Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Uso de banco de dados orientado a grafos na detecção de fraudes nas cotas para exercício da atividade parlamentar

Gabriel M. Araujo

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Orientadora
Prof.a Dr.a Maristela Terto de Holanda

Brasília
2018

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Uso de banco de dados orientado a grafos na detecção de fraudes nas cotas para exercício da atividade parlamentar

Gabriel M. Araujo

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Prof.a Dr.a Maristela Terto de Holanda (Orientadora)
CIC/UnB

Prof. Dr. Donald Knuth
Stanford University

Dr. Leslie Lamport
Microsoft Research

Prof. Dr. Rodrigo Bonifácio
Coordenador do Bacharelado em Ciência da Computação

Brasília, 7 de setembro de 2018

Dedicatória

Eu dedico esse trabalho aos colegas de curso, que ajudaram a sustentar o sofrimento constante que esse curso proporciona. Dedico também para a galera da central e da CJR, onde aprendi muito mais do que assistindo aulas de cálculo 3.

Agradecimentos

Gostaria de agradecer a todos os meus colegas de curso que me ajudaram nesse período, a todos os professores que se empenharam para ensinar seus alunos e melhorar a cada dia, e principalmente ao Stack Overflow.

Resumo

Este trabalho propõe o uso da tecnologia de banco de dados orientado a grafos para a detecção de fraudes na Cota para o Exercício da Atividade Parlamentar – CEAP. Além do uso do banco de dados orientado a grafos, será feita uma plataforma web para expor informações importantes a população e um estudo sobre o impacto dessas informações na política brasileira. O uso dessas tecnologias facilita muito a manipulação de dados bastante relacionados entre si, tanto em questão de complexidade na consulta, quanto em relação a visualização da informação. A proposta em questão foi validada com um estudo de caso, utilizando os dados abertos da Cota para o Exercício da Atividade Parlamentar da câmara dos deputados. Foi desenvolvido um ETL para extrair os dados e popular o banco, em seguida as consultas foram realizadas para detectar as fraudes e obter informações a respeito dos dados, finalmente foi desenvolvido um sistema web que se comunica via REST com o banco de dados para expor as informações a população de forma mais clara e simples. Para trabalhos futuros, seria interessante o uso de aprendizagem de máquina para obter mais informações valiosas sobre a CEAP.

Palavras-chave: Banco de dados orientado a grafos, fraude, política

Abstract

This work proposes the use of graph oriented databases to detect frauds in Quota for the Exercise of Parliamentary Activity. In addition to the use of the graph oriented databases, a web platform will be developed to expose important information to the population and a study on the impact of this information on Brazilian politics. The use of these technologies, greatly facilitates the manipulation of closely related data, both in terms of query complexity, and information visualization. The proposal in question was validated with a case study, using the open data of the Quota for the Exercise of the Parliamentary Activity of the Chamber of Deputies. It was developed an ETL to extract the data and fill the database, then the queries were made to detect the fraud and to obtain information about the data, finally a web system was developed that communicates via REST with the database to expose the information to the population more clearly and simply. For future work, it would be interesting to use machine learning to obtain more valuable information about CEAP.

Keywords: Graph oriented databases, fraud, politics

Sumário

Capítulo 1

Introdução

O Brasil é reconhecido mundialmente por seus aspectos culturais, por ser membro do grupo de países BRICS e por ser líder da américa latina. Porém, nosso país também é bastante conhecido por ser um país corrupto, e por possuir um cenário político conturbado. Um estudo feito por Abramo [1] compara as relações entre índices de percepção de corrupção e outros indicadores de alguns países latino americanos, e o Brasil se encontra na 49ª (quadragésima nona) posição em um ranking de corrupção dentre 90 países. Já o estudo feito por Filgueiras [2] mostra que de acordo com a percepção dos brasileiros, a Câmara dos vereadores e a Câmara dos deputados são as instituições com maior presença de corrupção.

Atualmente, operações como a Lava Jato, mostram que o país vive um período político muito sensível, que acabam causando vários problemas nos mais diversos setores do país e principalmente na economia. Dessa forma, o objetivo geral deste trabalho é utilizar a tecnologia de banco de dados orientado a grafos, para auxiliar na detecção de possíveis fraudes em um determinado conjunto de dados. Como foi mencionado acima, a câmara dos deputados está entre as instituições com maior presença de corrupção no país, portanto, o conjunto de dados utilizado neste trabalho é a Cota para o Exercício da Atividade Parlamentar – CEAP (antiga verba indenizatória), que é uma cota única mensal destinada a custear os gastos dos deputados exclusivamente vinculados ao exercício da atividade parlamentar.

