Comparação de Metodologias de Migração de Bancos de Dados Relacionais para Bancos Orientados a Documentos

Vanessa C. O. Souza*
UNIFEI - Universidade Federal de Itajubá
Itajubá, MG, Brasil
vanessasouza@unifei.edu.br

Melise M. V. Paula
UNIFEI - Universidade Federal de Itajubá
Itajubá, MG, Brasil
melise@unifei.edu.br

Tiago C. G. M. Barros
UNIFEI - Universidade Federal de Itajubá
Itajubá, MG, Brasil
tiagocgorgulho@gmail.com

ABSTRACT

Companies have migrated data from relational databases to NoSQL databases to improve their business through more active services at a lower operating cost, especially by the adoption of cloud services. This process is called Data Migration and is considered by some authors one of the biggest challenges in systems engineering today. Although it is advantageous, the process of migrating data from the relational model to NoSQL models is not trivial and has led to the development of different methodologies for this purpose. The objective of this work was to analyze and compare three different migration methodologies between Relational and NoSQL Document Oriented databases under the following aspects: algorithm input, method documentation, migration process and generated documents. For that, different relational models were empirically migrated using such methodologies, allowing the analysis of the evaluated aspects. The results show that there is no consolidated way to perform the migration and that the method to be chosen depends on the context of the application. So, scenarios that show when to use each method are presented. Although not performing computational tests, this work provides suggestions and insights through the evaluation of the migration processes under the theoretical models. It expected that the results presented here will help IT managers decide on the best data model migration methodology to follow in their actual projects.

KEYWORDS

Data migration metodology, NoSQL, Oriented Document

1 INTRODUÇÃO

Questões como escalabilidade, impedância entre modelos e a rigidez das propriedades ACID fazem com que o modelo relacional não seja o mais apropriado para algumas aplicações, como àquelas relacionadas à web 2.0, *business intelligence* e *Big Data*, que podem apresentar diferentes requisitos para consistência, escalabilidade e segurança, por exemplo [1, 18, 22, 24].

Como alternativa de persistência de dados, surgiram os chamados Bancos de Dados NoSQL (*Not Only SQL*) que atendem aos requisitos de computação distribuída em larga escala, oferecendo escalabilidade, alta disponibilidade, desempenho e confiabilidade. Os bancos NoSQL são mecanismos de persistência não relacionais, distribuídos, de código aberto e projetados para lidar com dados não estruturados [23]. A tendência de utilizar os mecanismos de persistência NoSQL é refletida no *ranking* oferecido pelo *site* DB-Engines [1], no qual, entre os 10 SGBDs mais populares, 4 são NoSQL. O SGBD orientado a documentos MongoDB ocupa o primeiro lugar entre os NoSQL. Uma revisão atual sobre os sistemas NoSQL pode ser vista em [7].

Com o mercado NoSQL em ascensão, empresas têm explorado tais mecanismos de persistência com o intuito de melhorar seus negócios por meio de serviços mais ágeis a um custo operacional menor [16]. Em especial, as organizações têm utilizado cada vez mais serviços em nuvem, uma vez que reduz os custos operacionais em comparação com a hospedagem interna [8, 12, 13]. Neste contexto, a engenharia dos sistemas organizacionais, no que tange o armazenamento e desenvolvimento de *software*, precisa se adequar a esse novo cenário e torna-se necessário converter dados do modelo relacional para os modelos NoSQL. A este processo, dá-se o nome de Migração de Dados.

Motivações adicionais que levam organizações e desenvolvedores a realizar o processo de migração incluem aumento na quantidade de dados da organização, mudanças em sua infraestrutura, problemas econômicos e recursos insuficientes no mecanismo de persistência utilizado, principalmente no que diz respeito ao desempenho [2, 14, 21]. Esse último aspecto, em especial, refere-se à complexidade de consultas SQL que podem ser simplificadas nos mecanismos NoSQL. Ademais, projetos de migração são realizados porque apoiam os objetivos do negócio [15, 16].

Apesar de oferecer vantagens, o processo de migração de dados do modelo relacional para modelos NoSQL não é trivial [12, 16] e impulsionou o desenvolvimento de diferentes metodologias para esse fim [3, 14, 16, 19, 20, 26]. Tal diversidade corrobora a complexidade do problema em questão e cria uma nova demanda prática :

[1] https://db-engines.com/en/ranking, acessado em Outubro/2019.

como escolher uma metodologia em detrimento de outra? Alguns autores consideram o mapeamento entre modelos de banco de dados relacional e NoSQL um dos maiores desafios atuais. Isso porque o processo precisa garantir que todos os relacionamentos foram corretamente mapeados, não resultando em uma estrutura de dados semanticamente pobre [20, 21].

Sendo assim, o objetivo deste trabalho foi analisar e comparar três diferentes metodologias de migração entre bancos relacionais e NoSQL Orientado a Documentos sob os seguintes aspectos: entrada do algoritmo, documentação do método, execução do processo de migração e documentos gerados. Para tanto, foram analisadas três metodologias diferentes: metodologia baseada em grafos [26], metodologia baseada em consultas [19] e a metodologia baseada na definição dos níveis físico e lógico dos dados [16]. A escolha de tais metodologias baseou-se no fato de elas considerarem o processo migratório dos dados sob perspectivas totalmente diferentes e apresentarem de forma clara os algoritmos de migração.

O restante deste artigo está dividido em cinco seções onde: a sessão 2 apresenta a revisão bibliográfica; a metodologia utilizada para execução desse trabalho é apresentada na seção 3. A sessão 4 apresenta as metodologias avaliadas; na sessão 5 são apresentados os resultados obtidos e, finalmente, a seção 6 apresenta as discussões e considerações finais.

2 MIGRAÇÃO DE DADOS

O termo migração de dados é definido como sendo um processo único que visa a migração de dados de uma estrutura de origem para uma estrutura de destino, enquanto ambas as estruturas diferem no conceito e/ou no nível físico [21]. No contexto deste trabalho, a migração de dados tem o objetivo de transcrever dados de bases de dados do modelo relacional, para o modelo NoSQL Orientado a Documentos.

