INTERFACE TECNOLÓGICA

# APLICAÇÃO HÍBRIDA DE BANCO DE DADOS

## *HYBRID DATABASE APPLICATION*

Pedro Henrique Nilson Mantovani – ph529865@gmail.com

Fernando Tiosso – fernando.tiosso@fatectq.edu.br

Faculdade de Tecnologia de Taquaritinga (Fatec) – Taquaritinga - São Paulo – Brasil

**DOI: 10.31510/infa.v17i1.711**

## RESUMO

Este artigo tem como objetivo proporcionar o entendimento a respeito do funcionamento de dois tipos de bancos de dados muito utilizados atualmente em meio às aplicações do segmento *e-commerce*: relacional e não relacional. Com o objetivo de vislumbrar alguns cenários e aplicações para os referidos bancos de dados, foi necessário revisar os conceitos que permeiam banco de dados relacionais e não relacionais, bem como estudar o ambiente por eles suportados. De modo a respaldar a pesquisa, foram consultadas diversas publicações envolvendo o objetivo proposto, baseada em artigos, documentações e livros. Já para o desenvolvimento prático, foi utilizado um ambiente para gerir os dois tipos de bancos de dados, relacional e não relacional, promovendo um cenário de banco de dados híbrido. Com os experimentos, pôde-se observar os resultados da aplicação, mais especificamente em relação ao armazenamento dos dados nos dois tipos de bancos de dados, ressaltando características como segurança e performance, além do destaque quanto à utilização do tipo de dado JSONB, responsável por prover um desempenho diferenciado na leitura dos dados, justificando sua possível utilização em aplicações do tipo *e-commerce*.

**Palavras-chave:** Banco de dados. SGBD. Relacional. NoSQL. JSONB.

## ABSTRACT

The purpose of this article is to provide an understanding of the operation of two types of databases that are currently widely used in e-commer*ce* applications: relational and non-relational. In order to glimpse some scenarios and applications for these databases, it was necessary to review the concepts that permeate relational and non-relational databases, as well as to study the environment they support. In order to support the research, several publications were consulted involving the proposed objective, based on articles, documentation and books. For the practical development, an environment was used to manage the two types of databases, relational and non-relational, promoting a hybrid database scenario. With the experiments, it was possible to observe the results of the application, more specifically in relation to the storage of data in the two types of databases, highlighting characteristics such as security and performance, in addition to the emphasis on the use of JSONB data type, responsible for providing a differentiated performance in reading the data, justifying its possible use in e-commerce type applications.

**Keywords:** Database. SGBD. Relational. NoSQL. JSONB.

## 1 INTRODUÇÃO

O desenvolvimento tecnológico proporcionado pelos bancos de dados provocou um considerável impacto no crescimento do consumo de dados pelas aplicações, representando um papel crucial nas áreas de negócios, comércio eletrônico, engenharia, medicina, direito, educação e as ciências da informação (ELMASRI E NAVATHE, 2010).

A definição de banco de dados consiste em um conjunto de informações ou uma coleção lógica e coerente designada para promover o povoamento dos dados atendendo a uma proposta específica de um determinado grupo de usuários. Como exemplo, podem ser citados coleções de informações sobre pessoas, lugares ou coisas. (ELMASRI E NAVATHE, 2010) (DATE, 1984).

Atualmente, a quantidade de dados armazenados em um banco de dados pode exceder bilhões de bytes, justificando valorosos investimentos realizados pelas empresas nesta área a fim de viabilizar a operação contínua e esplêndida dos sistemas que suportam suas operações. Como exemplo, pode-se citar as aplicações de comércio eletrônico, pois com o avanço da tecnologia e o advento da Internet das Coisas (IoT), elas exigem, cada vez mais, escalabilidade e alto desempenho, requisitos estes que impactam diretamente na seleção de bancos de dados mais robustos para armazenamento e segurança das transações realizadas por milhares de usuários.

