Tecnologia em Análise e Desenvolvimento de Sistemas

Disciplina - Banco de dados

Nome: Cassio Augusto Oliveira da Silva

Prontuário: CJ3019501

Banco de Dado NoSQL

INSTITUTO FEDERAL
São Paulo
Campus Campos do Jordão

Sumario

1. Resumo

2. Entrodução

3. Obejetivo

4. Metodologia

5. NoSQL

6. MongoDB

7. Exemplo de Projeto usando NoSQL

8. Exemplo de Implementação em MongoDB (Banco de Dados NoSQL Documental)

9. Motivos para Escolha do MongoDB

10. Conclusão

11. Bibliografia

Resumo

Este trabalho explora a utilização de bancos de dados NoSQL, especificamente o MongoDB, em um projeto de desenvolvimento de um aplicativo para reconhecimento de fotos de carne por uma IA . O aplicativo visa ajudar os consumidores a obter uma melhor compreensão sobre diferentes tipos de carne, permitindo aos usuários postar fotos, comentar e compartilhar receitas, além de categorizar informações e permitir a administração centralizada por um administrador. Este trabalho discute as vantagens do modelo NoSQL em lidar com as necessidades dinâmicas e de alta escalabilidade de aplicações modernas, como a proposta aqui apresentada.

Introdução

Com o crescimento exponencial na geração de dados por aplicações móveis, redes sociais e dispositivos IoT, tornou-se evidente a necessidade de plataformas de banco de dados capazes de lidar com grandes volumes de dados de forma eficiente e escalável. Os bancos de dados NoSQL surgiram como uma resposta a esses desafios, oferecendo flexibilidade em esquemas de dados, alta disponibilidade e capacidade de escalabilidade horizontal. Entre os modelos NoSQL, o MongoDB se destaca pela sua capacidade de armazenar dados em formato de documentos, suportando consultas ricas e estruturas de dados dinâmicas.

Este trabalho explora como o MongoDB pode ser aplicado em um projeto prático: um aplicativo de reconhecimento de foto de carne. O aplicativo permite aos usuários fotografar diferentes tipos de carne, identificar características e informações relevantes sobre cada tipo, além de interagir com uma comunidade por meio de compartilhamento de receitas, comentários e categorização de dados. O administrador terá a função de gerenciar o conteúdo, garantir a qualidade e a segurança das informações compartilhadas.

Objetivo do Trabalho

O objetivo deste trabalho é demonstrar como NoSQL junto com MongoDB pode ser implementado de maneira eficaz e eficiente em um cenário real de desenvolvimento de software. O projeto de visto proposto:

- Implementar um banco de dados MongoDB para armazenamento e gerenciamento de dados relacionados a fotos de carne, receitas, comentários e categorias.

- Desenvolver funcionalidades que permitam aos usuários interagir através do upload de fotos de carne, comentários em receitas e compartilhamento de novas receitas

- Integrar um sistema de administração para gerenciar usuários, conteúdo e gerar relatórios sobre a utilização do aplicativo.

- Avaliar as vantagens do MongoDB em relação aos bancos de dados relacionais tradicionais, especialmente em termos de flexibilidade de esquema, desempenho e escalabilidade.

Ao final deste trabalho, esperamos não apenas demonstrar a técnica prevista de uso do MongoDB, mas também fornecer melhores práticas para implementação de aplicações de banco de dados NoSQL em cenários semelhantes

Metodologia

E metodologia usada foi Realização de uma pesquisa detalhada das principais fontes de informação sobre bancos de dados NoSQL, incluindo documentos técnicos, artigos acadêmicos e documentação oficial de plataformas como Oracle, IBM, Microsoft Azure, Amazon Web Services.

Definição de tipos de bancos de dados NoSQL usando classificação e compreensão dos diferentes tipos de bancos de dados NoSQL, como valores-chave, orientados a documentos, colunas largas, baseados em gráficos e armazenamentos na memória. Isso inclui uma análise detalhada das características, estruturas de dados suportadas e casos de uso típicos para cada tipo.