O processo de migração possui duas etapas importantes: a migração do modelo de dados segundo os requisitos do sistema de destino, pois os esquemas de dados dos modelos são completamente diferentes; e, em seguida, a transferência desses dados para o banco de dados de destino [16]. A migração de um banco de dados relacional para um NoSQL caracteriza um problema chamado heterogeneidade estrutural, que requer a análise de fatores como tipos de estruturas, dados, relacionamentos e restrições, que são determinados pelo modelo de dados original e o de destino [6]. Neste trabalho o foco foi na primeira etapa (migração do modelo de dados). Não houve preocupação com a conversão de tipos de dados, restrições de integridade e *triggers*, por exemplo.

Na literatura são encontradas diferentes metodologias de migração do modelo de dados relacional para o NoSQL Orientado a Documentos. Em [14], cada tabela relacional torna-se um documento no modelo orientado a documentos. Havendo chave estrangeira na tabela, os dados da tabela relacionada são automaticamente embutidos. Esta abordagem é bastante parecida com a abordagem proposta por [3]. No algoritmo proposto em [20] é permitido ao usuário escolher entre embutir ou não dados relacionados. Em [26] é proposta uma metodologia baseada em grafos para realizar a migração do modelo de dados. Em [16] a migração é feita seguindo um tipo de engenharia reversa a partir do modelo relacional físico. Já em [19], uma metodologia baseada em consultas é proposta.

Algumas ferramentas têm sido desenvolvidas para realizar a migração de dados do banco de dados relacional para o banco de dados NoSQL, tais como o Apache Sqoop e o DataX [26]. Especificamente para o modelo orientado a documentos a literatura apresenta alguns protótipos [14, 20, 21]. A migração de dados sob a perspectiva de utilização de serviços em nuvem é também objeto de estudo de diversos artigos encontrados na literatura [9, 12, 13, 25].

2.1 Trabalhos Correlatos

Alguns trabalhos encontrados na literatura propõem-se a comparar metodologias de migração de bancos relacionais para NoSQL. Em [17], cinco diferentes metodologias são apresentadas. Os autores compararam as metodologias a partir dos seguintes parâmetros : banco de dados utilizado, aplicação, escopo futuro, vantagens e desvantagens.

Quatro diferentes metodologias de migração são apresentadas em [4]. Os autores apenas descrevem tais metodologias. Em [6], cinco metodologias de migração são apresentadas. O trabalho apresenta uma tabela comparativa, com os seguintes atributos : SGBD de origem, conversão (diz se a metodologia pode ser implementada de forma automática ou semi-automática), usuário (se a migração pode ser feita por um usuário comum ou especialista), e sistema destino (modelo de banco de dados NoSQL).

Nos três trabalhos acima citados, as metodologias não são testadas em esquemas hipotéticos, nem em aplicações reais. Os autores também não indicam cenários onde cada metodologia poderia ser melhor aplicada. Verifica-se, portanto, uma lacuna em trabalhos com o objetivo de auxiliar a tomada de decisão sobre qual metodologia utilizar no caso de um processo de migração de um modelo relacional para um modelo NoSQL orientado a documentos.É justamente esse ponto que diferencia este estudo dos trabalhos apresentados.

3 METODOLOGIA

As etapas realizadas foram:

Escolha das metodologias de migração

- Nessa etapa, realizou-se o levantamento bibliográfico sobre metodologias de migração de banco de dados relacional para NoSQL existentes. A pesquisa foi feita em artigos acessíveis a partir do Google Acadêmico, até o ano de 2018.
- Com base no levantamento obtido, avaliou-se metodologias que fizessem o mapeamento entre os modelos relacional e orientado a documentos; que apresentasse de forma clara o algoritmo de migração; e que apresentassem diferenças conceituais na forma de realizar a migração. Apesar de considerar os quesitos acima citados, essa foi uma etapa de análise qualitativa. Ou seja, fundamentalmente interpretativa [10].
- Três metodologias foram escolhidas : metodologia baseada em grafos [26], metodologia baseada em consultas [19] e a metodologia baseada na definição dos níveis físico e lógico dos dados [16]. A descrição dessas metodologias pode ser vista na seção 4.

Definição dos modelos relacionais teóricos e das consultas SQL.

- Esquemas hipotéticos foram levantados, considerando diferentes níveis de dependência referencial entre as tabelas,

representando diferentes cenários. O objetivo foi obter diversidade para analisar o comportamento das metodologias a partir de diferentes modelos.
- Ao todo, foram definidos dez modelos relacionais. O mais simples apresentava entidades sem nenhum relacionamento. A partir daí, as dependências e estruturas relacionais foram sendo introduzidas, tais como herança, auto-relacionamento, etc. O modelo mais complexo apresentava também relacionamento entre todas as entidades.
- Para cada modelo, foram definidas consultas que envolvessem junção de diversas entidades e que estivessem associadas aos requisitos do negócio representadas pelo modelo.
- A escolha dos modelos e de suas consultas associadas foi supervisionada por três profissionais com experiência em SGBD Relacional e NoSQL, tanto em atividades profissionais, quanto acadêmicas. Um dos especialistas também tinha ampla experiência em migração de dados.

Processo de migração.

- Nessa etapa, cada metodologia foi minuciosamente estudada. Os algoritmos foram reproduzidos e aplicados aos modelos relacionais previamente selecionados.

Avaliação das metodologias.

- Essa etapa foi realizada em paralelo com a etapa anterior, considerando os seguintes aspectos: entrada do algoritmo, documentação do método, execução do processo de migração e documentos gerados.
- No quesito **entrada do algoritmo**, avaliou-se como o algoritmo de migração é inicializado. Ou seja, se era a partir do Modelo Entidade-Relacionamento, do Modelo Relacional ou do Modelo Físico. Além disso, avaliou-se também se eram necessárias outras informações, como a definição à priori das consultas que seriam executadas no banco NoSQL.
- No quesito **documentação do método**, avaliou-se o artigo de referência onde a metodologia foi apresentada. A questão principal envolvida nesse quesito era sobre o quão reproduzível seria a metodologia a partir do artigo de referência. Tecnicamente, esse item foi avaliado em função da facilidade de compreender e reproduzir o algoritmo apresentado. Além disso, avaliou-se também a presença ou não de exemplos que ajudassem na compreensão do algoritmo; e a presença ou não de testes de performance para o método proposto.
- No quesito **execução do processo de migração**, avaliou-se a aplicabilidade da metodologia nos diferentes modelos relacionais teóricos e a as dificuldades envolvidas.
- No quesito **documentos gerados**, avaliou-se a quantidade de documentos gerados e a semântica dos mesmos. Neste caso, a semântica está relacionada ao cenário proposto por cada modelo.

