Projeto Big Data
Bancos de Dados 3

Gabriela Rech[1], **Gabriele A. do Nacimento**[1], **Giovanni L. M. Tenereli**[1], **Luan P. Marquetti**[1], **Marcelly M. de Oliveira**[1]

[1]Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas (TADS)
Universidade Federal do Paraná (UFPR) - Curitiba, PR - Brasil
Caixa Postal xx.xxx – 81520-260 – Curitiba – PR – Brazil

gabriela.rech@ufpr.br, gabrielealves1@ufpr.br, giovanni.tenereli@ufpr.br

luan.marquetti@ufpr.br, marcellymaciel@ufpr.br

Abstract. *This study addresses the development of a Big Data system for sentiment analysis on social media. From a JSON database, we extract and analyze sentiments using two specialized Python libraries. The processed data is then stored in two distinct database architectures: MongoDB (non-relational) and a relational database (PostgreSQL). We perform queries on the data to evaluate the characteristics and efficiency of each storage approach, allowing for a comparative analysis of the performance between non-relational and relational databases.*

Resumo. *Este estudo aborda o desenvolvimento de um sistema de Big Data para análise de sentimentos em redes sociais. A partir de uma base de dados em JSON, extraímos e analisamos sentimentos usando duas bibliotecas em Python especializadas. Em seguida, os dados processados são armazenados em duas arquiteturas de banco de dados distintas: MongoDB (não relacional) e um banco de dados relacional (PostgreSQL). Realizamos consultas sobre os dados para avaliar as características e a eficiência de cada abordagem de armazenamento, permitindo uma análise comparativa entre os desempenhos dos bancos de dados não relacionais e relacionais.*

1. Introdução

O crescimento exponencial dos dados em redes sociais tem criado novos desafios no armazenamento e análise de dados. Bancos de dados relacionais tradicionais, embora robustos e confiáveis, podem enfrentar limitações ao lidar com dados semiestruturados em larga escala, típicos da análise de mídias sociais. Este estudo compara o MongoDB com o PostgreSQL no gerenciamento de dados de análise de sentimentos, focando em aspectos críticos como desempenho, escalabilidade e facilidade de uso.

Este artigo é estruturado da seguinte forma: a Seção 2 apresenta a tecnologia de Big Data utilizada, suas características e vantagens. A Seção 3 detalha o estudo de caso, incluindo a descrição dos dados, perguntas de pesquisa e resultados esperados. Na Seção 4, são apresentados os resultados obtidos e uma análise comparativa das duas arquiteturas de banco de dados. A Seção 5 conclui com recomendações e considerações finais sobre as tecnologias empregadas, seguidas pelas referências e anexos, que incluem um guia prático para a reprodução do projeto.

2. Tecnologia Big Data Utilizada

2.1. MongoDB

O MongoDB é um banco de dados NoSQL orientado a documentos que armazena dados em documentos flexíveis no formato BSON (Binary JSON). Suas principais características incluem:

- Flexibilidade de esquema permitindo estruturas de documentos variáveis
- Escalabilidade horizontal através de sharding
- Suporte nativo para dados geoespaciais e busca textual
- Linguagem de consulta rica suportando agregações complexas

O MongoDB é particularmente adequado para:

- Aplicações com estruturas de dados em evolução
- Dados de alto volume com esquemas variáveis
- Casos que requerem escalabilidade horizontal
- Integração de dados em tempo real

2.2. PostgreSQL

O PostgreSQL é um sistema de banco de dados relacional robusto conhecido por:

- Conformidade com ACID
- Rico conjunto de tipos de dados e índices
- Otimização avançada de consultas
- Capacidades de busca full-text
- Operações de join extensivas

3. Estudo de Caso

3.1. Descrição do Dataset

O estudo utiliza um dataset de análise de sentimentos processado a partir do dataset original do Twitter. Durante o pré-processamento, realizamos várias etapas de limpeza e validação: Dataset Original:

- Tamanho inicial: 1.600.000 registros
- Fonte: Dados de sentimentos do Twitter

Processo de Limpeza:

