir a integridade do banco de dados. Enquanto os SGBDs *NoSQL* consideram essas propriedades muito restritivas e impossíveis de se alcançar em ambientes distribuídos. A Figura 3 demonstra essas propriedades, sendo posteriormente estendido para partição, disponibilidade, consistência, senão, latência, consistência (PACELC), mostrado na Figura 4 (INDRAWAN-SANTIAGO, 2012; ABADI, 2012).

Figura 3 – Propriedades do teorema de CAP e seus possíveis relacionamentos.

Fonte: :

Adaptado de (INDRAWAN-SANTIAGO, 2012).

1. *(C - Consistency) Consistência*: Todos os nós da rede necessitam ter a mesma versão do dados, ou seja, as aplicações veem a mesma versão dos dados mesmo depois de operações de inserção, remoção ou alteração nos dados.

2. *(A - Availability) Disponibilidade*: Todos os clientes podem sempre encontrar pelo menos uma cópia dos dados solicitados, mesmo que um nó do *cluster* esteja inacessível.

3. *(P - Partition Tolerance) Tolerância a partições (na rede)* : O sistema continua operacional funcionando corretamente com seus dados, propriedades e características mesmo que ocorra falhas que afetem um subconjunto de seus nós, ou seja, que afeta uma partição da rede.

O modelo PACELC sugere que a troca entre consistência e disponibilidade não se baseia, apenas, na tolerância à partição, mas também depende da existência da própria rede de partição. Em outras palavras, procura-se responder a seguinte pergunta: "*se houver uma partição (P), como o sistema troca a disponibilidade e consistência (A e C); caso contrário (E), quando o sistema estiver funcionando normalmente na ausência de partições, como o sistema troca a latência (L) e a consistência (C)?* "(ABADI, 2012).

No modelo PACELC, a latência é um fator de extrema importância, sendo que a maioria dos sistemas de banco de dados distribuídos usa a replicação para garantir a disponibilidade (Figura 4).

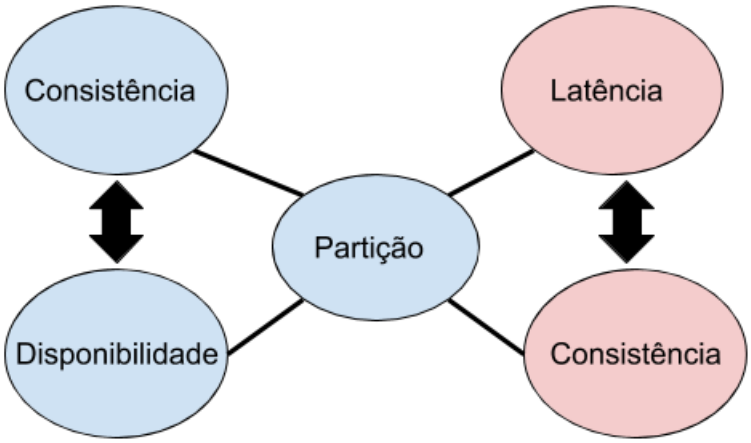


Figura 4 – Teorema PACELC.

Fonte: Adapatado de (INDRAWAN-SANTIAGO, 2012).

3.1.3 Classificação dos Bancos de Dados *NoSQL*

Os sistemas de BD *NoSQL* são classificados em quatro grandes grupos de acordo com a forma como organizam os dados: Orientados a Colunas, Pares Chave e Valor, Orientados a Documentos e Orientados a Grafos.

Os SGBDs NoSQL baseados no modelo pares chave-valor utilizam o conceito de uma chave e um valor, conhecida como tabela *hash*, para consistir os registros e garantir que não ocorra redundância. O modelo chave-valor preocupasse apenas com o armazenamento e recuperação de dados de forma eficiente, sem a responsabilidade por sua natureza ou uso. Apenas buscas pela chave para a recuperação dos demais dados são permitidas e implementadas de forma eficiente. Desta forma, esse modelo usa uma estrutura simples, porém eficiente para muitas aplicações, como buscadores.

Apesar de não conter o conceito de tabelas do modelo Relacional, o modelo orientado a colunas armazena as informações em famílias de colunas. Neste modelo as linhas são orde-

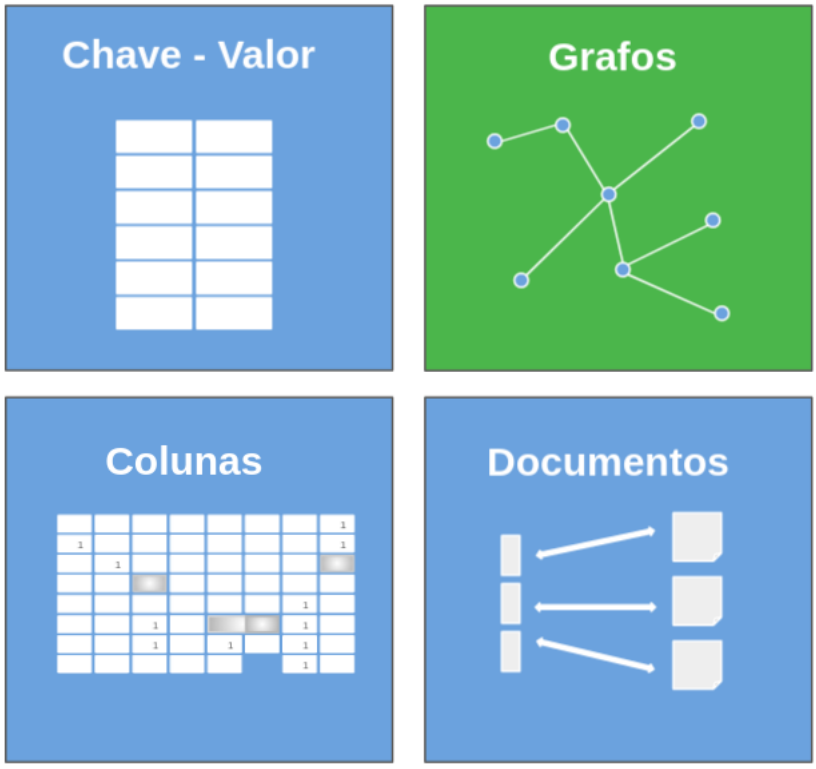


Figura 5 – Grupos *NoSQL*.

Fonte: Adaptado de Sasaki et al. (2019) .

nadas e agrupadas conforme seu contexto, daí a semelhança com tabelas. A unidade básica de armazenamento é a própria coluna, e consiste em um par chave-valor. Colunas podem ser combinadas, gerando uma supercoluna. Essas colunas são armazenadas em linhas, e uma linha contendo apenas colunas é chamada de família de colunas. A linha contendo supercolunas é chamada de família de supercolunas. A ideia básica no agrupamento de colunas é que consultas em geral acessam somente parte dos atributos e portanto armazenar os atributos sequencialmente conforme o acesso melhora o desempenho na recuperação dos dados. Portanto, o armazenamento não é realizado por linhas (dados de todas as colunas de uma tupla de um SGBD relacional) e ao contrário os dados são armazenados por coluna (ou supercoluna) de forma a recuperar rapidamente os dados de atributos específicos.

O modelo orientado a documentos é composto por documentos em vez de registros, onde estes normalmente são baseados nos formatos *eXtensible Markup Language* (XML) ou *JavaScript Object Notation* (JSON). Os SGBDs orientados a documentos armazenam e recuperam documentos como um arquivo eletrônico, podendo incluir mapas e listas. Em um nível mais simples, os documentos são armazenados e recuperados por ID, podendo se assemelhar ao modelo chave-valor. Entretanto, em geral, o modelo orientado a documentos conta com índices para facilitar o acesso dos documentos com base em seus atributos. Além disso, esse modelo permite o aninhamento de pares chave-valor e também o uso de listas de pares chave-valor e

referências a outros documentos. Documentos são organizados e armazenados conjuntamente em coleções que representam conjuntos de dados.

Por fim, o modelo orientado a grafos utiliza a estrutura de grafo para armazenar os dados com o uso dos conceitos de vértices (nós) e arestas (relacionamentos). Os dados são classificados e armazenados como entidades, assim como, suas relações são estabelecidas por meio de relacionamentos. Em vez de tabelas com linhas e colunas, este modelo é flexível e pode ser escalado por intermédio de várias máquinas.

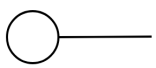
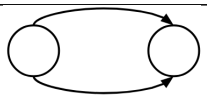
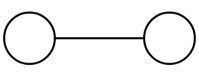
3.2 Tipos de Grafos

Assim como os SGBDRs, os SGBDs *NoSQL* Orientados a Grafos são fundamentados em uma base teórica sólida. Um ramo da matemática estabelecido há muito tempo, a teoria dos grafos, se aplica em diversas áreas, tais como medicina, física, sociologia e ciência da computação (HARRISON, 2015).

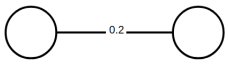
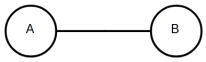
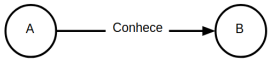
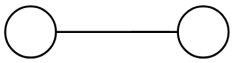
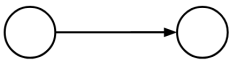
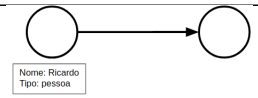
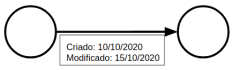
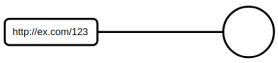
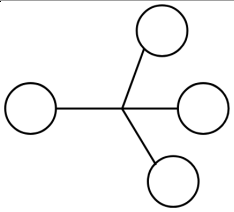
Um grafo é uma estrutura composta por nós (vértices) e arestas (ou bordas ou relacionamentos), sendo uma forma matemática para caracterizar uma rede de relacionamentos. Essa estrutura de uso geral permite modelar todos os tipos de cenários, de um sistema de estradas a uma rede de dispositivos, ao histórico médico de uma população ou a qualquer outro domínio no qual os dados possam ser definidos por relacionamentos (ROBINSON et al., 2015; HARRISON, 2015).

São diversos os tipos de grafos existentes, alguns recebem nomes diferentes em âmbitos diferentes, consequentemente é possível a utilização desses tipos combinados uns com os outros. O modo mais básico é o grafo simples, onde uma aresta conecta dois nós, e nenhum *loop* é permitido. A Tabela 3 apresenta alguns dos tipos de grafos de acordo com Rodriguez e Neubauer (2010).

Tabela 3 – Tabela de classificação dos Tipos de Grafos

Nome	**Descrição**	**Exemplo**
Grafo de Meia Aresta	Uma aresta unária, ou seja, que "conecta"um vértice. Tem aplicabilidade prática restrita e é discutido principalmente em matemática.	
Multi-Grafo:	Em diversas situações existe a necessidade de ter várias arestas entre os mesmos dois nós.	
Grafo Simples	É o grafo base, onde uma aresta conecta dois nós e nenhum *loop* é permitido.	

Continuação da Tabela 3

Nome	Descrição	Exemplo
Grafo Ponderado	Usado para representar a força dos laços ou probabilidades de transição por meio do uso de pesos nas arestas.	
Grafo marcado com nós	Quase todo grafo faz uso de nós rotulados, por exemplo um identificador.	
Grafo marcado na Aresta	Usado para denotar a maneira pela qual dois nós estão relacionados, por exemplo, amizades, parentesco, etc.	
Grafo não direcionado	O grafo típico que é usado quando o relacionamento é simétrico, por exemplo, amizade.	
Grafo direcionado	Ordena os nós de uma aresta para denotar a orientação da aresta.	
Grafo atribuído ao vértice	Usado em aplicações onde é desejável anexar metadados não relacionais a um vértice.	
Grafo atribuído à aresta	Usado em aplicativos onde é desejável acrescentar metadados não relacionais à aresta.	
Pseudo Grafo	Usado para denotar uma relação reflexiva.	
Grafo da estrutura de descrição de recursos	Um padrão de grafo desenvolvido pelo consórcio da *World Wide Web* que denota nós e arestas por identificadores uniformes de recursos.	
Grafo Semântico	Usado para modelar estruturas cognitivas, como a relação entre conceitos e as instâncias desses conceitos.	
Hipergrafo	Generaliza uma aresta binária pela qual uma aresta conecta um número arbitrário de nós.	

Fim da Tabela 3

Fonte: Adaptado de Rodriguez e Neubauer (2010).

Um tipo de grafo que atende grande parte dos sistemas de grafos (RODRIGUEZ; NEU-

BAUER, 2010; ANGLES, 2018) é o multigrafo, rotulado, direcionado, chamado de "grafo de propriedades". Nesse tipo de grafo, cada nó ou aresta pode assegurar um conjunto de pares de chave-valor, no qual a chave representa uma propriedade ou característica do dados armazenados em nós ou arestas.

Um grafo de propriedades contém nós e relacionamentos. Do ponto de vista da modelagem de dados, os nós representam entidades, e as arestas representam os relacionamentos. Tanto nós quanto arestas podem possuir propriedades, que são pares de chave-valor. Além disso, os nós podem ser rotulados, com um ou mais rótulos, e os relacionamentos são nomeados e direcionados, ou seja, sempre haverá um nó de início e um nó de fim (ANGLES, 2018). Rótulos servem para definir subconjuntos dos nós de um banco de dados de grafos.

Por exemplo, os dados do Twitter são facilmente representados como um grafo. Na Figura 6, vemos uma pequena rede de usuários do Twitter. Cada nó é rotulado como usuário, indicando seu papel na rede. Esses nós são então conectados aos relacionamentos (SEGUE), que ajudam a estabelecer ainda mais o contexto semântico.

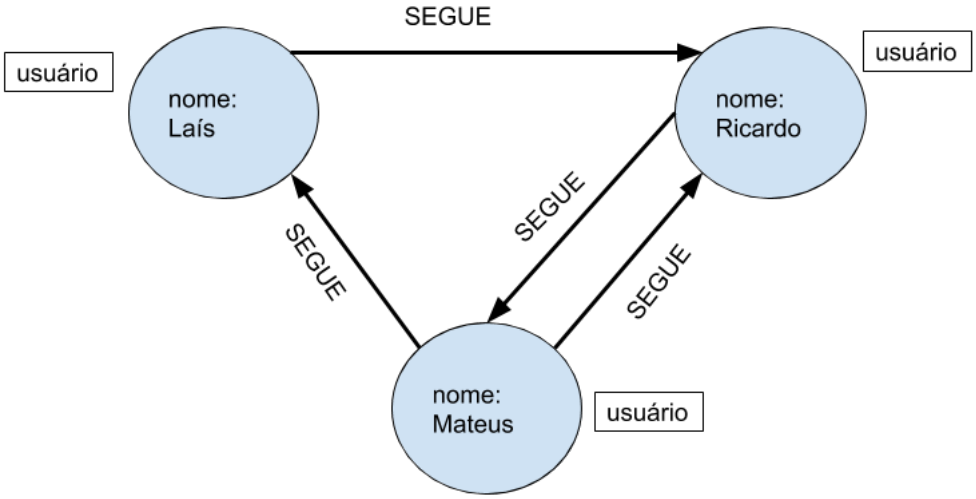


Figura 6 – Pequena rede de usuários do Twitter

Fonte: Autoria Própria

O modelo de dados para um SGBDs *NoSQL* Orientados a Grafos também é significativamente mais simples e mais expressivo do que os de SGBDRs ou outros modelos de SGBDs *NoSQL* (SASAKI et al., 2019).

3.3 Sistema de gerenciamento de banco de dados orientados a grafos

Os SGBDs *NoSQL* Orientados a Grafos são criados para uso com sistemas transacionais (OLTP) e são projetados de forma a garantir consistência por meio de integridade transacional e prover alta disponibilidade operacional. Um SGBD orientado a grafo é um sistema de gerenciamento de dados *on-line* com as operações Criar, Ler, Atualizar e Excluir (CRUD), trabalhando em um modelo baseado em grafos (ROBINSON et al., 2015).

Existem duas propriedades dos bancos de dados de orientados grafos que devem ser consideradas ao investigar as tecnologias de banco de dados de grafos:

- *Armazenamento subjacente:* Alguns bancos de dados orientados a grafos usam armazenamento de grafos "nativo", projetado especificamente para armazenar e gerenciar grafos, enquanto outros usam bancos de dados relacionais ou orientados a objetos. O armazenamento não nativo geralmente é mais lento que uma abordagem nativa.
- *Mecanismo de processamento:* O processamento nativo de grafos (também conhecido como “adjacência livre de índice”) é o meio mais eficiente para processar dados em um grafo, significando que os nós conectados fisicamente “apontam” um para o outro no banco de dados. No entanto, os mecanismos de processamento de grafos não nativos usam outros meios para processar operações de CRUD.

O mundo dos SGBDs Orientados a Grafos é um dos setores mais diversos do ecossistema *NoSQL*. Além de detalhes sobre armazenamento e processamento, os bancos de dados orientados a grafos também adotam modelos de dados distintos. A Figura 7 mostra uma visão geral de alguns SGBDs orientados a grafos no mercado atualmente, baseado em seus modelos de processamento e armazenamento.

Um dos principais motores para se trabalhar com banco de dados orientados a grafos de propriedades atualmente é o *Neo4J*. Assim como o SQL é a linguagem de consulta padrão para banco de dados relacionais, o *Cypher* é uma linguagem de consulta para tecnologias que utilizam grafos.

3.3.1 *Neo4j*

Neo4j [1] é um SGBD desenvolvido em *Java* e *Scala*, *open-source* e com versões para diversos sistemas operacionais (FERRO; SINICO, 2018), sendo o principal exemplo do modelo de Bancos de Dados Orientado a Grafos.

[1] https://neo4j.com

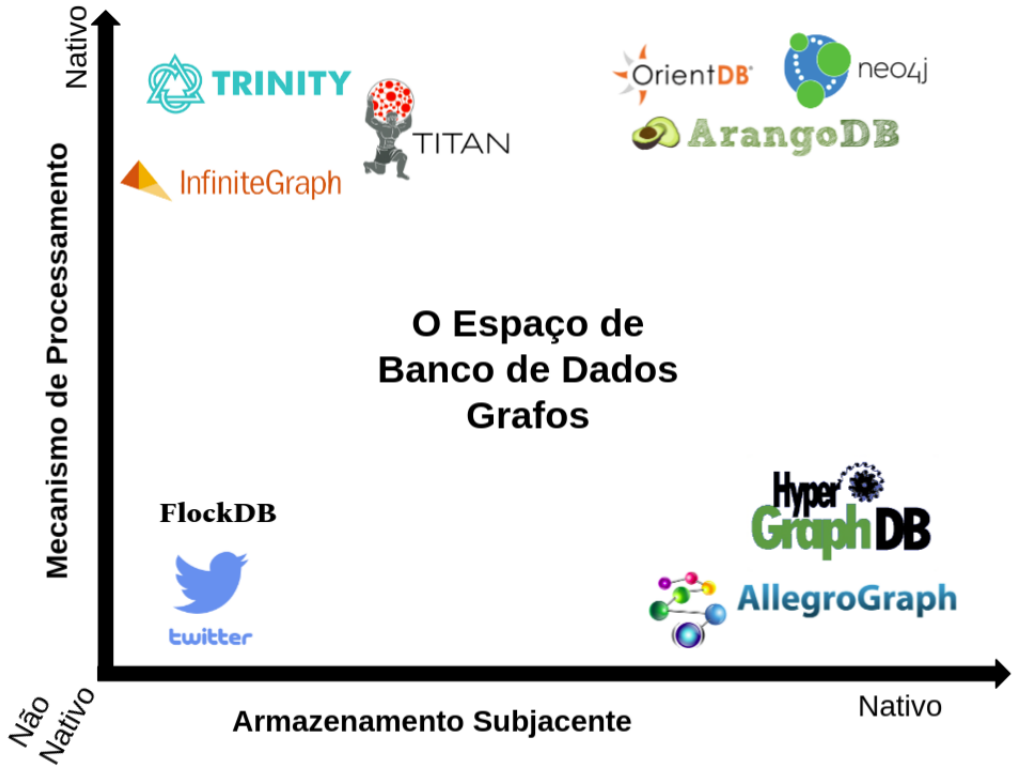


Figura 7 – Uma visão geral do espaço de SGBDs orientados a grafos.