O Ato da Mesa nº 43 de 2009, detalha as regras para o uso da CEAP, entretanto um deputado pode realizar algumas transações que não são observadas facilmente pelos responsáveis em fiscalizar essas transações. Por exemplo, o artigo 4, parágrafo 13 do Ato da Mesa nº 43 de 2009, diz: *Não se admitirá a utilização da Cota para ressarcimento de despesas relativas a bens fornecidos ou serviços prestados por empresa ou entidade da qual o proprietário ou detentor de qualquer participação seja o Deputado ou parente seu até terceiro grau.* Dessa forma, o Deputado pode realizar transações que violam essa regra,

sendo inviável verificar as relações de parentesco de cada Deputado em cada transação, especialmente se utilizarem tecnologias inadequadas.

Portanto, justifica-se o uso de um banco de dados orientado a grafo para identificar os relacionamentos envolvendo cada transação de um Deputado. Um banco de dados relacional também consegue resolver esse problema, entretanto, com um custo e complexidade bem maior em relação a um banco de dados orientado a grafo. Isso se deve porque os relacionamentos são evidenciados na estrutura de um grafo de forma muito mais natural e simples, onde cada entidade é representada como um nó do grafo e se relaciona com outras entidades por meio de arestas. Devido a essas particularidades, os bancos de dados em grafo vem ganhando bastante popularidade ultimamente [3], registrando a maior taxa de mudança de popularidade de 2013 até 2017.

1.1 Objetivos

O objetivo geral deste trabalho é utilizar o SGBD OrientDB para evidenciar relacionamentos nas transações dos Deputados que violam o artigo 4, parágrafo 13 do Ato da Mesa nº 43 de 2009, que regula a CEAP. Para essa análise, será feito um sistema web que expõe essas informações a população de forma mais simples e clara. Além de estudar como essas tecnologias impactam a política brasileira.

Os objetivos específicos deste trabalho são:

- Implementar o banco de dados em grafo com os dados da CEAP.
- Executar consultas que evidenciem relacionamentos fraudulentos.
- Identificar vantagens e desvantagens no uso de banco de dados orientado a grafo para a detecção de fraudes em conjuntos de dados genéricos.
- Desenvolver um sistema web para expor informações a respeito dos dados.
- Estudar o impacto dessas tecnologias na política e sociedade brasileira.

1.2 Metodologia

Este trabalho foi dividido em duas partes, a primeira teórica e a segunda prática. Na parte teórica foi realizado um estudo baseado em livros, artigos e páginas da *web* sobre os assuntos relacionados a banco de dados, SGBD orientado a grafo, NoSQL, as leis que regem a CEAP e os impactos dessas iniciativas na politica. Já na parte prática foi desenvolvido um ETL para ler os arquivos que contém os dados das transações e popular o banco, em seguida foram realizadas consultas que buscam evidenciar os relacionamentos

atrelados a cada transação, posteriormente foi desenvolvido um sistema web de forma que forneça as informações claramente a todos. Por fim, foi realizado uma análise dos resultados obtidos e conclusões finais.

1.3 Estrutura do Trabalho

Este trabalho está dividido nos seguintes capítulos:

- Capítulo 2: Introduzo os conceitos relacionados a grafos, necessários para a compreensão de um SGBD orientado a grafo. Forneço uma visão geral de SGBD NoSQL, apontando suas principais características e utilidades. Finalmente explico as características de um SGBD orientado a grafo e sua evolução.
- Capítulo 3: Nesse capítulo apresento os meios utilizados para resolver o problema, e uma explicação mais detalhada de todo o processo de desenvolvimento e como as tecnologias auxiliaram nessa resolução.
- Capítulo 4: Apresento a análise dos resultados obtidos ao realizar as consultas e do sistema desenvolvido para fornecer as informações. Essa análise abrange tanto os resultados das consultas, ou seja, se foi possível identificar uma transação ilícita, quanto o impacto da divulgação desse tipo de informação na política brasileira.
- Capítulo 5: Finalmente, apresento minhas conclusões do trabalho realizado e sugestões para trabalhos futuros relacionados com a área.

Capítulo 2

Fundamentação Teórica

Este capítulo descreve toda a fundamentação teórica por trás das tecnologias utilizadas na implementação do estudo de caso. Inicialmente, explico os conceitos gerais relacionados a teoria de grafos, em seguida, aponto as principais características e utilidades dos SGBDs NoSQL. Posteriormente, aponto as características e particularidades de um SGBD orientado a grafos, junto com uma explicação acerca do SGBD escolhido para o trabalho que é o OrientDB. Por fim, explico um pouco sobre a tecnologia REST utilizada para a comunicação entre o sistema desenvolvido e o OrientDB.

