Ana Paula Andrade Sodré

Grafo de citações de processos jurídicos relacionados à COVID-19

(*versão pré-defesa, compilada em 22 de janeiro de 2024*)

Trabalho de graduação apresentado como requisito à obtenção do grau de Bacharel em Ciência da Computação, no Departamento de Informática, setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Ciência da Computação*.

Orientador: Carmem Satie Hara.

Curitiba PR
2023

Resumo

A pandemia do coronavírus (Covid-19) gerou uma grande quantidade de dados relacionados à doença, tornando de suma importância a implementação de ferramentas que facilitem a análise desses dados. Na área jurídica, uma dessas ferramentas é o grafo de citações. Esses grafos proporcionam uma visão abrangente das relações entre documentos legais, identificando não apenas sua relevância, mas também as influências que exercem em decisões judiciais. Nesse contexto, este trabalho visa gerar grafos de citações a partir de processos obtidos do Portal de Jurisprudência do Supremo Tribunal Federal (STF) que possuem relação com a pandemia. A implementação envolve a criação de um data warehouse robusto, que vai além de associar processos a ministros do STF, estados e períodos de tempo, integrando também as relações de citações entre os documentos. Essa abordagem, com o intuito de explorar a hierarquia presente nas citações, revela-se essencial para analisar as interações intricadas presentes nos dados jurídicos. Além disso, o trabalho explorou o Neo4j como uma alternativa ao PostgreSQL, buscando comparar o desempenho entre abordagens de banco de dados relacional e de grafos. A visualização gráfica das relações legais permite uma compreensão mais abrangente do contexto, ajudando os profissionais do direito a identificar de forma mais eficaz precedentes relevantes e as influências específicas presentes em decisões judiciais.

Palavras-chave: Data Warehouse, grafo, citações.

Abstract

The coronavirus pandemic (Covid-19) has generated a vast amount of disease-related data, making it crucial to implement tools that facilitate the analysis of this information. In the legal field, one such tool is the citation graph. These graphs provide a comprehensive view of the relationships between legal documents, identifying not only their relevance but also the influences they exert on judicial decisions. In this context, this work aims to generate citation graphs from processes obtained from the Portal de Jurisprudência do Supremo Tribunal Federal (STF) that are related to the pandemic. The implementation involves the creation of a robust data warehouse, extending beyond associating processes with STF ministers, states, and time periods, integrating citation relationships between documents. This approach, intending to explore the hierarchy present in citations, proves essential for analyzing the intricate interactions in legal data. Additionally, the work explored Neo4j as an alternative to PostgreSQL, seeking to compare the performance between relational database and graph database approaches. The graphical visualization of legal relationships allows for a more comprehensive understanding of the context, helping legal professionals more effectively identify relevant precedents and the specific influences present in judicial decisions.

Keywords: Data Warehouse, graph, citations.

Lista de Figuras

Lista de Acrônimos

DINF	Departamento de Informática
PPGINF	Programa de Pós-Graduação em Informática
UFPR	Universidade Federal do Paraná
DW	Data Warehouse
STF	Supremo Tribunal Federal
ETL	Extração, Transformação e Carga

Sumário

Capítulo 1

Introdução

O surgimento de uma epidemia, mesmo em locais remotos, desperta a atenção global. Originado na cidade de Wuhan, na China, o vírus SARS-CoV-2, conhecido como COVID-19, foi rapidamente disseminado, desencadeando uma crise de saúde pública sem precedentes. O vírus se espalhou principalmente através do contato próximo entre indivíduos infectados, tanto por meio de gotículas respiratórias quanto pela interação com superfícies contaminadas. Essa alta transmissibilidade resultou em uma disseminação rápida do vírus globalmente. Segundo estimativas da Organização Mundial de Saúde (OMS)[1], há cerca de 773 milhões de casos reportados e aproximadamente 7 milhões de pessoas morreram, em todo o mundo, devido a pandemia de Covid-19 até o fim de 2023.

A pandemia deixou marcas profundas em todos os setores da sociedade, gerando impactos abrangentes. Como abordado em Morais [2023], na área da saúde, o aumento exponencial de casos graves de COVID-19 sobrecarregou os sistemas hospitalares em muitos países, acarretando em escassez de recursos médicos, leitos hospitalares e equipamentos de proteção. Além disso, os reflexos da crise se estenderam à economia global. O artigo ainda destaca que as medidas restritivas implementadas para conter a propagação do vírus resultaram no fechamento de empresas, demissões em massa e uma recessão generalizada. Segmentos como turismo, aviação, entretenimento e varejo sofreram impactos particularmente intensos, ressaltando os desafios enfrentados em meio a essa conjuntura única.

No contexto desafiador enfrentado pelo sistema judiciário durante a pandemia de COVID-19, novos obstáculos se tornaram evidentes. No artigo online STJ [2020], o Superior Tribunal de Justiça (STJ) destaca que o órgão prontamente adotou medidas de isolamento social, implementando o trabalho remoto para servidores e magistrados, além de manter os julgamentos exclusivamente virtuais. Apesar dessas medidas, Morais [2023] comenta que o contexto pandêmico resultou em atrasos processuais, aumento do acúmulo de casos e dificuldades exacerbadas no acesso à justiça.

[1] https://data.who.int/dashboards/covid19/cases?n=c

1.1 Motivação

A necessidade de realizar uma análise detalhada, tanto qualitativa quanto quantitativa, dos textos relacionados a decisões e processos sobre a pandemia no Supremo Tribunal Federal (STF) do Brasil se torna evidente. Para auxiliar nessa nova demanda, foi desenvolvido o Portal COVID-19[2]. O portal coleta processos judiciais e seus documentos associados por meio da busca de termos como "pandemia"ou "covid"no Portal de Jurisprudência do STF[3]. Como mostrado em Sodré et al. [2021], foram aplicadas técnicas de processamento de linguagem natural (NLP) e inteligência artificial para a realização das análises quantitativas e semânticas nos dados coletados.

As funcionalidades apresentadas pelo Portal incluem o conjunto de palavras mais frequentes e a proximidade entre as palavras. Estes resultados podem ser filtrados por diversos critérios, tais como por estado e por ministro do STF. Porém, percebeu-se que não havia no Portal nenhuma funcionalidade que relacionasse diferentes processos. Um tipo de relacionamento importante é a citação. Ou seja, quais processos são citados por outros e que podem refletir na argumentação utilizada por um processo.