Fonte: Adaptado de Robinson et al. (2015).

Figura 8 – Logo Neo4j

Fonte: Retirado de Neo4j, 2020.

A plataforma gráfica *Neo4j* possui ferramentas prontas para uso que permitem acessar grafos, provê suporte a *clusters* que oferecem alta disponibilidade e escalabilidade para acesso de leitura aos dados. O banco de dados *Neo4j* é um grafo de propriedades. Pode-se adicionar propriedades a nós e relacionamentos para enriquecer ainda mais o modelo de grafo.

A Figura 9 mostra um esquema de um Banco de Dados *NoSQL* Orientado a grafos para dados cinematográficos, gerado pelo *Neo4j*. Os nós rotulados Escritor, Diretor, Ator, Companhia possuem os relacionamentos ESCREVE, DIRIGE, ATUA_EM e PRODUZ, respectivamente, com os nós rotulados Filme. Isso permite que se alinhe estreitamente dados e conexões no

grafo com seu aplicativo do mundo real. Por exemplo, um nó Filme pode ter uma propriedade título e um nó Ator pode ter as propriedades nome, endereço. Além disso, um relacionamento, ATUA_EM , pode ter uma propriedade ano_Atuação.

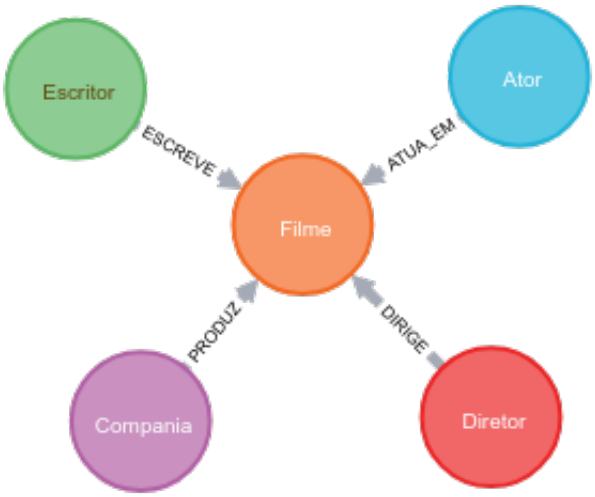


Figura 9 – Esquema de dados cinematográficos

Fonte: Autoria própria.

O mecanismo grafo *Neo4j* é usado para interpretar as instruções *Cypher* e também executa o código no nível do *kernel* para armazenar e recuperar dados, estejam eles em disco ou armazenados em cache na memória. O mecanismo grafo foi aprimorado a cada versão do *Neo4j* para fornecer o acesso mais eficiente aos dados grafos de um aplicativo. Existem várias maneiras de ajustar o desempenho do mecanismo para atender às suas necessidades específicas de aplicativos.

A Figura 10 mostra o panorama geral da plataforma gráfica *Neo4j*. O banco de dados *Neo4j* fornece suporte para transações e análises de grafos. Os desenvolvedores usam o *Neo4j Desktop*, juntamente com o *Neo4j Browser*, para desenvolver grafos e testá-los, além de implementar seus aplicativos em vários idiomas, usando *drivers*, ferramentas e APIs compatíveis. Os administradores usam ferramentas para gerenciar e monitorar bancos de dados e *clusters* do *Neo4j*. Os usuários corporativos usam ferramentas prontas de visualização de grafos ou ferramentas personalizadas (FERRO; SINICO, 2018). Analistas de dados e cientistas usam os recursos de análise nas bibliotecas do algoritmo de grafo ou usam bibliotecas personalizadas para entender e relatar descobertas à empresa. Os aplicativos também podem integrar-se aos bancos de dados existentes (SQL ou NoSQL), colocando o *Neo4j* em camadas sobre eles para fornecer acesso rico e habilitado para os dados de grafos.

3.3.1.1 Linguagem *Cypher*

Construída sobre os conceitos básicos e as cláusulas do SQL, *Cypher* é uma linguagem de consulta declarativa, porém com funções específicas para grafos. Uma linguagem intuitiva

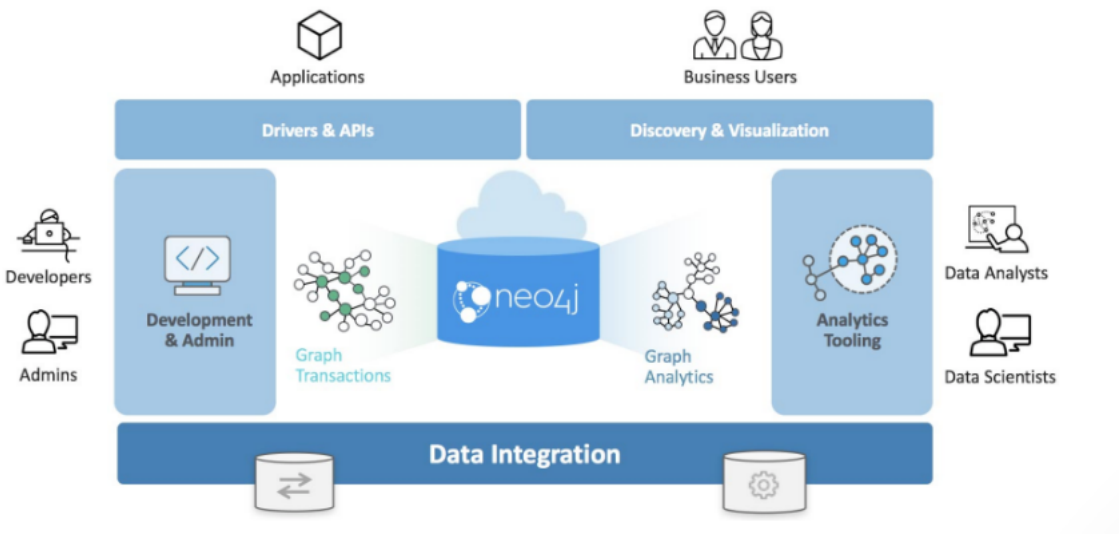


Figura 10 – Arquitetura da plataforma gráfica Neo4j.

Fonte: Retirado de NEO4J (2020)

e fácil de usar, a *Cypher* foi projetada para que desenvolvedores, profissionais de bancos de dados e partes interessadas de negócios possam ler e compreender a maneira como os grafos são descritos usando diagramas. Essa linguagem tem como princípio básico o uso de padrões para representar nós e relacionamentos. Um padrão de nó é representado entre (e), quanto um padrão de relacionamento é representado por um caminho não direcionado ()-() ou direcionado ()–>().

Um exemplo de comando *Cypher* para criação dos rótulos Escritor, Ator, Diretor e Companhia juntamente com seus respectivos atributos pode ser visualizado na Figura 11.

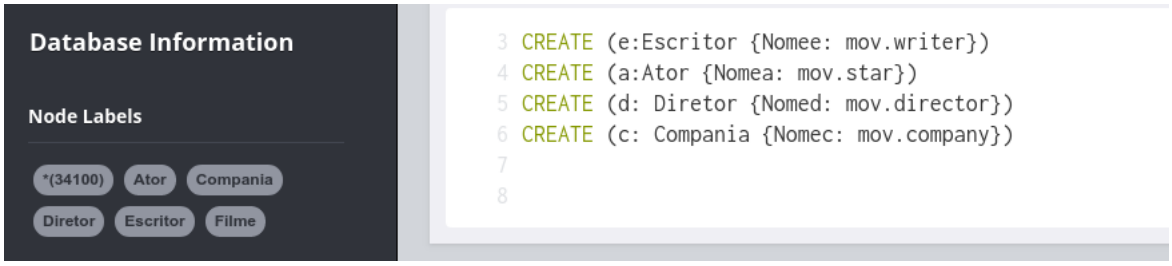


Figura 11 – Criação dos Rótulos e Atributos para o esquema de dados cinematográficos.

Fonte: Autoria própria.

3.3.2 *ArangoDB*

O *ArangoDB* [2] é um SGBD, desenvolvido na linguagem de programação C++, multimodelo e *open-source*. Esse SGBD permite modelos de dados *NoSQL* flexíveis para documentos, pares chave-valor e grafos. Ou seja, o ArangoDB é um SGBD NoSQL multimodelo. Utiliza

[2] https://www.arangodb.com/

Figura 12 – Logo ArangoDB

Fonte: Retirado de ArangoDB, 2020.

de uma mesma linguagem de consulta própria, a *Arango Query Language* (AQL), que se assemelha ao SQL e à extensões *JavaScript*, para todos os modelos de dados. Com versões nas edições *Community* e *Enterprise*, o *ArangoDB* oferece dois diferentes tipos de mecanismos de armazenamento: *memory-mapped files* (MMFiles), sendo o único mecanismo de armazenamento até a versão 3.1 do SGBD. Posteriormente, foi adicionado suporte para mecanismos de armazenamento plugáveis e um segundo mecanismo, baseado no *RocksDB* do *Facebook*. A partir da versão 3.4 o *RocksDB* se tornou o mecanismo de armazenamento padrão do *ArangoDB*, enquanto o MMFiles foi completamente removido na versão 3.7 (TUOMINEN, 2018).

As principais características do *ArangoDB* são:

- Gerenciamento de múltiplos modelos de dados com um único núcleo e linguagem de consulta (AQL);
- *API HTTP* para gerenciar bancos de dados;
- Multiarquitetura - instância única, *cluster* ou serviços mistos;
- Integração do *JavaScript Foxx Framework*;
- Fragmentação;
- Pronto para nuvem no *Azure Cloud*;
- Consolidação - minimiza os componentes, reduzindo a complexidade da pilha de tecnologia. Isso significa um menor custo total de propriedade, crescente flexibilidade para consolidar necessidades técnicas;
- Escalonamento de desempenho simplificado - pode reagir facilmente às necessidades crescentes de desempenho e armazenamento, escalando independentemente dos diferentes modelos de dados. O *ArangoDB* é dimensionado tanto verticalmente quanto horizontalmente, e se a necessidade de desempenho diminuir, pode-se facilmente reduzir o sistema de *back-end* para economizar em *hardware* e requisitos operacionais;

- Complexidade Operacional Reduzida - O conceito de Persistência Poliglota trata da escolha do modelo de dados certo para o trabalho certo. Um banco de dados multimodelo nativo permite ter dados poliglotas sem a complexidade, mas com consistência de dados em um sistema tolerante a falhas;
- Forte consistência de dados - um único *back-end* gerencia diferentes modelos de dados com suporte para transações ACID. Ele fornece consistência forte em uma única instância e operações atômicas ao operar no modo de *cluster*;
- Tolerância a falhas - usa um banco de dados de vários modelos e uma pilha de tecnologia consolidada. Por *design*, *ArangoDB* permite arquiteturas modulares com diferentes modelos de dados em execução e também foi projetado para ser usado de forma eficienet em *clusters* de computadores.

3.3.2.1 Linguagem *AQL*

Uma linguagem de consulta própria, a AQL é utilizada tanto para recuperar quanto para modificar dados armazenados no *ArangoDB*. Considerada principalmente uma linguagem declarativa, ou seja, é uma linguagem que expressa qual resultado deve ser alcançado, entretanto, não expressa como alcançar esse resultado.

A AQL se assemelha à SQL em seu objetivo, porém a sintaxe das consultas é diferente de uma linguagem para a outra. Proporciona suporte a operações de leitura e modificação de dados, mas não a operações de definição de dados. Isto se deve ao fato de a AQL ser uma linguagem de manipulação de dados (DML), e não uma linguagem de definição de dados (DDL), ou uma linguagem de controle de dados (DCL).

3.3.3 *OrientDB*

Figura 13 – Logo OrientDB

Fonte: Retirado de OrientDB, 2020.

O *OrientDB* [3] é um SGBD *NoSQL* multimodelo, *open-source*, implementado em *Java*, com suporte para os modelos de dados em documentos, pares chave-valor, grafos e objetos. Possui versões nas edições *Community* e *Enterprise*, sendo utilizado por várias empresas, tais como *Dell*, *Cisco*, *Comcast*, *Lufthansa*, *Thales* e *Fox Sports*.

Lançado em 2010, o SGBD *OrientDB* possui suporte para arquitetura distribuída com replicação e é totalmente transacional, garantindo o processamento confiável de todas as transações do banco de dados por meio do suporte a transações ACID. O banco de dados pode ser manipulado em Java, SQL ou linguagem própria *Gremlin* (PŁUCIENNIK; ZGORZAŁEK, 2017).

3.3.3.1 *OrientDB* SQL

Com foco na simplicidade e eficiência do sistema, a equipe de desenvolvimento do *OrientDB* preferiu não inventar uma nova linguagem, mantendo o SQL como linguagem de consulta. Mesmo com o propósito de manter a versão habitual do SQL, existe a necessidade de adaptação aos conceitos *NoSQL* .

Uma das principais mudanças feitas do SQL clássico para o SQL utilizado no *OrientDB* foi a remoção do operador *JOIN*, pois os SGBDs *NoSQL* não trabalham com tabelas. Dessa forma, o *OrientDB* SQL funciona com *links*, agindo como se fossem duas tabelas unidas, adicionando apenas um "."entre os dois atributos.

Além disso, *OrientDB* apresenta suporte a dados sem esquema, com esquema completo e misto, além de oferecer segurança em todos os dados confidenciais que estão presentes com o uso de autenticação, senha e criptografia de dados.

O *OrientDB* faz uso do projeto *Hazelcast Open Source*[4] objetivando a descoberta automática dos nós, armazenamento de configuração de *cluster* em tempo de execução e sincronização de certas operações entre nós.

De acordo com (69102) as principais vantagens e recursos apresentados pelo *OrientDB* são:

- Suporte à linguagem SQL;
- Suporte às tecnologias *Web - HTTP, RESTful*
- Definição de protocolo, bibliotecas JSON;
- Ser distribuído com suporte nativo à replicação *multi-master*;
- Estar pronto para ser usado em ambintes de computação em nuvem;

[3] https://www.orientdb.org/
[4] https://hazelcast.org/

- Manipulação de banco de dados utilizando *Java*;
- Permite a incorporação de documentos como qualquer outro banco de dados de documentos, mas também permite adcionalmente relacionamentos;
- Arquitetura *multi-master* com particionamento dos dados, fornecendo escalabilidade horizontal e confiabilidade;
- Instalação rápida;
- Versão gratuita com licença *Apache 2*.

3.3.4 Considerações Finais

Neste capítulo foram definidos os principais conceitos relacionados aos sistemas de bancos de dados *NoSQL*, incluindo as famílias que representam esse tipo de sistemas. Foram detalhados aspectos que definem as características de consistência e transações, tais como teorema CAP, teorema PACELC e propriedades BASE. Na Seção 3 é detalhado o paradigma de armazenamento orientado a grafos, o qual é usado como base para o desenvolvimento desta pesquisa de Mestrado.

O mundo *NoSQL* possui uma diversidade capaz de atender a grande parte das necessidades da era *Big Data*, e deve estar em constante atualização. O ecossistema de SGBD's *NoSQL* orientados a grafos é um dos mais ricos em diversidade, trazendo vários tipos de grafos e modelos de dados baseados em grafos. A indústria de grafos vem crescendo rapidamente, com o advento das redes sociais e redes em geral. Dessa forma, o surgimento de novos SGBD's *NoSQL* orientados a grafos, ou multimodelos, que adotam como um de seus modelos o grafo, tem se tornado cada vez mais frequente.

Um exemplo de SGBD *NoSQL* orientado a grafo bastante utilizado pelas aplicações é o *Neo4j*, que fornece ferramentas de manipulação de dados em formato de grafos, com grande facilidade, enriquecendo a experiência do usuário.

A escolha de um SGBD *NoSQL* orientado a grafo para uma aplicação depende de vários fatores, que vão desde o modelo de dados ao volume desses dados. Por isso, os *benchmarks* para SGBD's *NoSQL* orientados a grafo tem papel fundamental na avaliação dos SGBDs e estão sendo frequentemente estudados.

No próximo Capítulo 4 são apresentados os trabalhos relacionados escolhidos de acordo com a revisão sistemática (Apêndice A) bem como o estado da arte em relação a *benchmarks* para bancos de dados orientados a grafos.

Capítulo 4

TRABALHOS RELACIONADOS

4.1 Considerações iniciais

Devido ao crescente e recente desenvolvimento de SGBDs *NoSQL* orientados a grafos, diferentes soluções para a implementação deste tipo de sistemas têm sido propostas, tornando indispensável a utilização de algum mecanismo que possa medir e comparar a eficiência destas soluções para auxiliar na condução de projetos de pesquisa futuros. Geralmente utiliza-se a técnica experimental de *benchmark* para a análise de desempenho de SGBDs.

A Tabela 4 resume o estado da arte de pesquisas que propõem *benchmarks* para SGBDs *NoSQL* de grafos de propriedade, RDF, além de esforços de *benchmarking* entre estes domínios.

Tabela 4 – Estado da arte dos *Benchmarks* para banco de dados orientados a grafos

Nome	Ano	Propri.	RDF	Descrição
GooDBye	2020	X		*Benchmark* de banco de dados de grafos desenvolvido para atender a requisitos específicos de uma empresa internacional de TI (MATYJASZCZYK et al., 2020).
LDBC SNB	2020	X	X	Estrutura em que diferentes tecnologias baseadas em grafos possam ser testadas e comparadas, visando direcionar a identificação dos gargalos dos sistemas e das funcionalidades necessárias, além de ajudar os pesquisadores a abrir novas fronteiras de pesquisa (ANGLES et al., 2020).

Continuação da Tabela 4

Nome	**Ano**	**Propri.**	**RDF**	**Descrição**
BenchHub	2018	X	X	BenchCouncil BenchHub é um sistema de gerenciamento de código que pode ser usado para hospedar código-fonte e gerenciar projetos. Todos os códigos-fonte dos *benchmarks* e desafios do BenchCouncil estão hospedados no BenchHub (GUO, 2018).
Graph Benchmark	2018	X	X	Avaliação experimental extensiva de sistemas de BD de grafos de última geração. Fornecido uma metodologia de avaliação sistemática e de princípios com base em marcas de microbenchmark. Materializado em um conjunto de avaliação, projetado com extensibilidade em mente e contendo conjuntos de dados, consultas e *scripts* (LISSANDRINI et al., 2018).
Smart City	2018		X	O modelo de avaliação proposto permite entender se uma loja RDF pode ser usada com lucro como base para a modelagem e aplicações de cidades inteligentes. (BELLINI; NESI, 2018).
Train Benchmark	2018	X	X	Um macrobenchmark entre tecnologias que visa medir o desempenho da validação contínua de modelos com grafos e restrições capturadas como consultas. (SZÁRNYAS et al., 2018).
HOBBIT	2016		X	A plataforma serve como uma estrutura para comparar sistemas de *Big Linked Data*. Os *benchmarks* que se concentram na avaliação da qualidade de um sistema usando solicitações únicas e consecutivas podem ser executados na plataforma, bem como os *benchmarks* com o objetivo de eficiência, por exemplo, gerando muitas solicitações paralelas que levam a uma alta carga de trabalho (RÖDER et al., 2019).
IGUANA	2016		X	É uma estrutura genérica de execução de *benchmark* SPARQL focada na análise de SGBD's RDF e consultas federadas (CONRADS et al., 2017)

Continuação da Tabela 4

Nome	**Ano**	**Propri.**	**RDF**	**Descrição**
LITMUS	2016	X	X	Uma estrutura aberta extensível que permite aos usuários comparar o SGBD para RDF e modelos de dados grafos de propriedade em um determinado conjunto de dados e carga de trabalho de consultas. Promove a interoperabilidade, a reutilização e a replicabilidade dos *benchmarks* existentes por meio da visualização dos resultados. (THAKKAR et al., 2017).
TGDB	2016	X		Um projeto genérico de um *benchmark* de grafos que consiste em consultas e nos conjuntos de dados que atendem aos critérios: trabalhar com todos os tipos de grafos; a carga de trabalho deve incluir consultas de uma ampla gama de operações de grafos para avaliar o desempenho de diferentes componentes (ABUL-BASHER et al., 2016).
Cyclone	2016	X		*Cyclone Benchmark* possui três recursos principais. Primeiro, o *benchmark* tem dois modelos de dados grafos. Segundo, o *benchmark* vem com um programa para gerar grafos de dados sintéticos. Terceiro, o *benchmark* possui vários tipos principais de operações CRUD (Create, Read, Update e Delete), comuns em bancos de dados orientados a grafos (TANG, 2016).
FEASIBLE	2015		X	Estrutura de *benchmarking* SPARQL baseada em recursos (orientada a dados e estrutural) para BDs de grafos de RDF. Emprega uma abordagem automática para a geração de *benchmarks* usando *logs* de consulta (SALEEM et al., 2015).
WatDiv	2014		X	O *Waterloo SPARQL Diversity TEST Suite* (WatDiv) avalia os bancos de dados RDF usando seus geradores de dados e consultas para analisar a correlação entre o desempenho do DMS e as variáveis estruturas e complexidades de consulta (tipologia de consulta) (ALUÇ et al., 2014).