Assim, os dados contidos em sistemas de banco de dados podem ser distribuídos em um ou mais bancos de dados, podendo, até mesmo, serem de diferentes tipos, visando suportar os inúmeros acessos simultâneos realizados pelos usuários, formando um grande repositório de dados compartilhado utilizado para diferentes finalidades (DATE, 1984).

De acordo com a análise deste cenário, surgiu a ideia da escrita deste artigo, que consiste em definir e utilizar dois tipos de bancos de dados muito utilizados comercialmente, relacional e não relacional, que podem ser utilizados para suportar uma aplicação do segmento de *e-commerce* empregando as melhores características de cada um deles, visando proporcionar alta disponibilidade dos serviços ofertados para os usuários.

Assim, o presente trabalho foi composto por 5 seções, sendo elas: a Seção 2 (dois) que apresenta a metodologia de pesquisa utilizada, bem como a fundamentação teórica sobre

banco de dados relacional e não relacional, a seção 3 (três) que apresenta a o desenvolvimento do projeto utilizado para evidenciar os resultados, a seção 4 (quatro) que discorre sobre a discussão e análise dos resultados obtidos e, por fim, a seção 5 (cinco) que expõe as considerações finais do artigo.

## 2 FUNDAMENTAÇÃO TEÓRICA

De acordo com Gil (2002), a revisão bibliográfica é a base que sustenta uma pesquisa, sendo assim, tem o objetivo de proporcionar respostas às questões por ela apresentada.

Por meio dessa premissa, neste artigo, a revisão bibliográfica foi uma tarefa de considerável importância, evidenciando um cenário propício para a utilização de um ambiente de banco de dados híbrido em sistemas de *e-commerce*.

Em meio a uma aplicação híbrida de banco de dados, faz-se uso de conceitos relacionados aos bancos de dados relacionais e não relacionais suportados por Sistemas Gerenciadores de Banco de Dados (SGBDs) que possibilitam o acesso aos dados por aplicações e usuários. Considerando essa afirmação e visando um melhor entendimento e funcionamento deste ambiente, os tópicos subsequentes promovem o conhecimento de alguns conceitos que o permeiam.

### 2.1 Banco de Dados Relacionais (BDR)

Segundo Costa (2011, p. 33) e Bonfioli (2006, p. 4) o modelo relacional foi apresentado primeiramente em um artigo de 1970, porém ele só foi implementado de fato nos anos 80. A pesquisa inicial consistia em um estudo teórico realizado por Codd (1970) e tinha como base a teoria dos conjuntos e álgebra relacional. O objetivo que a pesquisa propôs era suprir as seguintes necessidades: o aumento da emancipação de dados incorporado nos SGBDs, além de proporcionar um agrupamento de funções baseado na álgebra relacional para grandes volumes de dados, assim como proporcionar o encadeamento das redes ad hoc.

A base fundamental que é utilizada no modelo relacional é a relação, pois é definida por um ou mais atributos (campos) que armazenam os tipos de dados informados pelo usuário. Cada instância efetuada é considerada uma linha ou tupla (registro) e, desta forma, o modelo relacional implementa esses dados organizados em entidades (tabelas) (BONFIOLI, 2006, p. 15).

Para se trabalhar com as entidades é preciso impor algumas restrições de modo a evitar a repetição de dados e perdas parciais ou completas de dados. As restrições mais relevantes são de integridade de entidade e referencial, obtidas pela utilização de chaves primárias e estrangeiras respectivamente (ELMASRI E NAVATHE, 2010).

A restrição de integridade de entidade impõe que o valor da *Primary Key* (PK) em nenhum momento pode ser nulo, pois é utilizada para identificar tuplas individuais em uma relação. Já a de integridade referencial, definida entre duas relações existentes, é aplicada para manter a consistências das tuplas nas duas relações. (ELMASRI E NAVATHE, 2010)

No entanto, o conjunto de programas responsáveis por aplicar as restrições e outros tipos de controle sobre os bancos de dados é denominado Sistema Gerenciador de Banco de Dados (SGBD).