1.2 Proposta

Esta monografia tem como objetivo investigar a viabilidade de incorporar grafos de citações nos processos relacionados à COVID-19 como uma nova funcionalidade do Portal. Essa iniciativa foi motivada pela busca de uma compreensão mais profunda das relações entre os processos, indo além das análises já feitas no Portal. Os grafos de citações são frequentemente empregados para organizar e mapear referências em documentos legais, representando uma possível linha de argumentação utilizada no processo.

1.3 Organização do documento

A estrutura deste trabalho segue o seguinte formato. O Capítulo 2 abrange definições de termos utilizados durante o documento. O processo de projeto e desenvolvimento do Data Warehouse (DW) é detalhado no Capítulo 3. Em seguida, o Capítulo 4 descreve a implementação do DW em um banco de dados de grafos e análise comparativa de desempenho em relação ao banco de dados relacional.

[2]http://portalcovid-cbio-cd.herokuapp.com/
[3]https://portal.stf.jus.br/jurisprudencia/

Capítulo 2

Fundamentação teórica

2.1 Introdução

No atual cenário jurídico trabalhista, a crescente complexidade dos processos e a influência do contexto pandêmico exigem uma abordagem inovadora para compreender as inter-relações entre os casos. Este capítulo visa estabelecer as bases conceituais e teóricas necessárias para a criação de um Data Warehouse voltado à montagem de grafos de citações entre processos jurídicos no âmbito trabalhista relacionados à pandemia do COVID-19.

2.2 Data Warehouse

O conceito de Data Warehouse (DW) vai além de ser apenas um repositório centralizado de dados. Ele abrange a ideia de armazenar informações históricas e oferecer suporte à tomada de decisões estratégicas, tornando as informações de uma organização facilmente compreensível Kimball and Ross [2013]. No contexto de um DW, existem os esquemas, que, como definido em Bhatia [2019], são as descrições lógicas de um banco de dados, de modo a definir como eles estão organizados. Também em Bhatia [2019] são citados os tipos de esquemas a serem utilizados no Data Warehouse: Estrela (*Star*), Floco de Neve (*Snowflake*) e Constelação de Fatos (*Fact Constellation*). No caso deste trabalho, foi escolhido o esquema em estrela para a arquitetura, visto que esta organiza os dados em tabelas fato e dimensões para otimizar a consulta e a análise.

2.2.1 Tabelas Fato

Bhatia [2019] define tabela fato como um grupo de itens de dados associados, composto por valores de dimensões e medidas. Essas medidas, geralmente, estão associadas a séries temporais, sendo comum a inclusão de timestamps e chaves estrangeiras.

Kimball and Ross [2013] destacam a importância de definir a granularidade como a primeira ação na modelagem, permitindo a especificação precisa do que cada tabela fato representa.

Essa granularidade pode ser definida em termos atômicos ou de alto nível, representando tanto os detalhes individuais quanto as agregações significativas.

2.2.2 Tabelas Dimensão

Enquanto as Tabelas Fato armazenam as medidas quantitativas e os dados relacionados aos eventos de negócio, as Tabelas de dimensão fornecem uma rica descrição textual do contexto que envolve essas medidas. Como definido em Bhatia [2019], a tabela de dimensão é uma coleção de informações de referência sobre um evento mensurável, onde suas colunas são os atributos descritivos. Além disso, as tabelas de dimensão são conhecidas por abrigar uma quantidade significativa de atributos. Como abordado em Kimball and Ross [2013], em muitos casos, essas tabelas podem conter entre 50 a 100 atributos, proporcionando uma visão abrangente e detalhada do contexto circundante aos eventos.

Essa abordagem robusta das tabelas de dimensão não apenas enriquece as análises de dados, mas também proporciona uma base sólida para a tomada de decisões informadas.

2.2.3 Esquema Estrela

Como definido em Bhatia [2019], o esquema estrela é uma arquitetura onde a tabela fato está no centro e as tabelas de dimensões estão nos nós da estrela. O mesmo autor ainda pontua também que cada dimensão em um esquema estrela representa apenas uma tabela unidimensional, que, nesse caso, é composta por um conjunto de atributos. Por sua vez, as tabelas fato consistem em fatos númericos e chaves estrangeiras para dados dimensionais.

A principal característica dessa arquitetura, que leva ao nome "Esquema estrela", é a disposição dos elementos de maneira que as tabelas de dimensão envolvam a tabela fato central. A representação visual final do esquema é semelhante a uma estrela. Este formato é frequentemente referido como "junção em estrela", conforme destacado por Kimball and Ross [2013].

2.3 Processo de Extração, Transformação e Carga (ETL)

O processo de Extração, Transformação e Carga (ETL) é um procedimento fundamental na integração dos dados. As três etapas que o envolvem são essenciais para a preparação dos dados para análise ou armazenamento em um sistema e serão descritas a seguir de acordo com as definições em Bhatia [2019].

- **Extração (Extract):** Nesta fase, os dados são coletados a partir de diferentes fontes, como banco de dados, arquivos, entre outros.
- **Transformação (Transform):** Após a extração, os dados passam por um processo de transformação. Nele, os dados são consolidados ou transformados em formas padrão

diferentes que são mais adequadas para a mineração de dados. Isso pode incluir a remoção de dados duplicados, correção de erros, entre outras operações para garantir a consistência e a qualidade dos dados.

- **Carga (Load):** Na etapa final, os dados transformados são carregados no destino final, no caso deste trabalho, um data warehouse. Isso prepara os dados para serem acessados, consultados e analisados de maneira eficiente.

2.4 Grafo

Como definido em Trudeau [1993], um grafo é um objeto composto por dois conjuntos chamados conjunto de vértices e conjunto de arestas. Neste trabalho, os grafos são utilizados para representar relações entre processos, com os vértices representando os processos e as arestas representando conexões (no caso, as citações) entre eles.

2.4.1 Grafo de Citações

Buneman et al. [2021] define grafo de citações como uma construção computacional amplamente utilizada para representar o domínio da literatura publicada. No contexto deste trabalho, os nós do grafo representam os processos, e as arestas indicam as citações, ou seja, quando um processo refere-se ou cita outro. Um exemplo disso é abordado em Carmichael et al. [2017], onde apresentam especificamente os precedentes da Suprema Corte e do Tribunal Federal de Apelação, utilizando métricas, como as de centralidade dos nós, para identificar decisões relevantes. Surpreendentemente, descobriram que as decisões que citam um maior número de precedentes têm maior probabilidade de serem citadas no futuro.