Continuação da Tabela 4

Nome	**Ano**	**Propri.**	**RDF**	**Descrição**
Graphium	2013	X	X	É um comparativo que inclui a visualização de resultados, comparando lojas RDF com lojas *Graph* (ou seja, Neo4J, Sparksee / DEX, HypergraphDB, RDF-3X) em conjuntos de dados de grafos personalizados, incluindo uma tripla de 10M no conjunto de dados (usando o gerador de dados BSBM) (FLORES et al., 2013).
Quertzal	2013		X	Estrutura de *benchmarking* RDF e Graph DMS que oferece uma nova tradução de SPARQL para SQL por meio de um mecanismo para vários *back-ends*. Sua versão atual provê suporte para *benchmarking* DB2, PostgreSQL e Jena Apache. Oferece carregamentos de consulta personalizados para os conjuntos de dados DBpedia (real) e LUBM (sintético) (BORNEA et al., 2013).
XGDBench	2012	X		Plataforma de *benchmarking* de bancos de dados de grafos para sistemas de computação em nuvem, que é uma extensão do famoso Yahoo! Referência de serviço em nuvem para o domínio do grafo. Os autores compararam os sistemas de BD de grafos AllegroGraph, Fuseki, Neo4j e OrientDB usando XGDBench no ambiente em nuvem Tsubame 2.0 HPC. (DAYARATHNA; SUZUMURA, 2012).
DBPSB	2011		X	O *DBpedia SPARQL Benchmark* (DBPSB) avalia RDMS DMSs usando o DBpedia, criando uma carga de trabalho derivada dos logs de consulta do DBpedia (MORSEY et al., 2011).
Graph 500	2010	X		é uma referência para sistemas de supercomputação intensiva em dados e suas aplicações. Não considera comparando bancos de dados grafos típicos (MURPHY et al., 2010).

Continuação da Tabela 4

Nome	**Ano**	**Propri.**	**RDF**	**Descrição**
BSBM	2009	X		O *Berlin SPARQL Benchmark* (BSBM) é um cenário de caso de uso de comércio eletrônico sintético baseado em dados para *benchmarking* RDF e DMS relacionais. Ele fornece geradores personalizados para criar conjuntos de dados e consultas de tamanho e tipologia personalizados (BIZER; SCHULTZ, 2009).
HPC -SGAB	2009	X		Esse *benchmark* foi desenvolvido por pesquisadores da academia, além de membros de várias empresas industriais para capturar as operações de grafos mais representativas, como carregamento de grafos, navegações curtas ou percursos em grafos completos (DOMINGUEZ-SAL et al., 2010).
SP2Bench	2009		X	É um dos *benchmarks* RDF DMS sintéticos baseados em dados mais comumente usados, que usa o esquema do conjunto de dados bibliográficos DBLP para gerar conjuntos de dados de tamanho personalizado (SCHMIDT et al., 2009).
LUBM	2005		X	O *benchmark* da Universidade de Lehigh (LUBM) avalia BDs de grafos de RDF em um grande conjunto de dados sintéticos que está em conformidade com uma ontologia de domínio universitário (GUO et al., 2005).

Fim da Tabela

Fonte: Autoria própria

Foram encontrados 22 *benchmarks* para SGBD's orientados a grafos na literatura, dos quais 9 sao voltados à grafos de triplas/RDF, 7 para grafos de propriedades e 6 para ambos os modelos de dados grafos. O LDBC-SNB é o mais completo dos *benchmarks* estudados, tanto se tratando das cargas de trabalho, quanto das medidas de desempenho e várias outras características como suporte, *driver* de teste, domínio da aplicação,infraestrutura, e etc., seguido do *Cyclone Benchmark* que complementa o LDBC-SNB nas cargas de trabalho que este não apresenta.

Dominguez-Sal et al. (2011) faz uma discussão sobre os principais aspectos a serem considerados em um *benchmark* para SGBD's *NoSQL* orientados a grafos. Como mencionado na Seção 3.2, existem diferentes modelos de bancos de dados orientados a grafos, levantando um

grau diferente de complexidade para cada um. Dessa forma, as características do grafo a serem aplicadas no processo de *benchmark* é de extrema importância. Assim, Dominguez-Sal et al. (2011) definem três pontos que um *benchmark* para SGBDs orientados a grafos deve analisar e, portanto, serem incluídos no processo:

1. Grafos rotulados, permitindo identificar os nós e as arestas do grafo;
2. Grafos direcionados, onde são definidos o relacionamento e à qual este aponta;
3. Suporte a atributos, empregado por aplicativos para definir o peso das arestas.

Quanto às operações, é possível obter um rico conjunto de operações específicas por meio de operações genéricas compartilhadas por todos os contextos. Ainda que não seja necessário considerar todas as operações em um *benchmark*, operações analíticas e de transformação devem fazer parte do conjunto de operações.

Para as consultas, é destacado que, por essas representarem a carga de trabalho de um ambiente real em que o *benchmark* será executado, então, precisam ser adaptadas ao perfil específico do sistema a ser testado.

No que se diz respeito às medidas de desempenho, os autores definem algumas medidas gerais a serem aplicadas em um *benchmark* para bancos de dados orientados a grafos:

(A) o tempo de carregamento, que mede o tempo decorrido para carregar e pré-processar o conjunto de dados;

(B) o tamanho do grafo;

(C) tempo de resposta da consulta, que contabiliza o tempo decorrido entre a emissão da consulta até a saída dos resultados;

(D) taxa de transferência, que mede o número de consultas concluídas em um intervalo de tempo;

(E) o preço do ambiente computação, incluindo custos de *hardware*, licença e manutenção, se aplicável;

(F) o consumo de energia, que mede os requisitos de energia do equipamento de computação e fornece uma medida indireta de seus requisitos de refrigeração.

São muitas as configurações e medidas disponíveis que podem ser designadas para um *benchmark* para SGBDs orientados a grafos. O que vai definir a seleção final são as características do *benchmark*, dos objetivos e da aplicação alvo.

4.2 LDBC Social Network Benchmark

O *Linked Data Benchmark Council* (LDBC) é um projeto formado por academia e indústria, inspirado pelo impacto dos *Benchmarks* do *Transaction Processing Performance Council* (TPC), o qual reúne diversos atores no campo do gerenciamento de dados semelhantes a grafos. Com o objetivo de desenvolver *benchmarks* de código aberto, com os quais diferentes tecnologias baseadas em grafos possam ser testadas e comparadas, levando ao direcionamento na identificação de gargalos dos sistemas e ainda contribuir com os novos rumos para a pesquisa dentro do estado da arte (ANGLES et al., 2020; BONCZ et al., 2013).

O projeto disponibiliza três *benchmarks* distintos: *1)* LDBC *Graphalytics*, um *benchamrk* de nível industrial, utilizado para medir operações de análise de grafos (IOSUP et al., 2016). *2)* LDBC *Semantic Publishing Benchmark* (SPB), voltado ao teste de desempenho de mecanismos RDF, motivados pela indústria de mídia/publicação (KOTSEV et al., 2016). *3)* LDBC *Social Network Benchmark*, que tem por objetivo testar SGBDs orientados a grafos que combinam atualização transacional com recursos de consulta(ANGLES et al., 2020). Este último, é um trabalho relacionado a esta pesquisa de Mestrado.

Como as redes sociais são ocorrências significativas para o uso de grafos, o LDBC SNB foi desenvolvido com o intuito de ser o mais representativo de todos os aspectos das operações de uma rede social. Desta forma, seu conjunto de dados corresponde a uma rede de amizade que conecta pessoas, sendo que grande parte dos dados está contida nas mensagens postadas em fóruns de discussão (ERLING et al., 2015), como apresentado no esquema da Figura 14.

O LDBC SNB apresenta duas categorias de cargas de trabalho, a *Interactive Workload*, representada por consultas interativas do tipo transacional. Outra categoria é a *Businees Inteligence Workload*, voltada à questões críticas de negócios, engloba consultas analíticas que respondam a essas questões. Além disso, possui uma carga de trabalho de análise de grafos, atribuída ao *Graphalytics* (ANGLES et al., 2020), que em combinação com o LDBC SNB abrange uma variedade de sistemas com características e domínios distintos (ERLING et al., 2015).

O LDBC SNB disponibiliza um gerador de dados sintéticos escalável e baseado no paradigma *MapReduce*, denominado *DATAGEN*, capaz de gerar grafos de tamanho variável, seguindo o fator de escala de entrada (0.1, 0.3, 1, 3, 10, 30, 100, 300, 1 000), equivalente ao volume de dados em GB, armazenados em arquivos com formato *.csv*. Com *design* baseado em ponto de estrangulamento, o *DATAGEN* possibilita a mineração de dados que permite encontrar parâmetros de substituição com desempenho equivalente. Alguns exemplos de pontos de estrangulamento são: **Pedidos interessantes**, testa a capacidade do otimizador de consulta de explorar as ordens interessantes induzidas por alguns operadores. **Projeção atrasada**, testa a capacidade do otimizador de consulta de atrasar a projeção de atributos desnecessários até o final da execução. Dentre vários outros pontos de estrangulamento que são discutidos por Erling et al. (2015).

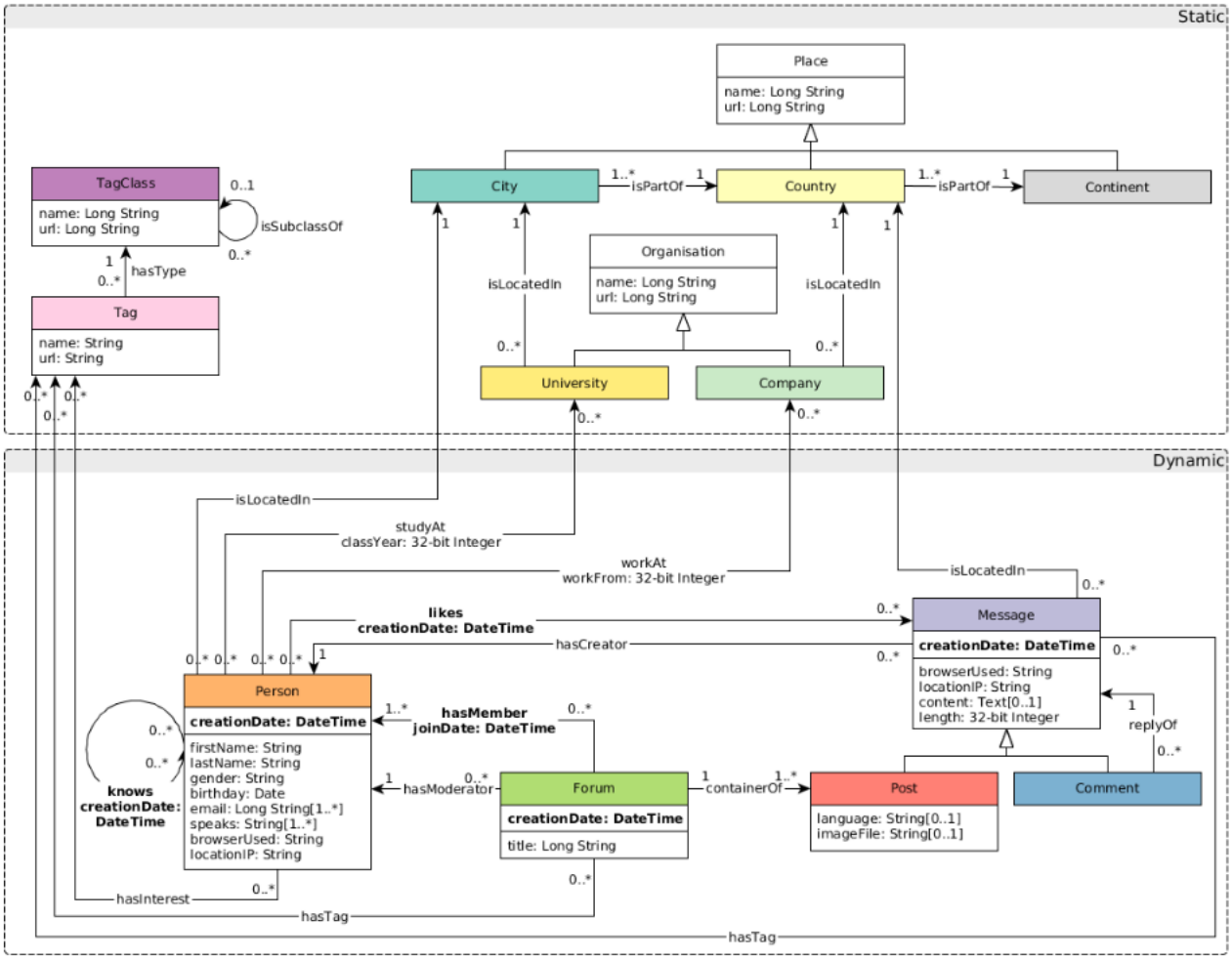


Figura 14 – Esquema de dados LDBC SNB

Fonte: Retirado de Angles et al. (2020)

O esquema do LDBC SNB é composto por 14 entidades diferentes, dentre as quais estão *City, University, Person* e *Forum*, juntamente com seus atributos e relações, representando uma rede social e suas características durante um determinado período de tempo. Ou seja, reproduz a interação das pessoas, mediante as relações de amizade que possuem, além da troca de mensagens e *likes*.

A *Interactive Workload* possui três categorias de consultas, *a)* Consultas complexas somente leitura, com um conjunto de 14 consultas somente leitura relativamente complexas, que abrangem uma quantidade considerável de dados, geralmente a vizinhança de amizade em duas etapas e as mensagens relacionadas a ela. *b)* Consultas curtas somente leitura, um conjunto de 7 consultas que geralmente exploram apenas uma entidade (por exemplo, uma pessoa). E por último *c)* Consultas de atualização transacional, inserindo um único nó de um certo tipo, junto com seus relacionamentos para outros nós existentes ou um único relacionamento de um certo tipo entre dois nós existentes, esse conjunto contém 8 consultas. A combinação das consultas é

feita pelo chamado Mix de Consultas, que compreende a declaração da quantidade de ocorrências de cada tipo de consulta, com o intuito de tornar a combinação o mais realista possível.

4.3 HPC Scalable Graph Analysis

Desenvolvido por membros de diversas empresas industriais em parceria com pesquisadores da academia, o *HPC Scalable Graph Analysis* (HPC-SGAB) é um *benchmark* que tem por objetivo a análise de estruturas de dados em formato de grafos dirigidos e ponderados Bader et al. (2006). Dividido em 4 *kernels*, sendo um *kernel* a operação da qual o desempenho é testado. O primeiro *kernel* é responsável pela construção do grafo (esparso), e os 3 restantes são encarregados por realizar operações sobre o grafo, sendo definidos da seguinte maneira:

1. *Kernel 1* - Construção de Grafo: Neste primeiro *kernel* é feita a leitura do arquivo do banco de dados, no formato definido e os dados são carregados, e assim realizada a medida do desempenho da inserção de nós e arestas. O grafo gerado nessa etapa é utilizado em todos os *kernels* seguintes, e não pode ser modificado por eles.

2. *Kernel 2* - Classificar grandes conjuntos: O segundo *kernel* busca um conjunto de arestas que atendam a uma condição, assim sendo, obtém as arestas de maior peso. Desta forma, mede o tempo gasto para encontrar o conjunto de arestas. Como saída, ele retorna uma lista de arestas e nós que se conectam, armazenando essa lista para que seja utilizada no *Kernel 3*.

3. *Kernel 3* - Extração de Grafo: Este *kernel* constrói subgrafos na vizinhança de um nó, medindo assim o tempo gasto para essa construção. Realizando, a operação de *K-hops*, por intermédio da implementação do percurso de pesquisa em largura, usando as arestas produzidas no *kernel* anterior.

4. *kernel 4* - Algoritmo de análise de grafo: O último *kernel* tem por objetivo identificar o conjunto de nós no grafo com maior centralidade de intermediação, que se baseia na enumeração de caminhos mais curtos. É responsável por aferir o desempenho de travessias em todo o grafo. A medida é realizada pelo *Travessed Edges Per Second* (TEPS), uma nova métrica introduzida pelos autores, que mede a taxa de arestas atravessadas por segundo.

A geração dos dados é feita por meio de um gerador de dados escalável, baseado no algoritmo de geração de grafo de matriz recursiva chamado MATrix (R-MAT) que emprega uma estrutura de dados do tipo matriz de adjacência (DOMINGUEZ-SAL et al., 2010), operando recursivamente sobre essa matriz (CHAKRABARTI et al., 2004).

Os grafos são gerados no primeiro *kernel*, com base em uma lista de tuplas predefinidas, onde cada tupla inclui orientações de nó inicial e nó final, direcionando a aresta que os relacionam, e ainda o peso que corresponde aos dados atribuídos à aresta (BADER et al., 2009).

4.4 XGDBench

O *XGDBench* é uma estrutura de *benchmarking* de banco de dados grafos extensível e distribuída, sendo uma extensão do *Yahoo! Cloud Serving Benchmark* (YCSB), baseada no modelo de grafos de propriedades, criada para avaliar sistemas *cloud Exascale* e escrita em X10, uma linguagem destinada a programação de sistemas HPC. Utiliza o modelo de criação de grafo sintético para grafos de atributos (propriedades), o *Multiplicative Atribute Graph* (MAG). O *XGDBench* tem por objetivo ser um *benchmark* que modele cenários de aplicativos de grafos escaláveis e que sejam realistas, com foco em serviços de redes sociais (DAYARATHNA; SUZUMURA, 2012).

O MAG modela estruturas de redes que contenham propriedades nos nós, assim como as relações da estrutura de rede com as propriedades dos nós. Os grafos de propriedades gerados pelo MAG são extremamente realistas, o que os tornam apropriados para o *benchmarking* de SGBDs *NoSQL* orientados a grafos (DAYARATHNA; SUZUMURA, 2014). O MAG é capaz de armazenar com facilidade as relações existentes entre a estrutura da rede e as propriedades do nó, permitindo uma combinação de nós, o que resulta na origem de *links* entre esses nós (KIM; LESKOVEC, 2011).

A arquitetura do *XGDBench* possui, como componetes fundamentais, o *Graph Generator, Graph Data Structure, Workload Executor, Graph DB Workload* e *Graph DB Interface Layer*, como mostra a Figura 15.