- Remoção de datas inválidas: 1 registro removido
- Eliminação de duplicatas: 531 registros removidos
- Dataset final: 1.048.044 registros únicos e validados

3.2. Modelos de Dados

3.2.1. Esquema MongoDB

```
{
    "tweet_id": Long,
    "date": String,
    "user": {
```

```
        "username": String,
        "flag": String
    },
    "content": {
        "original_text": String,
        "cleaned_text": String,
        "original_sentiment": String
    },
    "sentiment_analysis": {
        "target": Integer,
        "textblob_sentiment": String,
        "vader_sentiment": String,
        "textblob_polarity": Float,
        "vader_compound": Float
    }
}
```

3.2.2. Esquema PostgreSQL

```
CREATE TABLE users (
user_id SERIAL PRIMARY KEY,
username VARCHAR(50) NOT NULL,
flag VARCHAR(50)
);

CREATE TABLE tweets (
tweet_id BIGINT PRIMARY KEY,
user_id INTEGER REFERENCES users(user_id),
date TIMESTAMP NOT NULL,
original_text TEXT NOT NULL,
cleaned_text TEXT,
original_sentiment VARCHAR(20)
);

CREATE TABLE sentiment_analysis (
sentiment_id SERIAL PRIMARY KEY,
tweet_id BIGINT REFERENCES tweets(tweet_id),
target INTEGER NOT NULL,
textblob_sentiment VARCHAR(20),
vader_sentiment VARCHAR(20),
textblob_polarity FLOAT,
vader_compound FLOAT
);
```

4. Resultados e Avaliações

4.1. Ambiente de Teste

- Sistema Operacional: Pop!OS 22.04
- Hardware:
  - CPU: 11th Gen Intel i5-1145G7 (8) @ 4.400GHz
  - GPU: Intel TigerLake-LP GT2 [Iris Xe Graphics]
  - RAM: 16GB (15697MiB)
  - Modelo: ThinkPad T14 Gen 2i
- Ambiente Desktop:
  - DE: GNOME 42.9
  - Terminal: gnome-terminal
  - Resolução: 1920x1080

4.2. Métricas de Performance

4.2.1. Performance de Importação

MongoDB:

- Tempo total: 93.40 segundos
- Taxa de inserção: 11,220.47 documentos/segundo
- Uso de memória: Pico de 2,313.05 MB

PostgreSQL:

- Tempo total: 153.85 segundos
- Taxa de inserção: 6,812.11 documentos/segundo
- Uso de memória: Pico de 693.85 MB

4.2.2. Performance de Consultas

Consultas Simples:

- MongoDB Find: 0.0022s
- PostgreSQL Select: 0.0026s

Busca Textual:

- MongoDB Text Search: 0.0037s
- PostgreSQL Text Search: 0.0095s

Agregações:

- MongoDB Aggregation: 0.9383s
- PostgreSQL Aggregation: 0.0576s

Joins/Lookups:

- MongoDB Lookup: 0.0060s
- PostgreSQL Join: 0.0028s

4.3. Análise dos Resultados

4.3.1. Importação de Dados

A análise da performance de importação revelou aspectos interessantes sobre o comportamento dos dois sistemas. O MongoDB demonstrou uma clara vantagem na velocidade de importação, completando a tarefa em 93.40 segundos, enquanto o PostgreSQL necessitou de 153.85 segundos. Esta diferença pode ser atribuída principalmente à natureza schema-less do MongoDB, que elimina a necessidade de validações de esquema durante a importação.

Entretanto, esta vantagem em velocidade vem com um custo significativo em termos de consumo de memória. O MongoDB apresentou um pico de uso de memória de 2,313.05 MB, mais que três vezes o consumo do PostgreSQL, que manteve um uso mais moderado de 693.85 MB. Este comportamento é particularmente relevante para ambientes com recursos limitados de hardware.

4.3.2. Consultas Simples

Nas operações de consultas simples, observamos uma diferença moderada entre os sistemas. O MongoDB apresentou um tempo médio de resposta de 0.0022s, superando o PostgreSQL em 13.9% (0.0026s). Esta vantagem pode ser atribuída à estrutura de armazenamento orientada a documentos do MongoDB, que permite acesso direto aos dados sem necessidade de joins ou reconstruções de relacionamentos.