Outro estudo que utiliza grafo de citações no âmbito jurídico é abordado em Macohin [2019]. Nele, foram utilizados dados de jurisprudência do Conselho Nacional de Justiça para analisar 9.106 decisões judiciais entre 2005 e 2019. Utilizando técnicas de extração de conhecimento em bases de dados e redes complexas, foi criada uma rede de citações entre precedentes judiciais. Diferente da proposta deste trabalho, que tem por finalidade identificar níveis de hierarquia de citações entre os processos armazenados, em Macohin [2019] objetivava-se identificar os precedentes mais relevantes e a similaridade entre eles, empregando métricas de rede e de nós.

2.4.2 Sistema de gerenciamento de banco de dados de grafos

Como definido em Laudon and Laudon [2009], um sistema de gerenciamento de banco de dados (SGBD) é uma categoria de software especializado que tem a finalidade de desenvolver, armazenar, estruturar e recuperar dados de um banco de dados. No caso de um SGBD de grafos, em Penteado et al. [2014] é abordado que estes modelam seus dados utilizando vértices e arestas,

com o intuito de facilitar a modelagem de contextos complexos e definir as relações existentes entre as entidades de uma base.

Neste trabalho foi utilizado também o SGBD Neo4j para o armazenamento da estrutura dos grafos de citações. Ele utiliza nós para representar entidades, arestas para indicar as relações entre elas, e atributos associados a nós e arestas para fornecer informações adicionais. Desenvolvido pela Neo4j, Inc., ele é descrito pelos seus desenvolvedores, em Technology, como um banco de dados transacional compatível com ACID, assegurando confiabilidade em transações.

2.5 Consultas recursivas

Como explicado em Microsoft [2012], uma consulta recursiva é um tipo especial de consulta que utiliza uma expressão de tabela comum (CTE). A CTE, ao se referenciar a si mesma, possibilita a criação de consultas recursivas, onde a CTE inicial é executada de maneira iterativa para retornar subconjuntos de dados progressivos até a obtenção do conjunto de resultados completo. No mesmo artigo, é abordado que essas consultas são frequentemente empregadas para lidar com dados hierárquicos, como a representação de organogramas ou estruturas de listas de materiais. Isso permite uma abordagem eficiente para a exploração de estruturas complexas de dados.

Capítulo 3

Desenvolvimento do Data Warehouse

Este capítulo apresenta o desenvolvimento do Data Warehouse para dar suporte ao armazenamento e consulta do grafo de citações. A Seção 3.1 detalha sobre a arquitetura desenhada para o DW. A Seção 3.2 descreve como foi realizado o processo de extração, transformação e carga dos dados utilizados. A Seção 3.3 aborda a consulta recursiva utilizada nos testes de desempenho. Por fim, a Seção 3.4 explica como os dados sintéticos foram gerados.

3.1 Arquitetura do Data Warehouse

A maior parte das informações já armazenadas no Portal é baseada na análise da frequência e proximidade das palavras, as quais são apresentadas com base em alguns parâmetros de entrada do usuário, como o estado de interesse ou uma classe jurídica específica. Dentro do Data Warehouse utilizado no Portal eram armazenados os processos, seu conteúdo (texto completo), ministro responsável pelo caso, localização, data, classificação jurídica, palavra, contagem de palavras e grau de similaridade entre elas. Como o objetivo do trabalho é gerar grafos de citações, foram criadas novas tabelas fato (*citation, citation_type, synonyms*), que serão descritas na Seção 3.1, e desconsideradas as tabelas dimensão de contagem de palavras e proximidade, visto que a granularidade das tabelas dimensão foi redefinida para o objetivo atual. Essa redefinição foi necessária para o foco nas citações entre processos.

A modelagem do Data Warehouse foi focada em permitir tanto consultas que identifiquem conexões entre os processos (internos e externos) quanto consultas genéricas, que permitem o acesso a informações básicas de cada processo, como localização e ministro responsável, por exemplo. Para isso, foram criadas as tabelas dimensão "Citação Interna"e "Citação Externa". A de Citação Externa (EXT_CITATION) foi criada com o objetivo de possibilitar a análise de correlação entre processos armazenados no DW e documentos externos, como leis e medidas provisórias. Como o DW armazena apenas processos relacionados à pandemia, junto com seus atributos descritivos, é importante mapear as citações que não estão inclusas nesse universo de dados, com o intuito de expandir a contextualização legal. Para as citações entre processos já

presentes no DW, foi criada a tabela dimensão Citação Interna (INT_CITATION). Com ela, é possível consultar quais processos estão relacionados a um processo específico, além de permitir identificar uma possível hierarquia de citações.

O esquema proposto da constelação de fatos é representado na Figura 3.1. As tabelas de fatos são indicadas por linhas grossas e amarelas, enquanto as tabelas de dimensão são indicadas por linhas finas e azuis.

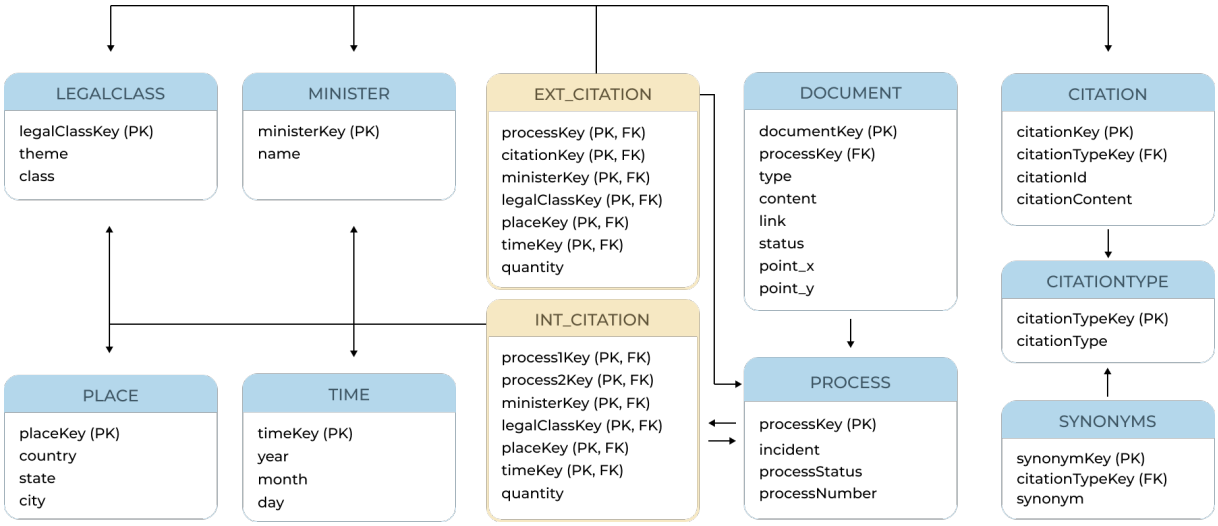

Figura 3.1: Constelação de fatos do data warehouse.