Ressalta-se que, em relação aos aspectos considerados nessa etapa da metodologia, parte da análise foi qualitativa e subjetiva. Itens como facilidade de compreender o algoritmo e dificuldades obtidas no processo de migração refletem as percepções dos especialistas e podem ser influenciadas pela percepção e experiência de quem analisa a metodologia.

A próxima seção apresenta resumidamente as metodologias avaliadas neste estudo.

4 METODOLOGIAS AVALIADAS

Apesar de seguir caminhos diferentes, as metodologias de conversão de modelo de dados relacional para NoSQL apresentam uma mesma linha de execução, que consiste no processo de desnormalização das informações por meio de aninhamentos entre tabelas.

As metodologias baseada em grafos [26] e baseada em consultas [19] podem ser utilizadas para migrar sistemas relacionais para os modelos NoSQL chave-valor, orientado a colunas e orientado a documentos. No entanto, neste trabalho as análises consideraram apenas o modelo orientado a documentos.

4.1 Metodologia Baseada em Grafos

Proposta por Zhao e colaboradores [26], a metodologia baseada em grafos foi desenvolvida para a realização da conversão do modelo de dados relacional para um modelo de dados NoSQL qualquer. A metodologia utiliza os conceitos de grafos e baseia-se no processo de aninhamento de tabelas, a fim de melhorar o desempenho das consultas.

O início do processo de migração se dá com a construção de um grafo $G =< V, E >$, onde V é o conjunto de vértices, representando as tabelas do banco de dados relacional, e E é o conjunto de arestas direcionadas no grafo, representando todas as dependências entre as tabelas do banco de dados relacional. Dependência significa que uma tabela faz referência a outra por chave estrangeira.

De posse do grafo G, aplica-se o algoritmo. O laço principal do algoritmo a partir do conjunto P, que contém os nós que são folhas, ou seja, não possuem arestas chegando nele. Nesse laço os vértices (u) que fazem parte do conjunto P são visitados. Caso ele tenha predecessor, ou seja, arestas que chegam nele, as tabelas são aninhadas e a aresta que realizava a ligação entre os nós é eliminada. Assim, o algoritmo segue recursivamente até percorrer todos os vértices do grafo.

Como saída, o algoritmo apresenta o conjunto S das chaves estrangeiras que relacionavam as tabelas do modelo relacional, e serviram para a realização do aninhamento dessas tabelas no modelo NoSQL. Um documento pode ser aninhado diversas vezes. Percebe-se portanto que a redundância de dados é o ponto fraco dessa metodologia. O ponto forte é a otimização das consultas, uma vez que haverá documentos específicos para todo tipo de consulta necessária.

4.2 Metodologia Baseada em Consultas

Li e colaboradores [19] sugerem a migração dos modelos através da metodologia baseada em consultas (do inglês *Query-Oriented Data Modeling (QODM)*). Nessa abordagem de migração, deve-se considerar quais consultas serão realizadas no banco de dados, a fim de aumentar o desempenho da busca, uma vez que operaçoes de junção não são aconselháveis em bancos de dados NoSQL.

Além dos requisitos de consulta, a metodologia proposta recebe como entrada a estrutura de dados armazenados, descrita pelos autores a partir de um diagrama UML, representando o banco de dados relacional por meio de classes. Essas classes são divididas em duas categorias : classes de consultas (*query classes*) e classes incluídas (*included classes*). As classes incluídas são aquelas que precisam de outras para serem representadas.

O algoritmo proposto inicia-se com a criação do conjunto T formado pelas tuplas ($< C_k, C_t >$). A partir das consultas que serão executadas no SGBD NoSQL, extrai-se C_k e C_t. C_k é um conjunto de classes que participam como chave da consulta e C_t é um conjunto de classes que participam do resultado da consulta. Após inicializar os conjuntos que armazenarão as classes agregadas ($A_classes$) e as classes de índice (I_{set}), o conjunto T é ordenado.

O diferencial dessa metodologia é a possibilidade de gerar documentos referenciados (classes de índice), além dos aninhados. Sendo assim, o objetivo da ordenação de T é fazer com que as tuplas com menos classes participantes da consulta, ocupem os primeiros lugares do conjunto e, com isso, agregar dados para consultas simples e criar entidades de índice para consultas complexas.

O eixo principal de funcionamento do algoritmo é o laço que irá verificar cada tupla $< C_k, C_t >$. Se alguma das classes desta tupla ainda não pertencer ao conjunto $A_{classes}$, então todas deverão ser agregadas e a tupla será armazenada no conjunto A_{set}. Se todas as classes da tupla já fizerem parte do conjunto $A_{classes}$, as classes da consulta deverão ser referenciadas por índices, e a tupla deverá ser armazenada no conjunto I_{set}. Por fim, tem-se o modelo de dados criado, com as entidades agregadas a partir das consultas simples e entidades de índices para consultas mais complexas.

O aspecto relativo às consultas é, por vezes ignorado nos trabalhos de migração de dados. Em geral, bancos NoSQL são orientados à consulta, ou seja, a modelagem dos dados e consequente organização física são dependentes das consultas realizadas na base [5, 23]. Esse aspecto, portanto, possui relação direta com a performance das consultas.

4.3 Metodologia baseada na definição dos níveis físico e lógicos dos dados

Nesta abordagem, a migração dos dados passa por três passos: nível físico dos dados, primeiro nível lógico dos dados e segundo nível lógico dos dados [16].

Na etapa do nível físico, as chamadas *Table-like-structure - TLS* são identificadas e caracterizadas. Cada tabela física e cada *view* são TLS. Na etapa de primeiro nível lógico dos dados, os metadados do nível físico são acrescidos de informações lógicas, em linguagem natural. As chaves *surrogate* são substituídas pelas chaves primárias originais, mesmo que essas sejam de campos longos, como texto. Nesta etapa as TLS também são classificadas como: a) codificadoras, que armazenam entidades que codificam alguma informação; b) entidade simples, que, sozinhas, armazenam todos os dados de um objeto; c) entidade complexa, que necessita de um conjunto de tabelas para armazenar os dados de um objeto; N:N-link, que armazena tabelas oriundas de relacionamento N:N.