NoSQL

NoSQL, que significa "Not Only SQL" (Não Apenas SQL), é uma abordagem de gerenciamento de banco de dados que oferece uma alternativa aos tradicionais bancos de dados relacionais (SQL). Com o crescimento explosivo dos dados gerados por aplicações modernas, como redes sociais, dispositivos IoT e grandes sites de e-commerce, surgiu a necessidade de uma solução de banco de dados que pudesse lidar com grandes volumes de dados, alta velocidade de processamento e

estruturas de dados flexíveis. É aqui que os bancos de dados NoSQL entram em cena, oferecendo uma abordagem mais ágil e escalável para o gerenciamento de dados.

A necessidade de bancos de dados NoSQL surgiu da explosão de dados e do aumento das demandas das aplicações modernas. Com a proliferação de dispositivos móveis, redes sociais, IoT (Internet das Coisas), e aplicações web que geram grandes volumes de dados em alta velocidade, os sistemas tradicionais de banco de dados relacional começaram a mostrar suas limitações. A estrutura rígida de esquemas, a dificuldade em escalar horizontalmente e o desempenho insuficiente para grandes volumes de dados levaram ao desenvolvimento de alternativas mais adequadas para esses novos desafios.

Vantagens do NoSQL

1. **Escalabilidade**

    - **Horizontal**: Os bancos de dados NoSQL são projetados para escalar horizontalmente, o que significa que você pode adicionar mais servidores ao seu sistema para aumentar a capacidade de processamento e armazenamento. Isso é diferente dos bancos de dados relacionais, que geralmente escalam verticalmente, exigindo hardware mais potente.

    - **Desempenho**: Essa escalabilidade horizontal permite que os bancos de dados NoSQL gerenciem grandes volumes de dados e suportem altas taxas de leitura e escrita, essenciais para aplicações com muitos usuários simultâneos.

2. **Flexibilidade**

    - **Esquema Flexível**: Ao contrário dos bancos de dados relacionais que exigem um esquema predefinido, os bancos de dados NoSQL permitem a inserção de dados sem um esquema fixo. Isso facilita a adaptação às mudanças nos requisitos de dados ao longo do tempo, permitindo a adição de novos campos e tipos de dados sem necessidade de migrações complexas.

- **Diversidade de Modelos**: Bancos de dados NoSQL suportam diversos modelos de dados, como chave-valor, documentos, colunas largas e grafos, oferecendo maior flexibilidade para desenvolver aplicações que exigem diferentes estruturas de dados.

3. **Desempenho**

- **Leitura e Escrita Rápida**: Muitos bancos de dados NoSQL são otimizados para operações de leitura e escrita rápidas. Isso os torna ideais para aplicativos que exigem respostas rápidas, como redes sociais, sistemas de e-commerce e aplicações de análise de dados em tempo real.

- **Cache Integrado**: Alguns bancos NoSQL, como o Redis, são projetados para operar inteiramente na memória, oferecendo tempos de resposta extremamente rápidos.

4. **Grande Volume de Dados**

- **Big Data**: Os bancos de dados NoSQL são particularmente adequados para lidar com grandes volumes de dados, conhecidos como Big Data. Eles podem armazenar e processar grandes quantidades de informações geradas por sensores IoT, logs de servidores, transações de e-commerce, entre outros.

Desvantagens do NoSQL

1. **Consistência**

- **Modelo CAP**: Muitos bancos de dados NoSQL seguem o Teorema CAP, que afirma que um sistema distribuído pode oferecer apenas duas das três garantias: Consistência, Disponibilidade e Partição Tolerante. Muitos optam por sacrificar a

  consistência para garantir alta disponibilidade e partição tolerante, o que pode ser um problema para aplicações que exigem dados sempre corretos e atualizados.