O esquema contém as seguintes tabelas de fatos:

- **EXT_CITATION (CITAÇÃO EXTERNA):** armazena uma citação entre um processo armazenado no DW e um documento externo, como lei, medida provisória, entre outros.
- **INT_CITATION (CITAÇÃO INTERNA):** armazena uma citação entre dois processos armazenados no DW.

E existem nove tabelas de dimensão:

- **TIME (TEMPO):** armazena a informação sobre o momento em que o processo foi protocolado.
- **PLACE (LOCAL):** contém informações sobre o local em que o processo foi protocolado.
- **PROCESS (PROCESSO):** armazena dados relacionados aos processos legais.
- **DOCUMENT (DOCUMENTO):** contém dados dos documentos, como tipo, conteúdo (texto completo), link de acesso e status. Um exemplo de documento que pode ser anexado a um processo é a decisão judicial.
- **LEGALCLASS (CLASSE JURÍDICA):** contém a classificação de cada processo, que pode ser categorizada por uma classe e um tema.

- **MINISTER (MINISTRO):** armazena informações sobre os ministros do STF responsáveis pelo processo.
- **CITATION (CITAÇÃO):** armazena informações sobre uma citação presente no processo.
- **CITATION_TYPE (TIPO DE CITAÇÃO):** contém o tipo de citação a que se refere, por exemplo, Projeto de Lei (PL), Medida Provisória (MP), entre outros.
- **SYNONYMS (SINÔNIMOS):** contém os sinônimos das siglas e termos presentes na tabela CITATION_TYPE.

O esquema representado na Figura 3.1 foi implementado em um banco de dados relacional, utilizando o PostgreSQL como sistema gerenciador de banco de dados (SGBD).

3.2 Processo de Extração, Transformação e Carga (ETL)

Como comentado no capítulo 2, o processo de extração, transformação e carga refere-se a três etapas fundamentais no fluxo de trabalho para a preparação de dados para análise ou armazenamento em um sistema. Em seguida, será descrito de que forma esse processo foi dividido no trabalho.

3.2.1 Extração

Os dados coletados foram extraídos do Data Warehouse utilizado pelo Portal COVID-19, cuja modelagem foi desenvolvida para uma análise qualitativa e quantitativa das palavras mais usadas nos processos jurídicos. Como abordado na Seção 3.1, o DW anterior armazenava, no PostgreSQL, dados sobre os processos, documentos associados, classe, tema, ministro, tempo e lugar. Como essas tabelas foram mantidas na modelagem atual, foi feito um *dump* no banco de dados, ou seja, foi gerada uma cópia de dados brutos presentes no DW. Como o SGBD usado foi o PostgreSQL, foi utilizado o comando `pg_dump` para criar a extração.

No caso das tabelas referentes a citações (CITATION e CITATION_TYPE), foi necessário extrair dentro do texto do processo as citações feitas. Para isso, foi utilizado o campo *content*, presente na tabela DOCUMENT, que armazena todos os textos relacionados ao processo referenciado pela chave processKey. Nesse sentido, foi necessário criar uma expressão regular para criar um padrão de busca. Foi notado que as citações seguiam o padrão *tipo_de_citacao numero [UF_do_documento_citado]*, ou seja, era necessário identificar uma cadeia de caracteres seguida de um número, podendo ou não ser seguido da UF do estado do documento citado. Isso resultou na expressão regular mostrada na Figura 3.2.

O **\b** é um delimitador de palavra utilizado no início, para assegurar que a correspondência comece no início de uma palavra, e no fim, para garantir que corresponda com o final de uma

```
\b(\w+) (\d+)(?: ([A-Z]{2}))?\b
```

Figura 3.2: Expressão regular utilizada para identificação das citações no texto

palavra. O **(\w+)** é um grupo de captura que corresponde a uma ou mais ocorrências de caracteres de palavra. Em seguida, é considerado o espaço entre a palavra seguida de um número. O **(\d+)** é o grupo de captura que corresponde a uma ou mais ocorrências de dígitos. Por fim, tem-se o **(?: ([A-Z]{2})**, onde a parte **([A-Z]{2})** indica outro grupo de captura que corresponde a exatamente dois caracteres maiúsculos de A a Z. O **?:** indica que essa parte é opcional.

Para essa extração foi utilizado um script em Python, que gera como resultado um arquivo em formato json, onde o processo que cita é uma chave e suas citações, os valores.

3.2.2 Transformação

Nesta fase, os dados extraídos são limpos e formatados antes de serem carregados no Data Warehouse. Isso incluiu também remover citações duplicadas, visto que o mesmo processo jurídico pode citar mais de uma vez um mesmo documento. Além disso, um dos problemas encontrados foi que a primeira palavra da expressão poderia ser a última parte do tipo de uma citação. Por exemplo, do termo "Medida Provisória 220022001"era obtido apenas "Provisória 220022001"como resultado da expressão regular definida. Também foram encontradas variações nas flexões gramaticais, gêneros e falta de padronização no uso de letras maiúsculas e minúsculas, referentes ao mesmo tipo de citação. Para resolver esse problema, foi criada uma planilha com todas as primeiras palavras da expressão. Nela, foi solicitado ao Dr. Dimmy Magalhães, que atua no Laboratório de Inovação do Tribunal de Justiça (OpalaLab), o preenchimento dos campos "Significado"e "Indicar Referência"para cada um das 500 palavras encontradas. O primeiro campo a ser preenchido corresponde ao significado da palavra, utilizado para agrupar termos repetidos e para mapear qual seria o termo anterior a ele (como o caso de Medida Provisória), ou se seria necessário verificar no documento as palavras vizinhas, no caso de termos mais abrangentes. O segundo campo indica se aquele tipo de citação é relevante o bastante para ser carregado no DW. Uma amostra do resultado dessa planilha é mostrada na Figura 3.3.

Com o agrupamento dos termos com mesmo significado e a eliminação das duplicatas, feitos em código em Python, a quantidade de termos reduziu para 154. Eles foram utilizados para o preenchimento das tabelas relacionadas as citações.

Os dados para as tabelas TIME, PLACE, PROCESS, DOCUMENT, LEGAL_CLASS e MINISTER foram extraídos diretamente do DW do Portal, sem precisar de alterações.