No segundo nível lógico dos dados as TLS são utilizadas para gerar o esquema inicial dos documentos e, consequentemente, a semântica do negócio. Para tanto elege-se uma TLS especial, nomeada *TLS-in-focus*. A escolha da *TLS-in-focus* pode ser feita com base em algum requisito de consulta e, nesse caso, a migração pode acontecer em apenas uma parte do banco; ou pode-se realizar a migração total do banco elegendo cada TLS como uma *TLS-in-focus*. Nessa fase, por meio das relações entre as TLS são gerados os aninhamentos. Os autores apresentam diferentes algoritmos para realizar essa etapa, incluindo uma busca em largura (*Breadth-First Search - BFS*).

Este algoritmo recebe como parâmetro o *noRaiz*, que é a *Tls-in-focus*, e os *links* que estão relacionados a este nó, ou seja, as chaves estrangeiras. Em seguida, cria-se uma fila e o *noRaiz* é enfileirado. A partir disso, enquanto a fila não estiver vazia, acontece a busca em largura, onde cada *link* desse nó é visitado. Como este algoritmo permite que cada *link* seja explorado apenas uma vez, se ele ainda não foi explorado, a TLS, chamada de *noFilho*, é visitada e adicionada na fila. No final da execução deste algoritmo, tem-se a relação de todos os nós visitados. Esses nós correspondem as tabelas do modelo relacional que serão agregadas formando o documento no modelo NoSQL. Esse documento é chamado de *template* e é sugerido pelos autores que o mesmo seja refinado por um especialista.

5 RESULTADOS

Embora tenham sido analisados diferentes modelos relacionais hipotéticos, neste trabalho a ilustração dos resultados foi obtida considerando o esquema relacional da Figura 1: uma aplicação para o gerenciamento de informações sobre vendas produtos, categorias, clientes e pedidos. Para melhor visualização, os atributos foram omitidos. Ressalta-se, portanto, que os resultados apresentados não são oriundos da análise desse único modelo. Ao apresentar um dos modelos estudados, pretende-se contribuir para o entendimento das metodologias, com um passo a passo da execução dos algoritmos e apresentar discussões acerca da semântica e quantidade dos documentos gerados.

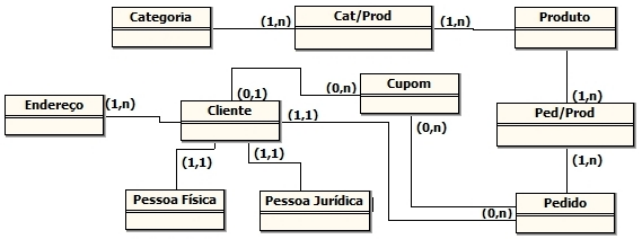

Figura 1: Diagrama relacional relativo a um cenário de vendas.

As Figuras 2, 3 e 4 apresentam, respectivamente, os resultados obtidos pelos processos de migração das metodologias baseada em grafos (M1), baseada em consultas (M2) e baseada na definição dos níveis físico e lógicos dos dados (M3). O lado esquerdo das Figuras representa a entrada do algoritmo e o lado direito, o modelo orientado a documentos obtido. Cada elipse indica um documento. Elipses que contém outras elipses significa que o documento incorpora as informações, isto é, houve aninhamentos.

A Figura 2a representa a modelagem em grafo sugerida pela metodologia M1. Os nós em vermelho são os chamados nós-folhas, ou seja, possuem grau de entrada, mas não possuem grau de saída. O algoritmo começa então pelo conjunto $P = (1, 2, 6)$ e visita os nós predecessores de cada um deles, realizando os aninhamentos. No caso do nó 1, o predecessor é apenas o nó 3. Sendo assim, o nó 1 é aninhado ao nó 3 e a aresta é removida (Figura 2b). O mesmo acontece com o nó 2 que, além do nó 3, possui como predecessor também o nó 5. Depois de processados os nós 1 e 2, o algoritmo irá processar o nó 6, que possui os predecessores 4, 7, 8, 9 e 10. Ao

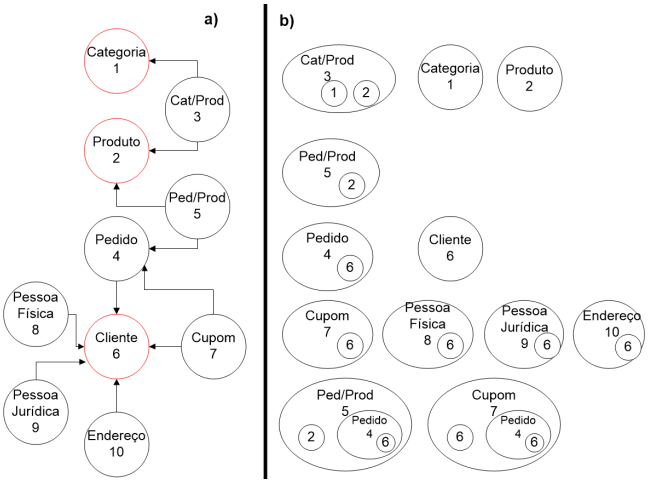

Figura 2: Resultado da migração utilizando a M1. 2a) Grafo das relações existentes no esquema da Figura 1. Cada nó corresponde a uma tabela do modelo relacional, as arestas indicam que há uma chave estrangeira do nó origem para o nó destino. Os nós folhas são identificados pela cor vermelha. 2b) Modelo orientado a documentos gerado.

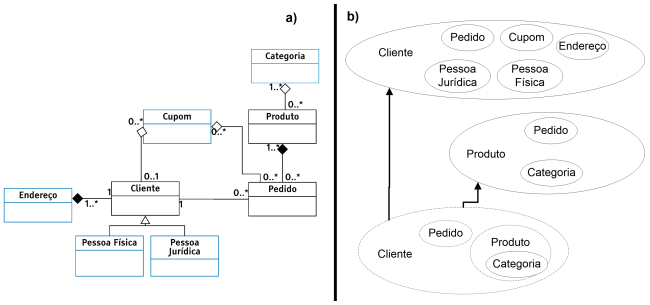

Figura 3: Resultado da migração utilizando a M2. 3a) Diagrama de Classes UML derivado do esquema relacional da Figura 1. As classes em azul são classes incluídas. 3b) Modelo orientado a documentos gerado a partir dos requisitos de consulta.

embutir o nó 6 no nó 4, a aresta é removida e o nó 4 passa a pertencer ao conjunto P. No entanto, neste momento, o nó 4 já embute o nó 6 e novos aninhamentos são feitos sobre esse nó já aninhado.