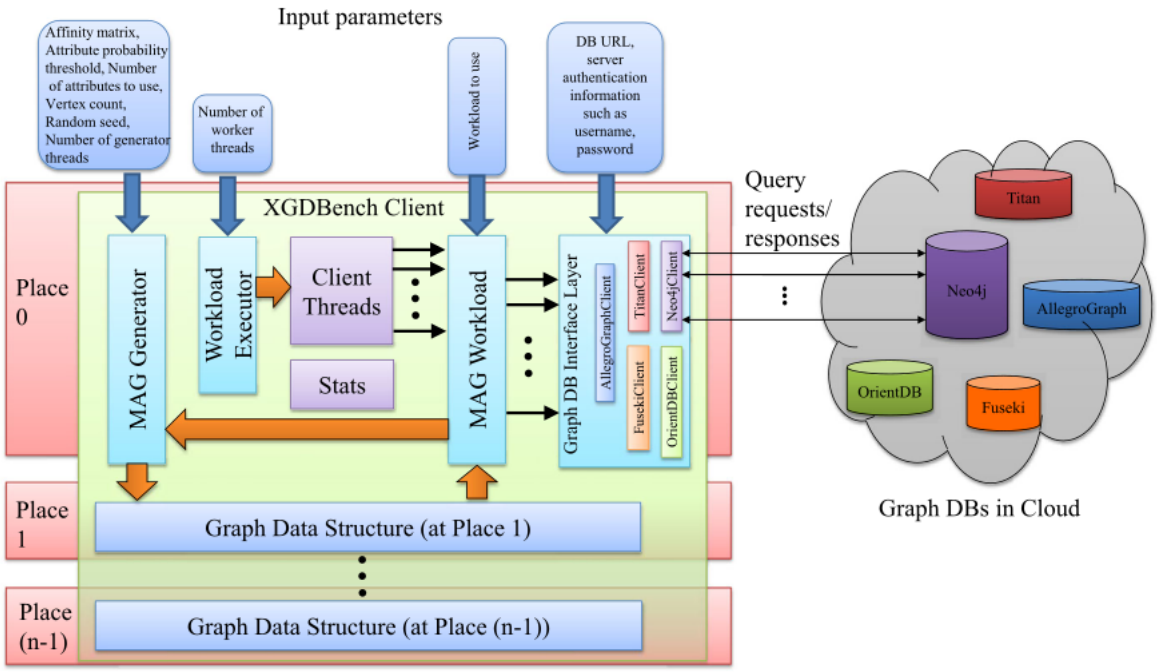


Figura 15 – Arquitetura XGDBench

Fonte: Retirado de Dayarathna e Suzumura (2014)

Tabela 5 – Operações Básicas

Operação	**Descrição**
Leitura	Ler um nó e suas propriedades
Inserção	Insere um novo nó
Atualização	Atualizar todos os atributos de um nó
Exclusão	Excluir um nó do banco de dados
Varredura	Carrega a lista de vizinhos de um nó
Travessia	Percorre o grafo de um determinado nó usando BFS. Isso representa encontrar amigos de uma pessoa nas redes sociais.

Fonte: Adaptado de Dayarathna e Suzumura (2012)

O *XGDBench Client* é o *software* responsável por executar as cargas de trabalho do *benchmark*, gerando os dados a serem carregados, por meio do modelo MAG, apresentado previamente. A execução do *XGDBench* é dividida em duas fases: Carregamento e Transações. A fase de carregamento é onde o grafo de propriedades é gerado, utilizando o modelo MAG e carregado no SGBD. Enquanto na fase de transação, o método *Transaction()* é invocado, carregando assim as operações básicas (leitura, atualização, inerção, varredura e travessia, definidas na Tabela 5, de acordo com Dayarathna e Suzumura (2012).

As cargas de trabalho do *XGDBench* são uma mistura, de acordo com algumas proporções predefinidas, das operações básicas citadas na Tabela 5, sendo classificadas em 5 grupos:

(A) *Atualização pesada:* A carga de trabalho A é uma mistura das cargas de trabalho de leitura e atualização, em uma proporção de 50/50. As operações de leitura consultam um vértice V e leem todos os atributos de V. A operação de atualização altera o último atributo de tempo de *login* dos nós. Os atributos relacionados à relação do nó não são alterados.

(B) *Leia principalmente:* Uma mistura das cargas de trabalho de leitura e atualização 95/5. As operações de leitura e atualização são semelhantes a A.

(C) *Somente leitura:* Corresponde em 100% de operações de leitura. As operações de leitura são semelhantes a A.

(D) *Leia as últimas:* Esta carga de trabalho insere novos nós no grafo. As inserções são feitas de forma que as relações de potência do grafo original sejam preservadas.

(E) *Curtas distâncias:* Esta carga de trabalho lê todos os nós vizinhos e seus atributos de um nó A. Isso representa o cenário de carregamento da pessoa A mais amigável em uma rede.

O *XGDBench* utiliza uma metodologia para implementar a geração de picos de carga de trabalho por meio de uma técnica de emulação de picos de carga de trabalho baseada em fator de multiplicação. A emulação é feita alterando o número de *threads* de cliente dinamicamente, dessa forma gerando um pico de carga de trabalho durante uma sessão de *benchmarking*.

4.5 TGDB

Uma proposta feita por pesquisadores da Universidade de Toronto, o *Toronto Graph Database Benchmark* (TGDB) é um *benchmark* desenvolvido para avaliar o desempenho de SGBDs orientados a grafos, por meio de cargas de trabalho e conjuntos de dados do mundo real. O TGDB foi projetado para lidar com todo tipo de grafo em aplicativos do mundo real, ponderado ou não ponderado, direcionado ou não direcionado, grafos de propriedades, etc (ABUL-BASHER et al., 2016).

As cargas de trabalho do *benchmark* abrangem um extenso conjunto de operações sobre grafos, que possibilitam avaliar o desempenho de diversos elementos do SGBD. Divididas nas categorias Cargas de trabalho de Carregamento de Dados, Carga de Trabalho Analítica, contendo seis subconjuntos de operações analíticas sobre os dados (Q3, Q4, Q5, Q6, Q7 e Q8), e Carga de Trabalho Atômica, com dois subconjuntos de operações atômicas sobre os dados (Q1 e Q2), como apresenta a Figura 16.

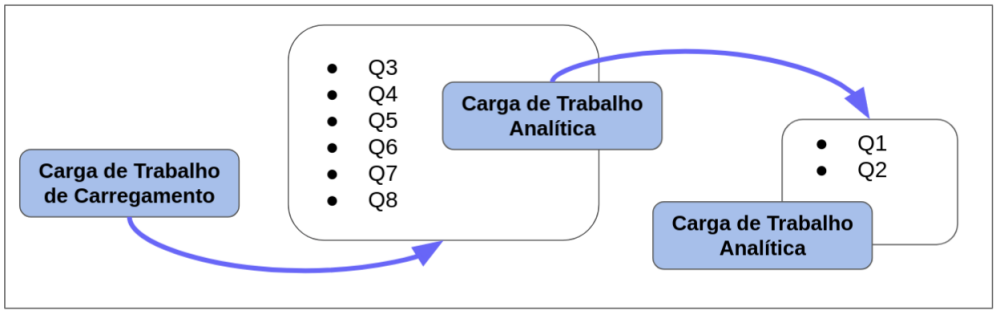


Figura 16 – Cargas de trabalho TGDB

Fonte: Adaptado de Abul-Basher et al. (2016)

O TGDB utiliza dois conjuntos de dados reais, *Wiki-Talk* e *Slashdot*, obtidos mediante o repositório *SNAP*[1]. A implementação das consultas é feita utilizando uma API *Java* para bancos de dados orientados a grafos, denominada *Blueprints*, recorrendo aos seus métodos para lidar com inserção e remoção de nós e arestas, métodos para recuperação de nós e arestas por meio de um ID ou um valor de atributo de um grafo, dentre vários outros que são capazes de suportar operações sobre grafos.

4.6 Cyclone Benchmark

Como resultado de uma extensa pesquisa na literatura sobre *benchmarks* voltados ao desempenho de SGBDs orientados a grafos e suas diferentes particularidades, Bonifati et al. (2018) propuseram o *Cyclone Benchmark*, visando suprir as lacunas encontradas na literatura

[1] http://snap.stanford.edu/data/

sobre o tema. O *Cyclone Benchamrk* oferece dois modelos de dados orientados a grafos, o primeiro possuindo apenas um tipo de nó para todos os nós do grafo e cinco tipos diferentes de arestas, como mostra a Figura 17.

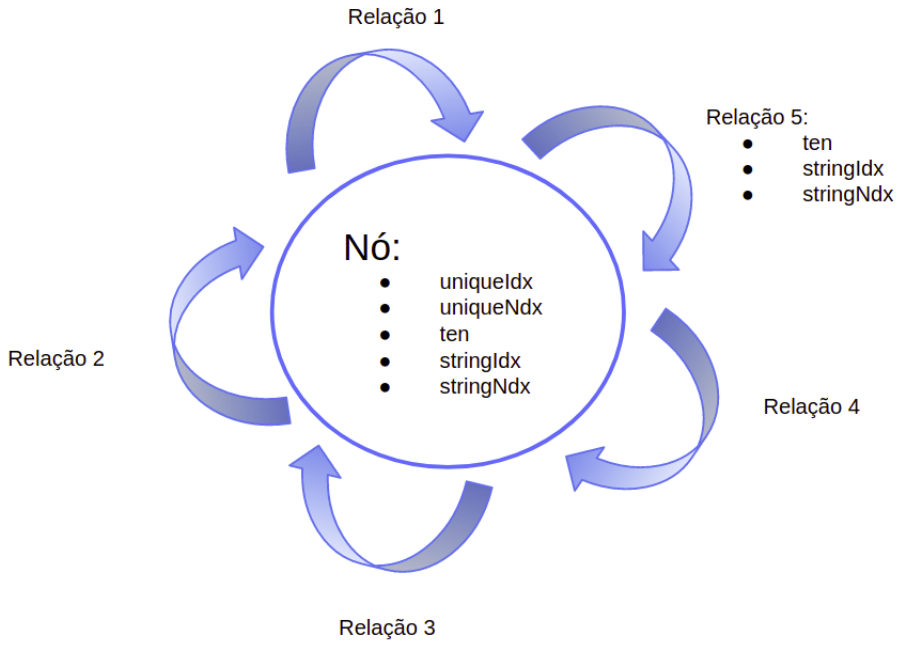


Figura 17 – Modelo com apenas um tipo de nó

Fonte: Adaptado de Bonifati et al. (2018)

Esse modelo permite a construção de duas estruturas diferentes, sendo uma de grafos aleatórios e a outra de grafos *Kronecker*. Esta última estrutura simula graus de distribuição dos nós da lei de potência em redes do mundo real. O modelo apresenta sete tipos de operações CRUD como carga de trabalho, sendo elas: Inserção em massa, exclusão em massa, seleção em massa com fator de seletividade, seleção em massa com vários tipos de arestas e consultas à estrutura do grafo.

Diferentemente do modelo com apenas um tipo de nó, o segundo modelo de grafos apresenta sete tipos diferentes de nós e dez tipos de arestas, conforme apresentando na Figura 18. Esse modelo emprega a distribuição de grau de nós de *Zipf*, utilizando um parâmetro ajustável, o que torna o modelo flexível. Essa distribuição calcula o número de nós que deve ser distribuído para cada tipo de nó.

Além dos dois modelos de dados, o *Cyclone Benchmark* é composto por um gerador de dados sintéticos, que é responsável por gerar os dados das propriedades de nós e arestas. Os nós possuem sete propriedades, enquanto as arestas possuem três propriedades, como pode ser visto na Figura 18.

As cargas de trabalho para esse segundo modelo de dados foram planejadas de modo que abrangessem uma gama de consultas complexas para alcançar um número considerável de nós

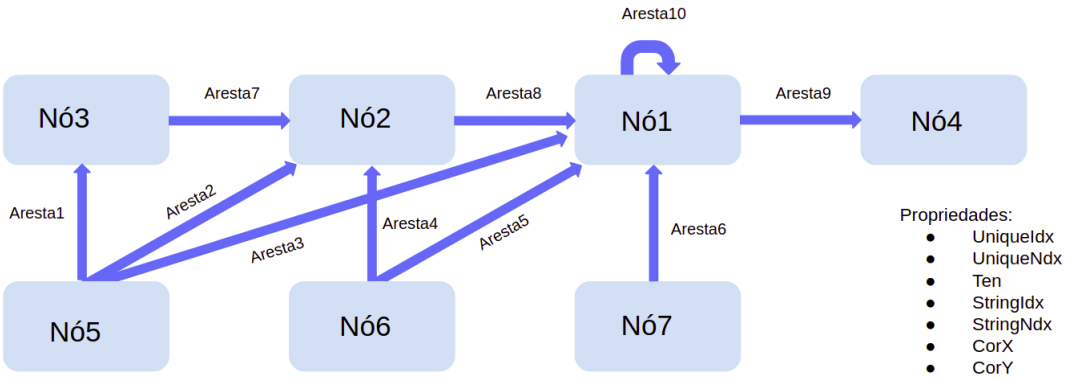


Figura 18 – Modelo com vários tipos de Nós

Fonte: Adaptado de Bonifati et al. (2018)

dentro do grafo. Este conjunto é denominado Consultas Complexas Interativas, e totalizam sete tipos de consultas (Q1, Q2,Q3, Q4, Q5, Q6 e Q7).

Uma das principais características do *Cyclone Benchmark* é que ele permite a análise do impacto da indexação em uma propriedade e também na correlação entre valores de propriedades. Dessa forma, permite comparar o efeito da indexação versus a não indexação de diferentes tipos de dados.

4.6.1 Propostas para comparar diferentes bancos de dados orientados a grafos

Kovács et al. (2019) compara os SGBDs orientados a grafos *Neo4j*, *Blazegraph* e *JanusGraph* utilizando dados reais da base de conhecimento *Wikidata*, com o objetivo de determinar as características de desempenho da utilização de SGBDs orientados a grafos em vários ambientes complexos. Foi necessário definir um fluxo de medição unificado para os sistemas. Executou-se os mesmos conjuntos de consultas, sendo estas geradas aleatoriamente. Os resultados mostraram que o *Neo4j* foi o sistema com pior desempenho, onde todas as consultas foram forçadas a encerrar, devido ao fim do prazo de execução estipulado.

Em seu trabalho Pacaci et al. (2017) apresenta uma arquitetura de *benchmark* baseada no LDBC, como foco em sistemas com cargas de trabalho transacionais interativas que possam representar, de melhor forma, a natureza em tempo real dos aplicativos de rede social. Dessa forma, o objetivo é comparar SGBDs orientados a grafos em cenários semelhantes a redes sociais para processamento de transações em tempo real. Os SGBDs escolhidos para o experimento foram *Neo4j* e *Titan*. Os resultados apresentados concluem que as linguagens de consulta do *Titan* ainda não estão preparadas para implantações em tempo real. Enquanto o *Neo4j*, com seu armazenamento nativo de grafos, obteve uma maior taxa de transferência.

Lissandrini et al. (2017) uilizou de uma metodologia de avaliação baseada em *microben-*

chmark com 35 operações diferentes, com o objetivo de avaliar os principais SGBDs orientados a grafos (*ArangoDB, BlazeGraph, Neo4j, OrientDB, Sparksee, Titan*). A metodologia utiliza dados reais e sintéticos em diferentes escalas, distribuição e complexidade. Ainda apresenta uma interface padrão, o que permite usar os mesmos conjuntos de operações para testar cada sistema. Os testes mostraram que, no geral, o *Neo4j* é o sistema com melhor desempenho, seguido do *OrientDB*, enquanto o *BlazeGraph* obteve o pior desempenho.

Navolskyi (2018) propôs uma versão estendida do *Yahoo! O Cloud Service Benchmark* (YCSB) para analisar o uso de SGBDs orientados a grafos com cargas de trabalho referentes a cenários industriais. O YCSB é um *benchmark* que não foi propriamente projetado para SGBDsde grafos e sim para armazenamento de chave/valor e nuvem. Porém sua arquitetura cliente é extensível para permitir suporte a diferentes cargas de trabalho, assim bancos de dados com qualquer estrutura podem ser integrados. Para a realização do experimento foram escolhidos os SGBDs*JenaApache, Neo4j, OrientDB* e *Sparksee*. O objetivo do experimento foi descobrir se os SGBDs orientados a grafos são adequados para uso em internet das coisas no domínio industrial. De acordo com os resultados obtidos a maioria dos sistemas testados não são adequados para utilização no domínio especificado. Dentre os sistemas, apenas o *Sparksee* conseguiu atingir a taxa de transferência designada para operações de inserção. Enquanto o *Neo4j* e *Jena Apache* não chegaram nem perto de armazenar a quantidade de dados no tempo estipulado, o *OrientDB* produziu um desempenho pouco abaixo do necessário.

No trabalho de Fernandes e Bernardino (2018) foi feita uma análise dos SGBDs orientados a grafos *AllegroGraph, ArangoDB, InfinitGraph, Neo4j* e *OrientDB* com o objetivo de comparar os aspectos de esquema flexível, linguagem de consulta, *sharding* e escalabilidade, as vantagens e casos de uso. Mesmo tendo os mesmos objetivos, tais como armazenamento de grandes volumes de dados com inúmeras conexões, os autores destacam que os sistemas testados são muito diferentes em termos de funcionalidades. Desta forma, a escolha de um desses sistemas irá depender de necessidades especificas para a implantação. Em conclusão os autores relatam que o *Neo4j* e o *ArangoDB*, quando se trata da implementação de bancos de dados grafos, oferecem funcionalidades superiores aos outros sistemas.

Rusu e Huang (2019) implementa o *benchmark* LDBC SNB nos sistemas *Neo4j* e *TigerGraph*. 46 consultas foram executadas em quatro fatores de escala e três arquiteturas de computação. Foram avaliados o desempenho das consultas, tempo de carregamento e o tamanho do armazenamento. De acordo com os resultados, os autores destacam que o *TigerGraph* apresentou melhor desempenho que o *Neo4j* em 95% das consultas, principalmente com o aumento do volume de dados. Porém, quanto ao carregamento dos dados grafos, o *Neo4j* tem um melhor desempenho.

4.6.2 Propostas para analisar um único SGBD orientado a grafos

Guia et al. (2017) escolheu analisar apenas um SGBD orientado a grafos, o *Neo4j*. Para isso foi utilizado o *Social Network Benchmark* do LDBC, com objetivo de expor as principais vantagens e características do sistema. Os conjuntos de dados foram gerados pelo *Datagen*, gerador de dados sintéticos do LDBC SNB, com diferentes tamanhos, possibilitando o teste de diferentes tempos de carregamento e execução das consultas. De acordo com o resultado dos testes, ficou evidente que o *Neo4j* é um sistema apropriado para trabalhar com tamanhos diferentes de bancos de dados grafos, sendo este, considerado pelo autor uma ferramenta poderosa.

Em Candell et al. (2020), Célula de trabalho é definida como: *"Sistemas de controle em rede altamente interconectados, nos quais vários dispositivos interagem e colaboram para realizar ordens de produção complexas e adaptáveis"*. Para avaliar esse tipo de sistema é necessário muito mais que a análise de desempenho da rede, mas também o impacto das redes nos sistemas de manufatura. Neste cotexto, o autor propõe uma abordagem para análise do desempenho das células de trabalho, sendo o *Neo4j* o SGBD orientado a grafos escolhido para tal análise. De acordo com os resultados dos testes, o autor conclui que o uso de banco de dados grafos é adequado para a aplicação em casos de células de trabalho. O *Neo4j* se mostrou eficiente na construção do banco de dados, e também no cálculo das medidas de desempenho da rede. Erdemir et al. (2019) se concentra na comparação do desempenho de 12 consultas diferentes no SGBDs orientado grafo *Neo4j*, usando modelos diferentes de grafos, sendo grafos e hiper-grafos. Para o armazenamento, o modelo de hiper-grafo obteve um custo mais alto, por conta dos nós complementares, necessários para a modelagem das hiper-arestas. Porém, esses mesmos nós extras levam o modelo a uma vantagem nas consultas que abrangem diversos tipos de nós, onde o tempo de execução é reduzido pela metade. O modelo de grafos se destacam consideravelmente em alguns tipos de consulta, tal como travessia.

A proposta de Wiese et al. (2018) é de uma integração de diversos conjuntos de dados *(FANTOM, Navegador do genoma UCSC e Servidor de Análise PC-TraFF)* de componentes genômicos em um SGBD orientado a grafos, o *Neo4j*. A partir dessa integração, um *benchmark* foi executado, para análise e visualização de uma rede reguladora de genoma, com conjuntos de dados pequenos e grandes, executando as mesmas consultas em todos os conjuntos. Foi possível verificar a escalabilidade de execução de consultas em uma carga de trabalho, usando tanto conjuntos de dados pequenos quanto grandes, e ainda analisar o impacto causado no desempenho pelo aquecimento da memória *cache*.

4.7 Considerações Finais

Com a importância da era *Big Data*, não faltaram esforços para avaliar as tecnologias emergentes de SGBDs *NoSQL* orientados a grafos. Este capítulo abordou alguns desses esforços

considerando principalmente os mais recentes, ou mais relacionados com o presente tema de pesquisa.

