VISUALIZANDO UM MODELO RELACIONAL ATRAVÉS DE GRAFOS E IDENTIFICANDO PADRÕES E ANOMALIAS NA MODELAGEM – UMA EXPERIMENTAÇÃO

Lucas Porto Rosa
MBA em Ciência de Dados
Professor orientador: Ezequiel Mendes

Resumo

Considerando que o modelo tradicional de dados - modelo relacional - é constituído de entidades (tabelas) que se relacionam entre si, podemos fazer uma comparação com o modelo orientado à grafos com seus vértices e arestas. Um banco de dados orientado à grafos (Neo4J) é especializado em tratar objetos e seus relacionamentos e possui uma gama de algoritmos que buscam encontrar padrões nesses relacionamentos.

Foi objetivo deste trabalho realizar uma experimentação aplicando os conceitos da teoria de grafos em um modelo relacional de banco de dados. Com isso, buscando-se entender melhor a modelagem de um software tradicional, identificando seus padrões e anomalias, e visualizado seus relacionamentos de uma maneira mais fácil.

Utilizando uso da pesquisa aplicada e qualitativa, o desenvolvimento deste trabalho possibilitou um maior entendimento sobre os relacionamentos de tabelas e objetos de um banco de dados dentro de um esquema de grafos. Foram levantados os dados de descobertas dos modelos experimentados e entendido que essa análise dos relacionamentos pode ser muito benéfica para a saúde dos sistemas que desenvolvemos. Ao final do trabalho foram identificados pontos importantes no modelo utilizado como possibilidades de melhorias, possíveis gargalos de performance e constatações sobre suas entidades, além de proporcionar um grande avanço nos estudos sobre Banco de Dados de Grafos em especial o Neo4J e sua lingaguem Cypher.

Palavras-chave: Banco de Dados, Grafos, Neo4J, Modelagem, Relacionamentos, Performance

Abstract

Considering that the relational model is constituted of entities (tables) that are related to each other, we can compare this model with graph theory with its vertices and edges. A graph-oriented database (Neo4J) is specialized in treating objects and their relationships and has a range of algorithms that search find patterns in these relationships.

It was the objective of this work to carry out an experimentation applying the concepts of graph theory in a relational model of database. With this, we seek to better understand the modeling of traditional software, identifying its patterns and anomalies, and visualizing its relationships in an easier way.

Using the use of applied and qualitative research, the development of this work allowed a greater understanding about the relationships of tables and objects of a database within a graph scheme. Data were collected on the discoveries of the used models and understood that this analysis of the relationships can be very beneficial for the health of the systems that we developed. At the end of the work, important points were identified in the model used as possibilities for improvements, possible performance bottlenecks and findings about their entities, as well as providing a great advance in studies on Graph Database especially Neo4J and its Cypher language.

Keywords: Database, Graphs, Neo4J, Modeling, Relationships, Performance

1. INTRODUÇÃO

No fluxo de desenvolvimento de software, antes da etapa de codificação do sistema, é necessário realizar um estudo das estruturas de banco de dados que vão ser necessárias para concluir o desenvolvimento do produto. À essa etapa damos o nome de Modelagem de Dados.

Segundo RAMEZ ELMASRI e SHAMKANT NAVATHE os objetivos da Modelagem de Dados são diversos: Satisfazer os requisitos de informações dos usuários e aplicações;

prover uma estrutura natural e fácil de entender das informações e suportar o processamento necessário e performance requirida como tempo de resposta, tempo de processamento e espaço de armazenamento.

A etapa de Modelagem de Dados deve ser realizada pelo profissional com conhecimentos técnicos sobre Banco de Dados e Análise de Sistemas. Os impactos negativos para o software, de uma Modelagem de Dados mal feita, podem ser muito significativos. Sendo que muitas vezes, esses impactos só aparecem algum tempo depois do desenvolvimento do software, seja por questões de crescimento do banco de dados ou até posteriormente, quando for necessário realizar alterações na aplicação que a Modelagem atual inviabiliza.

Considerando que o Modelo Relacional constitui de entidades - tabelas - e suas relações entre si, podemos fazer uma comparação com o modelo baseado em grafos. Esse modelo consiste na análise de objetos e seus relacionamentos. Na teoria de grafos existem certos padrões que podem ser explicados através de algoritmos e experimentos práticos. Um banco de dados orientado à grafos é otimizado para processar diversos tipos de relações entre suas entidades. Esse design permite a construção de modelos preditivos, detecção e correlação de padrões (MILLER, J. J.).

Através da análise dos relacionamentos entre as tabelas de um modelo relacional é possível conhecer melhor uma aplicação, entender quais são suas tabelas principais e seus possíveis pontos de gargalos em performance. Sendo assim, como questão norteadora deste trabalho, quais seriam as possibilidades de descobertas de padrões e anomalias em um banco de dados relacional, se visualisássemos seus objetos a partir de diagramas de grafos aplicando os algoritmos matemáticos da teoria dos grafos?

Mesclando esses dois modelos e utilizando a teoria de grafos através do banco de dados Neo4j, o objetivo deste trabalho é identificar e listar as descobertas sobre o modelo relacional e utilizar para obter maior conhecimento sobre o modelo e para tomar decisões de alterações do mesmo.

