

****Potencializando o desempenho com NoSQL****

**Andre Luiz Sazana Waleczki | RM:559685**

**Guilherme Vinícius dos Santos | RM:560564**

**Henrique Caproni Siqueira | RM:560105**

**Renan Thiago Aviz e Silva | RM:560849**

**Thiago Evangelista Dias | RM:559403**

**Versão 3**

**HISTÓRICO DE VERSÕES**

|  |  |  |  |
| --- | --- | --- | --- |
| Versão | Data | Responsável | Descrição |
| 1 | 14/06/2024 | Patrícia Maura Angelini | Versão Inicial Template PBL Fase 5 - CAP 01 - POTENCIALIZANDO O DESEMPENHO COM NOSQL |
| 2 | 18/06/2024 | Rita de Cássia Rodrigues | Revisão acadêmica |
| 3 | 19/03/2025 | Andre Luiz Sazana Waleczki | Criação de conteúdo |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

FICHA CATALOGRÁFICA   
**[NÃO PREENCHER - PARA USO DO DEPTO DE EAD E BIBLIOTECA]**

A000a Sobrenome, Nome

Título [livro eletrônico] / Nome Sobrenome. -- São Paulo : Fiap, 2016.

x MB ; ePUB

Bibliografia.

ISBN 000-00-00000-00-0

Categoria. 2. Subcategoria. S., Nome. II. Título.

CDU 000.000.00

RESUMO

Template para atividade de PBL fase 5 1º ano TSC.

Palavras-chave: PBL. FASE 5. TEMPLATE

LISTA DE Figuras

[Texto

Descrição gerada automaticamente Figura 1 – Algoritmo construído em python com as bibliotecas pandas, seaborn, matplotlib, numpy, math e scipy 15](#_Toc193313146)

[Figura 2 - Algoritmo construído em python com as bibliotecas pandas, seaborn, matplotlib, numpy, math e scipy Fonte: Elabvorado pela equipe (2025) 15](#_Toc193313147)

[Figura 3 – Algoritmo construído em python com as bibliotecas pandas, seaborn, matplotlib, numpy, math e scipy 16](#_Toc193313148)

[Figura 4 – Algoritmo construído em python com as bibliotecas pandas, seaborn, matplotlib, numpy, math e scipy 17](#_Toc193313149)

[Figura 5 – Algoritmo construído em python com as bibliotecas pandas, seaborn, matplotlib, numpy, math e scipy 18](#_Toc193313150)

[Figura 6 – Matriz de correlação construído em python com a biblioteca matplotlib 19](#_Toc193313151)

LISTA DE QUADROS

[Quadro 1 – Quadro resumo das tarefas do PBL **Erro! Indicador não definido.**](#_Toc169351308)

LISTA DE TABELAS

No table of figures entries found.LISTA DE CÓDIGOS-FONTE

No table of figures entries found.LISTA DE ComandoS de prompt do sistema operacional

No table of figures entries found.

Sumário

[Potencializando o desempenho com NoSQL 11](#_Toc193312483)

[1.Prova de conceito de banco de dados NoSQL 11](#_Toc193312484)

[1.1 Análise de cenários 11](#_Toc193312485)

[1.2 Cenário 1 11](#_Toc193312486)

[1.2.1 Justificativa do cenário 1 11](#_Toc193312487)

[1.2.2 Empresa que usa o cenário 1 12](#_Toc193312488)

[1.3 Cenário 2 12](#_Toc193312489)

[1.3.1 Justificativa do cenário 2 12](#_Toc193312490)

[1.3.2 Empresa que usa o cenário 2 13](#_Toc193312491)

[1.4 Cenário 3 13](#_Toc193312492)

[1.4.1 Justificativa do cenário 3 13](#_Toc193312493)

[1.4.2 Empresa que usa o cenário 3 14](#_Toc193312494)

[2 ANÁLISE DOS DADOS DE VENDAS 14](#_Toc193312495)