- **Eventual Consistency**: Em alguns sistemas NoSQL, a consistência dos dados é "eventual", ou seja, os dados se tornam consistentes após um período de tempo, o que pode não ser aceitável para todas as aplicações.

2. **Suporte a Transações**

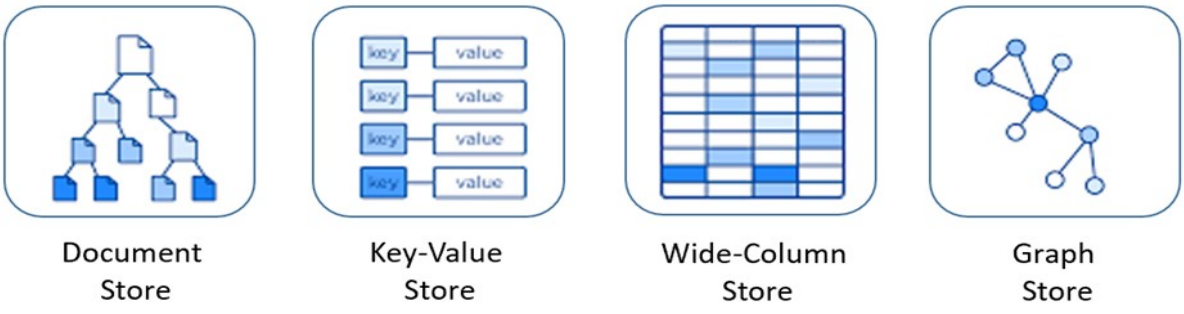
- **Transações Limitadas**: Enquanto bancos de dados relacionais oferecem suporte robusto a transações ACID (Atomicidade, Consistência, Isolamento e Durabilidade), muitos bancos de dados NoSQL oferecem suporte limitado ou inexistente a essas transações, o que pode complicar o desenvolvimento de aplicações que exigem operações complexas e consistência de dados garantida.

3. **Menos Maturidade**

- **Ferramentas de Suporte**: Embora os bancos de dados NoSQL estejam ganhando popularidade, alguns deles ainda são relativamente novos e podem não ter o mesmo nível de maturidade e conjunto de ferramentas de suporte que os bancos de dados relacionais. Isso pode incluir ferramentas de backup e recuperação, monitoramento, e integração com outras plataformas.

- **Comunidade e Documentação**: Dependendo do banco de dados NoSQL escolhido, pode haver uma comunidade menor e menos documentação disponível, o que pode dificultar a resolução de problemas e o aprendizado.

Tipos de Bancos de Dados NoSQL e Exemplos



https://learn.microsoft.com/pt-br/dotnet/architecture/cloud-native/relational-vs-nosql-data

1. **Key-Value Storage (Armazenamento Chave-Valor)**

    - **Descrição**: Bancos de dados chave-valor armazenam dados como pares de chave e valor, onde cada chave é única e está associada a um valor. Esse modelo é extremamente simples e eficiente para operações básicas de leitura e escrita, tornando-o ideal para aplicações que não exigem consultas complexas.

    - **Exemplos de Bancos**: Redis, Riak, Amazon DynamoDB

        - **Redis**: Um banco de dados em memória altamente performático, usado como cache, banco de dados e broker de mensagens. Redis suporta estruturas de dados como strings, hashes, listas, conjuntos e sorted sets.

        - **Riak**: Um banco de dados distribuído altamente disponível, conhecido por sua capacidade de escalar horizontalmente e oferecer alta disponibilidade.

        - **Amazon DynamoDB**: Um serviço de banco de dados NoSQL totalmente gerenciado que oferece desempenho rápido e previsível com escalabilidade perfeita.

    - **Uso Ideal**: Cache (armazenamento temporário de dados para acesso rápido), sessões de usuário, configurações de aplicativos, e qualquer aplicação onde a simplicidade e a velocidade são prioritárias.