Para tal, na seção Metodologia serão descritos os processos executados para condução deste trabalho; na seção Revisão de Literatura serão listados os principais conceitos para entendimento geral do artigo; na seção Apresentação da pesquisa será apresentado o estudo de caso utilizado; na seção Discussão dos Resultados, os resultados encontrados e por fim, em Considerações Finais serão apresentados as reflexões sobre o desenvolvimento do artigo e as possibilidades de continuidade da pesquisa.

2. METODOLOGIA

O trabalho proposto fará uso da pesquisa aplicada e qualitativa. A pesquisa qualitativa não se preocupa com representatividade numérica, mas sim, com o aprofundamento da compreensão de um grupo social, de uma origanização, etc. O método qualitativo busca explicar o porquê das coisas, exprimindo o que convém ser feito, mas não quantificam os valores e as trocas simbólicas nem se submetem à provas de fatos, pois os dados analisados são não-métricos e se valem de diferentes abordagens. Assim como a pesquisa Aplicada objetiva gerar conhecimentos para aplicação prática, dirigidos à solução de problemas específicos (UFRGS, 2009).

Com o intuito de alcançar o objetivo proposto, a metodologia consiste em escolher um modelo relacional de um sistema tradicional e exportar seus metadados que contenham informações de suas tabelas, índices, constraints e demais informações importantes para análise. Esses metadados deverão ser importados para um banco de dados de grafos (Neo4j). Essa importação deverá ser feita utilizando linguagem SQL e Cypher.

Após os dados inseridos no banco de dados de grafos, serão executados algoritmos utilizando a linguagem Cypher, que é a linguagem própria para manipulação de dados no Neo4j. Para os grafos com grande número de vértices, será utilizado o software Gephi para análise. As descobertas feitas em cima dos dados analisados serão visualizadas no artigo através de grafos, tabelas explicativas e relatórios descritivos.

Abaixo segue figura com o fluxo do processo a ser realizado.

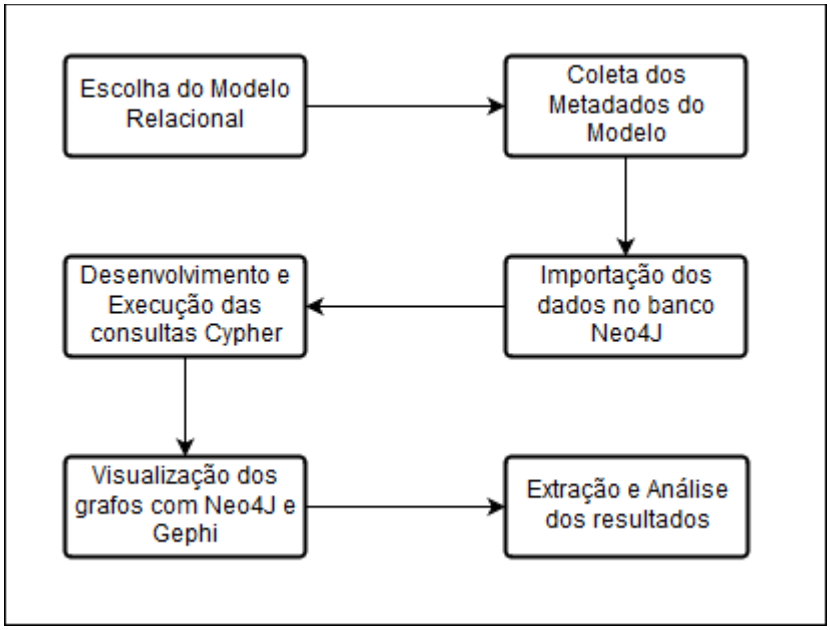

Figura 1: Fluxo do Processo de Metodologia

3. REVISÃO DE LITERATURA

Os conceitos sobre Teoria dos Grafos, Banco de Dados e Modelagem de Dados são importantes para o entendimento deste artigo. Nesta seção serão apresentados os seus possíveis benefícios perante o Desenvolvimento de Software e Análise dos seus Modelos de Dados.

3.1 Teoria dos Grafos

A Teoria dos Grafos consiste na análise de conjuntos de elementos que se relacinam entre si. São nomeados grafos pois podem ser representados graficamente onde cada vértice é representado por um ponto e cada aresta por uma linha que os liga. Um exemplo de grafo pode ser a representação de pessoas (vértices) e suas amizades com outras pessoas (arestas) (BONDY, J. A.; MURTY, U. S. R., 1976).

3.2 Banco de Dados Orientado à Grafos - Neo4J

Banco de Dados Orientado à Grafos (GDB) são modelos de sistemas NoSQL, que ao contrário do modelo tradicional-relacional, são especialistas em analisar e tratar os relacionamentos entre elementos em um grupo. No desenvolvimento deste artigo foi utilizado o banco de dados Neo4J.