4.3.3. Busca Textual

A performance em buscas textuais revelou uma das maiores vantagens do MongoDB, com tempo de resposta de 0.0037s contra 0.0095s do PostgreSQL, representando uma diferença de 61.6%. Este resultado é particularmente relevante para aplicações de análise de sentimentos, onde buscas textuais são operações frequentes. O mecanismo de indexação textual nativo do MongoDB mostrou-se mais eficiente para este tipo específico de operação.

4.3.4. Operações Complexas

A análise das operações complexas revelou o ponto mais significativo de diferença entre os dois sistemas. Nas operações de agregação, o PostgreSQL demonstrou uma superioridade impressionante, sendo 1529.4% mais rápido que o MongoDB (0.0576s vs 0.9383s). Esta diferença substancial pode ser explicada pelo motor de processamento de consultas maduro do PostgreSQL e sua otimização específica para operações de agregação.

Similarmente, nas operações de join/lookup, o PostgreSQL manteve sua vantagem, sendo 114.2% mais rápido (0.0028s vs 0.0060s). Este resultado era esperado, considerando que o modelo relacional foi especificamente projetado para otimizar operações de junção entre tabelas.

4.3.5. Considerações sobre Consistência

Um aspecto importante observado durante os testes foi a consistência dos resultados. O PostgreSQL apresentou menor variação entre execuções, especialmente em operações complexas, sugerindo um comportamento mais previsível em ambiente de produção. O MongoDB, embora mais rápido em certas operações, mostrou maior variabilidade nos tempos de resposta.

4.3.6. Impacto do Volume de Dados

É importante notar que estes resultados foram obtidos com um dataset de 1.048.044 registros, um volume significativo mas não extremo. O comportamento dos sistemas pode variar com volumes maiores de dados, especialmente em aspectos como uso de memória e tempo de resposta em operações complexas.

5. Conclusão

Com base em nossa análise detalhada, podemos concluir que cada sistema possui seus pontos fortes específicos: MongoDB se destaca em:

- Importação rápida de dados
- Consultas simples
- Buscas textuais
- Flexibilidade de esquema

PostgreSQL se destaca em:

- Operações de agregação complexas;
- Joins e relacionamentos;
- Eficiência no uso de memória;
- Consistência de performance.

Para o caso específico de análise de sentimentos, a escolha entre MongoDB e PostgreSQL dependerá das prioridades do projeto:

- Se a prioridade é ingestão rápida de dados e consultas simples: MongoDB;
- Se a prioridade é análise complexa e joins: PostgreSQL.

6. Referências

References

[1] MongoDB Documentation. Disponível em: https://docs.mongodb.com/

[2] PostgreSQL Documentation. Disponível em: https://www.postgresql.org/docs/

7. Anexos: Guida de Reprodução do Projeto

Este anexo apresenta um guia detalhado para reprodução do experimento, incluindo configurações, scripts e procedimentos necessários.

7.1. Ambiente de Desenvolvimento

7.1.1. Requisitos de Hardware Recomendados

- CPU: Processador quad-core ou superior
- RAM: Mínimo 8GB
- Armazenamento: 2GB de espaço livre em SSD

7.1.2. Requisitos de Software

- Sistema Operacional: Ubuntu 22.04 LTS ou similar (recomendado utilizar Linux)
- Python 3.8+
- pip (gerenciador de pacotes Python)
- Git

7.2. Configuração do Ambiente

7.2.1. Instalação do MongoDB

```
# Adicionar chave do repositório MongoDB
wget -qO - https://www.mongodb.org/static/pgp/server-5.0.asc |
    sudo apt-key add -

# Adicionar repositório
echo "deb [ arch=amd64,arm64 ] https://repo.mongodb.org/apt/ubuntu
    focal/mongodb-org/5.0 multiverse" |
    sudo tee /etc/apt/sources.list.d/mongodb-org-5.0.list

# Atualizar repositórios e instalar
sudo apt-get update
sudo apt-get install -y mongodb-org

# Iniciar serviço
sudo systemctl start mongod
sudo systemctl enable mongod
```