2. **Document Storage (Armazenamento de Documentos)**

    - **Descrição**: Bancos de dados de documentos armazenam dados em documentos, geralmente no formato JSON, BSON ou XML. Cada documento é uma coleção de pares chave-valor, e documentos em uma mesma coleção podem ter diferentes campos e estruturas. Isso oferece uma grande flexibilidade na maneira como os dados são armazenados e consultados.

    - **Exemplos de Bancos**: MongoDB, CouchDB

- **MongoDB**: Um dos bancos de dados NoSQL mais populares, MongoDB armazena dados em documentos BSON (uma versão binária de JSON), oferecendo suporte a consultas ricas e indexação.

- **CouchDB**: Um banco de dados de documentos que armazena dados em JSON e oferece uma API RESTful para interação, facilitando a integração com aplicações web.

- **Uso Ideal**: Aplicativos web que precisam de flexibilidade para armazenar dados variados, gerenciamento de conteúdo, e-commerce, e qualquer aplicação onde a estrutura dos dados pode mudar frequentemente.

3. **Wide-Column Storage (Armazenamento de Colunas Largas)**

- **Descrição**: Bancos de dados de colunas largas armazenam dados em tabelas, mas, ao contrário dos bancos de dados relacionais, cada linha pode ter um conjunto diferente de colunas. As colunas são agrupadas em famílias, e os dados são armazenados e acessados por famílias de colunas, o que permite a leitura e escrita eficientes de grandes volumes de dados.

- **Exemplos de Bancos**: Apache Cassandra, HBase

  - **Apache Cassandra**: Um banco de dados distribuído projetado para gerenciar grandes quantidades de dados através de muitos servidores, oferecendo alta disponibilidade sem ponto único de falha.

  - **HBase**: Um banco de dados não relacional escalável e distribuído, baseado no Google Bigtable, e projetado para armazenar grandes tabelas esparsas.

- **Uso Ideal**: Análise de dados, registros de log, redes sociais, e qualquer aplicação que precisa processar grandes volumes de dados rapidamente.

4. **Graph Storage (Armazenamento de Grafos)**

- **Descrição**: Bancos de dados de grafos são projetados para armazenar e navegar relações complexas entre dados. Eles utilizam estruturas de grafos, com nós (entidades), arestas (relações) e propriedades (informações sobre nós e arestas), tornando-os ideais para aplicações onde as relações entre os dados são tão importantes quanto os próprios dados.

- **Exemplos de Bancos**: Neo4j, OrientDB

    - **Neo4j**: Um dos bancos de dados de grafos mais conhecidos, Neo4j oferece uma linguagem de consulta poderosa chamada Cypher, que facilita a criação e a consulta de grafos.

    - **OrientDB**: Um banco de dados multi-modelo que suporta documentos, grafos, chave-valor e objetos, oferecendo grande flexibilidade e desempenho.

- **Uso Ideal**: Redes sociais, sistemas de recomendação, gerenciamento de redes, e qualquer aplicação que precisa lidar com dados interconectados de forma eficiente.

5. **In-Memory Storage (Armazenamento em Memória)**

    - **Descrição**: Bancos de dados em memória armazenam dados na memória RAM em vez de em disco, oferecendo acesso extremamente rápido aos dados. Isso os torna ideais para aplicações que exigem baixa latência e alto desempenho.

    - **Exemplos de Bancos**: Redis, Memcached

        - **Redis**: Além de ser um banco de dados chave-valor, Redis é usado como cache, broker de mensagens, e suporta várias estruturas de dados em memória.

        - **Memcached**: Um sistema de cache de memória distribuída usado para acelerar aplicativos dinâmicos ao alocar dados em cache na RAM.

    - **Uso Ideal**: Cache, sessões de usuário, filas de mensagens, e qualquer aplicação onde a velocidade de acesso aos dados é crucial.