É importante destacar o trabalho de Juričić e Radošević (2020), que apresenta uma revisão de *Benchmarking* de SGBDs orientados a grafos e a caracterização das formas que devem ser consideradas durante o *benchmarking*. O trabalho descreve uma comparação de seis *benchmarks* para SGBDs orientados a grafos, com ênfase em consultas e estrutura de dados. Comparação esta, que se relaciona com a proposta deste projeto de pesquisa de mestrado, que visa a análise comparativa de *benchmarks* para SGBDs orientados a grafos de propriedade. Dos seis *benchmarks* comparados pelos autores, três estão sendo analisados neste projeto de pesquisa de mestrado, sendo eles o LDBC SNB, XGDBench e o HPC-SGAB. O presente trabalho de pesquisa de Mestrado traz uma gama de operações e consultas que não foram destacadas por Juričić e Radošević (2020), além de outros aspectos considerados importantes na escolha de um *benchmark*. Os autores replicaram o trabalho de Tang (2016), destacando alguns recursos comuns encontrados em grafos de grandes volumes de dados, e também realizando um levantamento dos *benchmarks* populares para cada tipo de SGBD's (Relacional, XML, Orientado a Objetos e Grafos).

Bonifati et al. (2018) em seu trabalho realizou um levantamento dos principais benchmarks para SGBS's *NoSQl* orientados a grafos, separados por tipos de grafos. Sendo eles: grafos RDF, grafos de propriedades e sistemas de processamento de grafo para sistemas paralelos e distribuídos. Os autores realizam uma análise da maturidade histórica dos *benchmarks* de cada tipo de grafo, além da diversidade de arquiteturas de sistemas que aparecem em cada área. A análise comparativa realizada teve como foco características tais como: tipos de entrada e saída, tipo de carga de trabalho que os *benchmarks* consideram, modelos ou estruturas de dados que os *benchmarks* atendem, linguagem de consulta que são utilizadas e pontos de estrangulamento. Porém, os autores não consideram as medidas de desempenho e as diversas características que são importantes levar em consideração ao se escolher um *benchmark*, tais como domínio da aplicação, volume de dados, e várias outras, como discutido no Capítulo 5. Dentre todos os tipos de *benchmarks* comparados por Bonifati et al. (2018), destacam-se aqueles para bancos de dados *NoSQL* orientados a grafos de propriedades, por serem o tipo de *benchmark* estudados no presente trabalho de pesquisa de Mestrado, sendo *LDBC SNB*, *WatDiv*, *gMark* e *GSCALER*. Destes, apenas o *LDBC SNB* está sendo tratado nesta pesquisa de Mestrado.

Tang (2016) realiza uma revisão na literatura, apresentada na Seção 4.6, sobre alguns *benchmarks* para SGBD's *NoSQL* orientados a grafos. Sua pesquisa teve como objetivo conhecer as lacunas apresentadas pelo estado da arte, e a partir dessas, propor o desenvolvimento de um novo *benchmark* que atendesse aos requisitos levantados que não foram investigados na literatura. O autor atentou-se a desenvolver um *benchmark* com dois modelos de dados, e que inclui-se operações CRUD em massa, o que até então não havia sido desempenhado por nenhum outro *benchmark*. Porém, a análise comparativa é realizada epenas entre um SGBD *NoSQL* orientado

a grafo (*Neo4j*) e um SGBD relacional (*MySQL*).

Tabela 6 – Trabalhos correlatos

Trabalhos	**Tipos de Grafos**	**Ênfase**	***Benchmarks***
Silva (2021)	Propriedades	Cargas de trabalho, medidas de desempenho, recursos, outras características	LDBC SNB, XGDBench, HPC-SGAB, TGDB , Cyclone
Juričić e Radošević (2020)	Propriedades	Cargas de trabalho e estrutura de dados	LDBC SNB, HPC-SGAB, McColl, XGDBench, WGB, Sotirios
Bonifati et al. (2018)	Propriedades, RDF, Grafos paralelos e distribuídos	Tipos de entrada e saída, Cargas de trabalho e estrutura de dados	LDBC SNB, WatDiv, gMark e GSCALER
Tang (2016)	Propriedades	Cargas de trabalho e estrutura de dados	LDBC SNB, HPC-SGAB, McColl, XGDBench, WGB, Sotirios

Fonte: Autoria Própria.

Uma visão comparativa entre os trabalhos apresentados e a pesquisa realizada pode ser analisada por meio da Tabela 6.

Um grande diferencial desta pesquisa de Mestrado é a apresentação detalhada das diferentes cargas de trabalho (5.2) e recursos de dados (5.3) para *benchmarks* de SGBD's *NoSQL* orientados a grafos. Além de um rico levantamento sobre as medidas de desempenho (5.4) mais utilizadas e diferentes características (5.5) dos *benchmarks* que podem influenciar na escolha de qual será mais adequado para cada domínio de aplicação.

Capítulo 5
ANÁLISE COMPARATIVA

5.1 Considerações iniciais

Consultas representam uma parte significativa e importante de testes experimentais usando um *benchmark* como parte da avaliação do *software* sendo analisado. Mesmo existindo uma diversidade de consultas e operações sobre grafos, em geral, um padrão é seguido nas aplicações típicas que fazem uso de SGBD's *NoSQL* orientados a grafos. As informações mais importantes de um grafo, na maioria das vezes, está armazenada em nós, dessa forma, grande parcela das consultas e operações focam nessa parte do grafo (JURIČIĆ; RADOŠEVIĆ, 2020).

De acordo com Juričić e Radošević (2020), para que as operações consigam ser finalizadas, o SGBD's *NoSQL* orientados a grafos deve ter a possibilidade de armazenar diferentes tipos de resultados temporariamente. Algumas das consultas e operações podem ser complexas, podendo ser muito custoso e até inviável de se obter um resultado preciso, deste modo, valores aproximados são considerados úteis e satisfatórios.

Este capítulo concentra as principais contribuições dessa pesquisa de Mestrado no âmbito de originalidade. Neste capítulo serão abordados os principais aspectos das consultas e operações executadas pelos *benchmarks* pesquisados, além das medidas utilizadas para avaliar esses *benchmarks* e várias outras características inerentes aos *benchmarks*. Mais especificamente, este Capítulo realiza uma análise comparativa dos principais *benchmarks* para SGBD's *NoSQL* orientados a grafos de propriedades existentes na literatura, a saber: LDBC SNB, HPC-SGAB, XGDBench, TGDB e Cyclone. Com isso, características importantes exigidas por aplicações que fazem uso destes *benchmarks* e funcionalidades presentes nestes *benchmarks* são comparadas entre si com o intuito de identificar qual ou quais *benchmarks* são mais aptos a lidar com certas funcionalidades e serem mais adequados aos requisitos de aplicações de diferentes domínios.

5.2 Categorias de Carga de Trabalho

A Tabela 7 mostra os *benchmarks* alvo desta análise comparativa, com ênfase em suas cargas de trabalho. O suporte a operações individuais e em massa *(bulk-loading)* (DOMINGUEZ-SAL et al., 2011), tais como inserção, exclusão, seleção e atualização de dados são mostradas nas 8 primeiras linhas. As células na cor verde são referentes às operações realizadas pelos *benchmarks*, enquanto as células em vermelho indicam que o *benchmark* não oferece essa funcionalidade.

Dos cinco *benchmarks* estudados, apenas o *Cyclone Benchmark* tem suporte a todas as operações CRUD em massa, enquanto o HPC-SGAB oferece suporte a inserção e exclusão em massa. Enquanto isso, os demais *benchmarks* não realizam operações CRUD em massa, o que limita a diversidade de consultas por eles realizadas. Quanto às operações CRUD individuais, essas são realizadas de forma ampla apenas pelos *benchmarks* LDBC SNB, XGDBench e TGDB. *Cyclone Benchmark* possui apenas a funcionalidad de seleção individual. Em contrapartida, *Cyclone Benchmark* enfoca operações CRUD em massa, visando analisar o desempenho em função da escalabilidade com diferentes fatores de seletividade, sendo um fator de seletividade do nó a proporção de nós de saída como resultado da consulta, em relação ao número de nós de entrada. Já o fator de seletividade da aresta é definido pelo *Cyclone Benchmark* como a razão entre o número de arestas de saída e o número total de arestas de entrada.

Além das operações individuais e em massa, várias outras consultas são citadas na literatura como essenciais para se trabalhar com dados organizados em grafos. Uma delas é a consulta de caminho mais curto, que encontra o caminho mais curto entre um nó inicial e um nó final, retornando o comprimento do caminho ou em alguns nós o próprio caminho composto por nós e relacionamentos(TANG, 2016). Apenas o *XGDBench* não apresenta consultas de caminho mais curto em sua carga de trabalho.

Consultas de travessia são comumente utilizadas para avaliar informações obtidas por meio do grafo. As principais representantes dessas consultas são a Pesquisa em Largura (BFS) e a Pesquisa em Profundidade (DFS), que efetua uma busca ou travessia em um grafo e estrutura de dados do tipo árvore. A BFS é iniciada no nó raiz, explorando todos os seus vizinhos. Assim, para cada nó mais próximo, seus nós vizinhos inexplorados são percorridos, até que seja encontrado o alvo da busca. Enquanto a DFS percorre todos os nós filhos do nó raiz até a profundidade máxima, e apenas depois de todos os nós filhos percorridos, retrocede aos nós vizinhos, efetuando o mesmo procedimento nestes nós (MALHOTRA et al., 2014).

Com relação às consultas de travessia, todos os *benchmarks* implementam BFS, e apenas o *XGDBench* não implementa DFS.

De acordo com Tang (2016), as consultas de componentes buscam um subconjunto de nós no grafo que se encontram isolados de todos os demais, formando um componente conectado. Em outras palavras, um componente conectado de um grafo é um subgrafo conectado desse grafo.

Enquanto as consultas de centralidade, buscam indicar qual nó de um grafo é mais importante em relação aos outros, ou seja, quanto maior o número de relações e articulações um nó possuir em um grafo, maior é a sua importância dentro do padrão estrutural da rede. Apenas o *XGDBench* não apresenta consultas de componentes na sua carga de trabalho.

Nas consultas de Vizinhos do *k-hop*, busca-se todos os nós a uma distância *k* do nó inicial. O subgrafo encontrado é a vizinhança do nó inicial. Todos os *benchmarks* analisados neste projeto de pesquisa adotam o *k-hop* em seu conjunto de consultas, representando a importância que esse tipo de consulta tem para os modelos de dados baseados em grafos. Uma das características dos grafos de propriedade é possuir atributos (ou propriedades) tanto em seus nós quanto nas arestas. Dessa forma, um outro grupo de consultas importantes são aquelas que consideram o gerenciamento de atributos de nós e arestas, responsável por gerenciar informações contidas nos mesmos. Quanto a consultas de gerenciamento de atributos de nós, apenas o HPC-SGAB não apresenta esse tipo em sua carga de trabalho, pois este utiliza o modelo de grafo ponderado, onde o único atributo que o nó possui é o peso usado para cálculo de travessias. Já as consultas referentes ao gerenciamento de atributos de arestas, não está presente no conjunto de carga de trabalho do *XGDBench*, tendo em vista que o *benchmark* não apresenta propriedades em suas arestas.

Além dos atributos em nós e arestas, os grafos de propriedades possuem a característica de rotulagem, separando os grupos de nós de acordo com um tipo definido. Assim, as consultas referentes à rotulagem dos nós é de grande relevância em pesquisas para grafos, contribuindo nas buscas a partir dos tipos de nós. Os *benchmarks* HPC-SGAB e TGDB não realizam a distinção de nós por meio da rotulagem, dessa forma não possuindo consultas que trabalhem com a rotulagem de nós. Ainda sobre a carga de trabalho de um *benchmark* para SGBD orientado a grafos, existe o *Mix* de consultas, que diz respeito à combinação de consultas de leitura e atualização e a frequência com que são enviados (ANGLES et al., 2020). Esse *Mix* de consultas, é apresentado no LDBC-SNB e no *XGDBench*, onde cada um define a proporção dos tipos de consultas a serem combinadas. Os outros *benchmarks* não combinam consultas. O uso de mix de consultas permite representar mais fielmente um ambiente real de aplicações que realizam conjuntamente diferentes tipos de consultas e operações nos grafos.

Finalizando a Tabela 7 no que se refere aos tipos de carga de trabalho, tem-se a Tendência de Picos, que retrata-se a eventos importantes que impulsionam o volume de atividades em determinados nós em uma rede. Apenas o LDBC-SNB e o *XGDBench* se preocuparam com os eventos que podem impulsionar as atividades da rede.

E por último é apresentada a quantidade de consultas e operações presentes na carga de trabalho definidas por cada *benchmark*, sendo o LDBC-SNB o *benchmark* com o maior número.

Foi possível identificar que os *benchmarks* LDBC-SNB e *Cyclone* abordam uma ampla gama das principais consultas para SGBD's orientados a grafos, sendo esses os mais completos no aspecto carga de trabalho. Nota-se que as consultas que não estão presentes no LDBC-SNB

fazem parte do conjunto de consultas que descrevem a carga de trabalho do *Cyclone*, enquanto as consultas que faltam no *Cyclone* estão presentes no LDBC-SNB, ou seja, eles se completam. Nesse sentido, o uso conjunto dos *benchmark* LDBC-SNB e *Cyclone* tem o potencial de ser representativo e abrangente com relação a carga de trabalho composto de operações e consultas em SGBD *NoSQL* orientados a grafos.

O TGDB apresenta uma variedade significativa de consultas, não contendo apenas as operações em massa, consultas referentes à rotulagem dos nós, *mix* de consultas e tendências de pico. É o único *benchmark* comparado que trabalha com dados reais, retirados de um repositório de dados, e não com dados sintéticos, gerados por meio de um gerador de dados.

O *XGDBench* não apresenta em sua craga de trabalho consultas de pesquisa em profundidade, centralidade e componentes, consideradas por Tang (2016), consultas relacionadas ao grafo de grande importância. Esse *benchmark* concentra suas consultas em torno da manipulação de um nó e suas propriedades, tais como inserir um nós, excluir um nó ou atualizar as propriedades de um nó.

E por fim, o HPC-SGAB não inclui operações CRUD em massa ou individual, além da inserção em massa, e também não possui consultas que trabalhem com gerenciamento de propriedades de nós e arestas, pois esse contém apenas uma propriedade em suas arestas.

5.3 Recursos de dados

A Tabela 8 apresenta alguns dos principais recursos de dados que são oferecidos pelos *benchmarks* que analisam SGBD *NoSQL* orientados a grafos. As células na cor verde são referentes aos recursos de dados oferecidos pelos *benchmarks*, enquanto as células em vermelho indicam que o *benchmark* não oferece essa funcionalidade. As células preenchidas com N/D significa que não foi possível identificar a definição do recurso, e N/A quando o recurso não se aplica ao *benchmark*.

Dentre os tipos de grafos definidos na Seção 3.2, tem-se o grafo ponderado, que considera valores que representam a intensidade de um relacionamento entre nós, e o grafo direcionado, que aponta uma direção tomada pelo relacionamento (RODRIGUEZ; NEUBAUER, 2010).

O HPC-SGAB e o TGDB adotam os modelos de grafos ponderado e direcionado. O LDBC-SNB e o *Cyclone* adotam apenas o modelo de grafos direcionado, enquanto o *XGDBench* não adota nenhum dos dois modelos, utilizando o modelo de grafos não direcionado e de propriedades, este último modelo também adotado por todos os outros *benchmarks*, o que os tornaram parte desta pesquisa, que tem por enfoque grafos de propriedades.

Uma das vantagens de se usar um modelo de dados baseado em grafos é a grande diversidade de tipos de grafos existentes, como descrito nas Seções 3.2. Os modelos de dados baseados em grafos permitem uma notável flexibilidade quanto à sua representação, podendo ser

Tabela 7 – Comparação das Cargas de trabalho dos *Benchmarks* orientados a grafos

Cargas de Trabalho	**LDBC SNB**	**HPC-SGAB**	**XGDBench**	**TGDB**	**Cyclone**
Inserção em massa (*bulk loading*)					
Exclusão em massa					
Seleção em massa					
Atualização em massa					
Inserção individual					
Exclusão individual					
Seleção individual					
Atualização individual					
Caminho mais curto					
Pesquisa em largura (BFS)					
Pesquisa em profundidade (DFS)					
Centralidade					
Componente					
Vizinhos do k-hop					
Gerenciamento de atributos de nós					
Gerenciamento de atributos de arestas					
Rotulagem de nós					
Mix de consultas					
Tendências de Pico					
Total de consultas	29	4	7	8	11
Quantidade de tipos de consultas	15	8	10	12	14

Fonte: Autoria Própria.

grafos com apenas um tipo de nó, onde os relacionamentos são definidos entre nós do mesmo tipo. Ou ainda, grafos com múltiplos tipos de nós, com relacionamentos entre os diferentes tipos de nós, como ocorre em uma rede social, onde os nós do tipo *Pessoa* podem ter relacionamentos entre si, mas também com outros tipos de nós, tais como *Universidade* ou *Cidade*. O HPC-SGAB é o único *benchmark* que não utiliza o modelo de dados grafos com vários tipos de nós, enquanto o *Cyclone* apresenta os dois modelos em sua definição.

Um dos diferenciais do *Cyclone* é o uso de indexação nas propriedades dos nós, com o intuito de avaliar o impacto da utilização de índices em comparação a não utilização dos mesmos nas propriedades dos nós geradas pelos modelos de dados grafos empregados pelo *benchmark*. Dentre os *benchmarks* analisados, *Cyclone* é o único que apresenta tal característica.

Para executar um *benchmark* necessita-se de um conjunto de dados. Esses dados podem

ser sintéticos, gerados artificialmente a partir de um gerador de dados, ou dados reais, extraídos de alguma base de dados. Esses termos estão definidos na Seção 2.2.2. De todos os *benchmarks*, apenas o TGDB trabalha com dados reais, extraídos do repositório SNAP, enquanto os outros *benchmarks* utilizam dados gerados artificialmente por meio de um gerador de dados sintéticos específico para cada um.

Dentre os *benchmarks* analisados, foi possível identificar três diferentes geradores de dados sintéticos. O LDBC-SNB possui um gerador de dados próprio, denominado *DATAGEN*, baseado no modelo *MapReduce*, capaz de gerar uma rede social com estrutura definida pela Lei da potência. O HPC-SGAB utiliza o algoritmo de geração de grafos de matriz recursiva, o R-MAT, que pode gerar grafos realistas rapidamente. Enquanto o *XGDBench* faz uso do MAG, um modelo projetado para gravar a homofilia (amizade com semelhante) e heterofilia (amizade com diferentes) que ocorre dentro da rede (KIM; LESKOVEC, 2011). *Cyclone* não especifica qual gerador de dados é utilizado e o TGDB utiliza dados reais, não necessitando de um gerador de dados.

A escalabilidade em SGBDs orientados a grafos é um conceito importante a ser considerado (PAPER, 2019) para permitir compreender o desempenho dos SGBDs *NoSQl* orientados a grafos e também as suas limitações. Desta forma escalabilidade é de grande relevância, pois são indispensáveis para posteriores processamentos de transações *on-line* (OLTP) de consultas em grafos. Apenas o TGDB não possui a funcionalidade de escalabilidade de dados, por se tratar de um *benchmark* que usa um conjunto de dados reais, o que impossibilita a característica de escalabilidade.

Em um *benchmark* com carga de trabalho proposta para reproduzir fielmente interações do mundo real, o número de transações a serem executadas carece estar associado ao tamanho do conjunto de dados (ANGLES, 2013). O tamanho do conjunto de dados utilizados na execução do *benchmark* é determinado pelo fator de escala (ANGLES et al., 2020). Os *benchmarks* LDBC-SNB e HPC-SGAB definiram em sua documentação quais são os fatores de escala que define o tamanho do conjunto de dados. No entanto, não foi possível identificar os fatores de escala para os *benchmarks XGDBench* e *Cyclone*. Quanto ao TGDB, por não ser um *benchmark* escalável, não apresenta fatores de escala.

A apresentação do volume máximo e mínimo de dados é definido de forma diferente para cada *benchmark*, o LDBC-SNB apresenta usando a medida de capacidade GB, o TGDB em KB e o *Cyclone* em número de nós. Enquanto para os *benchmarks* HPC-SGAB e o *XGDBench* não foi possível encontrar esse valor.