7.2.2. Instalação do PostgreSQL

```
# Instalar PostgreSQL
sudo apt-get install postgresql postgresql-contrib

# Iniciar serviço
sudo systemctl start postgresql
sudo systemctl enable postgresql

# Criar usuário e banco de dados
sudo -u postgres psql -c "CREATE USER benchmark WITH
```

```
    PASSWORD 'benchmark';"
sudo -u postgres psql -c "CREATE DATABASE sentiment_analysis;"
sudo -u postgres psql -c "GRANT ALL PRIVILEGES ON DATABASE
    sentiment_analysis TO benchmark;"
```

7.3. Dependências Python

7.3.1. Instalação de Pacotes

```
# Instalar dependências Python
pip install pandas numpy pymongo psycopg2-binary textblob
    nltk tqdm seaborn matplotlib

# Baixar recursos NLTK necessários
python -c "import nltk; nltk.download('vader_lexicon')"
```

7.4. Execução dos Testes

7.4.1. Preparação do Ambiente

```
# Realizar o fork e clonar repositório
git clone https://github.com/gab-i-alves/mongodb-postgres-comparison.gi
cd mongodb_postgres_comparison

# Configurar variáveis de ambiente
export MONGO_URI="mongodb://localhost:27017"
export POSTGRES_URI="postgresql://benchmark:benchmark@
    localhost:5432/sentiment_analysis"
```

7.4.2. Execução dos Scripts

Execute os scripts na seguinte ordem:

1. `improved-sentiment-analysis.py`: Processamento inicial dos dados
2. `improved-mongodb.py`: Importação para MongoDB
3. `improved-postgres.py`: Importação para PostgreSQL
4. `performance-benchmark.py`: Execução dos benchmarks

7.5. Monitoramento e Análise

7.5.1. Arquivos de Log

Os seguintes arquivos de log são gerados durante a execução:

- `sentiment_analysis.log`: Log do processamento de dados
- `mongodb_import.log`: Log de importação MongoDB
- `postgres_import.log`: Log de importação PostgreSQL
- `benchmark_results.log`: Métricas de performance

7.5.2. Análise dos Resultados

Os resultados são disponibilizados em:

- `benchmark_report.json`: Resultados detalhados
- Gráficos gerados no diretório `charts/`:

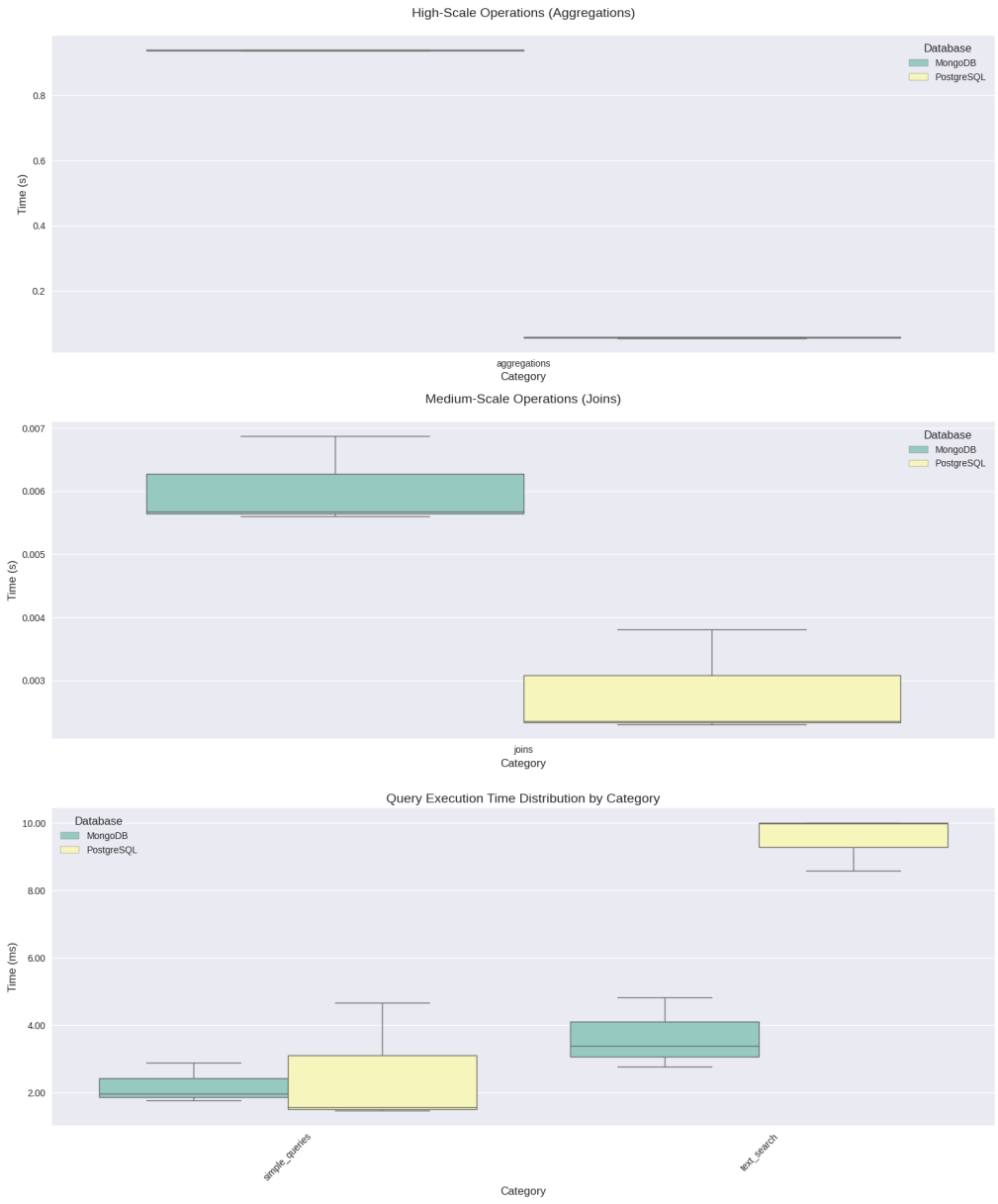

Figure 1. Distribuição dos tempos de execução por tipo de operação

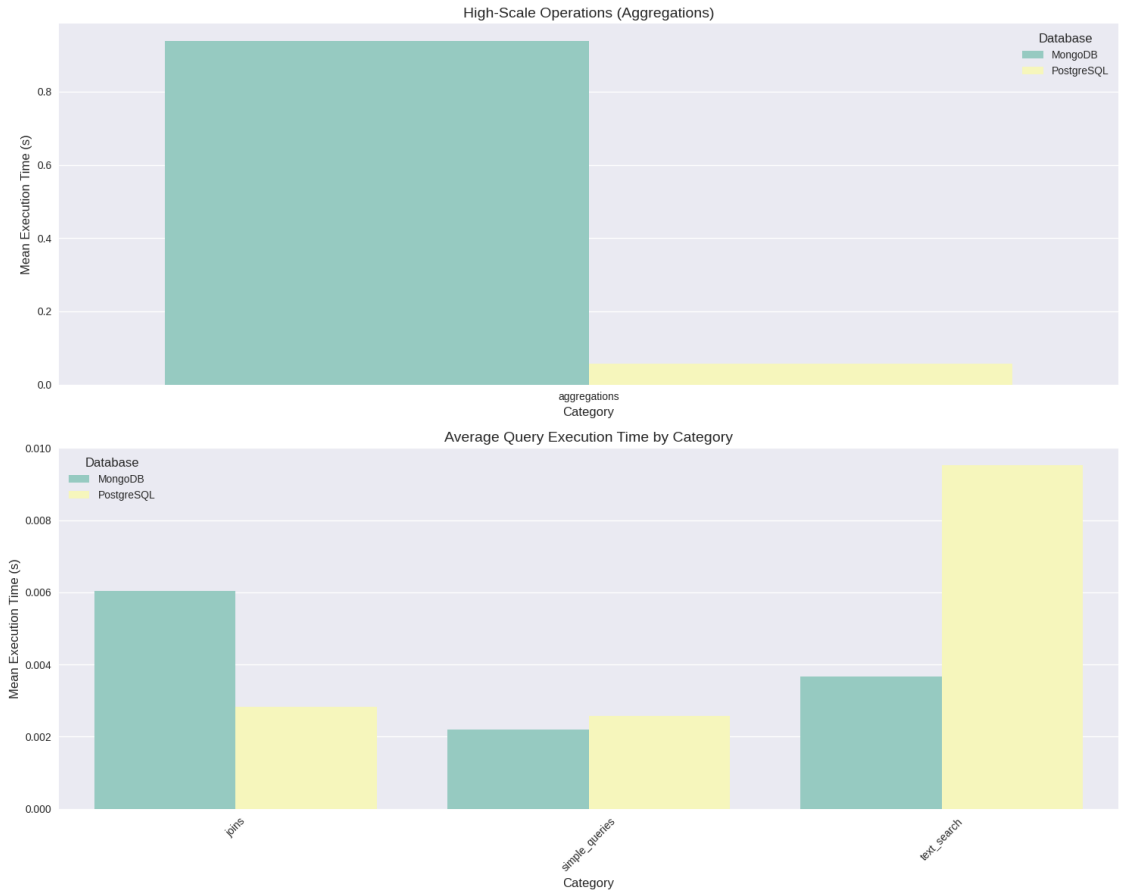

Figure 2. Tempos médios de execução para diferentes operações