3.2.3 Carga

Após a execução do *dump* comentado na Subseção 3.2.1, foram gerados arquivos em csv com os dados de cada tabela fato em comum com o DW do Portal (TIME, PLACE, PROCESS,

	A	B	C
1	**Termo**	**Significado**	**Indicar Referência**
2	7Lei	Deve indicar o número de uma lei. Verificar palavras vizinhas para determinar a origem da lei	SIM
3	AC	Ação Cautelar ou Apelação Cível (verificar contexto)	SIM
4	ACO	Ação Cível Ordinária	SIM
5	ACO's	Ação Cível Ordinária	SIM
6	Acórdão	Referencia algum acórdão	SIM
7	acórdão	Referencia algum acórdão	SIM
8	Acórdãos	Referencia algum acórdão	SIM
9	ACORQO	Provavelmente um erro de grafia para 'ACORDO'	SIM
10	ACP	Ação Civil Pública	SIM
11	ACPCiv	Ação Civil Pública	SIM
12	ACr	Apelação Criminal	SIM
13	ADC	Ação declaratória de constitucionalidade	SIM
14	ADCs	Ação declaratória de constitucionalidade	SIM
15	ADI	Ação direta de inconstitucionalidade	SIM
16	ADI's	Ação direta de inconstitucionalidade	SIM
17	ADI´s	Ação direta de inconstitucionalidade	SIM
18	ADIN	Ação direta de inconstitucionalidade	SIM
19	ADIs	Ação direta de inconstitucionalidade	SIM

Figura 3.3: Planilha de análise dos termos

DOCUMENT, LEGAL_CLASS e MINISTER). Esses arquivos foram utilizados como entrada do script em Python a seguir para a população dos dados nas referidas tabelas no DW atual.

Listing 3.1: Script em Python para o carregamento da tabela TIME

```
def time_data():
    try:
        connection = psycopg2.connect(user=user,
                                      password=password,
                                      host=host,
                                      port=port,
                                      database=database)
        cursor = connection.cursor()
        print("Connected")

        dateQuery = getTime()

        with open(resources_path + '/time_dimension.csv', 'r') as f:
            reader = csv.reader(f)
            for row in reader:
                date = datetime.strptime(row[0], "%Y-%m-%d")
                cursor.execute(
                "INSERT INTO time (year, month, day, created_at, updated_at)
                VALUES (%s, %s, %s, %s, %s)",
                (int(date.year), int(date.month), int(date.day), dateQuery,
                          dateQuery))
                connection.commit()


    except (Exception, psycopg2.Error) as error:
        print("Error while fetching data from PostgreSQL", error)

    finally:
```

```
29        # closing database connection.
30        if connection:
31            cursor.close()
32            connection.close()
33            print("PostgreSQL connection is closed")
```

Após a conexão ao banco de dados ser feita com sucesso utilizando a biblioteca *psycopg2*, cada linha do arquivo é lida e inserida na tabela. Como cada tabela possui campos diferentes, foi necessário criar uma função de carregamento para cada tabela, mas seguindo o mesmo padrão do código mostrado acima.

No caso da tabela CITATION_TYPE, foi utilizada a planilha final abordada na Subseção 3.2.2 para carregar os termos, que são os tipos de citações. Para a tabela CITATION, utilizando a planilha citada e o json que continha os processos e suas citações foi feito um mapeamento no banco de dados para identificar quais tipos de citações cada processo possuia. O script gerava um arquivo em csv com o id do processo e os tipos de citações presentes nele.

Para a tabela fato INT_CITATION, foi necessário criar dados sintéticos para realizar um teste de desempenho de consultas. Como a pandemia referente ao COVID-19 é um assunto recentes, foram encontrados menos de 5 citações internas entre os processos. O processo para a geração dos dados sintéticos é descrito na Seção 3.4. No caso da tabela fato EXT_CITATION, foi mapeado cada citação que não era do tipo referente a processos internos e os processos que a citam. Assim, foi possível consultar os dados dos processos e criar um novo csv também com as chaves referentes a cada citação.

3.3 Implementação da consulta recursiva

A modelagem desenvolvida para o Data Warehouse possibilita a criação tanto de consultas genéricas quanto consultas mais específicas, relacionadas a citações sobre processos. Com as citações, é possível obter uma contextualização legal sobre determinado processo, além de possibilitar identificação de precedentes. Isso permite que profissionais do direito entendam melhor como decisões passadas influenciam e orientam o desfecho de casos semelhantes no presente.

Para uma análise mais aprofundada, afim de compreender as hierarquias presentes nas citações, foi criada uma função para a consulta recursiva mostrada abaixo. Ao passar o id do processo como parâmetro da função, o resultado obtido é a hierarquia completa a partir do processo consultado. Ou seja, cada citação é um nível, onde um processo cita outro processo, que, por sua vez, cita outro, e assim por diante.

Listing 3.2: Consulta recursiva em SQL

```
1 CREATE OR REPLACE FUNCTION get_hierarchy(p_id integer)
2 returns setof int_citation
3 LANGUAGE SQL
```

```
AS
with recursive result(cita, citado, h_level) as
    (select p.process1_id, p.process2_id, 1
     from int_citation p
     where p.process1_id = p_id
    union
    select r.citado, p.process2_id, h_level+1
     from result r, int_citation p
     where r.citado=p.process1_id)


    select r.cita, r.citado, r.h_level
    from result r
    where r.cita in ( select p.process1_id
                    from int_citation p);
;
```

Um exemplo de resultado da consulta é mostrado na Figura 3.4, utilizando o id de processo 1 como parâmetro.

cita	citado	h_level
1	1025	1
1025	1537	2
1537	1793	3
1793	1921	4
1921	1985	5
1985	2017	6
2017	2033	7
2033	2041	8
2041	2045	9
2045	2047	10
2047	2048	11

(11 rows)

Figura 3.4: Resultado para chamada de função para consulta recursiva

Com 88891 registros na tabela INT_CITATION, a consulta demonstrou um desempenho notável ao completar a análise em apenas 0.7 segundo, o que destaca a eficiência da abordagem adotada. Além disso, com o objetivo de identificar a quantidade de citações por nível na hierarquia, foi feita a seguinte consulta recursiva:

Listing 3.3: Consulta recursiva em SQL

```
with base as (
   with recursive result(cita, citado, h_level) as
   (select p.process1_id, p.process2_id, 1
    from int_citation p
```

```
    union
    select r.cita, p.process2_id, h_level+1
     from result r, int_citation p
     where r.citado=p.process1_id)


    select r.cita, r.citado, r.h_level
    from result r
    where r.cita in ( select p.process1_id
                    from int_citation p)
)
select h_level, count(*)
from base
group by h_level
order by h_level asc;
```

O resultado obtido é mostrado na Figura 3.5. É notório que a cada nível descresce pela metade no seu anterior, o que é resultado do algoritmo mostrado na Seção 3.4.

```
 h_level | count
---------+-------
       1 | 88891
       2 | 44443
       3 | 22218
       4 | 11106
       5 |  5551
       6 |  2774
       7 |  1386
       8 |   692
       9 |   345
      10 |   172
      11 |    86
(11 rows)
```

Figura 3.5: Resultado da segunda consulta recursiva

Por fim, torna-se evidente que a utilização da abordagem recursiva não apenas aprimora a compreensão das dinâmicas de citação entre processos, mas também executa essa tarefa com notável rapidez. Essa eficácia contribui significativamente para uma análise ágil das relações entre os elementos presentes no conjunto de dados armazenado no DW.