De acordo com Elmasri e Navathe (2010), o SGBD é um conjunto de aplicações que permite aos usuários criar e manter um ou mais banco de dados. Portanto, é um sistema que facilita os processos de definição, construção, manipulação e compartilhamento de bancos de dados com múltiplos usuários e aplicações. Sua maior característica e função é ser responsável por atuar no controle de redundância, consistência e acesso dos dados pelos usuários. Como exemplos de SGBDs relacionais encontrados no mercado podem ser citados: Oracle, Microsoft SQL Server, MySQL, PostgreSQL, entre outros.

## 2.2 Bancos de Dados Não Relacional (BDNR)

De acordo com Vilela (2015), os bancos de dados não relacionais ou comumente chamados de *Not Only SQL* (NoSQL) foram inicialmente abordados por Carlos Strozzi em 1999, com o objetivo de oferecer novas maneiras de armazenamento de grandes volumes de dados proporcionando mais disponibilidade e escalabilidade nas aplicações quando comparados com os bancos de dados relacionais.

Tornando-se mais conhecidos a partir de 2009 por meio de Johan Oskarsson em um evento sobre banco de dados distribuídos, empresas como Google, Facebook e Amazon passaram a adotar o modelo NoSQL para desenvolverem suas aplicações e soluções (VILELA, 2015).

Os bancos de dados NoSQL são subdivididos em quatro categorias: chave-valor, orientado a colunas, orientado a documentos e orientado a grafos, brevemente citadas a seguir:

- Chave/Valor: apresentado como o modelo de dados mais simples, sendo cada valor correspondente a uma chave, permitindo o armazenamento de grande volume de dados e modificações nos valores através das chaves (HAN, J. *et al.*, 2011). Redis é um dos bancos de dados mais famosos desta categoria.
- Documentos: modelo encarregado de armazenar um par de chave/documento nos arquivos do tipo JSON ou XML. O documento, dentro do banco de dados, é identificado por uma chave única e especial chamada Id (HECHT; JABLONSKI, 2011). CouchDB e MongoDB são exemplos de bancos de dados não relacionais orientados a documentos.
- Grafos: orientados a grafos e especializados no gerenciamento de dados fortemente ligados e direcionados para aplicações de geolocalização e de recomendações. Os nós e arestas são formados por objetos que contém atributos chave-valor embutidos (HECHT; JABLONSKI, 2011). Neo4J, InfoGrid e AllegroGraph são bancos de dados que representam esta categoria.
- Colunas: orientados por colunas sendo o modelo que mais se assemelha a um banco de dados relacional. Cassandra e Amazon SimpleDB são exemplos de bancos de dados de colunas (VILELA, 2015).

## 3 PROJETO

Com o intuito de apresentar um possível cenário escalável e de alta performance para um sistema de *e-commerce* utilizando as melhores características de diferentes tipos de bancos de dados, neste caso representados por relacionais e não relacionais, visando suportar acessos simultâneos realizados por usuários, foi proposto um ambiente utilizando algumas tecnologias *open source* com grande poder de impacto no mundo atual de desenvolvimento de sistemas, devidamente explicadas a seguir.

Pretendendo garantir um vasto fluxo de dados e um ótimo desempenho em consultas e inserções dos dados de um produto, o projeto fez uso do banco de dados MongoDB por meio do seu modelo de documentos, devido a sua característica original da ausência do conceito transacional e excessivas restrições. Todavia, para garantir o armazenamento de dados transacionais contidos de forma temporária, o projeto também fez uso do SGBD PostgreSQL.

Ressalta-se a utilização do tipo de dados *JavaScript Object Notation Binary* (JSONB) para o armazenamento dos dados no banco de dados relacional, devido a sua considerável

performance na inserção e na consulta dos dados, uma vez que este tipo de dados apresenta suporte de indexação.

A construção do código para suportar o projeto foi realizada por meio da ferramenta *Visual Studio Code* (VSCODE) em conjunto com ambiente sustentado pelo Node.js visando prover as instalações dos pacotes e executar a linguagem JavaScript do lado do servidor.