A distribuição de nós dentro de um grafo, durante a geração de dados sintéticos, pode ser feita de forma uniforme ou não uniforme. A distribuição uniforme baseia-se em uma função randômica uniforme, onde os dados são gerados mantendo uma regularidade para cada valor específico no decorrer do tempo (CIFERRI, 1995). As distribuições uniformes, como analisado por Zipf, não fazem parte da distribuição dos dados que ocorre em aplicações do mundo real. Pois,

ao observar comportamentos das distribuições, como exemplo o uso de palavras em populações de cidades, nomes de pessoas, etc, ele verificou a existência de distribuições assimétricas. Dessa forma, idealizou uma lei para caracterizar esta ação, a chamada Lei de Zipf, ou Lei de distribuição de Zipf. A Lei de Zipf é uma derivação das Leis de potência, que define que uma função de distribuição de probabilidade, que ocorrem em uma diversidade de situações, desde a quantidade de acessos a uma página web, aos tamanhos de terremotos (NEWMAN, 2005).

Todos os *benchmarks* seguem a lei de potência, exceto o TGDB que é um *benchmark* com dados reais. Além da lei de potência, o *Cyclone*, segue também, no seu modelo de dados de vários nós, a Lei de Zipf, com o intuito de tornar o modelo de grafos flexível, calculando o número de nós distribuídos para cada tipo de nó.

Tabela 8 – Recursos de dados oferecidos por *Benchmarks* orientados a grafos

Recursos	**LDBC SNB**	**HPC-SGAB**	**XGDBench**	**TGDB**	**Cyclone**
Grafo Ponderado					
Grafo Dirigido					
Tipo único de nó					
Vários tipos de nós					
Indexação de nós					
Fator de Seletividade					
Dados sintéticos/reais	Sintéticos	Sintéticos	Sintéticos	Reais	Sintéticos
Gerador de Dados	Datagen	R-MAT	MAG	N/A	N/D
Escalabilidade do Volume de dados					
Fatores de escala	1, 3, 10, 30, 100, 300, 1000, 3000 (em GB)	10(1k), 15(32K), 20(1M)	N/D	N/A	N/D
Volume Min/Max	0,813GB/ 900GB	N/D	N/D	500K/ 10000K	10.000, 100.000, 1.000.000 (nós)
Distribuição de grau de nó	Lei da potência	Lei da potência	Lei da potência	N/D	Zipf/ Lei da potência
Recursos OLTP					
Quantidade de recursos	9	8	6	5	9

Fonte: Autoria Própria.

Dos recursos de dados apresentados na Tabela 8, o *Cyclone* se destaca por oferecer uma quantidade expressiva de recursos de dados. Porém, um dos recursos mais importantes para SGBD's orientados a grafos, os recursos OLTP, não se encontra dentre o conjunto de recursos oferecidos pelo *Cyclone*. Entretanto, é o único *benhcmark* que utiliza de dois tipos de distribuição de graus de nós, a Lei da Potência e a Lei de Zipf. O Gerador de dados e fatores de escala não

foram identificados para este o *Cyclone*.

O *benchmark* que apresenta o menor número de recursos de dados é *XGDBench*, que possui apenas vários tipos de nós, escalabilidade do volume de dados e recursos OLTP, seguindo do TGDB com grafo ponderado, grafo dirigido e vários tipos de nós.

O LDBC-SNB e o HPC-SGAB apresentam diferentes recursos de dados, porém a mesma quantidade destes. Onde o HPC-SGAB também não apresenta os recursos OLTP.

5.4 Medidas de desempenho

A comparação das principais medidas de desempenho para *benchmarks* que avaliam o desempenho de SGBD's *NoSQL* orientados a grafos, definidas na literatura, são mostradas na Tabela 9, onde as células em verde são medidas que o *benchmark* utiliza para avaliar o desempenho, enquanto as células em vermelho indicam que o *benchmark* não usa a medida.

Assim como definido na Seção 5.4, existem três tipos básicos de medidas de desempenho para *benchmarks*: Tempo de resposta, Produtividade *Throughput* e Utilização (CIFERRI, 1995; SCABORA, 2016). Adicionalmente, um *benchmark* pode oferecer medidas que capturem custos e consumo de energia. Porém, a seleção dessas medidas é decorrente dos objetivos e das características do próprio *benchmark*.

Em um *benchmark* para SGBDs orientados a grafos, as medidas mais usuais são o tempo gasto(JURIČIĆ; RADOŠEVIĆ, 2020), tendo como exemplos o tempo de carregamento em massa, que retorna o tempo decorrido para o carregamento do conjunto de dados, o tempo de resposta de consulta, que fornece o tempo necessário para que uma consulta forneça os resultados, e a latência que retorna o tempo que uma requisição está aguardando para ser atendida. Uma segunda métrica é requisito de armazenamento ou volume de dados, pode-se citar o tamanho total do grafo, que retorna o tamanho total do grafo gerado como exemplo dessa métrica. Uma terceira métrica é a produtividade *(throughput)*. Como exemplo dessa métrica tem-se a taxa de transferência que fornece o número de registros processados por segundo ou tempo total que leva-se para executar um trabalho em um conjunto de dados de um determinado tamanho.

Com o advento das redes sociais, surgem as medidas de centralidade. Tem-se ainda a medida, que tem por objetivo apontar a posição de um nó em comparação aos outros nós da rede (LARANJEIRA; CAVIQUE, 2018). Como exemplo deste tipo de medida, tem-se a centralidade de intermediação (*Betweenness Centrality*), que mede a importância de um nó em um grafo, baseando-se na especificação de caminhos mais curtos para cada nó do grafo.

Alguns *benchmarks* definem suas próprias medidas, por exemplo o TEPS, definido pelo *benchmark* HPC-SGAB (DOMINGUEZ-SAL et al., 2010), que calcula o esforço médio para percorrer os relacionamentos com relação ao tamanho total do grafo. Essa medida de desempenho também é adotada pelo LDBC-SNB, porém não utiliza o mesmo nome.

O LDBC-SNB é o mais completo no quesito medidas de desempenho, não apresentando apenas uma das oito medidas listadas na Tabela 9, sendo essa o tamanho do grafo, uma medida que somente o HPC-SGAB contempla. A única medida de desempenho que todos os *benchmarks* apresentam é o tempo de execução.

Tabela 9 – Comparação das Medidas dos *Benchmarks*

Nome	**Medidas**	**LDBC SNB**	**HPC-SGAB**	**XGD-Bench**	**TGDB**	**Cyclone**
Tamanho do grafo	Requisito de armazenamento					
Tempo de execução	Tempo de Resposta					
TEPS (Bordas atravessadas por segundo)	Produtividade (*Throughput*)					
Tempo de carregamento em massa	Tempo de Resposta					
Duração total da carga de trabalho	Tempo de Resposta					
Latência	Tempo de Resposta					
Taxa de transferência	Produtividade (*Throughput*)					
Centralidade de intermediação	Centralidade					
Quantidade de medidas		7	4	3	4	4

Fonte: Autoria Própria.

5.5 Diferentes Características dos *Benchmarks*

A Tabela 10 apresenta algumas características dos *benchmarks* voltdos à análise de desempenho de SGBD *NoSQL* orientados a grafos que podem influenciar na escolha do mais apropriado para uma determinada aplicação.

O LDBC-SNB[1] e o HPC-SGAB[2], por serem *benchmarks* de nível industrial, disponibilizam para seus usuários todo um suporte, por meio dos seus respectivos *websites*, além de diversos trabalhos na literatura, que descrevem experiências de uso, relatando detalhadamente resultados da execução do *benchmark*. Por outro lado, o *XGDBench*, TGDB e *Cyclone*, não

[1] http://ldbcouncil.org/benchmarks/snb
[2] http://graphanalysis.org/

oferecem nenhum suporte ao usuário, a não ser os trabalhos em que foram publicados, sendo estes resultantes de pesquisas acadêmicas.

Alguns *benchmarks* oferecem um programa que é responsável pela execução das diferentes cargas de trabalho e pela coleta dos resultados de desempenho. Esse programa é denominado Driver de teste (ANGLES et al., 2020). Dos *benchmarks* analisados, apenas o LDBC-SNB e o *XGDBench* disponibilizam um *driver* de teste. Os outros *benchmarks* não possuem um programa distinto para a execução das cargas trabalho, fazendo isso internamente.

Os *benchmarks* estudados tem em vista serem *benchmarks* genéricos. Porém, estes serão utilizados de diferentes formas, por domínios diferentes, que irão desde as redes sociais, *IoT*, redes de transporte, às ciências biológicas (LARANJEIRA; CAVIQUE, 2018). Onde cada domínio possui a necessidade de suprir características específicas a este. Dessa forma, os *benhcmarks* devem ser específicos ao domínio de destino, ou, os *benchmarks* genéricos devem ser adaptados ao domínio tornando-se relevante a ele.

Um domínio bastante utilizado é o das redes sociais, por ser uma área que tem sido muito estudada nos dias atuais, devido à grande importância que os relacionamentos possuem nesse domínio, sendo o relacionamento uma característica muito forte oferecida por SGBDs orientados a grafos. O LDBC-SNB e o *XGDBench* adotam o domínio das redes sociais, enquanto os outros *benchmarks* não adotam um domínio específico, podendo ser usados em diferentes domínios.

O gerador de dados do LDBC-SNB é implementado baseado no paradigma *MapReduce*, o que permite a escalabilidade, e por sua vez a geração de grandes conjuntos de dados, e para isso são utilizados *clusters* de computadores. O HPC-SGAB tem como foco a infraestrutura de computação de alto desempenho, para a execução de cargas de trabalho de processamentos de grandes volumes de de dados. Enquanto isso, o *XGDBench* utiliza uma estrutura de servidores de bancos de dados orientados a grafos em sistemas de computação em nuvem, sendo este um paradigma em constante evolução, e que oferece grandes avanços na implantação de aplicativos com uso intensivo de dados. O *XGDBench* e o *Cyclone* não especificam qual a infraestrutura é usada para a sua implementação.

O LDBC-SNB e o *Cyclone* foram desenvolvidos na linguagem *Java*, e o *XGDBench* em X10 uma linguagem destinada a programação de sistemas HPC. Não foi possível identificar em qual linguagem o HPC-SGAB e o TGBD foram desenvolvidos.

Dentre os SGBDs analisados pelos *benchmarks*, apenas o *Neo4j* está presente em todos. O *OrientDB* faz parte da avaliação de três desses *benchmarks*, sendo o LDBC-SNB, *XGDBench* e TGDB. O *Cyclone* é o único que realiza a comparação de um SGBD orientado a grafo, com um SGBD relacional.

Tabela 10 – Comparação de Características dos *Benchmarks*

Características	**LDBC SNB**	**HPC-SGAB**	**XGDBench**	**TGDB**	**Cyclone**
Suporte					
Driver de teste					
Domínio	Redes Sociais	Diversos	Redes Sociais	Diversos	Diversos
Infraestrutura	Computação em cluster	Computação de alto desempenho	Computação em nuvem	N/D	N/D
Linguagen de programação	Java	N/D	X10	N/D	Java
SGBDs comparados	Neo4j, InfiniteGraph, Sparksee, Titan/JanusGraph, OrientDB	Neo4J, Jena, HyperGraphDB e DEX	OrientDB, AllegroGraph, Neo4j, Fuseki e Titan	Neo4j, Titan, OrientDB	MySQL e Neo4j
Quantidade de características	6	4	5	2	3

Fonte: Autoria Própria.

5.6 Considerações Finais

Ao se projetar um *benchmark* para SGBD's *NoSQL* orientados a grafos, deve-se levar em consideração as características que são importantes em um grafo. É necessário ainda ter conhecimento sobre a aplicação e desta forma estabelecer os atributos a serem medidos.

Neste capítulo os *benchmarks* LDBC-SNB, HPC-SGAB, *XGDBench*, TGDB e *Cyclone* foram comparados em aspectos como carga de trabalho, recursos de dados, medidas de desempenho e diferentes características inerentes a estes.

Na Seção 5.2 foram definidas dezenove categorias de consultas que constituem as cargas de trabalho dos *benchmarks* analisados neste projeto de pesquisa de Mestrado, incluindo as operações CRUD em massa e individuais. Em relação às cargas de trabalho, os *benchmarks* LDBC-SNB e *Cyclone* se destacam quando comparados aos outros, pois possuem, respectivamente, 15 e 14 tipos de consultas, das quais, abrangem uma ampla diversidade de consultas que trabalhem com os variados aspectos do grafo, como travessia, gerenciamento de atributos e acesso a vizinhança. Neste caso, a decisão de qual *benchmark* mais apropriado para a aplicação, deve-se levar em consideração as operações CRUD, individual para o LDBC-SNB e em massa para o *Cyclone*, sendo que as outras consultas são semelhantes em ambos. Outra distinção entre as cargas de trabalho desses *benchmarks* são o *Mix* de consultas, que combina as consultas de leitura e atualização de acordo com uma determinada proporção, e a tendência de pico, para

retratar eventos importantes que impactam no volume de atividades da rede, estas duas categorias não estão presentes no *Cyclone*.

O TGDB, único *benchmark* para dados reais analisado neste projeto de pesquisa de Mestrado, retrata consultas que são de grande importância para se trabalhar com SGBD's *NoSQL* orientados a grafos, deixando de fora as operações CRUD em massa, *Mix* de consultas, Tendências de Pico e Rotulagem de nós.

Quanto aos recursos de dados oferecidos pelos *benchmarks*, descritos na Seção 5.3, o *Cyclone* é o único *benchmark* que disponibiliza um modelo de dados com vários tipos de nós, e outro modelo com apenas um tipo de nó. Ainda, é o único que trabalha com índices de propriedades de nós, para a comparação do desempenho do uso de indexação *versus* a não indexação e utiliza duas leis de distribuição de grau de nós, a Lei de Potência e a Lei de Zipf. Entretanto, os recursos OLTP, de grande importância para se trabalhar com transações, não faz parte do conjuntos de recursos oferecidos pelo *benchmark*.

Oito medidas de desempenho foram definidas na Seção 5.4, das quais sete são utilizadas pelo LDBC-SNB para avaliar o desempenho dos diversos SGBD's *NoSQL* orientados a grafos por ele analisados, onde apenas a medida de Tamanho do Grafo não é usada. Desta forma, o LDBC-SNB é o mais completo *benchmark*, dos comparados nesta pesquisa de Mestrado, na questão de tipos de medidas de desempenho.

O TGDB e o *Cyclone* utilizam exatamente as mesmas medidas de desempenho, num total de quatro medidas, sendo Tempo de execução, Tempo de carregamento em massa, Duração total da carga de trabalho e centralidade de intermediação.

Apenas o HPC-SGAB utiliza a medida de Tamanho do grafo em seu conjunto de medidas de desempenho. Foi o *benchmark* responsável pela criação da medida TEPS, que calcula o esforço médio para percorrer os relacionamentos com relação ao tamanho total do grafo, medida esta que veio a ser adotada posteriormente pelo LDBC-SNB.

Por fim, na Seção 5.5 são apresentadas diferentes características importantes ao se definir um *benchmarks*, dentre as quais se encontra a definição do domínio da aplicação, *driver* de teste e infraestrutura.

Em termos gerais, o LDBC-SNB se sobressaiu em relação aos outros *benchmarks*, principalmente quando se trata dos tipos de consultas e medidas de desempenho e por comparar diversos SGBD's *NoSQL* orientados a grafos. Porém, o *benchmark* LDBC-SNB se limita ao domínio das redes sociais, algo que é suprido pelo *Cyclone*, que é o segundo *benchmark* com maior destaque, e foi desenvolvido visando diversos domínios. Desta forma, quando se trata do domínio das redes sociais, o LDBC-SNB é mais indicado que o *XGDBench*, enquanto o *Cyclone* tem uma melhor adequação quando não se tem um domínio único. No entanto, uma fraqueza do *Cyclone* é não realizar a comparação entre SGBD's *NoSQL* orientados a grafos, que é o foco desta pesquisa de Mestrado, mas sim entre um SGBD *NoSQL* e um SGBD Relacional.

Capítulo 6
CONCLUSÃO

6.1 Considerações Iniciais

Neste capítulo são apresentadas as conclusões sobre a pesquisa realizada de Mestrado. O Capítulo está dividido em três seções, sendo que na primeira são apresentadas todas as contribuições resultantes deste projeto de pesquisa de Mestrado. Na segunda seção são apresentadas todas as dificuldades e limitações que foram surgindo no decorrer do projeto. Por fim, na terceira e última seção são apresentados os trabalhos futuros indicados para serem realizados como continuação a esta pesquisa de Mestrado.

6.2 Contribuições

Apesar da importância dos *benchmarks* para a análise de desempenho de SGBD's *NoSQL* orientados a grafos, existem poucos trabalhos de pesquisa que realizam uma análise mais crítica e minuciosa das características e funcionalidades desses *benchmarks*, que pode revelar pontos de melhoria nos *benchmarks* e nos seus processos de auditoria, execução e publicação.

Desta forma, este projeto de pesquisa de Mestrado teve por objetivo realizar uma análise comparativa dos principais *benchmarks* voltados à análise de desempenho de SGBD's *NoSQL* orientados a grafos, com ênfase em grafos de propriedades.

Portanto, as principais contribuições decorrentes da realização deste trabalho são:

1. Pesquisa bibliográfica abrangente com uso da técnica de revisão sistemática no assunto de *benchmarks* para SGBDs *NoSQL* Orientados a Grafos.
2. Detalhamento das características dos principais *benchmarks* de SGBD's *NoSQL* Orientados a Grafos.
3. Amplo estudo, descrição e comparação das principais características dos *bechmarks*, com ênfase no esquema, tipos de dados, carga de trabalho, medidas de desempenho, parâmetros e forma de execução dos *benchmarks*.

4. Análise comparativa de diferentes sistemas de *benchmarks* de SGBD's *NoSQL* Orientados a Grafos, com ênfase em grafos de propriedades.
5. Identificação dos *benchmarks* mais completos.
6. Identificação de qual *benchmark* melhor se adéqua a determinados domínios de aplicação.

Com o propósito de fornecer um contexto teórico ao trabalho, os Capítulos 2 e 3 trouxeram as principais definições referentes à técnica de *benchmark* e SGBD's *NoSQL* orientados a grafos.

Como estudo inicial para este trabalho, foram apresentados no Capítulo 4, alguns *benchmarks* mais relevantes para SGBD's *NoSQL* orientados a grafos e destacados os trabalhos relacionados a este projeto de pesquisa de Mestrado.

Por meio dos estudos e análise comparativa realizada na pesquisa, foi possível confirmar a hipótese abaixo:

A análise comparativa de benchmarks para SGBDs NoSQL Orientados a Grafos, com ênfase em grafos de propriedade, pode auxiliar usuários na escolha do benchmark mais apropriado para suas aplicações, assim como direcionar futuros trabalhos de pesquisa relacionados à proposição de novos benchmarks.

6.3 Dificuldades e Limitações

Como em todos os projetos de pesquisa, esta pesquisa possui limitações e, portanto, não foge à regra. A principal limitação desta pesquisa de Mestrado foi o fato de não conseguir reproduzir os testes das cargas de trabalho dos *benchmarks*. Após muitas tentativas, não foi possível a execução dos *benchmarks*, pois a máquina utilizada para a realização dos testes não suportou a execução dos mesmos. Além disso, a documentação de muitos *benchmarks* é falha e dificulta a correta reprodução dos experimentos. Muitas vezes, os *benchmarks* usam versões de *software* que já estão desatualizadas e que não possuem suporte atualmente.

Outra limitação para a execução dos testes foi a não disponibilização dos repositórios ou arquivos de código dos *benchmarks* TGDB e *Cyclone*, além disso pouco material de referência pôde ser encontrado para a execução desses *benchmarks*.

6.4 Trabalhos futuros

Como trabalhos futuros espera-se poder:

- Incluir novos *benchmarks* para SGBD's *NoSQL* orientados a grafos de propriedades que eventualmente surjam na literatura.