MongoDB

MongoDB representa uma revolução no campo dos sistemas de gerenciamento de banco de dados não relacionais (NoSQL). Em contraste com os tradicionais bancos de dados relacionais que utilizam tabelas rigidamente estruturadas, MongoDB adota um modelo baseado em documentos

JSON (JavaScript Object Notation), oferecendo flexibilidade excepcional para desenvolvedores e empresas.

Modelagem Flexível de Dados

Ao contrário dos sistemas baseados em SQL, onde o esquema dos dados é pré-definido, MongoDB permite esquemas dinâmicos. Isso significa que os desenvolvedores podem criar registros de dados sem a necessidade de estruturas rígidas, adaptando-se facilmente às necessidades em evolução dos aplicativos. A capacidade de armazenar e consultar uma ampla gama de tipos de dados de forma eficiente faz do MongoDB uma escolha ideal para ambientes que lidam com dados não estruturados ou semiestruturados, como redes sociais, análises de big data e IoT (Internet of Things).

Comparação com Bancos de Dados Relacionais e Outros NoSQL

MongoDB se destaca em comparação com bancos de dados relacionais como MySQL devido à sua capacidade de escalar horizontalmente e lidar com grandes volumes de dados de maneira mais eficiente. Enquanto bancos de dados tradicionais como MySQL são ideais para aplicações que exigem transações ACID e integridade de dados, MongoDB brilha em cenários onde a flexibilidade e a velocidade de desenvolvimento são prioritárias.

Em comparação com outros bancos de dados NoSQL, como Cassandra, MongoDB oferece um equilíbrio único entre flexibilidade e desempenho. Enquanto Cassandra utiliza uma estrutura de tabela com linhas e colunas que favorece a uniformidade e a durabilidade dos dados, MongoDB é mais adequado para aplicativos que requerem consultas ad hoc e manipulação eficiente de documentos complexos.

Aplicações e Casos de Uso

MongoDB é amplamente utilizado em uma variedade de aplicações:

- **Aplicativos Móveis e IoT:** A flexibilidade do modelo de documento do MongoDB é ideal para armazenar e processar dados de aplicativos móveis e dispositivos IoT, facilitando a escalabilidade e o gerenciamento de dados distribuídos.

- **Análise em Tempo Real:** Empresas que dependem de análises em tempo real para tomar decisões estratégicas podem aproveitar a capacidade do MongoDB de lidar com grandes

volumes de dados de maneira eficiente, convertendo documentos JSON em estruturas de dados facilmente manipuláveis.

- **Sistemas de Gerenciamento de Conteúdo (CMS):** Plataformas de CMS podem se beneficiar da capacidade do MongoDB de suportar atributos e estruturas de dados dinâmicas, facilitando a personalização e expansão de funcionalidades sem grandes alterações no esquema.

Benefícios e Recursos

Além da flexibilidade e escalabilidade, MongoDB oferece uma série de benefícios:

- **Balanceamento de Carga e Escalabilidade:** A fragmentação horizontal permite distribuir dados em várias máquinas virtuais, mantendo alto desempenho e disponibilidade sem depender de escalabilidade vertical.

- **Consultas Ad Hoc e Linguagem de Consulta Familiar:** MongoDB suporta consultas ad hoc que não requerem esquemas predefinidos, utilizando uma linguagem de consulta semelhante ao SQL que facilita a interação com os dados.

- **Suporte a Múltiplas Linguagens:** Com suporte a várias linguagens de programação, MongoDB é acessível para desenvolvedores de diferentes backgrounds, incluindo Python, JavaScript, Node.js, entre outras.

Exemplo de Projeto usando NoSQL

O projeto que darei de exemplo e um aplicativo que desenvolverei futuramente chamado **"School of Meat",** a dificuldade em identificar e escolher tipos adequados de carne é uma questão recorrente para muitos consumidores Este projeto visa desenvolver uma solução tecnológica por meio de um aplicativo de reconhecimento de carne bovina, empregando técnicas de IA para auxiliar os usuários na identificação precisa das variedades de carne disponíveis no mercado. Neste contexto, o Google Cloud oferece uma série de ferramentas e serviços que facilitam a criação, treinamento e implantação de modelos de IA. Este trabalho se concentra na aplicação do AutoML

Vision e do Cloud Storage com IA no desenvolvimento de um aplicativo voltado para o reconhecimento de diferentes tipos de carne.