Para M2 e M3, foram definidas 3 consultas: ($q1$) listar os números e status dos pedidos do cliente a partir do CPF do cliente; ($q2$) listar os números dos pedidos e respectivos produtos (identificador, descrição e nome da categoria) a partir do CPF do cliente; ($q3$) listar os pedidos que contenham o produto com o identificador informado.

Na Figura 3a, o diagrama relacional da Figura 1 foi transformado no Diagrama de Classes UML apresentado. As classes na cor azul são classes incluídas. Apesar do diagrama relacional apontar que podem existir instâncias de Categoria que não estejam associadas a nenhum Produto, a classe Categoria foi considerada incluída porque cada produto precisa estar associado a pelo menos uma Categoria. Isto é, a informação sobre um Produto depende da classe Categoria para estar completa.

Para os requisitos de consulta acima mencionados, obtém-se o conjunto $T : t1 :< \{cliente\}, \{pedido\} >; t2 :< \{cliente\}, \{pedido, produto, categoria\} >; t3 :< \{produto\}, \{pedido\} >$. Depois da ordenação de T, a execução será feita na ordem $t1$, $t3$ e $t2$. Assim, ao processar $t1$, pedido será aninhado à Cliente produzindo um novo documento. Ao processar $t3$, a entidade Pedido será aninhada à Produto, uma vez que Produto não foi aninhado anteriormente. Por último, $t2$ será processada. Como as entidades cliente, pedido e produto já foram anteriormente aninhadas, a nova classe será uma classe de índice que irá referenciar os documentos resultantes da execução de $t1$ e $t3$ (Figura 3b).

Para a M3, as Figuras 4a e 4b apresentam o primeiro nível lógico das entidades Cliente e Produto, que são as TLS necessárias para responder aos requisitos de consulta definidos. Utilizando o metamodelo sugerido pela metodologia M3, mapeia-se os atributos e *links* de cada TLS. Os *links* passam pelo algoritmo de busca em largura, realizando os aninhamentos necessários. Para responder ao requisito de consulta ($q1$), a *TLS-in-focus* será Cliente. Essa TLS é enfileirada e seus *links* são visitados, realizando os aninhamentos. Ao visitar uma TLS por meio do *link*, os *links* dessa TLS (ou seja, as TLSs referenciadas) são enfileiradas. Por esse motivo, após visitar os *links* da TLS Cliente (Endereço, Cupom, Pessoa Física, Pessoa Jurídica e Pedido), as TLS definidas em Pedido que não foram ainda visitadas são enfileiradas, o que culmina na visita de Produto que, por sua vez, referencia Categorias. O resultado final é um documento que contém Cliente, Pedido e Produto. Esse documento responde ao requisito de consulta ($q1$) e ($q2$). Para responder à consulta ($q3$), a TLS Produto será processada e seus *links* (Pedido e Categoria) são aninhados. Executando esse processo, as informações de Cliente também são aninhadas, uma vez que Cliente refere-se à Pedido. Mas, considerando a semântica da consulta, realizou-se o refinamento sugerido pelos autores e o documento gerado não possui informações de Cliente. O resultado do algoritmo para os requisitos de consulta pode ser visto na Figura 4c.

As próximas subseções apresentam as análises das metodologias considerando a entrada do algoritmo, documentação do método, execução do processo de migração e documentos gerados.

5.1 Entrada do Algoritmo

Em M1 a migração parte do domínio lógico do banco de dados relacional, o qual é mapeado para um grafo direcionado (Figura 2a). O mesmo acontece com M2, que tem como entrada o Diagrama Relacional (DR) que é transformado em um diagrama de classes UML (Figura 3a). No entanto, percebe-se que mesmo partindo do DR, os algoritmos têm visões diferentes da migração, uma vez que ao gerar o diagrama de classes UML e rotular as classes como sendo de consulta ou incluída, a M2 embute aninhamentos à priori.

A M2 também tem como entrada as consultas. Essa é uma característica que pode ser vista como vantagem e/ou desvantagem. Nos bancos orientados à consulta, os documentos gerados poderão ter um desempenho melhor quando comparado àqueles que são gerados sem considerar as consultas. Por outro lado, a necessidade

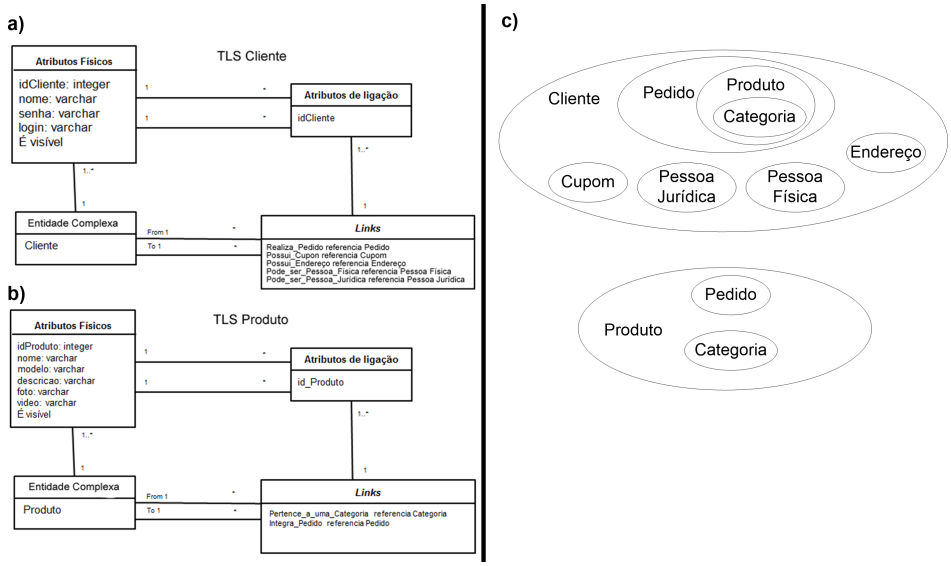

Figura 4: Resultado da migração utilizando a M3. 4a e 4b - 1º nível lógico para as TLS Cliente e Produto, respectivamente. 4c: Modelo orientado a documentos gerado a partir dos requisitos de consulta.

de conhecer as consultas à priori pode fazer com que novos processos de migração sejam necessários posteriormente. Isso porque ao elencar consultas, pode ser que nem toda a base de dados migre.