Assim, pretendendo proporcionar um melhor entendimento do desenvolvimento de todo o projeto, as seções subsequentes apresentam seu processo de construção em detalhes.

**3.1 Desenvolvimento**

Para o desenvolvimento do projeto, optou-se pela utilização do Node.js devido ao seu suporte fornecido para a criação de aplicações de alta escalabilidade por meio de um servidor Web, além do gerenciamento de pacotes referentes aos bancos de dados MongoDB e PostgreSQL, como também ao suporte para a execução da linguagem JavaScript, utilizada para a escrita do código e responsável pelo intermédio da comunicação entre os dois bancos de dados.

A primeira etapa do projeto foi caracterizada pela instalação e configuração do ambiente em Node.js por meio do comando *npm init*.

A segunda etapa também foi executada no terminal Node.js, por meio da instalação dos pacotes dos bancos de dados MongoDB e PostgreSQL, utilizando respectivamente os comandos *npm install mongodb e npm install pg*.

Já, a terceira etapa, proporcionou o desenvolvimento de todos os códigos, por meio da linguagem JavaScript, encarregados de executar todas as funções para a integração dos dados nos bancos de dados MongoDB e PostgreSQL.

O código desenvolvido para a comunicação com o banco de dados não relacional MongoDB, desde a sua conexão até a inserção de documentos pode ser visualizado na Figura

Figura 1 - Código para acesso ao MongoDB




Fonte: próprio autor

Em contrapartida, a Figura 2 mostra o código desenvolvido para comunicação com o banco de dados relacional PostgreSQL, desde sua conexão até à inserção de dados.

Figura 2 - Código para o acesso ao PostgreSQL




Fonte: próprio autor

A Figura 3 mostra a etapa de desenvolvimento mais importante do projeto, caracterizada pelo código de inserção de um dado da coleção de produtos do banco de dados não relacional MongoDB no banco de dados relacional PostgreSQL, fazendo uso do tipo de dados JSONB.

Figura 3 - Código de integração entre o banco de dados MongoDB e PostgreSQL




Fonte: próprio autor

Por fim, a última etapa do desenvolvimento foi a utilização do ambiente Node.js para executar os códigos desenvolvidos e efetuar toda a integração dos dados por meio dos dois tipos de banco de dados.

## 4 RESULTADOS E DISCUSSÃO

A simulação realizada se deu a partir de uma aplicação em produção para analisar algumas das principais operações de um banco de dados, neste caso, inserções e consultas, para validar a utilização dos bancos de dados em uma ambiente híbrido e a importância da utilização do tipo de dados JSONB no banco de dados relacional para armazenamento dos dados. Visando ser o mais realista possível, os testes de carga continham uma imensa entrada de dados, neste caso 600 mil registros e/ou documentos, visto que no mundo de negócios o fluxo de transações é consideravelmente elevado para aplicações do tipo *e-commerce*.

Assim, a inserção dos registros utilizando o comando *copy* do PostgreSQL, levou 12,12 segundos para sua conclusão, conforme mostra a Figura 4. Neste caso, o teste de carga partiu de um arquivo previamente configurado no formato JSON.

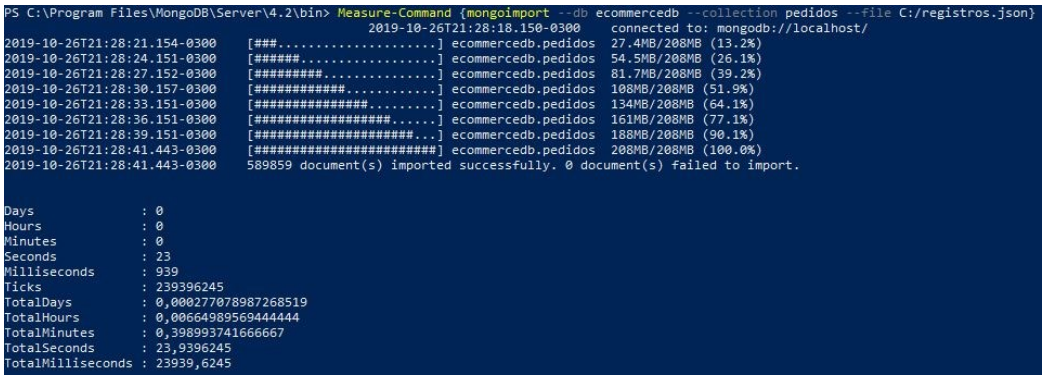
Figura 4- Inserção de dados no PostgreSQL