Comparando o modelo relacional com novos modelos que estão surgindo, notamos grande importância ao modelo baseado em grafos. Esse modelo consiste na análise de objetos e seus relacionamentos. Na teoria de grafos existem certos padrões que podem ser explicados através de algoritmos e experimentos práticos. Um banco de dados orientado à grafos é otimizado para processar diversos tipos de relações entre suas entidades. Esse design permite a construção de modelos preditivos, detecção e correlação de padrões. Banco de Dados de Grafos são alternativas viáveis para os Modelos Relacionais de Banco de Dados (RDBMS). Áreas de atuação como Química, Biologia, Web Semântica, Redes Sociais e Sistemas de Recomendação são exemplos de aplicações que podem ser representadas de uma maneira mais natural através de grafos (MILLER, J. J., 2013).

3.3 Linguagem para Consultas e Manipulaçao de Dados – SQL e Cypher

Quando lidamos com sistemas de banco de dados, precisamos entender a linguagem utilizada para acesso aos dados e informações contidas no software.

Uma Linguagem de Manipulação de Dados é uma coleção de operações e regras para serem aplicadas em estruturas de um modelo de dados, com o objetivo de manipular, combinar e consultar dados. Sistemas de banco de dados relacionais (RDBMS) utilizam linguagem SQL (Sructure Query Language) para inserções, exclusões e atualizações de dados, alterações e criações de estruturas do modelo. SQL é baseado em álgebra relacional e cálculo relacional baseado em linhas (tuplas). (MILLER, J. J., 2013).

Para manipulação de dados em um banco de dados orientado à grafos como o Neo4J, a linguagem busca informações através de transversais entre os elementos do grafo.

Para se consultar informações de um grafo é necessário se utilizar das transversais entre seus elementos (vértices e arestas), onde cada um terá informações indexadas dos demais elementos nos quais são conectados. Isso representa um ganho de performance em comparação ao SQL (MILLER, J. J., 2013).

Uma das principais linguagens para se trabalhar com dados de grafos é o Cypher: Cypher é uma linguagem de consulta e manipulação de dados inspirada no SQL, que procura facilitar o uso de transversais no código (MILLER, J. J., 2013).

3.4 Visualização e Análise de Grafos com Gephi

Para análise e visualização de um grafo com alto número de vértices e arestas, foi utilizado o software Gephi em detrimento do Neo4J. Nesse caso o processo consiste na execução das consultas dentro do banco de dados Neo4J e importação do resultado, em formato de grafo, através do própria linguagem Cypher, para dentro de um projeto do software Gephi.

Gephi é um softwarde de código livre para análise e visualização de grandes grafos e redes neurais. É possível importar, visualizar, filtrar, manipular e exportar dados de diversos tipos de grafos e redes neurais. Possui ainda tecnologia de renderização 3D para análise de grafos em tempo real (BASTIAN, MATHIEU; HEYMANN, SEBASTIEN, 2009).

3.5 Modelagem de Dados

No fluxo de desenvolvimento de software, antes da etapa de codificação do sistema, é necessário realizar um estudo da modelagem de dados que vai ser necessária para concluir a codificação. Entende-se como Modelagem de Dados:

“A criação ou alteração das estruturas lógicas e físicas de um ou mais bancos de dados, para receber as informações necessárias dos usuários em uma organização para um conjunto de aplicações” (ELMASRI, RAMEZ; NAVATHE, SHAMKANT, 1989).

Segundo RAMEZ ELMASRI e SHAMKANT NAVATHE os objetivos da Modelagem são diversos:

- Satisfazer os requisitos de informações dos usuários e aplicações;
- Prover uma estrutura natural e fácil de entender das informações;
- Suportar o processamento necessário e performance requirida como tempo de resposta, tempo de processamento e espaço de armazenamento.

3.5.1. Problemas de performance referentes a Modelagem

Problemas de Modelagem de Dados são muito comuns em quase todo software. Relacionamentos entre tabelas realizados de forma não otimizada, falta de chaves-primárias e índices em tabelas e redundância desnecessária de dados são alguns exemplos de problemas que podem acarretar em má performance das aplicações. Sobre redundância de dados e performance podemos citar a Normalização dos Dados como uma prática: A Normalização de Dados garante a consistência dos dados e economiza espaço em disco (storage). Porém as vezes é necessário se ter uma certa redundância controlada para melhorar a performance das consultas SQL (ELMASRI, AMEZ; NAVATHE, SHAMKANT, 1989). Ainda sobre índices: Banco de dados precisam ser capazes de executar SQL de forma performática. Como o banco de dados normalmente é armazenado no disco, o RDBMS precisa conter as estruturas de dados e técnicas de pesquisa capazes de agilizar a procura no disco pelos registros solicitados no SQL. Os índices são arquivos auxiliares utilizados para este propósito (ELMASRI, RAMEZ; NAVATHE, SHAMKANT, 1989).

4. APRESENTAÇÃO DA PESQUISA

A pesquisa foi realizada a partir de metadados levantados de um grande sistema de informação do segmento hospitalar disponível no mercado. Por se tratar de um software de saúde, seus dados são críticos e precisa possuir uma performance que atenda os requisitos operacionais do cliente (hospitais e clínicas de saúde). Para isso,

é importante termos um entendimento sobre seu modelo de dados, o que inclui suas tabelas e relacionamentos.