- Realizar a execução dos *benchmarks* para um conjunto amplo de SGBD's, com o objetivo de comparar o desempenho de cada SGBD.
- Realizar a análise comparativa de diferentes sistemas de *benchmarks* de SGBDs *NoSQL* Orientados a Grafos, com ênfase em outros modelos além de grafos de propriedades, como exemplo grafos RDF.
- Verificar o desempenho dos principais SGBDs *NoSQL* Orientados a Grafos, com ênfase em *Neo4j, OrientDB* e *ArangoDB*
- Destacar as diferenças dos testes experimentais dos *benchmarks* na ordenação relativa de desempenho dos SGBDs *Neo4j, OrientDB* e *ArangoDB*.

Apêndice A
REVISÃO SISTEMÁTICA

A.1 Considerações Iniciais

Este capítulo tem por objetivo descrever a revisão sistemática elaborada para o desenvolvimento das atividades deste projeto de Mestrado. Desta forma, na seção A.2 é feita a contextualização da revisão sistemática de acordo com os objetivos do projeto. O planejamento da revisão sistemática é descrito na seção A.3. Na seção A.4 é apresentado como foi realizada a condução da revisão sistemática. As considerações finais são abordadas na seção A.5, finalizando o capítulo.

A.2 Contextualização

Uma revisão sistemática foi feita para identificar o estado da arte relacionado a este projeto de Mestrado.

A revisão sistemática é uma maneira de identificar, interpretar, avaliar e sumarizar os principais aspectos em torno de um determinado tema de pesquisa. Dessa maneira, é possível apontar o estado da arte referente ao projeto, identificar possíveis lacunas para sugerir áreas que podem ser investigadas e observar evidências de tecnologias de acordo com o contexto do tema de pesquisa (KITCHENHAM; CHARTERS, 2007).

O processo de revisão sistemática foi realizado com o auxílio das ferramentas *Parsifal* e *Mendeley*. A ferramenta *Parsifal* fornece meios de gerenciar todas as fases que envolve o processo de revisão, além de disponibilizar dados estatísticos sobre o resultado final da revisão sistemática enquanto a ferramenta *Mendeley* auxilia no gerenciamento das referências bibliográficas.

A revisão sistemática elaborada para este projeto foi realizada de acordo com o processo indicado por (NAKAGAWA et al., 2017; KITCHENHAM; CHARTERS, 2007), com fases e atividades precisas. Esse processo é ilustrado na Figura 19, onde pode-se ver o fluxo de condução do mesmo. Sendo este flexível, ao necessitar de ajustes, uma fase pode ser reavaliada e redefinida.

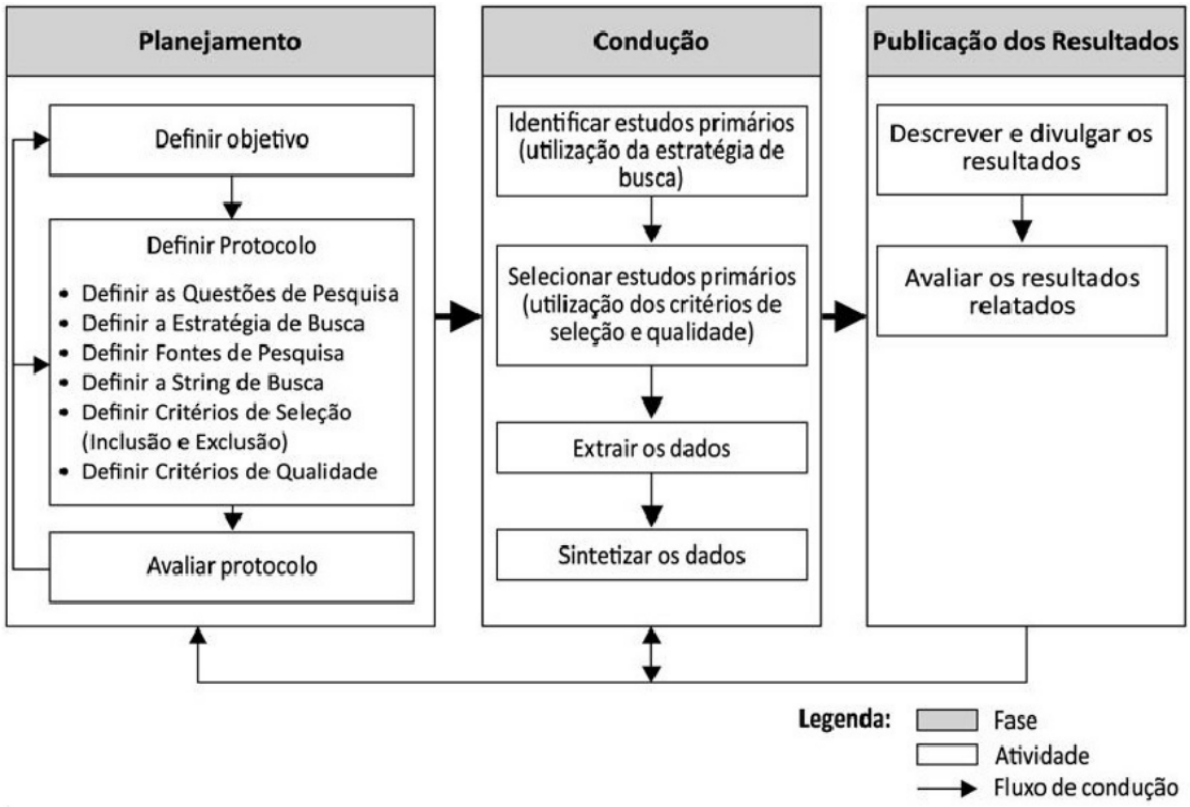


Figura 19 – Processo de revisão sistemática.

Fonte: :

(NAKAGAWA et al., 2017)

A.3 Planejamento

A primeira fase da revisão sistemática é o planejamento, onde é definido um protocolo, o qual servirá de roteiro para todo o processo. Esse protocolo consiste na definição dos objetivos (Seção A.3.1), das questões de pesquisa (Seção A.3.2), fontes de busca (Seção A.3.3), dos idiomas considerados (Seção A.3.4), das palavras-chave (Seção A.3.5) e dos critérios de seleção dos trabalhos (Seção A.3.6).

A.3.1 Objetivos

Os objetivos desta revisão sistemática são:

- Identificar arquiteturas, metodologias, abordagens e plataformas de *benchmarks* para bancos de dados *NoSQL* orientados à grafos já desenvolvidas;
- Identificar estudos que tratam da avaliação de desempenho de bancos de dados *NoSQL* orientados à grafos;

- Identificar arquiteturas, metodologias, abordagens e plataformas de *benchmarks* que deem prioridade à avaliação de desempenho de bancos de dados *NoSQL* orientados à grafos de propriedade;
- Identificar estudos que não proponham arquiteturas, metodologias, abordagens e plataformas propriamente ditas, mas que reflitam em novidades que podem ser incorporadas em arquiteturas de *benchmarks* bancos de dados *NoSQL* orientados à grafos.

A.3.2 Questões de Pesquisa

A formulação das questões de pesquisa é uma fase de extrema importância para o processo de revisão sistemática, pois permite estabelecer o foco de interesse da pesquisa. Servem como base para identificar os trabalhos que discutem e respondem essas questões. Essas respostas contribuem na realização de uma análise específica, para obter uma conclusão sobre o foco da pesquisa. Desta forma, as questões de pesquisa levantadas para a revisão sistemática são:

- **Questão 1**: Quais trabalhos realizaram levantamento bibliográfico (survey) de soluções para *benchmarks* de bancos de dados *NoSQL* orientados a grafos?
- **Questão 2:** Existem arquiteturas, metodologias, abordagens e plataformas de *Benchmark* destinadas à avaliação de Bancos de Dados NoSQL orientados à Grafos?
- **Questão 3:** Como analisar comparativamente *benchmarks* para bancos de dados NoSQL orientados à Grafos?

De acordo com Biolchini et al. (2007), Nakagawa et al. (2017) as questões de pesquisa podem obter a seguinte estrutura:

- **P - População:** grupo de interesse da revisão;
- **I - Intervenção:** o que deve ser investigado;
- **C - Comparação:** o que deve ser utilizado como base para comparação do que será investigado;
- **O - Resultados (*Outcomes*):** caracterização da pesquisa.

Diantes dessas definições, a estrutura PICO definida para a pesquisa foi:

- **P:** Pesquisadores com interesse em avaliar Bancos de Dados *NoSQL* orientados à Grafos.
- **I:** No Contexto da Revisão Sistemática serão analisados *Benchmarks* para Bancos de Dados *NoSQL* orientados à Grafos

- **C:** Avaliar e analisar diversos *Benchmarks* para Bancos de Dados *NoSQL* orientados à Grafos
- **O:** Indicar qual é o melhor *benchmark* para sistemas de bancos de dados *NoSQL* orientados a grafos de propriedade

A.3.3 Fontes de Busca

Alguns critérios devem ser levados em consideração ao definir as fontes de busca, esses envolvem condições e características que estas fontes devem atender. O primeiro critério diz respeito ao *a) alcance:* só serão consideradas as fontes que retornarem uma quantidade razoável de trabalhos. Quanto à *b) atualização:* o conteúdo retornado deve ser recente e corresponder ao tema de pesquisa. E por fim, *c) disponibilidade:* os trabalhos devem ser integralmente acessíveis. De acordo com estes critérios, foram definidas as seguintes fontes de busca:

- *IEEEXplore Digital Library*[1]
- *ACM Digital Lybrary*[2]
- *dpbl: computer science bibliography*[3]
- *Springer Link* [4]
- *Google Scholar*[5]

A.3.4 Idiomas dos Trabalhos

Os idiomas considerados para a revisão sistemática são o Português e o Inglês. O Português, aceito por ser o idioma de origem do trabalho, além disso, sua exclusão eliminaria todos os trabalhos científicos de pesquisadores brasileiros. O inglês é considerado, por ser um idioma internacionalmente utilizado para publicação de trabalhos científicos.

A.3.5 Palavras-Chave

Os termos comumente utilizados no contexto em que a revisão sistemática pretende investigar definem as palavras-chave. Com isso, com base nos objetivos e fontes de busca definidos, foram estabelecidos dois termos principais: Benchmark e Graph database. Além disso, é necessário definir os sinônimos desses termos, de modo a retornar outros trabalhos que se relacionam com os termos. Para a revisão sistemática, foram escolhidas as seguintes palavras-chave e respectivos sinônimos:

[1] https://ieeexplore.ieee.org/Xplore/home.jsp
[2] https://dl.acm.org/
[3] https://dblp.uni-trier.de/
[4] https://link.springer.com/
[5] https://scholar.google.com/

- Benchmark: *Avaliação de desempenho*, *Benchmarking*, *Performance Evaluation*;
- NoSQL
- Graph Database: *Banco de dados grafos*.

A.3.6 Critérios de Seleção

A última etapa da fase de planejamento e definição do protocolo é determinar como deve ser a seleção dos trabalhos relevantes. Nesta etapa são definidos os critérios de inclusão e exclusão e os procedimentos para selecionar os estudos. A definição dos critérios garante que os trabalhos considerados relevantes correspondem apenas aos objetivos da pesquisa.

A.3.6.1 Critérios de Inclusão

Tendo como base os objetivos da pesquisa, critérios de inclusão de estudos foram definidos para atender às questões de pesquisa:

- **I-1** - trabalhos realizaram levantamento bibliográfico (survey) de soluções para *benchmarks* de bancos de dados *NoSQL* orientados a grafos
- **I-2** - Trabalhos que apresentam arquiteturas, metodologias, abordagens, plataformas e estudos em geral sobre *Benchmark* aplicado em bancos de dados orientados à grafos.
- **I-3** - Trabalhos que realizaram análise comparativa de *benchmarks* para bancos de dados NoSQL orientados à Grafos

A.3.6.2 Critérios de Exclusão

Tendo como base os objetivos da pesquisa, os seguintes critérios de exclusão de trabalhos foram definidos:

- **E-1** - Estudos em que o idioma seja diferente do inglês ou português.
- **E-2** - Estudos que não estejam disponíveis na íntegra.
- **E-3** - Estudos que não respondem nenhuma das questões de pesquisa.

A.3.6.3 Procedimento de seleção

O procedimento para seleção dos estudos é definido em 3 etapas:

A.4 Condução

Esta fase consiste na execução das strings de busca nas fontes de busca e seleções dos estudos, de acordo com o protocolo definido na fase de planejamento (Seção A.3), para obter o maior número de trabalhos relevantes sobre o tema de pesquisa.

A.4.1 Construção da *string* de busca

As *strings* de busca são definidas para serem executadas nas fontes selecionadas (Seção A.3.3) usando-se como base a definição das palavras-chave (Seção A.3.5). Com isso, por meio de operações lógicas (AND e OR) as *strings* são construídas obedecendo as línguas consideradas para a seleção do trabalho (seção A.3.4). Dessa forma, a *string* de busca ficou assim definida:

"("Benchmark"OR "Performance Evaluation") AND ("NoSQL") AND ("Graph Database")".

"("Benchmark"OR "Avaliação de desempenho"OR "Análise de desempenho") AND ("NoSQL) AND("Banco de dados orientados a grafos")

A.4.2 Execução das Consultas

Ao executar uma consulta, é necessário indicar em qual parte do texto espera-se encontrar as palavras-chave definidas. Em outras palavras, é necessário indicar se os termos devem aparecer no título, no resumo, no corpo do texto ou em outras partes do texto. Além disso, cada fonte de busca tem um formato próprio para executar a *string* de busca. Dessa forma, é necessário adequá-la para o formato de cada fonte de busca. Assim, a *string* de busca dessa revisão sistemática considera apenas o título e o resumo dos trabalhos e ficou assim definida para cada fonte de busca:

- ***IEEEXplore Digital Library***

  ((((((("Document Title":benchmark) OR "Document Title":benchmarking) AND "Document Title":graph database) OR "Abstract":benchmark) OR "Abstract":benchmarking) AND "Abstract":graph database)

- ***ACM Digital Lybrary***

  [Publication Title: benchmark] OR [Publication Title: and] OR [Publication Title: "graph database"]] AND [[Abstract: benchmark] OR [Abstract: and] OR [Abstract: "graph database"]] AND [Publication Date: (01/01/2015 TO 04/30/2020)]

- ***DBLP: computer science bibliography***

  benchmark AND "graph database"

- ***Springer Link***

  (Benchmark OR Benchmarking OR "performance evaluation") AND ("graph database")

- ***Google Scholar***

  benchmark AND NoSQL AND "graph database"

A.4.3 Seleção dos Estudos

Na fase de seleção dos estudos é feita a filtragem dos trabalhos retornados das fontes de busca, através da execução das *strings* de busca. Essa seleção ocorre com base nos critérios de exclusão e inclusão. Dessa forma, esta fase pode ser dividida em quatro etapas:

A.4.3.1 Processo de Importação

Os trabalhos retornados na fase de execução foram extraídos para o formato *Bibtex* e importados para a ferramenta *Parsifal*. A quantidade de trabalhos retornados, de acordo com cada fonte de busca, é mostrada na Tabela 11.

Tabela 11 – Total de trabalhos retornados para cada fonte de busca pesquisada.

Trabalhos Retornados	
Fonte	**Qtde**
ACM Digital Lybrary	281
IEEEXplore Digital Library	200
Springer	128
DBLP	14
Google Scholar	103

A.4.3.2 Processo de Limpeza

Com todos os trabalhos importados é feita a limpeza desses. Neste processo, trabalhos duplicados são removidos, devendo ser considerado apenas um desses trabalhos, eliminando suas cópias (Tabela 12).

Tabela 12 – Total de trabalhos após o processo de limpeza de trabalhos.

Trabalhos Após Limpeza			
Fonte	**Qtde**	**Duplicados**	**Total**
ACM Digital Lybrary	281	4	277
IEEEXplore Digital Library	200	2	198
Springer	128	0	128
DBLP	14	1	13
Google Scholar	103	8	95

A.4.3.3 Processo de Seleção Inicial

A seleção inicial é feita de forma que, para cada trabalho mantido no processo de limpeza, foram lidos o título e *abstract*, baseados nos critérios de inclusão e exclusão de trabalhos definidos nas Seções A.3.6.2 e A.3.6.1.

Tabela 13 – Total de trabalhos relevantes e descartados para cada fonte de dados pesquisada.

Trabalhos Relevantes			
Fonte	**Total**	**Forte Relac.**	**Descarte**
ACM Digital Lybrary	277	7	270
IEEEXplore Digital Library	198	3	195
Springer	128	12	116
DBLP	13	2	11
Google Scholar	103	14	89
Total	**719**	**38**	**681**

A Tabela 13 apresenta a quantidade de trabalhos obtidos após a realização do processo de seleção inicial. Os trabalhos considerados como Fortemente Relacionados foram 41 trabalhos. Os trabalhos classificados como Descarte foram 681 trabalhos. Vale a pena destacar que os trabalhos descartados são trabalhos classificados conforme o critério de exclusão de trabalhos apresentado na Seção A.4.3.

O total de trabalhos selecionados estão descritos na Tabela 13 de acordo com cada fonte de pesquisa. Consta também a quantidade de trabalhos retornados e o total de trabalhos que serão utilizados para a revisão sistemática.

Na Tabela 14 constam os títulos dos 38 trabalhos selecionados e os códigos de critérios de inclusão aos quais eles correspondem de acordo com a descrição da Seção A.3.

Tabela 14 – Títulos dos Trabalhos Selecionados

Títulos	**Ano**	**Cod.**
GooDBye: a Good Graph Database Benchmark - an Industry Experience	2020	I-2
The LDBC Social Network Benchmark	2020	I-2
Characteristics of Graph Database Benchmarks	2020	I-2
A Graph Database Approach to Wireless IIoT Workcell Performance Evaluation	2020	I-2
HOBBIT: A platform for benchmarking Big Linked Data	2019	I-2
Which Category is Better?: Benchmarking Relational and Graph Database Managenent Systems	2019	I-2

Continuação da Tabela 14

Títulos	**Ano**	**Cod.**
SQL Database With Physical Database Turning Technique and NOSQL Graph Database Comparisons	2019	I-2
Benchmarking Graph Database Backends - What Works Well With Wikidata?	2019	I-2
Comparison of Querying Performance of Neo4j on Graph an Hypergraph Data Model	2019	I-2
Construction and Visualization of Dynamic Biological Networks: Benchmarkin the Neo4j Graph Database	2019	I-2
In-Depth Benchmarking of Graph Databse Systems With the Linked Data Benchmark Council (LDBC) Social Network Benchmark (SNB)	2019	I-2
Beyond Macrobenchmarks: Microbenchmark-based Graph Database Evaluation	2018	I-2
Performance assessment of RDF graph databases for smart city services	2018	I-2
BenchHub: store database benchmark result in database	2018	I-2
The Train Benchmark: cross-technology performance evaluation of continuous model queries	2018	I-2
Benchmarking of Graph Databases Suitability for the Industrial Internet of Things	2018	I-2
Graph Databases Comparison: AllegroGraph, ArangoDB, InfiniteGraph, Neo4j and OrientDB	2018	I-2
Graph Databases Benchmarking on the Italian Business Register	2018	I-2
Comparação entre MySQL e Neo4j para o Acesso a Dados Complexos Usando Linguagens Declarativas	2018	I-2
A Survey of Benchmarks for Graph-Processing Systems	2017	I-2
Graph Databases: Neo4j Analysis	2017	I-2
Do We Need Specialized Graph Databases? Benchmarking Real Time Social Networking Applications	2017	I-2
A Survey and Experimental Comparison of Distributed SPARQL Engines for Very Large RDF Data	2017	I-1
Benchmarking of Database for Big Data Exploration in the Social Graph Analysis	2017	I-2
Relational Database and Graph Database: A Comparative Analysis	2017	I-2

Continuação da Tabela 14

Títulos	**Ano**	**Cod.**
Predictive Performance Comparison Analysis of Relational & NoSQL Databases	2017	I-2
Iguana: a generic framework for benchmarking the read-write performance of triple stores	2017	I-2
Trying not to die benchmarking: Orchestrating RDF and graph data management solution benchmarks using LITMUS	2017	I-2
TGDB: towards a benchmark for graph databases	2016	I-2
Benchmarking Graph Databases with Cyclone Benchmark	2016	I-2
Comparative Performance Evaluation Relational and NoSQL Databases for Spatial and Mobile Applications	2015	I-2
Feasible: A feature-based sparql benchmark generation framework	2015	I-3
Diversified stress testing of rdf datamanagement systems	2014	I-2
Building an efficient RDF store over a relational database	2013	I-2
GRAPHIUM: Visualizing Performance of Graph and RDF Engines on Linked Data	2013	I-2
XGDBench: A benchmarking platform for graph stores in exascale clouds	2012	I-2
Survey of graph database performance on the hpc scalable graph analysis benchmark	2010	I-1
SP2Bench: A SPARQL Performance Benchmark	2010	I-2
Introducing the graph 500	2010	I-2
The berlin sparql benchmark	2009	I-2
LUBM: A benchmark for OWL knowledge base systems	2005	I-2

Fim da Tabela

Fonte: Autoria própria

A.5 Considerações Finais

Com o término da revisão sistemática, foi possível identificar todos os trabalhos mais correlatos ao tema deste projeto de pesquisa de mestrado, sobre a análise comparativa de *benchmarks* voltados para a avaliação de bancos de dados orientados a grafos de propriedade. Foi possível obter uma análise criteriosa sobre o estado da arte da análise comparativa de bancos de dados orientados a grafos, por meio da leitura de todos os trabalhos relacionados ao tema de pesquisa. Ainda, foi possível identificar os pontos fracos e limitações de cada trabalho, bem como do tema de pesquisa em geral.