A M3 difere das demais tendo como entrada o modelo físico do banco de dados. Um processo semelhante à engenharia reversa é executado, capturando a semântica do dado. Um aspecto discutido apenas em M3 é a migração de *views*, que podem ser consideradas *TLS-in-focus* e migradas conforme as tabelas-base do banco.

A metodologia M1 apresenta a forma de entrada mais intuitiva. A M2 demanda como entrada, além das consultas previamente realizadas na base, entendimento de diagramas de classe UML e da semântica das consultas para rotular as classes. Já a M3 é destinada a bancos de dados com pouca documentação, uma vez que parte do modelo físico do banco.

5.2 Documentação do Método

Para esse quesito considerou-se os artigos onde as metodologias foram apresentadas. Foram analisadas a apresentação dos métodos, dos algoritmos, dos exemplos práticos e se foram realizados e documentados os testes de performance. Vale ressaltar que uma documentação falha dificulta a replicação do método em sua totalidade.

Na documentação de M1, o exemplo conceitual apresentado é diferente do que foi utilizado nos testes de performance realizados, cujo modelo orientado a documentos não é apresentado. Com base nos teste de performance, os autores concluíram que a metodologia oferece baixa latência para consultas e alta latência para cargas [26].

Para M2 é apresentado um estudo de caso sobre uma base real. No entanto, os autores não apresentam testes de performance.

Já em M3, parte do algoritmo é descrita, mas alguns passos não são completamente definidos, sendo sugerido pelos autores que um refinamento seja aplicado ao resultado obtido. O estudo de caso foi feito utilizando uma ferramenta que não está disponível no *link* fornecido, sendo apresentado somente o resultado final. Não foram realizados testes de performance.

5.3 Execução do Processo de Migração

Nesse aspecto avaliou-se a facilidade de seguir o algoritmo proposto, a saída dos algoritmos e a complexidade de automatização do método.

Considerando a percepção dos pesquisadores envolvidos na análise, avaliou-se que M1 é uma metodologia de fácil compreensão e execução. Tanto o algoritmo fornecido, quanto o exemplo conceitual auxiliam o entendimento do método e sua aplicação. A geração do grafo e os aninhamentos são bem explicados pelos autores. A saída do algoritmo reflete os documentos com os respectivos aninhamentos das relações.

Um aspecto não detalhado pelos autores é sobre como migrar um esquema no qual todas as relações possuem uma chave estrangeira. Ou seja, um modelo onde não exista nó folha. Isso porque o laço principal do algoritmo opera sobre um conjunto de vértices com grau de saída igual a zero e grau de entrada diferente de zero. A Figura 5 exemplifica esse cenário. No modelo relacional apresentado na Figura 5a todas as entidades recebem uma chave estrangeira. O grafo gerado é apresentado em 5b. As arestas direcionadas indicam que há uma chave estrangeira do nó de origem para o nó destino. O grafo não apresenta nós folha, ou seja, um nó que não tenha arestas saindo dele. Em 5c, verifica-se a inicialização das variáveis do algoritmo. Com o conjunto P vazio, o laço principal do algoritmo não pode ser inicializado (linha 7 da Figura 5d).

O algoritmo proposto em M2 também foi considerado pelos pesquisadores desse trabalho simples de ser seguido. A saída é o conjunto de classes agregadas (aninhadas) e de classes chamadas

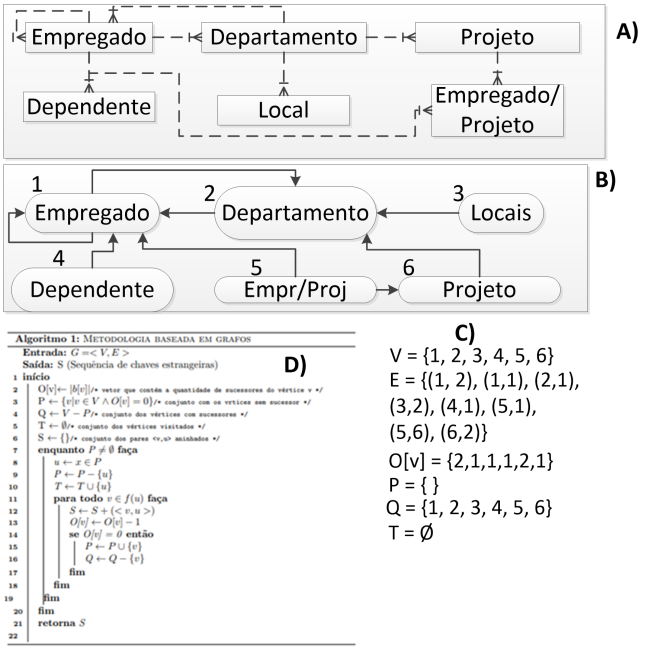

Figura 5: Cenário onde a metodologia baseada em grafos falha por não haver um nó folha no modelo.

de índice, que fazem referência às classes agregadas. Como dito anteriormente, o fato de ser orientada a consulta faz com que essa metodologia não necessariamente migre o banco todo, mas apenas as relações participantes da consulta.

Das três metodologias avaliadas, a M3 foi aquela considerada com maior complexidade de execução pelos pesquisadores desse trabalho. A engenharia reversa proposta pelos autores é dividida em diferentes níveis, o que torna o processo mais demorado. A execução completa da metodologia depende do entendimento de etapas que não são detalhadas pelos autores. Para a etapa final (segundo nível lógico), são propostos diferentes algoritmos. A metodologia proposta retorna um *template* do documento, que deverá ser refinado por especialistas.