2.1 Teoria de Grafos

2.2 SGBD NoSQL

Essa seção aborda a categoria de bancos de dados não relacionais. Inicialmente, realizo uma introdução sobre o assunto, para em seguida explicitar as principais características e categorias dentro desse grupo de bancos de dados.

2.2.1 Introdução

O termo NoSQL foi utilizado na literatura pela primeira vez por Carlo Strozzi em 1998 como o nome de um banco de dados que ele estava desenvolvendo na época [4]. Curiosamente, é um banco relacional que não possuia interface SQL, portanto, NoSQL. Entre o período dos anos de 2000 e 2005 o universo dos SGBD NoSQL começou a aumentar bastante, o SGBD baseado em grafo Neo4j começou a ser desenvolvido no ano de 2000, o SGBD da Google BigTable[5] começa em 2004 e em seguida o CouchDB se inicia por volta de 2005. Porém, todas essas tecnologias não eram na época categorizadas como NoSQL, somente em 2009 o termo foi reintroduzido por Johan Oskarsson ao organizar um evento

para discutir sobre banco de dados não relacionais, distribuídos e de código aberto, e a partir desse momento o termo passa a ser usado para categorizar os bancos de dados não relacionais.

O termo NoSQL gera muita confusão pois leva a entender que os bancos de dados nessa categoria não utilizam a linguagem SQL. Porém, o termo costuma ser utilizado como *Not only SQL*, e demonstra que tais sistemas podem suportar também linguagens de consultas baseadas em SQL. O aspecto mais importante por trás dos SGBD NoSQL é o motivo de seus surgimentos. No início dos anos 2000, a forma como os usuários começaram a interagir na internet começou a evoluir e consequentemente os sistemas e aplicações passaram a gerar e consumir cada vez mais dados e informações. Esse crescimento na geração e consumo de dados foi enorme, e por isso os especialistas necessitavam de tecnologias que atendessem melhor suas necessidades, desse cenário começaram a nascer os primeiros SGBD NoSQL como o BigTable da google e o Dynamo da Amazon[6].

2.2.2 Características de um SGBD NoSQL

Como foi mencionado acima, por volta dos anos 2000 começou a nascer a necessidade de novas tecnologias na área de banco de dados. Os sistemas precisavam de boa escalabilidade horizontal para operações de forma distribuída, entre outras coisas. Vários aspectos relacionados a categoria dos SGBD NoSQL são ainda abertos e não possuem um consenso. O trabalho feito por Rick Cattell[7] aglomerou algumas características que geralmente se encontram nesse tipo de banco de dados:

- Habilidade de escalar horizontalmente operações simples através de vários servidores
- Habilidade de replicar e distribuir os dados através de vários servidores
- Um modelo de concorrência mais flexível que as transações ACID presentes em sistemas relacionais.
- Uso eficiente de índices distribuidos e da memória RAM para armazenar os dados
- Habilidade de adicionar dinamicamente novos atributos a um registro

Porém, é importante ressaltar que nem todas essas características precisam ser atendidas para fazer parte dessa categoria. Sendo que também existem outras características importantes observadas em alguns sistemas como por exemplo, disponibilidade e consistência dos dados, níveis de flexibilidade para o esquema do banco de dados e etc.

2.2.3 Gerenciamento de Transações

Uma propriedade importante em qualquer SBGD é o controle de concorrência. Normalmente, vários usuários solicitam operações ao SGBD simultâneamente, sendo que esse sistema gerenciador de banco de dados precisa garantir aos usuários a integridades dos dados transacionados. A maioria dos bancos de dados relacionais utilizam transações ACID[8] para controlar essa concorrência, que impõe ao SGBD as seguintes propriedades:

- Atomicidade: Essa propriedade impõe que durante uma transação, todas as alterações feitas no banco de dados sejam efetivadas. Caso ocorra algum erro durante a transação o SGBD realiza um *rollback* para o estado consistente anterior a transação.
- Consistência: Essa propriedade impõe que ao término de uma transação o banco de dados está de forma consistente. Ou seja, a cada transação os dados devem estar consistentes e a garantia de que as restrições impostas aos dados não sejam violadas.
- Isolamento: A propriedade de isolamento de uma transação, busca impor ao SGBD que cada transação seja independente das demais. Ou seja, garante que cada transação seja vista como uma unidade e impede que outras transações acessem dados que possam levar o bancos de dados a um estado inconsistente.
- Durabilidade: A durabilidade garante que as informações salvas após cada transação permaneçam no banco de dados. Os dados não podem desaparecer em nenhum momento e devem permanecer persistidos independente de qualquer falha.