Foram levantandos dados relacionados ao modelo de dados do sistema escolhido, tais como informações de tabelas, índices e constraints; importados os dados dentro do banco de dados Neo4J e aplicados algoritmos para identificação de alguns padrões, anomalias e demais descobertas sobre o modelo de dados do sistema escolhido. Foi feito também a exportação do grafo para o sistema Gephi para criação do grafo completo com as devidas marcações e cores para diferenciar as estruturas e visualizar o modelo de uma forma mais fácil.

A seguir serão apresentados as descobertas da pesquisa, através de gráficos, tabelas e descrições. Os scripts e gráficos utilizados nesse trabalho estão disponibilizados no GitHub, no repositório *lucprosa/ProjectRelationalToGraph*.

4.1 Dados utilizados na pesquisa e Modelo de Grafos escolhido

Para extração dos metadados para aplicação na pesquisa, foi utilizado linguagem SQL em uma conexão com banco de dados Oracle. Para criação dos grafos e consulta no Neo4J, foram utilizadas queries com a linguagem Cyper. Os scripts utilizados se encontram disponíveis para consultas no repositório do GIT já mencionado.

Foram exportados 2 arquivos no formato CSV, separados por vírgula. Foram divididos em informações relativas à tabelas e às suas chaves-estrangeiras, como mostra a figura abaixo:

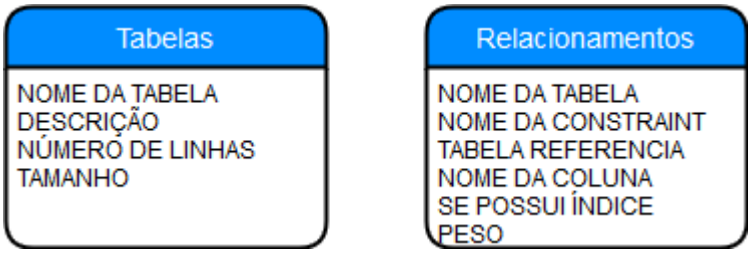

Figura 2: Estrutura de Dados Exportada do Banco Relacional

No banco de dados Neo4J foi criado o database e exportado os dados do CSV. Foram considerados os dados de Tabelas como “Vértices” e os seus Relacionamentos como “Arestas”:

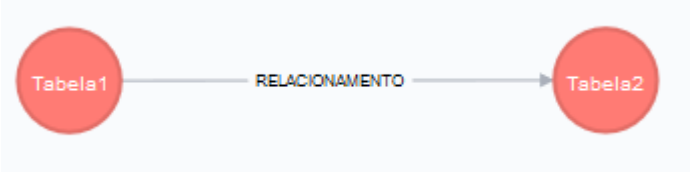

Figura 3: Modelo de Grafo Criado no Neo4J

No total foram importados 1946 nós e 3924 arestas para o banco de dados Neo4J.

O grafo gerado é do tipo não-direcionado.

Consulta(s) Utilizada(s):
qry01_import, qry02_tabs_cnt, qry03_rel_cnt

4.2 Principais Tabelas do Modelo (Tamanho, Quantidade de Relacionamentos e Índices Faltantes)

Na análise do modelo de dados, é importante conhecer quais são as suas principais tabelas e relacionamentos. Nesse item vão ser disponibilizados gráficos mostrando quais as tabelas com maior tamanho e as que possuem maiores relacionamentos (outras tabelas que fazem referência a ela).