3.4 Geração de dados sintéticos

Ao extrair os dados, foi notado que, entre os processos armazenados, havia menos de 10 citações internas. Considerando que são filtrados apenas processos jurídicos, no âmbito trabalhista, que possui relação com a pandemia, já era esperada a baixa quantidade de citações internas (entre processos dentro da base).

Nesse sentido, foi criado um script em Python para gerar as citações internas. Para isso, foi utilizado o seguinte algoritmo:

Listing 3.4: Script em Python para geração de citações internas

```
with open('./increase_database/new_process_data.csv', 'w') as newFile:
   with open('../resources/process_dimension.csv', 'r') as f:
      reader = csv.reader(f)
      for row in reader:
         for i in range (0, 10):
            newFile.write("%s,%s,False\n"%(int(row[0]),
                                           int(row[0])+5000000+1000000*i))
```

Ao percorrer o arquivo `process_dimension.csv`, o script gera, pra cada número de processo encontrado na lista, 10 novos processos. O código utiliza a soma do número do processo com o valor 5000000, mais 1000000 vezes o número da iteração. Dessa forma, é possível identificar um padrão nos últimos algarismos, facilitando a identificação do processo original, caso necessário. Inicialmente, havia 3486 processos, depois da execução do script foram totalizados 34860.

Após essa etapa, foi criado um novo código para a geração das citações entre todos os processos. O pseudocódigo 1 mostra como foram criadas 11 hierarquias de citações.

Considerando que o arquivo *'new_incident_processo.csv'* possui apenas um número de processo por linha, o algoritmo agrupa 11 níveis de hierarquia, de forma que um processo cite o próximo na linha do arquivo em csv. Ao finalizar 11 citações sequentes, o algoritmo pula uma linha para evitar a geração de uma hierarquia maior do que a esperada. Além disso, os dados sintéticos gerados também foram utilizados para testar o banco de dados de grafos. como apresentado no próximo capítulo.

Algoritmo 1 Algoritmo para geração de citação interna

```
0: Abrir internCitationCSV para escrita
0: count ← 0
0: Abrir 'new_incident_processo.csv' para leitura
0: for cada linha em datareader do
0:    if count mod 2^11 = 0 then
0:       if count + 2^10 < tamanho de arrayCitation then
0:          Escrever "%s,%s,%d" em internCitationCSV (arrayCitation[count], arrayCitation[count+2^10], 11)
0:       end if
0:    else if count mod 2^10 = 0 then
0:       if count + 2^9 < tamanho de arrayCitation then
0:          Escrever "%s,%s,%d" em internCitationCSV (arrayCitation[count], arrayCitation[count+2^9], 10)
0:       end if
   (...)
0:    else if count mod 2^1 = 0 then
0:       if count + 2^0 < tamanho de arrayCitation then
0:          Escrever "%s,%s,%d" em internCitationCSV (arrayCitation[count], arrayCitation[count+2^0], 1)      end if
0:
0:          Incrementar count
0:       end for
0:       Fechar internCitationCSV
=0
```

Capítulo 4

Neo4j

A medida que o campo da análise de dados evolui, surgem novas abordagens para lidar com estruturas complexas e que possam ser inter-relacionadas. Este capítulo apresentará uma nova forma de armazenamento de dados, movendo-se do tradicional banco de dados relacional para uma abordagem baseada em grafos, utilizando a plataforma Neo4j. Em vez de tabelas, o Neo4j utiliza grafos para representar e armazenar os dados.

4.1 Processo de Extração, Transformação e Carga (ETL)

O processo de ETL no Neo4j foi mais simples de ser implementado, comparado com o DW desenvolvido no PostgreSQL. Isso ocorreu devido a disponibilidade dos dados, já tratados, armazenados no Data Warehouse. Assim, para o processo de extração dos dados foi realizado um *dump* no DW atualizado. Com isso, foram gerados arquivos em formato csv para cada tabela presente na base. Em seguida, foi realizada a carga desses dados no banco de grafos. Para isso, foi criado um código em Cypher, que é uma linguagem de consulta declarativa desenvolvida pela Neo4j, como mostrado a seguir.

Listing 4.1: Carregamento dos dados no Neo4j

```
1  LOAD CSV WITH HEADERS FROM 'http://localhost:11001/project/table.csv' AS row
2  WITH row
3
4  MERGE (t:Time {id: toInteger(row.time_id)})
5  MERGE (r:Relator {id: toInteger(row.relator_id)})
6  MERGE (c:Classification {id: toInteger(row.classification_id)})
7  MERGE (l:Location {id: toInteger(row.location_id)})
8  MERGE (p1:Process {id: toInteger(row.process1_id)})
9  MERGE (p2:Process {id: toInteger(row.process2_id)})
10
11 MERGE (ic:IntCitation {id: toInteger(row.id)})
12 ON CREATE SET
13   ic.quantity = toInteger(row.quantity),
```

```
14   ic.created_at = datetime(replace(row.created_at, ' ', 'T')),
15   ic.updated_at = datetime(replace(row.updated_at, ' ', 'T'))
16
17 MERGE (t)-[:HAS_INTCITATION]->(ic)
18 MERGE (r)-[:HAS_INTCITATION]->(ic)
19 MERGE (c)-[:HAS_INTCITATION]->(ic)
20 MERGE (l)-[:HAS_INTCITATION]->(ic)
21 MERGE (p1)-[:CITES]->(ic)
22 MERGE (ic)-[:CITED_BY]->(p2);
```

A cláusula LOAD CSV WITH HEADERS foi usada para carregar dados do arquivo csv. O comando FROM especifica a URL do arquivo contendo os dados a serem importados. Além disso, a opção WITH HEADERS indica que a primeira linha do arquivo csv contém os cabeçalhos das colunas.