Esse formato é utilizado por bancos de dados relacionais, e atende bem uma boa parte de sistemas e aplicações nos dias de hoje, como é o caso de sistemas bancários por exemplo, que necessitam dessas propriedades para garantir a integridades das informações de seus clientes.

O mais importante a se perceber aqui é que nem toda aplicação precisa necessariamente desse controle de concorrência elaborado. No caso de bancos é de suma importância esse tipo de controle, agora em uma aplicação de alta disponibilidade uma propriedade como a consistência pode ser um pouco mais fraca. Portanto, para atender a essas novas demandas, nasceu o modelo de transações BASE[9] que busca fornecer maior flexibilidade para esses novos sistemas. O BASE tem como característica em vez de exigir consistência após cada transação, é suficiente para o banco de dados estar eventualmente consistente. Existe um teorema famoso feito por Eric Brewer conhecido como Teorema CAP[10], que demonstra o trade-off entre consistência, disponibilidade e tolerância a particionamento. Eric diz que se for necessário essas três características, é necessário escolher somente duas delas. Portanto no modelo BASE é possíve escolher tolerância a particionamento e

disponibilidade e abrir mão de uma alta consistência dos dados. O acrônimo BASE vem das três seguintes características:

- *Basic Availability*: Essa propriedade no fundo quer dizer que o banco de dados aparenta funcionar a maior parte do tempo.
- *Soft state*: Essa propriedade quer dizer que as bases não precisam estar sempre consistentes, nem as réplicas precisam estar consistentes o tempo inteiro. Ou seja, o estado do sistema é volátil.
- *Eventual consistency*: Essa propriedade diz que as bases de dados vão ficar eventualmente consistentes no futuro.

2.2.4 Categorias de NoSQL

Dentro do universo dos SGBD NoSQL, existem cinco grandes categorias que separam cada SGBD em relação a estratégia de armazenamento[11]. Tais categorias são:

SGBD NoSQL por chave/valor

Nessa categoria

SGBD NoSQL orientado por colunas

Nessa categoria

SGBD NoSQL orientado a documentos

Nessa categoria

SGBD NoSQL orientado a grafos

Nessa categoria

SGBD NoSQL orientado a objetos

Nessa categoria

2.3 SGBD NoSQL orientado a grafos

2.4 OrientDB

Essa seção tem por objetivo especificar as características do SGBD OrientDB, e realizar uma comparação com outros SGBD orientado a grafos como o Neo4j.

2.4.1 Introdução

O OrientDB, é um SGBD de código aberto sob a licensa Apache. Ele é o primeiro SGBD NoSQL multi modelo, com suporte a uma arquitetura distribuída e orientado a grafos. Sendo assim o OrientDB suporta operações com documentos, chave/valor e grafos. Essa característica garante uma enorme flexibilidade para manipular os dados dentro do OrientDB, sendo possível armazenar os dados tanto como grafos ou como documentos no mesmo banco de dados [12].

O OrientDB é implementado utilizando a linguagem java, tendo sua primeira versão disponível no ano de 2010. Ele possui alta flexibilidade para definir o esquema do banco de dados, podendo ser *Schema-free*, *Schema-hybrid* ou *Schema-full*. A sua linguagem de consulta é derivada do SQL o que é bastante vantajoso para aqueles que possuem experiência com bancos de dados relacionais, e além disso ele utiliza o modelo de transações ACID que como foi mencionado na seção 2.2.3 é algo mais comum no grupo dos SGBD relacionais, isso demonstra que o OrientDB presa pela integridade dos dados ao mesmo tempo que também fornece um suporte a particionamento dos dados [12].

Todas essas características fazem com que o OrientDB seja um SGBD bastante flexível e confiável para se utilizar em diversas aplicações. As operações utilizando grafos em específico, vem ganhando bastante visibilidade pois funciona muito bem em certos domínios de aplicação. Como foi mencionado no capítulo 1 e nesse artigo feito por Matthias Gelbmann [13] a popularidade dos SGBD orientados a grafos vem crescendo bastante nos últimos anos, e essas características ajudam a explicar o porque de banco de dados como o OrientDB e Neo4j estarem crescendo tanto em popularidade.