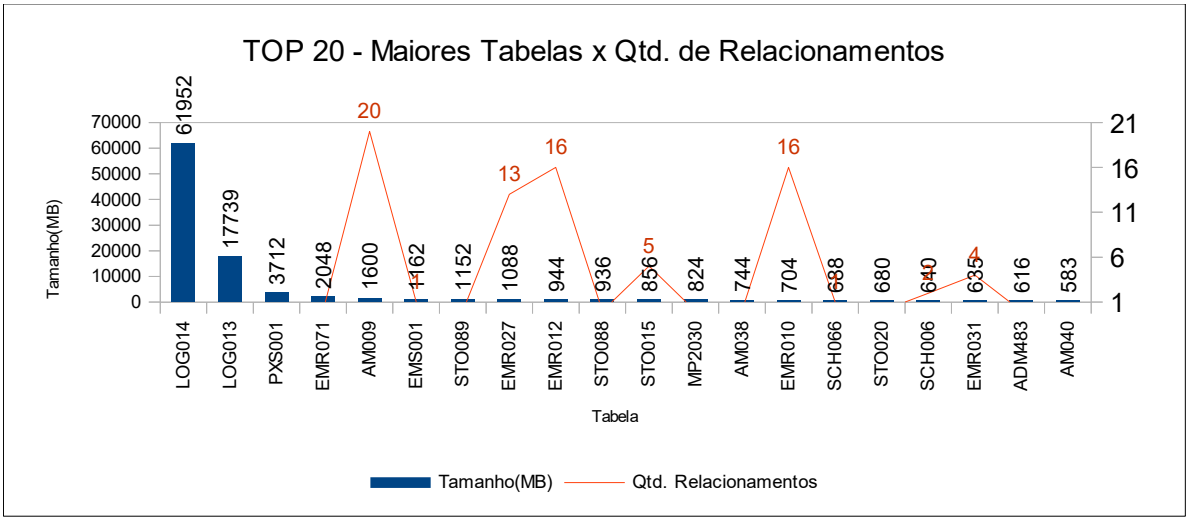

Figura 4: Gráfico Maiores Tabelas

Observando o gráfico da *Figura 3*, nota-se que as maiores tabelas do modelo são referentes à logs do sistema, e que essas não são referenciadas por outras tabelas. O ponto de atenção deve ficar em cima das tabelas que possuem muitos relacionamentos como a “AM009”, “EMR027”, “EMR012” e “EMR010.

O gráfico da *Figura 4 abaixo,* mostra a relação entre as tabelas que possuem maior quantidade de relacionamentos com a quantidade de índices faltantes em cada constraint.

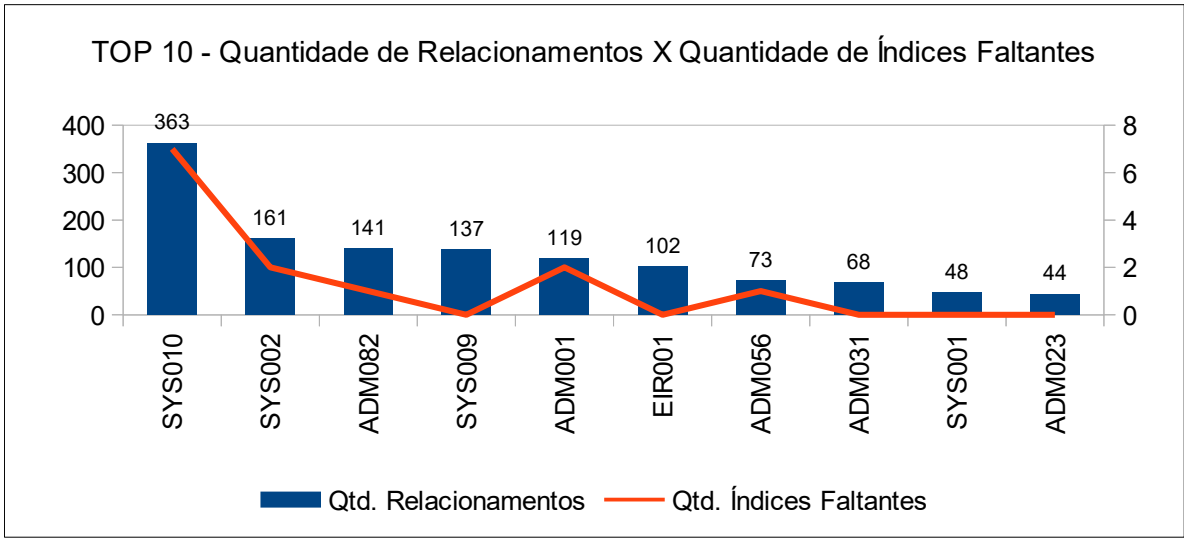

Figura 5: Gráfico Tabelas com Mais Relacionamentos

Nota-se que a tabela “SYS010” é a mais referenciada por outras tabelas do modelo, e que possui o maior número de índices faltantes em chaves-estrangeiras (6). Em cima dessa tabela foi gerado o grafo abaixo, que mostra os 363 relacionamentos do vértice “SYS010”, com destaque para as arestas que possuem atributo “HasIndex” com valor “N” (não possui índice na chave-estrangeira).