Além do carregamento dos dados, o código também estrutura as relações e nós conforme a hierarquia de citações internas do conjunto de dados jurídicos. As linhas de código que começam com MERGE são responsáveis por criar ou combinar nós no grafo para representar diferentes entidades. No nosso contexto, entidades como Tempo (Time), Relator (Relator), Classificação (Classification), Localização (Location), Processo 1 (Process), Processo 2 (Process), e Citação Interna (IntCitation) são representadas como nós no grafo. A cláusula ON CREATE SET é usada para definir propriedades específicas quando um novo nó é criado. As linhas subsequentes estabelecem relacionamentos entre esses nós. Por exemplo, a cláusula MERGE (t)-[:HAS_INTCITATION]->(ic) indica que o nó de Tempo (t) tem um relacionamento chamado HAS_INTCITATION com o nó de Citação Interna (ic). Esses relacionamentos representam as ligações entre diferentes entidades. Além disso, a cláusula ON CREATE SET foi utilizada para atribuir valores específicos às propriedades dos nós, como quantidade, data de criação e data de atualização.

Ao executar o código, o Neo4j transforma os dados tabulares do csv em um grafo, onde os nós representam diferentes entidades e os relacionamentos entre eles retratam a inter-relação entre os processos, além de suas informações. A estrutura do grafo é adaptada para capturar as relações complexas presentes nos dados jurídicos, proporcionando uma representação visual intuitiva dessas interações.

Um exemplo de grafo é mostrado na Figura 4.1. Para a geração desse grafo foi utilizada a seguinte consulta em cypher:

Listing 4.2: Exemplo de consulta entre as entidades Process e Relator

```
1 MATCH path = (r:Relator)-[:HAS_INTCITATION]->(:IntCitation)
2 -[:CITED_BY]->(p:Process)
3 RETURN path
4 LIMIT 5;
```

Esta consulta busca caminhos no grafo que começam com um nó de Relator, seguem um relacionamento HAS_INTCITATION até um nó de IntCitation, e então continuam através de

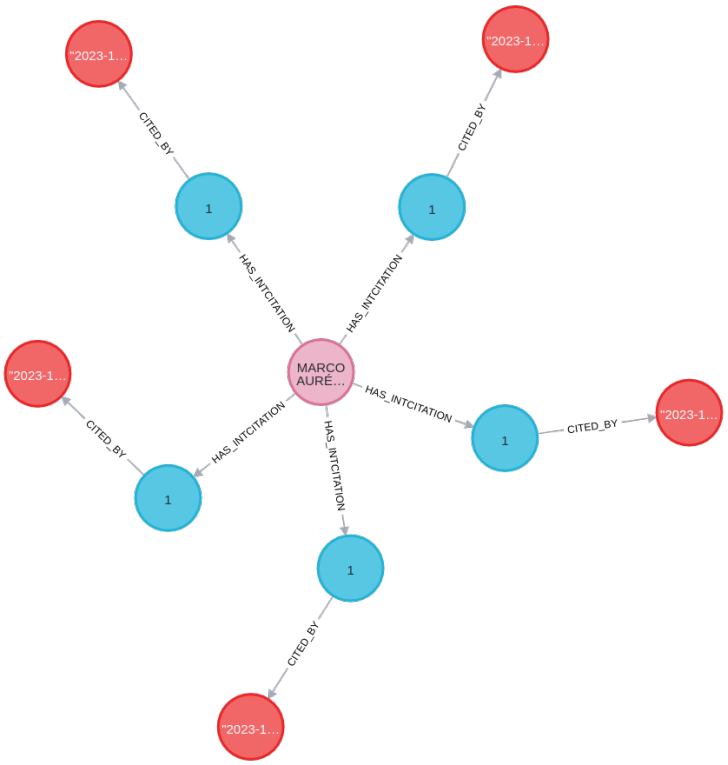

Figura 4.1: Exemplo de grafo utilizando as entidades Relator e Process

um relacionamento CITED_BY até um nó de Process. O resultado inclui esses caminhos, e a cláusula LIMIT 5 garante que apenas os primeiros 5 caminhos encontrados sejam retornados na consulta. É possível identificar as entidades Process e Relator como os nós vermelhos e a entidade que cria o relacionamente entre elas (IntCitation) em azul.

4.2 Tradução de consultas recursivas

Além de consultas simples em busca de informações sobre os processos e seus atributos descritivos, o Neo4j facilita também a criação de consultas de hierarquias dentro do grafo. Enquanto é necessário explicitamente criar uma recursão na consulta em SQL, como mostrado no código 3.3, em cypher é possível criar, de forma intuitiva, uma consulta que percorre o caminho completo das citações internas.

A tradução em cypher para a consulta 3.3 seria da seguinte forma:

Listing 4.3: Consulta para hierarquia de processos internos

```
1 MATCH (p1:Process)-[:CITES]->(ic:IntCitation)-[:CITED_BY]->(p2:Process)
2 RETURN p1, ic, p2;
```

O resultado da consulta é mostrado na Figura 4.2. Afim de facilitar a visualização dos grafos, a consulta foi limitada para que apresentasse apenas as 10 primeiras citações (relacionadas pela IntCitation, representada pelos nós azuis).

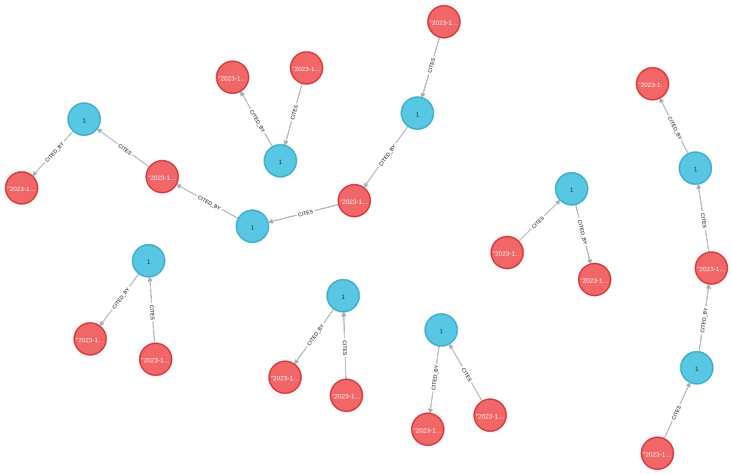

Figura 4.2: Exemplo de grafos de citações e sua hierarquia completa

É possível notar que, além de ser uma consulta mais intuitiva devido a facilidade de utilizar o caminho da hierarquia de citações para construir o código, o Neo4j possibilita a visualização gráfica dos grafos.