Em termos práticos, a automatização de M1 deve ser a menos complexa, uma vez que o algoritmo é simples, direto e depende apenas das tabelas e seus relacionamentos. A partir das tabelas e relacionamentos, a geração do grafo também é trivial. A implementação de M2 deve ter uma complexidade maior comparada à M1. Isso porque automatizar a criação do diagrama de classes UML, rotulando as classes como sendo de consulta ou incluída apresenta uma dificuldade semântica inerente ao processo, assim como os requisitos de consulta. Avalia-se, portanto, que o processo pode ser semiautomatizado e depende de um especialista no domínio para tal tarefa. Da mesma forma, a automatização de M3 também deve ser parcial, uma vez que as etapas para a definição do nível físico e primeiro nível lógico objetivam dar semântica ao dado e necessitam de um especialista no domínio. Além disso, a etapa final da metodologia é o refinamento por um especialista.

5.4 Documentos Gerados

Nesse quesito foram avaliados a quantidade de documentos gerados e a qualidade semântica dos mesmos.

Dentre as três metodologias avaliadas, a M1 é que apresenta a maior quantidade de documentos gerados (Figura 2b) e maior redundância. Nos testes realizados pelos autores [26], foi verificada uma baixa latência para as consultas e uma alta latência para as cargas de dados. A diferença é que essa metodologia migra todo o banco sem se basear em possíveis consultas e sem referência à modelagem dos dados no banco NoSQL de destino. Como teoricamente o agregado [5] pode ser qualquer um, o modelo gerado atende às diferentes demandas. Consequentemente, o espaço de armazenamento cresce com o aumento da quantidade de dados.

Algo importante de salientar é sobre a semântica dos documentos gerados pela M1. Ao avaliar a Figura 2b, verifica-se que o modelo gera um documento para Cliente (nó 6). Cada entidade associada (pessoa física, pessoa jurídica, endereço, pedido e cupom) geram documentos que incorporam os dados de Cliente. No entanto, a metodologia não gera um documento Cliente que incorpore todas as características dessa entidade. Isso acontece porque o aninhamento ocorre no nó predecessor e não no nó folha.

Dentre as metodologias avaliadas, a M2 é a única capaz de gerar tanto documentos aninhados, quanto referenciados. Com isso, a redundância é reduzida e, consequentemente, o espaço de armazenamento também. Para os requisitos de consulta definidos, M2 gerou dois documentos aninhados e um que faz referência aos dois anteriores (Figura 3b). Os documentos gerados por M2 apresentam semântica perfeitamente condizente com os requisitos de consulta definidos.

Como dito anteriormente, a M3 pode ser aplicada seguindo um requisito de consulta ou não. Neste trabalho, optou-se por seguir os requisitos de consulta aplicados a M2. O resultado gerou apenas dois documentos (Figura 4d) e mostrou-se pouco redundante. O especialista é o responsável pelo refinamento do modelo, a fim de adequá-lo às necessidades da aplicação. Comparando diretamente com M2, M3 foi mais eficiente em termos de armazenamento e a semântica também foi condizente aos requisitos de consulta definidos.

Nenhuma das metodologias estudadas faz referência direta ao caso de herança, como Cliente com Pessoa Física e Pessoa Jurídica (Figura 1). Na implementação dessas metodologias esse aspecto precisa ser levado em conta.

6 ANÁLISE DOS RESULTADOS

Com base nas discussões realizadas na sessão 5, sugere-se:

- Utilizar a M1 em bases de dados com poucas relações e baixa conectividade, visto a alta redundância gerada pela metodologia. Atentar para a restrição que exige uma relação sem chave estrangeira.
- Utilizar a M2 em bases altamente conectadas e cujos requisitos de consulta são previamente conhecidos e estáveis.
- Utilizar a M3 em sistemas legados, cuja semântica e entendimento real do modelo de dados sejam desconhecidos. Por ser um processo mais demorado, o tempo de migração pode ser minimizado utilizando requisitos de consulta bem definidos.

A metodologia M1 foi avaliada por [4, 6, 17]. Os trabalhos relacionados não apontaram o fato de a metodologia não ser aplicável aos modelos relacionais com alto grau de conectividade, ou seja, onde todas as tabelas tenham chaves estrangeiras. Tampouco questionam a semântica dos documentos gerados. Ressaltam apenas a alta redundância, aspecto já evidenciado pelos autores da metodologia [26].

A metodologia M3 também foi avaliada por [6, 17]. O trabalho [17] afirma que esta abordagem apresenta a vantagem de fácil implementação. Já no artigo [6], os autores afirmam a implementação de M3 deve ser parcial, ou seja, trata-se de uma abordagem semi-automática, onde há uma preocupação com a semântica dos dados originais. Com base nas avaliações realizadas neste trabalho, as afirmações de [6] foram corroboradas, enquanto discorda-se das conclusões obtidas em [17].

Este estudo aponta uma dificuldade real para empresas e pesquisadores que desejam migrar dados de seus sistemas de informação para mecanismos de persistência incluídos no contexto NoSQL. A falta de uma metodologia única, sistemática e que contemple diferentes cenários dificulta e atrasa os processos de migração. Como foi possível observar, a escolha de uma das diversas metodologias disponíveis na literatura depende do contexto da aplicação que será migrada e do objetivo final desejado.

Apesar de não terem sido realizados testes computacionais, este trabalho fornece sugestões e *insights* através da avaliação dos processos de migração sob os modelos teóricos. Espera-se que os resultados aqui apresentados auxiliem gerentes de TI a decidirem pela melhor metodologia de migração de modelo de dados seguir em seus projetos reais.

Artigos com foco na transferência dos dados, e não apenas na migração do modelo, podem ser vistos em [2, 11]. Nesses estudos foram definidos alguns requisitos de migração, métricas de análise e executados diversos testes para os processos de migração. Nesse sentido, um plano de testes com uma base de dados real está em andamento e seus resultados, que incluem métricas quantitativas como tempo de carga, vazão e latência, serão comunicados em breve.

REFERÊNCIAS

[1] Daniel J Abadi. 2009. Data management in the cloud: Limitations and opportunities. *IEEE Data Eng. Bull.* 32, 1 (2009), 3–12.