Fonte: próprio autor

A Figura 5 mostra a inserção dos documentos no MongoDB. Neste caso, utilizou-se o comando *measure-command* para medir o tempo da execução do comando *mongoimport*, comprovando que a execução levou aproximadamente 23,93 segundos para se completar.

Figura 5 - Inserção de dados no MongoDB




Fonte: próprio autor

A operação de consulta sem a utilização da indexação do tipo de dado JSONB no PostgreSQL levou 3 segundos para ser executada, conforme mostra a Figura 6, por meio da característica *Database sessions*, que permite analisar o quanto tempo (em segundos) que uma determinada operação permanece ativa (linha verde) em meio a uma sessão com o banco de dados.

Figura 6 - Consulta de dados sem utilização de um index no PostgreSQL




Fonte: próprio autor

A Figura 7 mostra a operação de consulta com indexação do tipo de dado JSONB no PostgreSQL, levando 2 segundos para ser executada, notada pelos dois ciclos de atividade da linha verde, cada um correspondendo 1 segundo.

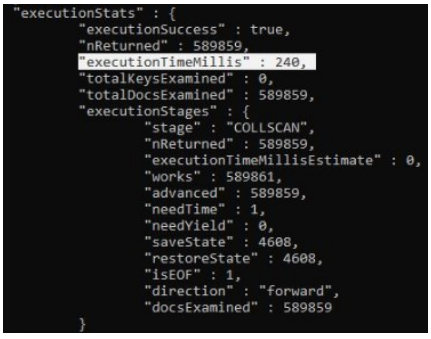
Figura 7 - Consulta de dados utilizando indexação PostgreSQL




Fonte: próprio autor

Com relação a mesma consulta executada no banco de dados MongoDB, pode-se observar que foi concluída em 240 ms, conforme mostra a característica destacada na Figura 8.

Figura 8 - Consulta de dado no MongoDB




Fonte: próprio autor

Portanto, de acordo com todas as características encontradas e analisadas envolvendo as operações de inserção e consulta para os diferentes tipos de bancos de dados, foi possível elaborar o Quadro 1, contendo um comparativo entre as operações.

**Quadro 1- Comparativo entre as operações dos bancos de dados**

| Banco de Dados | Inserção | Consulta | Consulta/indexação |
|---|---|---|---|
| PostgreSQL | 12,12 segundos | 3 segundos | 2 segundos |
| MongoDB | 23,93 segundos | 240 ms | - |

Fonte: próprio autor

Por meio da análise do Quadro 1, observa-se resultados significativos mediante aos dados apresentados nos testes de cargas, sendo possível concluir que a utilização do banco de dados relacional pode ser um tanto quanto eficiente nas operações de inserção quando utilizado o tipo de dados JSONB para armazenamento, visto que este tipo de operação pôde ser realizado com aproximadamente 50,64% do tempo quando comparado ao banco de dados não relacional.

No entanto, para operações de consultas de dados, é inegável o melhor desempenho do banco de dados não relacional, pois nota-se um tempo consideravelmente menor, em torno de 92% de diferença, para realizar a mesma operação quando comparado ao modelo relacional. Mas, por outro lado, nota-se, também, que a indexação do tipo de dados JSONB melhora a performance das operações de consulta em aproximadamente 40%.