2.4.2 Performance

O OrientDB possui uma ótima performance em operações utilizando grafos. Um aspecto técnico que ajuda nessa qualidade é que o OrientDB trabalha cada registro como um objeto e o *link* entre esses objetos não é feito por referência, e sim por *link* direto. Ou seja, é salvo um ponteiro que aponta diretamente para o objeto referenciado. Isso faz com que a velocidade para recuperar informações seja muito mais rápido em comparação com

os joins utilizados nos SGBD relacionais. Portanto, não existem operações de joins dentro do OrientDB para obter as relações entre os registros, sendo que ele consegue salvar certa de 120 mil registro por segundo.

O OrientDB utiliza mecanismos de indexação baseados em árvores-B e hash extendido, esses mecanismos garantem uma complexidade constante para obter relacionamentos entre um registro para muitos registros. Um estudo feito pelo instituto de tecnologia de tokio e pela IBM, mostra que o OrientDB chega a ser 10 vezes mais rápido que o seu maior concorrente que é o Neo4j [14].

2.4.3 Arquitetura Distribuída

Em relação ao suporte a uma arquitetura distribuída, o OrientDB trabalha com um método de particionamento conhecido como *sharding*, e um método de replicação conhecido como *Multi-master replication*. Esse modelo de replicão, permite que qualquer membro do cluster possa persistir dados no banco de dados. Isso evitar gargalos como occore no modelo *Master-slave replication* e permite uma maior escalabilidade em aplicações.

2.5 REST

Referências

[1] Abramo, Cláudio Weber: *Relações entre índices de percepção de corrupção e outros indicadores em onze países da américa latina*. SPECK, Bruno W. et al. Os custos da corrupção. Cadernos Adenauer, (10):47–62, 2000. 1

[2] Filgueiras, Fernando: *A tolerância à corrupção no brasil: uma antinomia entre normas morais e prática social*. Opinião Pública, 15(2):386–421, 2009. 1

[3] engines db: *Dbms popularity broken down by database model*. `https://db-engines.com/en/ranking_categories`. Acessado em setembro de 2017. 2

[4] Strozzi, Carlo: *Nosql a relational database management system*. `http://www.strozzi.it/cgi-bin/CSA/tw7/I/en_US/nosql`. Acessado em janeiro de 2018. 4

[5] Chang, Fay, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes e Robert E. Gruber: *Bigtable: A distributed storage system for structured data*. ACM Trans. Comput. Syst., 26(2):4:1–4:26, junho 2008, ISSN 0734-2071. `http://doi.acm.org/10.1145/1365815.1365816`. 4

[6] DeCandia, Giuseppe, Deniz Hastorun, Madan Jampani, Gunavardhan Kakulapati, Avinash Lakshman, Alex Pilchin, Swaminathan Sivasubramanian, Peter Vosshall e Werner Vogels: *Dynamo: Amazon's highly available key-value store*. SIGOPS Oper. Syst. Rev., 41(6):205–220, outubro 2007, ISSN 0163-5980. `http://doi.acm.org/10.1145/1323293.1294281`. 5

[7] Cattell, Rick: *Scalable sql and nosql data stores*. SIGMOD Rec., 39(4):12–27, maio 2011, ISSN 0163-5808. `http://doi.acm.org/10.1145/1978915.1978919`. 5

[8] Ramez Elmasri, Shamkant B Navathe: *Sistemas de banco de dados*. Pearson Addison Wesley, 2005. 6

[9] Pritchett, Dan: *Base: An acid alternative*. Queue, 6(3):48–55, maio 2008, ISSN 1542-7730. `http://doi.acm.org/10.1145/1394127.1394128`. 6

[10] Brewer, Eric: *A certain freedom: Thoughts on the cap theorem*. Em *Proceedings of the 29th ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing*, PODC '10, páginas 335–335, New York, NY, USA, 2010. ACM, ISBN 978-1-60558-888-9. `http://doi.acm.org/10.1145/1835698.1835701`. 6

[11] Nayak, Ameya, Anil Poriya e Dikshay Poojary: *Type of nosql databases and its comparison with relational databases*. International Journal of Applied Information Systems, 5(4):16–19, 2013. 7

[12] engines db: *Orientdb system properties.* `https://db-engines.com/en/system/OrientDB`. Acessado em janeiro de 2018. 8

[13] Gelbmann, Matthias: *Graph dbmss are gaining in popularity faster than any other database category.* `https://db-engines.com/en/blog_post/26`. Acessado em janeiro de 2018. 8

[14] Dayarathna, Miyuru e Toyotaro Suzumura: *Xgdbench: A benchmarking platform for graph stores in exascale clouds*. Em *Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on*, páginas 363–370. IEEE, 2012. 9