[2] Angeles Cruz Manjarrez Antaño, José Mario Martínez Castro, and René E Cuevas Valencia. 2014. Migracion de Bases de Datos SQL a NoSQL. *Revista Tlamati, Especial* 3 (2014), 144–148.

[3] Rupali Arora. 2013. *A Novel Approach for Transformation of Data from MySQL to NoSQL (MongoDB).* Dissertação (Mestrado). Thapar University.

[4] Aparna Babu and Subu Surendran. 2017. Relational to NoSQL Database Migration. *International Journal of Innovative Research in Science, Engineering and Technology* 6 (2017), 58–62.

[5] F.a Bugiotti, L.b Cabibbo, P.b Atzeni, and R.b Torlone. 2015. How I learned to stop worrying and love NoSQL databases. In *23rd Ital. Symp. Adv. Database Syst. SEBD 2015*. 216–223. https://www.scopus.com/inward/record.uri?eid=2-s2.0-84971463436{&}partnerID=40{&}md5=3f8627537eee7a94c150f6b854921804

[6] Myller Claudino, Damires Souza, and Ana Carolina Salgado. 2015. Mapeamentos conceituais entre os modelos Relacional e NoSQL: uma abordagem comparativa. *Revista Principia - Divulgação Científica e Tecnológica do IFPB* 28 (2015), 37–50.

[7] Alejandro Corbellini, Cristian Mateos, Alejandro Zunino, Daniela Godoy, and Silvia Schiaffino. 2017. Persisting big-data: The NoSQL landscape. *Information Systems* 63 (2017), 1–23.

[8] Augusto Verzbickas Costa, Patricia Vilain, and Ronaldo Santos Mello. 2016. A Layer for the Mapping Management of SQL DML Instructions to the Key-Value NoSQL Database Voldemort. In *Brazilian Symp. Inf. Syst.* Florianópolis/SC, 30:224—-30:231. https://doi.org/10.1177/0734242X09345599

[9] Caio H Costa, Lincoln S Rocha, and Nabor C Mendonc. 2014. Migração parcial de um banco dados relacional para um banco de dados NoSQL na nuvem através de adaptações não-intrusivas: um relato de experiência. In *Work. Softw. Vis. Evol. Maint.* Maceió/AL, 38–45.

[10] John W Creswell. 2010. Projeto de pesquisa métodos qualitativo, quantitativo e misto. In *Projeto de pesquisa métodos qualitativo, quantitativo e misto.*

[11] Fábio Vieira de Oliveira. 2017. *Migração de bases de dados relacionais para NoSQL - Métodos de Análise.* Dissertação(Mestrado). Instituto Universitário de Lisboa.

[12] Antonio Carlos Marcelino de Paula, Glauco de Figueiredo Carneiro, and Antonio Cesar Brandao Gomes da Silva. 2018. Evidences from the Literature on Database Migration to the Cloud. In *Information Technology - New Generations*, Shahram Latifi (Ed.). Springer International Publishing, Cham, 507–513.

[13] Martyn Ellison, Radu Calinescu, and Richard Paige. 2014. Re-engineering the Database Layer of Legacy Applications for Scalable Cloud Deployment. In *Proceedings of the 2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing (UCC '14)*. IEEE Computer Society, Washington, DC, USA, 976–979. https://doi.org/10.1109/UCC.2014.160

[14] Mohamed Hanine, Abdesakik Bengarag, and Omar Boutkhoum. 2015. Data Migration Methodology from Relational to NoSQL Databases. *World Acad. Sci. Int. J. Comput. Inf. Eng.* 9, 12 (2015), 2518–2522.

[15] Philip Howard. 2011. *Data Migration - White Paper.* Technical Report May. 1–15 pages. papers2://publication/uuid/C70C51D3-6718-4CE8-8FAF-7F9A62B285EF

[16] Girts Karnitis and Guntis Arnicans. 2015. Migration of Relational Database to Document-Oriented Database: Structure Denormalization and Data Transformation. In *Proc. - 7th Int. Conf. Comput. Intell. Commun. Syst. Networks, CICSyN 2015*. 113–118. https://doi.org/10.1109/CICSyN.2015.30

[17] Arati Koli and Swati Shinde. 2017. Approaches used in efficient migration from Relational Database to NoSQL Database. In *Proc. Second Int. Conf. Res. Intell. Comput. Eng.*, Vol. 10. 223–227. https://doi.org/10.15439/2017R76

[18] Neal Leavitt. 2010. Will NoSQL databases live up to their promise? *Computer* 43, 2 (2010).

[19] Xiang Li, Zhiyi Ma, and Hongjie Chen. 2014. QODM: A query-oriented data modeling approach for NoSQL databases. In *Proc. - 2014 IEEE Work. Adv. Res. Technol. Ind. Appl. WARTIA 2014*. 338–345. https://doi.org/10.1109/WARTIA.2014.6976265

[20] Alza A Mahmood. 2018. Automated Algorithm for Data Migration from Relational to NoSQL Databases. *Al-Nahrain J. Eng. Sci.* 21, 1 (feb 2018), 60–65. https://doi.org/10.29194/NJES21010060

[21] Florian Matthes, Christopher Schulz, and Klaus Haller. 2011. Testing & quality assurance in data migration projects. In *2011 27th IEEE Int. Conf. Softw. Maint.* 438–447. https://doi.org/10.1109/ICSM.2011.6080811

[22] K North. 2010. The nosql alternative. *Retrieved April* 24 (2010), 2011.

[23] Pramod J Sadalage and Martin Fowler. 2013. *NoSQL distilled: a brief guide to the emerging world of polyglot persistence.* Pearson Education.

[24] Michael Stonebraker. 2010. SQL databases v. NoSQL databases. *Commun. ACM* 53, 4 (2010), 10–11.

[25] Q. H. Vu and R. Asal. 2012. Legacy Application Migration to the Cloud: Practicability and Methodology. In *2012 IEEE Eighth World Congress on Services*. 270–277. https://doi.org/10.1109/SERVICES.2012.47

[26] Gansen Zhao, Qiaoying Lin, Libo Li, and Zijing Li. 2014. Schema conversion model of SQL database to NoSQL. In *Proc. - 2014 9th Int. Conf. P2P, Parallel, Grid, Cloud Internet Comput. 3PGCIC 2014*. 355–362. https://doi.org/10.1109/3PGCIC.2014.137