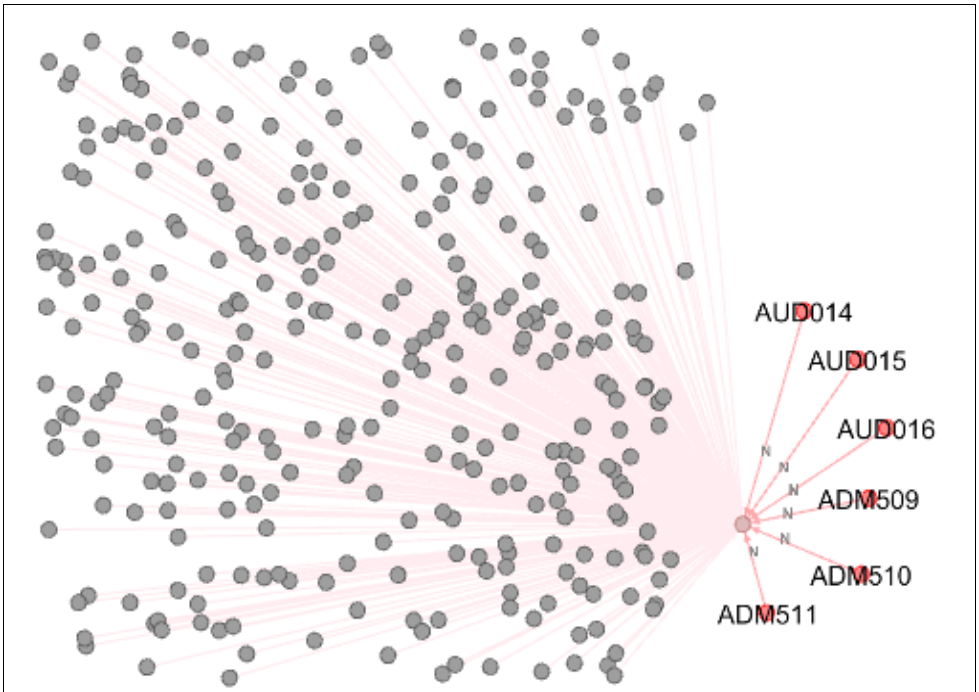

Figura 6: Grafo do vértice "SYS010" e seus relacionamentos

A criação de índices para colunas utilizadas como “Chaves-Estrangeiras” é boa prática para evitar perda de performance em consultas SQL e eliminar número de concorrência entre transações no banco de dados.

Especialmente para banco de dados Oracle, segundo DARL KUHN, SAM R. ALAPATI, e BILL PADFIELD (2012) a utilização de índices do tipo de Árvore (B-Tree) é muito importante para garantir:

- Aumento de performance de consutlas SQL;
- Unidicidade de Chaves Primárias e Chaves-Únicas;
- Reduzir problemas de contenção (locks) entre as tabelas pai e filho através de associação de chave-primária com chave-estrangeira.

No total foram levantadas 35 chaves-estrangeiras não indexadas e 159 tabelas sem nenhum relacionamento.

Consulta(s) utilizada(s):
qry06_tabs_isoladas , qry07_fks_sem_ind, qry12_top_tabs_tam, qry13_top_tabs_rel_list, qry14_top_tabs_rel_graph, qry15_10_top_tabs_rel.

4.3 Contagem de Triângulos e PageRank

Para analisar o nível de conectividade do grafo, foi aplicado algoritmo de contagem de triângulos, verificação de coeficiente de agrupamento e PageRank. Um triângulo vai ser o conjunto de 3 vértices interligados entre si a partir de arestas. O Coeficiente de Agrupamento mede o grau com que os nós de um grafo tendem a agrupar-se.
Foram encontrados 1496 triângulos no total.
Média de Coeficiente de Agrupamento ficou em 0.1187.

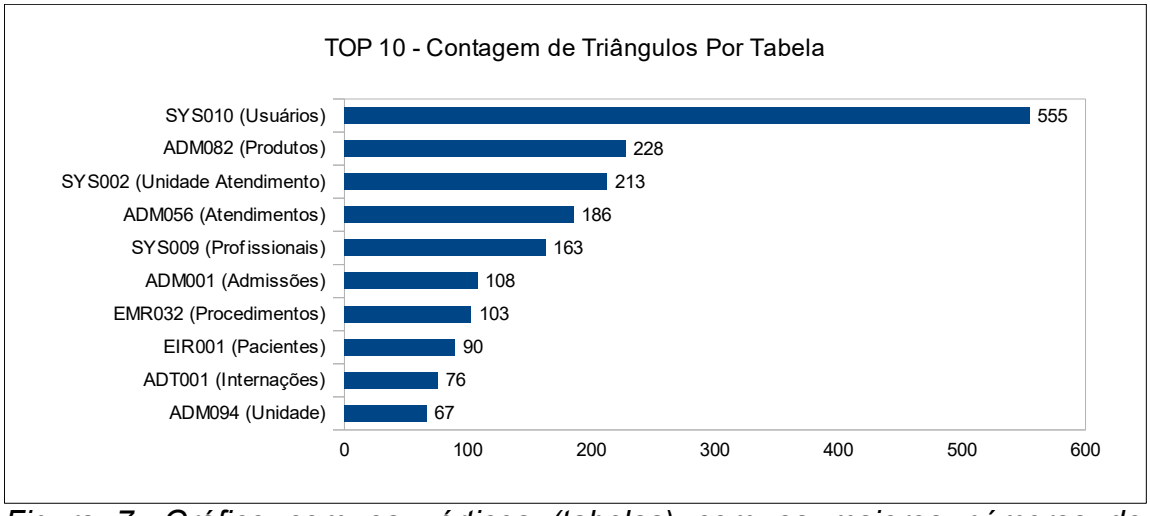

Figura 7: Gráfico com os vértices (tabelas) com os maiores números de triângulos

A figura 7 mostra a lista dos 10 vértices com maior quantidade de participações em triângulos no grafo. Nota-se uma alta disparidade em relação do primeiro da lista ao segundo. O vértice SYS010 faz referência à tabela de usuários do sistema estudado.

Essa tabela é extremamente importante pois é referenciada em diversas outras tabelas do modelo para garantir a consistência dos dados relacionados aos usuários que utilizam o sistema.

Em um modelo relacional, um triângulo, além de representar o nível de conexão entre três tabelas, pode ajudar a identificar certos padrões redundantes de relacionamento como colunas que possuem a mesma informação porém em tabelas diferentes ou ainda mais de uma chave-estrangeira para a mesma tabela, porém se referindo à colunas diferentes.