Modelo NoSQL Baseado em Documentos

Um banco de dados NoSQL que usarei como exemplo e MongoDB baseado em documentos é adequado para esse tipo de modelo, onde cada documento pode representar um usuário, uma receita, um comentário, uma foto, uma notificação, um relatório, um administrador ou uma categoria.

Exemplo de Implementação em MongoDB (Banco de Dados NoSQL Documental)

Usuário (Usuario)

```
{
   "_id": ObjectId(""),
   "nome": "Nome do Usuário",
   "email": "usuario@email.com",
   "senha": "hashed_password",
   "receitas": [
      {
         "_id": ObjectId(""),
         "titulo": "Título da Receita",
         "ingredientes": ["Ingrediente 1", "Ingrediente 2"],
         "modo_preparo": "Descrição do modo de preparo",
         "foto": "caminho/para/foto.jpg",
         "categorias": ["Categoria 1", "Categoria 2"],
         "comentarios": [
```

```
            {
                "_id": ObjectId(""),
                "texto": "Comentário sobre a receita",
                "avaliacao": 5
            },
            {
                "_id": ObjectId(""),
                "texto": "Outro comentário sobre a receita",

                "avaliacao": 4
            }
        ]
    }
],
"fotos": [
    {
        "_id": ObjectId(""),
        "caminho": "caminho/para/foto.jpg",
        "tipo_carne": "Tipo de carne"
    }
],
"notificacoes": [
    {
        "_id": ObjectId(""),
        "mensagem": "Conteúdo da notificação"
    }
]
}
```


Administrador (Administrador)


```
{
```

```
"_id": ObjectId(""),
"nome": "Nome do Administrador",
"email": "admin@email.com",
"senha": "hashed_password",
"relatorios": [
    {
        "_id": ObjectId(""),

        "tipo": "Tipo do relatório",
        "conteudo": "Conteúdo do relatório"
```

Categoria (Categoria)

```
{
    "_id": ObjectId(""),
    "nome": "Nome da Categoria",
    "receitas": [
        {
            "_id": ObjectId(""),
            "titulo": "Título da Receita",
            "ingredientes": ["Ingrediente 1", "Ingrediente 2"],
            "modo_preparo": "Descrição do modo de preparo",
            "foto": "caminho/para/foto.jpg",
            "comentarios": [
                {
                    "_id": ObjectId(""),
                    "texto": "Comentário sobre a receita",
```

```
            "avaliacao": 5
        }
    ]
}
]
}
```


Motivos para Escolha do MongoDB

1. **Flexibilidade na Modelagem de Dados:**

    - MongoDB permite armazenar documentos sem a necessidade de uma estrutura de dados, o que é ideal para um aplicativo onde as receitas podem variar amplamente em termos de ingredientes, instruções e outros detalhes.
    -

2. **Gerenciamento de Dados Não Estruturados:**

    - As receitas contêm informações variadas, como listas de ingredientes, preparações específicas podem e fotos. MongoDB facilita o armazenamento desses dados não estruturados de maneira eficiente em documentos JSON.

3. **Escalabilidade horizontal:**

    - À medida que a plataforma cresce e mais usuários adicionam receitas e interação no aplicativo, o MongoDB suporta facilmente a escalabilidade horizontal, distribuindo dados entre vários servidores para lidar com aumentos de carga.

4. **Desempenho em Leituras e Escritas Rápidas:**

- As consultas no MongoDB são otimizadas para operações de leitura e escrita rápidas, o que é crucial para um aplicativo onde os usuários podem acessar e atualizar frequentemente suas receitas, comentários e fotos.
-

5. **Integração com Aplicações Modernas:**

- MongoDB é amplamente utilizado em aplicações modernas devido à sua capacidade de integração com frameworks e linguagens de programação comuns, facilitando o desenvolvimento e a manutenção do aplicativo.