Figure 3. Distribuição de operações de alta escala (Agregações)

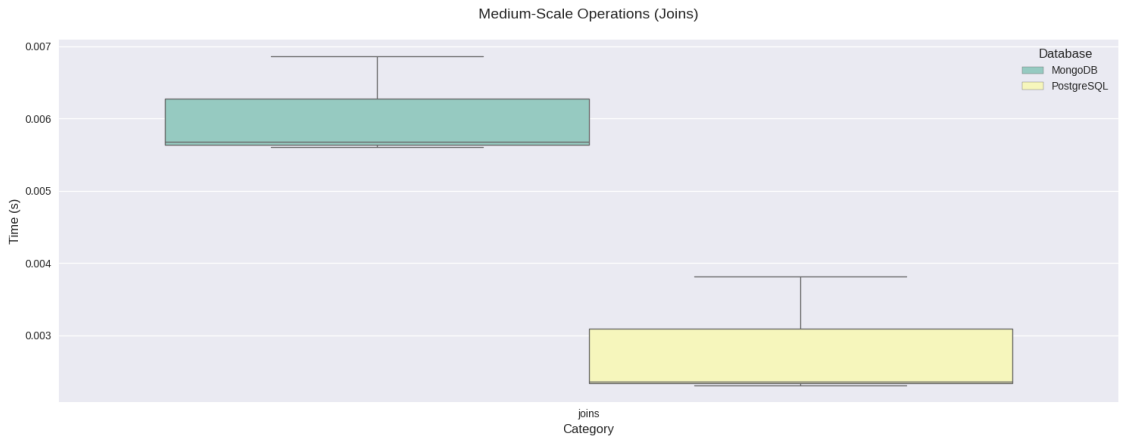

Figure 4. Distribuição de operações de média escala (Joins)