REFERÊNCIAS

ABADI, D. J. Design. n. February, p. 37–42, 2012. Citado 3 vezes nas páginas 28, 30 e 31.

ABUL-BASHER, Z.; CHIGNELL, M. H.; GODFREY, P.; YAKOVETS, N. Tgdb: towards a benchmark for graph databases. In: IBM CORP. *Proceedings of the 26th Annual International Conference on Computer Science and Software Engineering*. [S.l.], 2016. p. 257–267. Citado 3 vezes nas páginas 16, 46 e 55.

ALUÇ, G.; HARTIG, O.; ÖZSU, M. T.; DAUDJEE, K. Diversified stress testing of rdf data management systems. In: SPRINGER. *International Semantic Web Conference*. [S.l.], 2014. p. 197–212. Citado na página 46.

ANGLES, R. *Benchmark principles and methods*. [S.l.], 2013. Citado na página 67.

ANGLES, R. The property graph database model. In: *AMW*. [S.l.: s.n.], 2018. Citado na página 35.

ANGLES, R.; ANTAL, J. B.; AVERBUCH, A.; BONCZ, P.; ERLING, O.; GUBICHEV, A.; HAPRIAN, V.; KAUFMANN, M.; PEY, J. L. L.; MARTÍNEZ, N. et al. The ldbc social network benchmark. *arXiv preprint arXiv:2001.02299*, 2020. Citado 7 vezes nas páginas 16, 44, 50, 51, 64, 67 e 71.

ANGLES, R.; BONCZ, P.; LARRIBA-PEY, J.; FUNDULAKI, I.; NEUMANN, T.; ERLING, O.; NEUBAUER, P.; MARTINEZ-BAZAN, N.; KOTSEV, V.; TOMA, I. The linked data benchmark council: a graph and rdf industry benchmarking effort. *ACM SIGMOD Record*, ACM New York, NY, USA, v. 43, n. 1, p. 27–31, 2014. Citado na página 15.

BADER, D. A.; FEO, J.; GILBERT, J.; KEPNER, J.; KOESTER, D.; LOH, E.; MADDURI, K.; MANN, B.; MEUSE, T. Hpcs scalable synthetic compact applications 2 graph analysis. *SSCA*, Citeseer, v. 2, p. v2, 2006. Citado na página 52.

BADER, D. A.; FEO, J.; GILBERT, J.; KEPNER, J.; KOESTER, D.; LOH, E.; MADDURI, K.; MANN, B.; MEUSE, T.; ROBINSON, E. Hpc scalable graph analysis benchmark. *Citeseer. Citeseer*, v. 2009, p. 1–10, 2009. Citado na página 52.

BARATA, M.; BERNARDINO, J.; FURTADO, P. Survey on Big Data and Decision Support benchmarks. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, v. 8645 LNCS, n. PART 2, p. 174–182, 2014. ISSN 16113349. Citado na página 15.

BELLINI, P.; NESI, P. Performance assessment of rdf graph databases for smart city services. *Journal of Visual Languages & Computing*, Elsevier, v. 45, p. 24–38, 2018. Citado na página 45.

BERRY, J. K. Fundamental operations in computer-assisted map analysis. *International journal of geographical information system*, Taylor & Francis, v. 1, n. 2, p. 119–136, 1987. Citado na página 19.

BIOLCHINI, J. C. de A.; MIAN, P. G.; NATALI, A. C. C.; CONTE, T. U.; TRAVASSOS, G. H. Scientific research ontology to support systematic review in software engineering. *Advanced Engineering Informatics*, Elsevier, v. 21, n. 2, p. 133–151, 2007. Citado na página 79.

BIZER, C.; SCHULTZ, A. The berlin sparql benchmark. *International Journal on Semantic Web and Information Systems (IJSWIS)*, IGI Global, v. 5, n. 2, p. 1–24, 2009. Citado na página 48.

BONCZ, P.; FUNDULAKI, I.; GUBICHEV, A.; LARRIBA-PEY, J.; NEUMANN, T. The linked data benchmark council project. *Datenbank-Spektrum*, Springer, v. 13, n. 2, p. 121–129, 2013. Citado na página 50.

BONIFATI, A.; FLETCHER, G.; HIDDERS, J.; IOSUP, A. A survey of benchmarks for graph-processing systems. In: *Graph Data Management*. [S.l.]: Springer, 2018. p. 163–186. Citado 5 vezes nas páginas 55, 56, 57, 60 e 61.

BORNEA, M. A.; DOLBY, J.; KEMENTSIETSIDIS, A.; SRINIVAS, K.; DANTRESSANGLE, P.; UDREA, O.; BHATTACHARJEE, B. Building an efficient rdf store over a relational database. In: *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*. [S.l.: s.n.], 2013. p. 121–132. Citado na página 47.

CANDELL, R.; KASHEF, M.; LIU, Y.; MONTGOMERY, K.; FOUFOU, S. A Graph Database Approach to Wireless IIoT Workcell Performance Evaluation. *2020 IEEE International Conference on Industrial Technology (ICIT)*, IEEE, p. 251–258, 2020. Citado na página 59.

CELKO, J. *Graph Databases*. [S.l.: s.n.], 2014. 27–46 p. ISBN 9781491930892. Citado 2 vezes nas páginas 27 e 28.

CHAKRABARTI, D.; ZHAN, Y.; FALOUTSOS, C. R-mat: A recursive model for graph mining. In: SIAM. *Proceedings of the 2004 SIAM International Conference on Data Mining*. [S.l.], 2004. p. 442–446. Citado na página 52.

CIFERRI, R. R. Um benchmark voltado a análise de desempenho de sistemas de informações geográficas. *Campinas, SP, Brasil: UNICAMP*, 1995. Citado 8 vezes nas páginas 18, 19, 22, 23, 24, 25, 67 e 69.

CONRADS, F.; LEHMANN, J.; SALEEM, M.; MORSEY, M.; NGOMO, A.-C. N. I guana: a generic framework for benchmarking the read-write performance of triple stores. In: SPRINGER. *International Semantic Web Conference*. [S.l.], 2017. p. 48–65. Citado na página 45.

DAYARATHNA, M.; SUZUMURA, T. Xgdbench: A benchmarking platform for graph stores in exascale clouds. In: IEEE. *4th IEEE International Conference on Cloud Computing Technology and Science Proceedings*. [S.l.], 2012. p. 363–370. Citado 4 vezes nas páginas 16, 47, 53 e 54.

DAYARATHNA, M.; SUZUMURA, T. Towards emulation of large scale complex network workloads on graph databases with xgdbench. In: IEEE. *2014 IEEE international congress on big data*. [S.l.], 2014. p. 748–755. Citado na página 53.

DOMINGUEZ-SAL, D.; MARTINEZ-BAZAN, N.; MUNTES-MULERO, V.; BALETA, P.; LARRIBA-PEY, J. L. A discussion on the design of graph database benchmarks. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, v. 6417 LNCS, p. 25–40, 2011. ISSN 03029743. Citado 4 vezes nas páginas 15, 48, 49 e 63.

DOMINGUEZ-SAL, D.; URBÓN-BAYES, P.; GIMÉNEZ-VANÓ, A.; GÓMEZ-VILLAMOR, S.; MARTÍNEZ-BAZAN, N.; LARRIBA-PEY, J.-L. Survey of graph database performance on the hpc scalable graph analysis benchmark. In: SPRINGER. *International Conference on Web-Age Information Management*. [S.l.], 2010. p. 37–48. Citado 4 vezes nas páginas 16, 48, 52 e 69.

ERDEMIR, M.; GOZ, F.; MUTLU, A.; KARAGOZ, P. Comparison of querying performance of Neo4j on graph and hyper-graph data model. *IC3K 2019 - Proceedings of the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management*, v. 1, n. September, p. 397–404, 2019. Citado na página 59.

ERLING, O.; AVERBUCH, A.; LARRIBA-PEY, J.; CHAFI, H.; GUBICHEV, A.; PRAT, A.; PHAM, M.-D.; BONCZ, P. The ldbc social network benchmark: Interactive workload. In: *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*. [S.l.: s.n.], 2015. p. 619–630. Citado na página 50.

FERNANDES, D.; BERNARDINO, J. Graph databases comparison: Allegrograph, arangoDB, infinitegraph, Neo4J, and orientDB. *DATA 2018 - Proceedings of the 7th International Conference on Data Science, Technology and Applications*, n. Data, p. 373–380, 2018. Citado na página 58.

FERRO, N.; SINICO, L. Graph Databases Benchmarking on the Italian Business Register. *CEUR Workshop Proceedings*, v. 2161, 2018. ISSN 16130073. Citado 2 vezes nas páginas 36 e 38.

FLORES, A.; PALMA, G.; VIDAL, M.-E.; ABREU, D. D.; PESTANA, V.; PINERO, J.; QUEIPO, J.; SÁNCHEZ, J. Graphium: Visualizing performance of graph and rdf engines on linked data. In: *International Semantic Web Conference (Posters & Demos)*. [S.l.: s.n.], 2013. p. 133–136. Citado na página 47.

GATTI, S. D.; CIFERRI, R. R.; OLGUÍN, C. J. M. Análise comparativa dos principais benchmarks voltados a análise de desempenho de sgbdoos. *Revista UNIMAR*, v. 19, n. 4, p. 997–1016, 1997. Citado 3 vezes nas páginas 22, 23 e 24.

GUIA, J.; SOARES, V. G.; BERNARDINO, J. Graph databases: Neo4j Analysis. *ICEIS 2017 - Proceedings of the 19th International Conference on Enterprise Information Systems*, v. 1, n. Iceis, p. 351–356, 2017. Citado na página 59.

GUO, P. *BenchHub: store database benchmark result in database*. Tese (Doutorado) — UC Santa Cruz, 2018. Citado na página 45.

GUO, Y.; PAN, Z.; HEFLIN, J. Lubm: A benchmark for owl knowledge base systems. *Journal of Web Semantics*, Elsevier, v. 3, n. 2-3, p. 158–182, 2005. Citado na página 48.

HARRISON, G. *Next Generation Databases*. [S.l.: s.n.], 2015. ISBN 9781484213308. Citado 3 vezes nas páginas 13, 14 e 33.

INDRAWAN-SANTIAGO, M. Database research: Are we at a crossroad? Reflection on NoSQL. *Proceedings of the 2012 15th International Conference on Network-Based Information Systems, NBIS 2012*, p. 45–51, 2012. Citado 3 vezes nas páginas 28, 30 e 31.

IOSUP, A.; HEGEMAN, T.; NGAI, W. L.; HELDENS, S.; PRAT-PÉREZ, A.; MANHARDTO, T.; CHAFIO, H.; CAPOTĂ, M.; SUNDARAM, N.; ANDERSON, M. et al. Ldbc graphalytics: A benchmark for large-scale graph analysis on parallel and distributed platforms. *Proceedings of the VLDB Endowment*, VLDB Endowment, v. 9, n. 13, p. 1317–1328, 2016. Citado na página 50.

JURIČIĆ, V.; RADOŠEVIĆ, M. Characteristics of Graph Database Benchmarks. *Proceedings of the 2020 Central European Conference on Information and Intelligent Systems*, 2020. Citado 4 vezes nas páginas 60, 61, 62 e 69.

KIM, M.; LESKOVEC, J. *Multiplicative Attribute Graph Model of Real-World Networks (Author's Manuscript)*. [S.l.], 2011. Citado 2 vezes nas páginas 53 e 67.

KITCHENHAM, B.; CHARTERS, S. Guidelines for performing systematic literature reviews in software engineering. Citeseer, 2007. Citado na página 77.

KLEPPMANN, M. *Designing data-intensive applications: The big ideas behind reliable, scalable, and maintainable systems*. [S.l.]: "O'Reilly Media, Inc.", 2017. Citado 2 vezes nas páginas 28 e 29.

KOTSEV, V.; MINADAKIS, N.; PAPAKONSTANTINOU, V.; ERLING, O.; FUNDULAKI, I.; KIRYAKOV, A. Benchmarking rdf query engines: The ldbc semantic publishing benchmark. In: *BLINK@ ISWC*. [S.l.: s.n.], 2016. Citado na página 50.

KOVÁCS, T.; SIMON, G.; MEZEI, G. Benchmarking Graph Database Backends—What Works Well with Wikidata? *Acta Cybernetica*, v. 24, n. 1, p. 43–60, 2019. ISSN 0324-721X. Citado na página 57.

LARANJEIRA, P. A.; CAVIQUE, L. Métricas de centralidade em redes sociais. 2018. Citado 2 vezes nas páginas 69 e 71.

LISSANDRINI, M.; BRUGNARA, M.; VELEGRAKIS, Y. an Evaluation Methodology and Experimental Comparison of Graph Databases. n. April, 2017. Disponível em: <http://disi.unitn.it>. Citado na página 57.

LISSANDRINI, M.; BRUGNARA, M.; VELEGRAKIS, Y. Beyond macrobenchmarks: Microbenchmark-based graph database evaluation. *Proceedings of the VLDB Endowment*, v. 12, n. 4, p. 390–403, 2018. ISSN 21508097. Citado 2 vezes nas páginas 25 e 45.

MALHOTRA, A.; MALHOTRA, D.; KASHYAP, S. Optimized proposed algorithm for graph traversal. *International Journal of Computer Applications*, Citeseer, v. 104, n. 9, p. 38–43, 2014. Citado na página 63.

MATYJASZCZYK, P.; ROSOWSKI, P.; WREMBEL, R. Goodbye: a good graph database benchmark-an industry experience. 2020. Citado na página 44.

MORSEY, M.; LEHMANN, J.; AUER, S.; NGOMO, A.-C. N. Dbpedia sparql benchmark–performance assessment with real queries on real data. In: SPRINGER. *International semantic web conference*. [S.l.], 2011. p. 454–469. Citado na página 47.

MURPHY, R. C.; WHEELER, K. B.; BARRETT, B. W.; ANG, J. A. Introducing the graph 500. *Cray Users Group (CUG)*, v. 19, p. 45–74, 2010. Citado na página 47.

NAKAGAWA, E. Y.; SCANNAVINO, K. R. F.; FABBRI, S. C. P. F.; FERRARI, F. C. *Revisão sistemática da literatura em engenharia de software: teoria e prática*. [S.l.]: Elsevier Brasil, 2017. Citado 3 vezes nas páginas 77, 78 e 79.

NAVOLSKYI, C. Benchmarking of Graph Databases - Suitability for the Industrial Internet of Things. 2018. Citado na página 58.

NEO4J. *Neo4j Graph Platform – The Leader in Graph Databases*. 2020. Disponível em: <https://neo4j.com/>. Acesso em: 04 abril 2020. Citado na página 39.

NEWMAN, M. E. Power laws, pareto distributions and zipf's law. *Contemporary physics*, Taylor & Francis, v. 46, n. 5, p. 323–351, 2005. Citado na página 68.

PACACI, A.; ZHOU, A.; LIN, J.; Tamer Özsu, M. Do we need specialized graph databases? benchmarking real-time social networking applications. *5th International Workshop on Graph Data Management Experiences and Systems, GRADES 2017 - Co-located with SIGMOD/PODS 2017*, 2017. Citado na página 57.

PAPER, W. Why "Right Tool, Right Job" Applies to Graph Databases. 2019. Citado na página 67.

PŁUCIENNIK, E.; ZGORZAŁEK, K. The multi-model databases–a review. In: SPRINGER. *International Conference: Beyond Databases, Architectures and Structures*. [S.l.], 2017. p. 141–152. Citado na página 42.

ROBINSON, I.; WEBBER, J.; EIFREM, E. *Graph databases: new opportunities for connected data*. [S.l.]: "O'Reilly Media, Inc.", 2015. Citado 4 vezes nas páginas 14, 33, 36 e 37.

RÖDER, M.; KUCHELEV, D.; NGOMO, A.-C. N. Hobbit: A platform for benchmarking big linked data. *Data Science*, IOS Press, n. Preprint, p. 1–21, 2019. Citado na página 45.

RODRIGUEZ, M. A.; NEUBAUER, P. Constructions from dots and lines. *Bulletin of the American Society for Information Science and Technology*, Wiley Online Library, v. 36, n. 6, p. 35–41, 2010. Citado 4 vezes nas páginas 33, 34, 35 e 65.

RUSU, F.; HUANG, Z. In-Depth Benchmarking of Graph Database Systems with the Linked Data Benchmark Council (LDBC) Social Network Benchmark (SNB). p. 1–19, 2019. Disponível em: <http://arxiv.org/abs/1907.07405>. Citado na página 58.

SALEEM, M.; MEHMOOD, Q.; NGOMO, A.-C. N. Feasible: A feature-based sparql benchmark generation framework. In: SPRINGER. *International Semantic Web Conference*. [S.l.], 2015. p. 52–69. Citado na página 46.

SASAKI, B. M.; CHAO, J.; HOWARD, R. Graph DB for beginer. 2019. Citado 5 vezes nas páginas 13, 14, 27, 32 e 35.

SCABORA, L. D. C. Avaliação do Star Schema Benchmark aplicado a bancos de dados NoSQL distribuídos e orientados a colunas. 2016. Citado 4 vezes nas páginas 20, 22, 25 e 69.

SCHMIDT, M.; HORNUNG, T.; LAUSEN, G.; PINKEL, C. Spˆ 2bench: a sparql performance benchmark. In: IEEE. *2009 IEEE 25th International Conference on Data Engineering*. [S.l.], 2009. p. 222–233. Citado na página 48.

SZÁRNYAS, G.; IZSÓ, B.; RÁTH, I.; VARRÓ, D. The Train Benchmark: cross-technology performance evaluation of continuous model queries. *Software and Systems Modeling*, v. 17, n. 4, p. 1365–1393, 2018. ISSN 16191374. Citado 2 vezes nas páginas 25 e 45.

TANG, Y. Benchmarking graph databases with cyclone benchmark. *ProQuest Dissertations and Theses*, p. 67, 2016. Disponível em: <https://search.proquest.com/docview/1845054280?accountid=188395>. Citado 6 vezes nas páginas 16, 46, 60, 61, 63 e 65.

THAKKAR, H.; KESWANI, Y.; DUBEY, M.; LEHMANN, J.; AUER, S. Trying not to die benchmarking: Orchestrating rdf and graph data management solution benchmarks using litmus. In: *Proceedings of the 13th International Conference on Semantic Systems*. [S.l.: s.n.], 2017. p. 120–127. Citado na página 46.

TPC, T. P. P. C. Tpc benchmark™ e. Citeseer, 2010. Citado na página 20.

TUOMINEN, M. Arangodb database server for gathering data from social medias. Oulun ammattikorkeakoulu, 2018. Citado na página 40.

WIESE, L.; WANGMO, C.; STEUERNAGEL, L.; SCHMITT, A. O.; GÜLTAS, M. Construction and visualization of dynamic biological networks: Benchmarking the neo4j graph database. In: SPRINGER. *International Conference on Data Integration in the Life Sciences*. [S.l.], 2018. p. 33–43. Citado na página 59.