[2.1 Quantidade 14](#_Toc193312496)

[2.2 Preço 16](#_Toc193312497)

[2.2 Correlações 17](#_Toc193312498)

[GLOSSÁRIO 19](#_Toc193312499)

# ****Potencializando o desempenho com NoSQL****

## 1.Prova de conceito de banco de dados NoSQL

## 1.1 Análise de cenários

A realização de testes de cenários é essencial para validar a adequação de diferentes bancos de dados NoSQL a necessidades específicas do e-commerce da Melhores Compras. Abaixo, detalhamos os cenários analisados, justificando as escolhas e fornecendo exemplos de empresas que já utilizam as soluções sugeridas.

## 1.2 Cenário 1

Quando um cliente seleciona um produto, a plataforma de e-commerce exibe, adicionalmente, recomendações de outros itens, baseadas nas compras de quem comprou esse produto e em outras promoções correlatas. No contexto atual, esse cálculo está demorando muito tempo para ser feito utilizando estruturas relacionais, dado o volume de dados envolvidos.

## 1.2.1 Justificativa do cenário 1

Para esse cenário, um banco de dados NoSQL do tipo Grafo foi escolhido, pois permite modelar eficientemente relações complexas entre produtos e clientes. A estrutura de grafos possibilita consultas altamente otimizadas, eliminando a necessidade de JOINs e garantindo desempenho superior na recomendação de produtos.

O banco de dados em grafo Neo4j é a melhor solução, pois:

Lida com relações complexas de forma eficiente, Oferece consultas extremamente rápidas sem necessidade de JOINs; altamente escalável, suportando grandes volumes de dados e conexões;

## 1.2.2 Empresa que usa o cenário 1

Empresas como Netflix, Facebook e Amazon utilizam bancos de dados de grafos para recomendações personalizadas. O Neo4j é um dos bancos mais populares para esse tipo de aplicação.

Podemos citar a Amazon que utilizou o Neo4j para modelar relacionamentos complexos entre produtos, clientes e comportamentos de compra. Permitindo recomendações altamente personalizadas, como "clientes que compraram este item também compraram...". A estrutura de grafos facilita a análise de relações em tempo real, tornando as recomendações mais precisas e escaláveis.

## 1.3 Cenário 2

A definição da entrega de um produto em 24h depende da disponibilidade de estoque do centro de distribuição mais próximo do endereço de entrega. Se o cliente optar por essa entrega rápida, é necessário realizar a reserva no centro de distribuição e atualizar o estoque automaticamente. Nos testes preliminares, o modelo relacional apresentou baixo desempenho devido ao volume de dados e à alta frequência de atualizações.

## 1.3.1 Justificativa do cenário 2

O banco de dados NoSQL Colunar Apache Cassandra é a melhor solução, dentre vários benefícios, citamos alguns:

Escalabilidade Horizontal: O Cassandra é projetado para escalar horizontalmente, adicionando mais nós ao cluster conforme o volume de dados e transações aumenta.

Alta Disponibilidade: Com sua arquitetura distribuída, o Cassandra garante que os dados estejam sempre disponíveis, mesmo em caso de falhas de hardware ou de rede. Desempenho em Escrita: O Cassandra é otimizado para operações de escrita, o que é essencial para cenários de atualização frequente de estoque.

Consistência Ajustável: O Cassandra permite ajustar o nível de consistência dos dados, oferecendo flexibilidade para equilibrar desempenho e precisão.

## 1.3.2 Empresa que usa o cenário 2

Podemos citar a eBay, uma das maiores plataformas de e-commerce do mundo, enfrenta desafios semelhantes ao gerenciar estoques de milhões de produtos e garantir atualizações em tempo real para disponibilidade e reservas. A eBay utiliza o Apache Cassandra para armazenar e gerenciar dados relacionados a transações, estoque e disponibilidade de produtos. O Cassandra permite que a eBay atualize o estoque em tempo real e garanta que as informações estejam consistentes em todos os seus data centers distribuídos. O Cassandra oferece alta escalabilidade e tolerância a falhas, permitindo que a eBay lide com picos de tráfego e atualizações frequentes de estoque sem comprometer o desempenho.