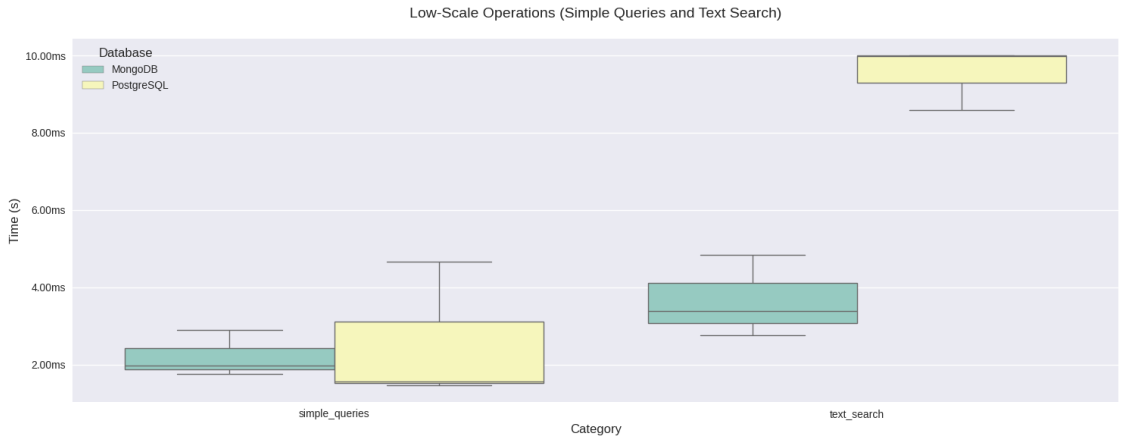

Figure 5. Distribuição de operações de baixa escala (Consultas Simples e Buscas Textuais)

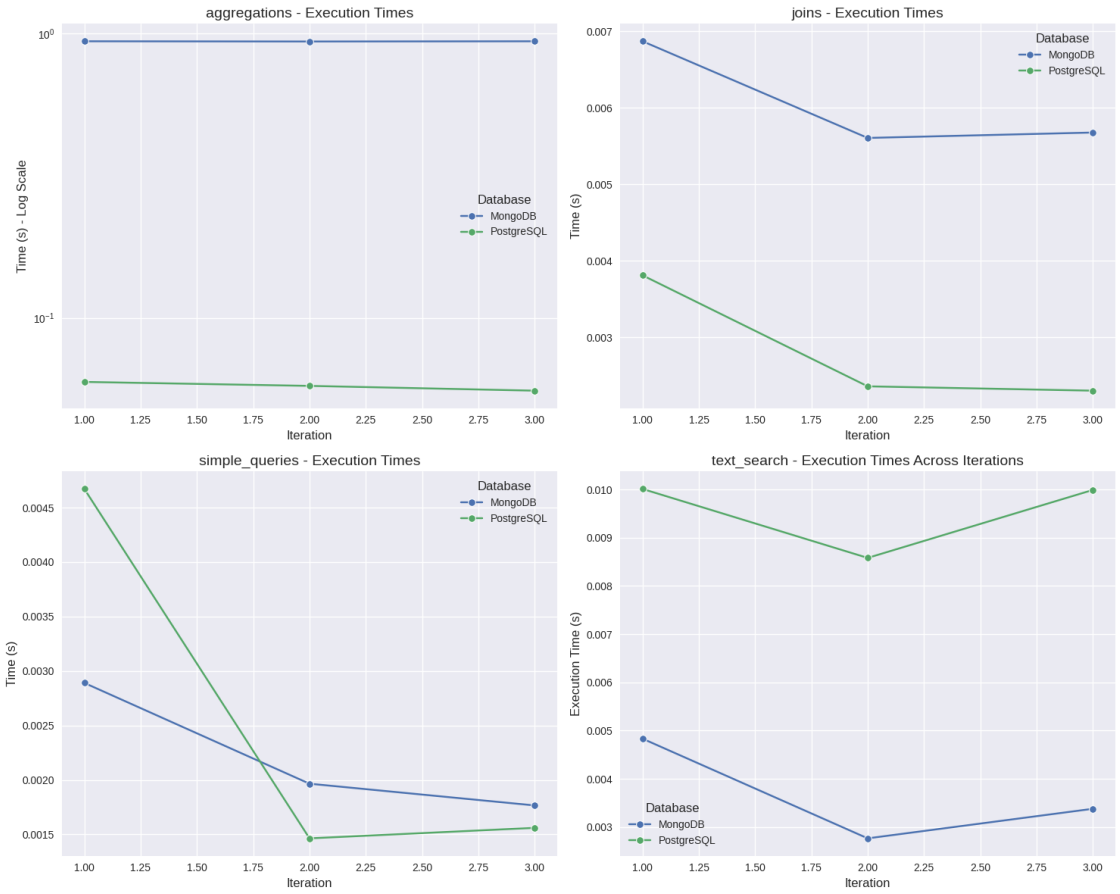

Figure 6. Comparação de tempos de execução ao longo das iterações