No exemplo da Figura 8 abaixo, entre os vértices “ADM082”, “ADM485”, “SYS002” e “SYS010” existem 4 triângulos formados:
ADM485 → ADM082 → SYS010 ← ADM485
ADM485 → ADM082 → SYS010 ← ADM485
ADM485 → SYS002 → SYS010 ← ADM485
ADM485 → SYS002 ← SYS010 ← ADM485

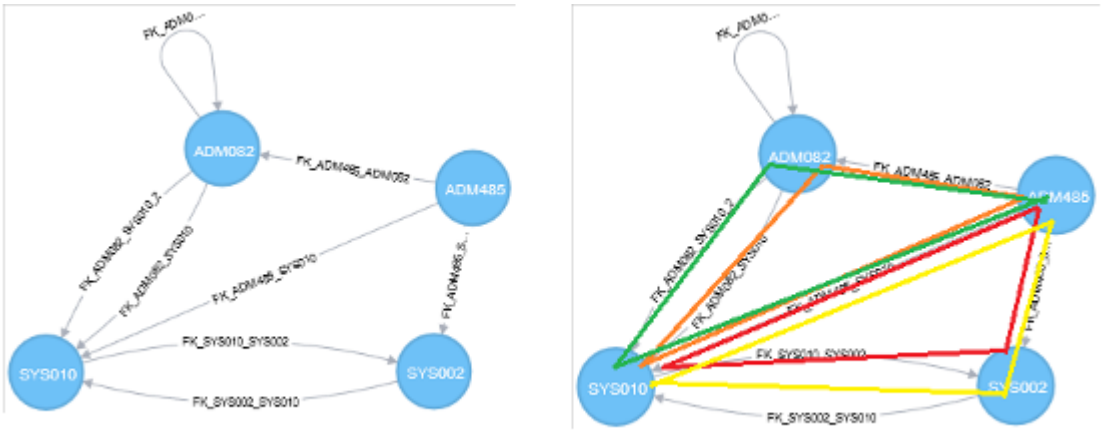

Figura 8: Representação de Grafo com Triângulos

Entre os vértices SYS010 e ADM082 são identificados duas arestas no mesmo sentido porém com atributos diferentes. Nesse caso, a tabela de produtos (ADM082) referencia a tabela de usuários (SYS010) duas vezes, com colunas diferentes.

Consultando todos os atributos das arestas desse relacionamento identificamos as colunas das chaves-estrangeiras. Nesse caso usa-se uma coluna referenciando o usuário que inseriu o registro e outra com o usuário que atualizou o registro da tabela de produtos (ADM082).

Figura 9: Resultado de consulta nos atributos das arestas entre dois vértices

Consulta(s) utilizada(s):

qry08_triang_cnt, qry09_triang_exe, qry10_rel_adm082_sys010

Foi utilizado também na análise o algoritmo de PageRank que mensura a importância de cada vértice no grafo, considerando o número e a qualidade de seus relacionamentos. Neste trabalho foi utilizado algoritmo de PageRank do pacote de funções APOC disponibilizado para o banco de dados Neo4J.

"Tabela"	"Triangulos"	"Coefficient"	"Betweenness"	"Score"
"SYS010"	555	0.008309005165057264	15754.455549067532	33.1612
"SYS002"	213	0.013990147783251231	18610.17524375205	24.20923
"SYS009"	163	0.016054368167044225	6483.239879429275	20.72696
"SYS126"	6	0.21428571428571427	0	15.18221
"SYS011"	13	0.19696969696969696	514.8101166807044	13.2356
"SYS028"	3	0.08333333333333333	1943	10.68883
"TSS001"	6	0.018461538461538463	1493.3965703092254	10.63847
"ADM001"	108	0.01328740157480315	3504.4050730435706	9.87354
"SYS001"	58	0.039057239057239054	10043.441446913734	9.30631
"EIR001"	90	0.015012510425354461	1977.141033156581	8.12138

Figura 10: Resultado de PageRank - TOP 10

Na figura acima temos a lista com as 10 principais tabelas do grafo de acordo com o PageRank. A tabela SYS010 foi considerada a tabela com mais importância no modelo, com um score de 33.16 de acordo com o PageRank.

Nessa tabela entende-se como Coeficiente (Coefficient) o grau de densidade de ligações dos vértices vizinhos à um vértice; e centralidade (Betweenness) mede o nível de importância de um vértice no grafo, baseado nos “Menores Caminhos” entre seus vértices e arestas. O algoritmo de PageRank utiliza essas medições para determinar qual os vértices ou nós mais importantes dentro de um grafo.

Consulta(s) utilizada(s): qry16_PageRank

5. DISCUSSÃO DOS RESULTADOS

A partir dos números levantados e dos gráficos criados é possível se ter um conhecimento sobre o modelo de dados do sistema estudado. A visualização de um modelo relacional, com suas tabelas e seus relacionamentos, em um formato de grafo facilita a análise da conectividade do modelo. O banco de dados de grafo, junto com a teoria dos grafos e seus algoritmos, possui a tecnologia ideal para fazer esse tipo de análise.