## 1.4 Cenário 3

A tela de detalhes de um produto recebe constantemente novas informações, como reviews, versões, dados de entrega, imagens e recomendações. Para armazenar esse conjunto dinâmico de informações, um banco relacional tradicional pode ser ineficiente devido à rigidez de seu esquema.

## 1.4.1 Justificativa do cenário 3

O banco de dados NoSQL orientado a documentos MongoDB é a melhor solução, dentre vários benefícios, citamos alguns:

Flexibilidade de Esquema: O MongoDB permite armazenar dados em formato de documentos JSON/BSON, o que facilita a adição de novos campos (como reviews, imagens ou informações de entrega) sem alterar a estrutura do banco de dados.

Desempenho em Leitura e Escrita: O MongoDB é otimizado para operações de leitura e escrita frequentes, essenciais para cenários de atualização constante de informações de produtos.

Escalabilidade Horizontal: O MongoDB permite escalar horizontalmente, adicionando mais nós ao cluster para lidar com o crescimento do volume de dados e tráfego.

Consultas Complexas: O MongoDB suporta consultas avançadas, incluindo buscas por texto, agregações e filtros, o que é útil para exibir informações dinâmicas na tela de detalhes do produto.

## 1.4.2 Empresa que usa o cenário 3

Empresas como eBay, Forbes, Cisco e SAP.

Podemos citar a eBay novamente, onde a empresa precisa gerenciar informações dinâmicas e complexas sobre milhões de produtos, incluindo reviews, imagens, detalhes de entrega e recomendações. A eBay utiliza o MongoDB para armazenar dados de produtos de forma flexível e escalável. O MongoDB permite que a eBay adicione novos campos (como reviews ou informações de entrega) sem alterar a estrutura do banco de dados, além de oferecer desempenho otimizado para consultas frequentes. A flexibilidade do MongoDB permite que a eBay atualize rapidamente as informações dos produtos e ofereça uma experiência personalizada aos usuários.

# 2 ANÁLISE DOS DADOS DE VENDAS

## 2.1 Quantidade

O método do intervalo interquartil (IQR) foi utilizado para identificar outliers na coluna quantidade. Onde foram detectados 11.283 outliers.

A média das vendas foi recalculada sem os outliers, resultando em uma média mais precisa (reduzindo de 568,16 para 319,91).

Uma estimativa de variabilidade foi calculada ignorando os outliers:

Desvio padrão das vendas **sem outliers**: 19.789,14.

Coeficiente de variação: 6.185,84%.

Amplitude interquartil (IQR) das vendas sem outliers: 62,00.

Texto

Descrição gerada automaticamente Figura 1 – Algoritmo construído em python com as bibliotecas pandas, seaborn, matplotlib, numpy, math e scipy

Fonte: Elaborado pela equipe (2025)

Texto

Descrição gerada automaticamente

Figura - Algoritmo construído em python com as bibliotecas pandas, seaborn, matplotlib, numpy, math e scipy  
Fonte: Elabvorado pela equipe (2025)

## 2.2 Preço

A média geral dos preços foi calculada, e um teste t de amostra única foi aplicado para comparar as médias por região em relação à média da população.

Para todas as regiões (Centro-Oeste, Nordeste, Norte, Sudeste, Sul), os p-valores foram maiores que 0.05, indicando que não há diferença estatisticamente significativa entre a média de preço de cada região e a média geral.

A mesma análise pode ser aplicada às modalidades de pagamento.

Texto

Descrição gerada automaticamente

Figura – Algoritmo construído em python com as bibliotecas pandas, seaborn, matplotlib, numpy, math e scipy

Texto

Descrição gerada automaticamente

Figura – Algoritmo construído em python com as bibliotecas pandas, seaborn, matplotlib, numpy, math e scipy

## 2.2 Correlações

Correlação alta entre valor e valor\_comissao (0.94)

Isso sugere que a comissão é fortemente influenciada pelo valor do produto. Quanto mais caro o produto, maior a comissão.