4.3 Comparativo de desempenho

Para analisar o desempenho da consulta em SQL e em cypher, foi utilizada a consulta abordada na seção anterior, porém para um processo específico. Dessa forma, seria possível percorrer o mesmo caminho e avaliar ambas abordagens de forma justa.

Para a análise da execução da consulta em SQL, foi utilizado um script em Python que executava 5 vezes a consulta presente na função get_hierarchy, mostrada no código 3.3. Ao finalizar, foi feita uma média dos 5 valores para obter um resultado mais acurado. Foi utilizado 5 processos diferentes para a média simples final do desempenho. A média final da consulta recusiva em SQL foi de 0.1 segundos.

O modelo de consulta em cypher utilizada é mostrado a seguir.

Listing 4.4: Consulta para hierarquia de citações internas a partir de um processo

```
1 MATCH p= (:Process { id:id_do_processo })
2     ((p1:Process)-[:CITES]->(:IntCitation)-[:CITED_BY]->(p2:Process)){1,}
3 RETURN p;
```

Nesse caso, foi seguida a mesma metodologia para a análise de desempenho. A consulta em cypher teve uma média final de 0.02 segundos.

Em síntese, a abordagem em cypher demonstrou uma eficiência notável, sendo cinco vezes mais rápida na obtenção dos grafos resultantes da consulta em comparação com a abordagem

em SQL. Esses resultados ressaltam a aptidão da linguagem e do SGBD para lidar eficientemente com estruturas de grafos, consolidando-se como uma opção ágil para consultas desse tipo.

Capítulo 5

Conclusão

Este trabalho representou um estudo sobre dois paradigmas fundamentais de gerenciamento de dados: o modelo relacional tradicional e o modelo de banco de dados de grafos. Ao implementar um Data Warehouse inicialmente baseado em um banco relacional e, posteriormente, replicar a estrutura no Neo4j, buscou-se compreender as nuances e os benefícios de cada abordagem.

Durante o processo, foi constatado que o Neo4j oferece vantagens significativas para representar e consultar dados altamente inter-relacionados, como no contexto jurídico. A flexibilidade do Cypher, linguagem de consulta do Neo4j, permitiu expressar relações complexas de maneira concisa e eficaz, destacando-se em cenários nos quais as conexões entre entidades são cruciais.

A análise comparativa entre o desempenho das consultas em SQL e Cypher ressaltou a eficiência deste último para operações que exploram a estrutura de grafos. Em conjunto, essas descobertas fornecem uma visão inicial sobre a escolha do modelo de dados mais apropriado para diferentes tipos de análises, evidenciando que a coexistência de modelos pode ser benéfica em ambientes complexos.

5.1 Trabalhos futuros

Para as próximas etapas deste trabalho, é esperado explorar mais profundamente as capacidades da ferramenta Neo4j. Pretende-se incorporar algoritmos disponibilizados pela plataforma que aprimorem a capacidade de análise e interpretação dos grafos presentes nos dados jurídicos. Além disso, visa-se expandir o escopo de conhecimento ao explorar outras metodologias de gerenciamento de dados. Dessa forma, seria possível adotar abordagens mais abrangentes que possam enriquecer ainda mais a compreensão das complexas relações presentes nos conjuntos de dados em questão.

Referências Bibliográficas

Muito além do novo coronavírus: a jurisprudência do stj em tempos de epidemia, 2020. URL https://www.stj.jus.br/sites/portalp/Paginas/Comunicacao/Noticias/Muito-alem-do-novo-coronavirus-a-jurisprudencia-do-STJ-em-tempos-de-epidemia.aspx. Publicado em 19/04/2020, Acessado em 27/12/2023.

Parteek Bhatia. *Data Mining and Data Warehousing: Principles and Practical Techniques*. Cambridge University Press, 2019.

Peter Buneman, Dennis Dosso, Matteo Lissandrini, and Gianmaria Silvello. Data citation and the citation graph. *Quantitative Science Studies*, 2(4):1399–1422, 12 2021. ISSN 2641-3337. doi: 10.1162/qss_a_00166. URL https://doi.org/10.1162/qss_a_00166.

Iain Carmichael, James Wudel, Michael Kim, and James Jushchuk. Examining the evolution of legal precedent through citation network analysis. *N.C. Law Review*, 96:227, 2017. https://scholarship.law.unc.edu/nclr/vol96/iss1/6.

Ralph Kimball and Margy Ross. *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling*. John Wiley & Sons, Inc., Indianapolis, 3rd edition, 2013.

Kenneth C. Laudon and Jane Price Laudon. *Sistemas de Informação Gerenciais*. Pearson Prentice Hall, São Paulo, 7 edition, 2009.

A. Macohin. Análise da jurisprudência do conselho nacional de justiça através de redes complexas. *Revista CNJ*, 3(2):19–26, 2019. doi: 10.54829/revistacnj.v3i2.79. URL https://www.cnj.jus.br/ojs/revista-cnj/article/view/79.

Microsoft. Recursive queries using common table expressions, 2012. URL https://learn.microsoft.com/en-us/previous-versions/sql/sql-server-2008-r2/ms186243(v=sql.105)?redirectedfrom=MSDN. Acesso em 26/11/2023.

Fernanda de Morais. Os impactos da pandemia do covid-19 no sistema judiciário e nas práticas jurídicas, 2023. URL https://direito.idp.edu.br/idp-learning/mercado-juridico/impactos-pandemia/. Publicado em 19/09/2023.

Raqueline R. M. Penteado, Rebeca Schroeder, Diego Hoss, Jaqueline Nande, Ricardo M. Maeda, Walmir O. Couto, and Carmem S. Hara. Um estudo sobre bancos de dados em grafos nativos. In *Anais da Escola Regional de Banco de Dados (ERBD)*, 2014.

A. Sodré, D. Magalhães, L. Floriano, A. Pozo, C. Hara, and S. Machado. Covid-19 portal: Machine learning techniques applied to the analysis of judicial processes related to the pandemic. In *25th East-European Conference on Advances in Databases and Information Systems, ADBIS 2021 co-located with Workshop on DOING 2021*, volume 1450 of *CCIS*, pages 109–120. Springer, 2021.

Neo Technology. Neo4j graph database. `https://neo4j.com/product/“‘`. Acessado em 26/12/2023.

R.J. Trudeau. *Introduction to Graph Theory*. Dover Books on Mathematics. Dover Pub., 1993. ISBN 9780486678702. URL `https://books.google.com.br/books?id=NunuAAAAMAAJ`.