Nessa pesquisa foram levantadas 33 chaves-estrangeiras não indexadas. Sendo que algumas delas estão nas tabelas principais do modelo, como a SYS010 que á a tabela de usuários do sistema estudado. Nesse caso existe uma possibilidade de melhoria que seria a criação desses índices, pois como visto ao longo do artigo, o índice faltante pode acarretar em problema de performance em uma consulta SQL do sistema. Além da tabela com maior quantidade de relacionamento, a SYS010 também foi o vértice mais envolvido em triângulos do grafo. Isso demonstra o nível de conectividade da tabela e pode levantar uma hipótese de haver relacionamentos redundantes no modelo.

Com a consulta dos vértices de maior tamanho e fazendo a conexão com os vértices com maiores números de arestas, foi possível identificar as maiores tabelas do modelo que podem aparecer com maiores custos de performance nas consultas SQL do sistema. É interessante dar mais atenção para essas tabelas pois podem ser gargalo para aplicação caso não sejam devidamente indexadas ou ainda utilizadas de forma não otimizadas dentro do SQL.

Apesar do alto nível de conectividade no grafo, foram encontrados 159 vértices isolados, sem nenhum relacionamento. As tabelas referentes à esses vértices podem ser analisadas manualmente para identificar se faltam chaves-estrangeiras no modelo ou até se as tabelas não estão sendo utilizadas de forma correta e performática.

Com os resultados obtidos, entende-se que é possível agregar conhecimento ao modelo a partir desse tipo de análise através de grafos e que existem possibilidades de melhorias na pesquisa que podem ser executadas visando melhorar os resultadas e as descobertas feitas.

6. CONSIDERAÇÕES FINAIS

Este artigo documentou a análise de um modelo de dados de um software tradicional através de um banco de dados de grafos, com o intuito de fazer uma experimentação, buscando obter novos conhecimentos assim como identificar padrões e anomalias no modelo. Esta análise visa conhecer melhor um modelo de dados, identificar padrões em seus relacionamentos, principais tabelas, possíveis gargalos de performance, etc, assim levantando possibilidades de melhorias em sua modelagem e no sistema como um todo. O termo experimentação foi colocado no titulo deste artigo, pois os resultados obtidos não seriam somente os práticos, mas também abordariam todo o estudo feito sobre o tema escolhido e as possíveis descobertas de informações que poderiam ou não ser levantadas nesse primeiro momento. Além do resultado objetivo, este trabalho visou também o aumento de conhecimento técnico sobre grafos, Neo4J e linguagem Cypher.

Como continuidade para este trabalho, pode-se incluir outros objetos na análise do modelo como procedures e functions de bancos de dados, para se analisar conectividade e dependência entre as tabelas utilizadas nesses objetos. Outra possibilidade seria importar dados de consultas SQL do software para dentro do Neo4J. Nesse caso se consideraria como vértices o identificador do SQL e suas tabelas utilizadas, e como arestas os joins do SQL entre as tabelas. Nesse grafo poderia ser analisado a existência de padrões entre consultas SQL de uma aplicação, identificar anomalias em seus relacionamentos ou mesmo o nível de dependência entre seus códigos.

REFERÊNCIAS

CODD, E.F. “Extending the Database Relation Model to Capture More Meaning” in ACM Trans. on Database Sys. Vol 4 No 4, 1979.

BONDY, J. A.; MURTY, U. S. R. "Graph Theory With Applications".

Departmnent·of Combinatorics and Optimization - University of Waterloo, Ontario, Canada, 1976.

ELMASRI, RAMEZ; NAVATHE, SHAMKANT. “Fundamentals Of Database Systems” - Sixth Edition, 1989.

GYSSENS, MARK; PAREDAENS, JAN; VAN GUCHT, DIRK. “A Graph-Oriented Object Model for Database End-User Interfaces”, 1990.

HEWITT, R. ”A Searching and Reporting System for Relational Databases Using a Graph-Based Metadata Representation”. Hewitt Consulting, San Diego, California, 2005.

MILLER, J. J. “Graph Database Applications and Concepts with Neo4j” Georgia Southern University, 2013.

O'NEIL, MARK. “Design Your Own Database Concept to Implementation or How to Design a Database Without Touching a Compute”, 2004.

TRAJANO, FRANK. “A Gentle Introduction to Relational and Object Oriented Databases”. The Olivetti & Oracle Research Laboratory - ORL Technical Report TR-98-2.

UFRGS – Organizado Por TATIANA GERHARDT e DENISE SILVEIRA. “Métodos de Pesquisa”. UFRGS, 2009.

WILSON, ROBIN J. “Introduction to Graph Theory“ - Fourth Edition, 1996.

BASTIAN, MATHIEU; HEYMANN, SEBASTIEN. “Gephi : An Open Source Software for Exploring and Manipulating Networks” - Gephi, WebAtlas, 2009 (Disponível em “<https://gephi.org/publications/gephi-bastian-feb09.pdf>”).

KUHN, DARL; ALAPATI, SAM R. E PADFIELD BILL. “Expert Indexing in Oracle Database 11g – Maximum Performance for you Database” - Apress, 2012.