Correlação alta entre valor\_total\_bruto e valor\_comissao (0.90)

Indica que o total bruto de vendas está intimamente ligado à comissão paga.

Correlação moderada entre lucro\_liquido e valor (0.76)

Produtos mais caros tendem a gerar mais lucro líquido, mas essa relação não é perfeita.

Correlação baixa entre quantidade e lucro\_liquido (0.24)

Sugere que vender mais unidades nem sempre se traduz em mais lucro, o que pode indicar variações de margem de lucro entre os produtos.

Correlação praticamente nula entre quantidade e valor\_total\_bruto (-0.00)

Isso pode significar que o total bruto de vendas não depende muito do número de unidades vendidas, mas sim do preço individual dos produtos.

Texto

Descrição gerada automaticamente

Figura – Algoritmo construído em python com as bibliotecas pandas, seaborn, matplotlib, numpy, math e scipy

Gráfico

Descrição gerada automaticamente

Figura – Matriz de correlação construído em python com a biblioteca matplotlib

# GLOSSÁRIO

|  |  |
| --- | --- |
| **NoSQL** | Tipo de banco de dados não relacional, projetado para armazenar e recuperar grandes volumes de dados de forma eficiente, sem a necessidade de esquemas rígidos. |
| **Banco de Dados de Grafos** | Modelo de banco de dados NoSQL que utiliza nós e arestas para representar e armazenar relações complexas entre entidades, como recomendações de produtos. |
| **Neo4j** | Banco de dados de grafos amplamente utilizado para modelagem de relações complexas e análise de redes sociais. |
| **Banco de Dados Colunar** | Tipo de banco de dados NoSQL otimizado para leitura e escrita de grandes volumes de dados estruturados em colunas, como o Apache Cassandra. |
| **Apache Cassandra** | Banco de dados colunar distribuído, altamente escalável, utilizado por empresas como Netflix e Twitter para gerenciar grandes volumes de dados. |
| **Google Bigtable** | Banco de dados colunar do Google, utilizado para armazenamento de dados massivos e escaláveis, como no Google Analytics. |
| **Banco de Dados de Documentos** | Modelo de banco NoSQL que armazena informações em documentos JSON ou BSON, permitindo alta flexibilidade e eficiência. |
| **MongoDB** | Banco de dados de documentos NoSQL que permite armazenamento escalável de dados sem estrutura fixa. |
| **Amazon DynamoDB** | Serviço de banco de dados NoSQL gerenciado pela AWS, otimizado para alta disponibilidade e escalabilidade. |
| **Recomendações Baseadas em Grafos** | Técnica de recomendação que utiliza bancos de dados de grafos para sugerir produtos ou serviços com base em interações e preferências de usuários. |
| **Intervalo Interquartil (IQR)** | Método estatístico utilizado para identificar outliers em um conjunto de dados, analisando a dispersão dos valores dentro dos quartis. |
| **Desvio Padrão** | Medida estatística que representa a variação ou dispersão dos dados em relação à média. |
| **Coeficiente de Variação** | Índice estatístico que mede a dispersão relativa dos dados em relação à média. |
| **Correlação** | Medida estatística que indica a relação entre duas variáveis, variando entre -1 (correlação negativa perfeita) e 1 (correlação positiva perfeita). |
| **Matriz de Correlação** | Representação gráfica da correlação entre diferentes variáveis dentro de um conjunto de dados. |
| **Teste T** | Método estatístico utilizado para comparar médias e verificar se há diferença estatisticamente significativa entre grupos de dados. |
| **JOINs** | Operação em bancos de dados relacionais utilizada para combinar registros de duas ou mais tabelas baseadas em uma chave comum, muitas vezes impactando o desempenho. |