A escolha do MongoDB para este projeto de aplicativo se baseia em sua capacidade de lidar com dados flexíveis e não estruturados de maneira eficiente, garantindo desempenho escalável e acesso rápido aos dados. Isso não apenas suporta a experiência dos usuários ao interagir mas também facilita o desenvolvimento e a expansão futura do aplicativo conforme mais funcionalidades são adicionadas.

Conclusão

Os bancos de dados NoSQL representam uma solução robusta e flexível para os desafios contemporâneos de gerenciamento de dados. Com a explosão de dados gerada por aplicações modernas, como redes sociais, dispositivos IoT e sistemas de e-commerce, torna-se evidente a necessidade de plataformas que ofereçam escalabilidade, flexibilidade e desempenho aos bancos de dados relacionais tradicionais superiores.

No contexto dos bancos de dados NoSQL, o MongoDB se destaca como um exemplo. Utilizando o modelo de armazenamento de documentos, o MongoDB permite uma manipulação eficiente e flexível de dados estruturados e não estruturados. Sua capacidade facilita a adaptação às mudanças frequentes nos requisitos das aplicações, sem comprometer o desempenho.

Para um projeto futuro, como o desenvolvimento de um sistema de gerenciamento de receitas e compartilhamento de receitas online, o MongoDB oferece uma solução ideal. A capacidade de armazenar e consultar documentos complexos em formato JSON/BSON permite a representação intuitiva de receitas com ingredientes, instruções e outros metadados relacionados. Além disso, a escalabilidade horizontal do MongoDB é crucial para suportar um grande número de usuários e volumes de dados crescentes conforme o sistema ganha popularidade.

Ao considerar a escolha de um banco de dados NoSQL para projetos futuros, como o mencionado acima, é importante avaliar não apenas as características técnicas, mas também a comunidade de suporte, a disponibilidade de ferramentas e a integração com outras tecnologias utilizadas no desenvolvimento de software moderno . O MongoDB se destaca não apenas pela sua tecnologia avançada, mas também pelo ecossistema robusto e pela vasta adoção na indústria.

Em resumo, o MongoDB exemplifica como os bancos de dados NoSQL podem capacitar desenvolvedores para construir aplicações escaláveis, flexíveis e de alto desempenho. Ao adotar uma abordagem NoSQL com o MongoDB, esperamos não apenas resolver os desafios atuais de gerenciamento de dados, mas também preparar o terreno para inovações futuras na era digital.

Bibliografia

Amazon Web Services. (nd). Bancos de dados de valores-chave NoSQL. Recuperado de **https://aws.amazon.com/pt/nosql/key-value/**

- Caçador Geek. (nd). Banco de Dados NoSQL: Um Manual Prático e Didático. Recuperado de https://blog.geekhunter.com.br/banco-de-dados-nosql-um-manual-pratico-e-didatico/#O_que_sao_bancos_de_dados_NoSQL

- IBM. (nd). MongoDB. Recuperado de https://www.ibm.com/br-pt/topics/mongodb

- IBM. (nd). Bancos de dados NoSQL. Recuperado de https://www.ibm.com/br-pt/topics/nosql-databases

- Microsoft Azure. (nd). O que é banco de dados NoSQL? Recuperado de https://azure.microsoft.com/pt-br/resources/cloud-computing-dictionary/what-is-nosql-database

- Microsoft Aprenda. (nd). Dados relacionais vs. dados NoSQL. Recuperado de https://learn.microsoft.com/pt-br/dotnet/architecture/cloud-native/relational-vs-nosql-data

- Oráculo. (nd). O que é NoSQL? Recuperado de https://www.oracle.com/br/database/nosql/what-is-nosql/