## 5 CONCLUSÃO

Sabe-se que a Web está em constante evolução e diferentes abordagens de tecnologias são aplicadas nos diversos projetos, mas um dos principais agentes para a aplicação ser dinâmica e constantemente atualizada é banco de dados.

A contribuição deste artigo diz respeito à possível utilização de um ambiente híbrido de banco de dados para aplicações, principalmente as do segmento *e-commerce* com um certo grau de exigência quanto ao quesito de escalabilidade quando analisado o tempo de resposta para diversos usuários conectados de forma simultânea. O ambiente híbrido pode permitir a

utilização das melhores características de cada tipo de banco de dados, relacional e não relacional, mediante a necessidade da operação a ser realizada.

Conforme demonstrado nos testes de carga realizados na seção 4, observou-se que banco de dados relacional PostgreSQL se portou de forma mais eficiente para inserção do que para consultas utilizando o tipo de dados JSONB, quando comparado ao banco de dados não relacional MongoDB. No entanto, para este mesmo cenário, o MongoDB apresentou uma maior capacidade de consulta aos dados do que o PostgreSQL.

Assim, após às análises sistêmicas e a conclusão dos seus resultados, fica evidente que este artigo apresentou, de forma demonstrativa, significativas diferenças em uma pequena aplicação híbrida de banco de dados voltado para o segmento *e-commerce*, ressaltando que a construção de uma aplicação utilizando dois modelos, relacional e não relacional, pode atender diferentes necessidades estipuladas por um determinado projeto, viabilizando uma solução flexível para o desenvolvimento dos sistemas visando proporcionar alta disponibilidade dos serviços ofertados para os usuários.

Para trabalhos futuros, propõe-se um estudo envolvendo um cenário de consultas mais específicas nos dois tipos de banco de dados, ou seja, desenvolver um ambiente onde seja possível obter dados de um determinado produto ou de uma determinada categoria de produtos e comparar os resultados referentes ao desempenho obtido pelas diferentes abordagens, relacional e não relacional.

## REFERÊNCIAS

BONFIOLI, Guilherme Ferreira Bonfili. **Banco de dados relacional e objeto-relacional: uma comparação usando PostgreSQL**. Disponível em: < http://repositorio.ufla.br/bitstream/1/8354/1/MONOGRAFIA_Banco_de_dados_relacional_e_objeto_relacional_uma_compara%C3%A7%C3%A3o_usando_postgresql.pdf>. Acesso em: 01 mar. 2019.

COSTA, E. R. da. (2011). *Elisângela Rocha da Costa* **BANCOS DE DADOS RELACIONAIS**. 64. Acesso em: 15 jul. 2019.

DATE, C. J. ***Introdução a sistemas de bancos de dados***. Rio de Janeiro: Campus, 1984. 513 p.

ELMASRI, R. e NAVATHE, S. B. (2010). ***Sistemas de Banco de Dados*** *- 6ª Edição.pdf*. p. 790. Acesso em: 01 mar. 2019.

GIL, A. C. **Métodos e Técnicas de Pesquisa Social.** 5. ed. São Paulo: Atlas, 1999. Acesso em: 20 jul. 2019.

HAN, J. *et al*. **Survey on nosql *database***. In Proceedings of 6th International Conference on Pervasive Computing and Applications (ICPCA 2011), 1:363–366, outubro 2011. Acesso em: 10 ago. 2019.

HECHT, R.; JABLONSKI, S. ***Nosql evaluation - a use case oriented survey***. 2011 International Conference on Cloud and Service Computing, 1:336–341, dezembro 2011.

LAKATOS, E. M; MARCONI, M. A. **Fundamentos de Metodologia Científica**. 5. ed. São Paulo: Atlas, 2003. Acesso em: 22 jul. 2019.

VILELA, F. A., SILVA, J. C. (2015). **Um método de integração de dados armazenados em bancos de dados relacionais e NOSQL**. Disponível em: <https://pdfs.semanticscholar.org/badc/1769779cf862adee49cce011c02359c1e92e.pdf>. Acesso em: 15 jul. 2019.