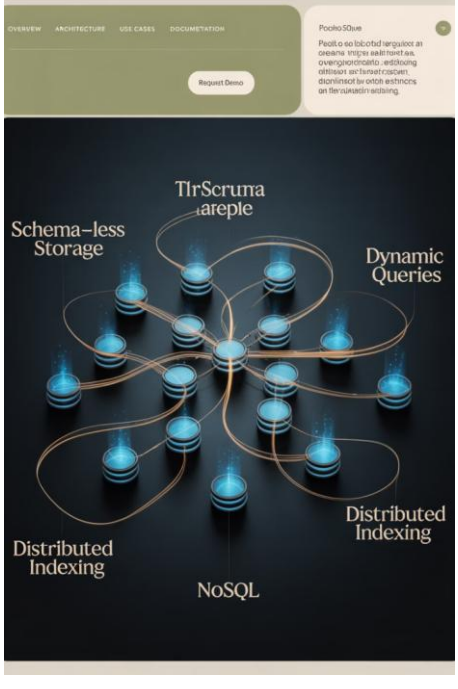


NoSQL e Cassandra para Dados Abertos Governamentais

Bem-vindos a esta apresentação sobre a aplicação de tecnologias NoSQL, especificamente Apache Cassandra, no contexto de dados abertos governamentais. Exploraremos como estas tecnologias oferecem soluções eficientes para o gerenciamento de grandes volumes de dados, superando limitações tradicionais dos bancos relacionais.

Abordaremos desde conceitos fundamentais de dados abertos e suas classificações, até a implementação prática de um estudo de caso utilizando dados do programa Bolsa Família. Veremos como o Cassandra, com sua arquitetura distribuída, pode oferecer desempenho e escalabilidade para iniciativas de transparência governamental.

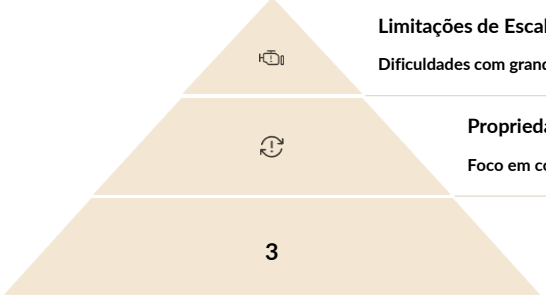
DT **por Dimas Tomadon**

Importante !

As informações e opiniões apresentadas pelo Prof.
Dimas Rogério Tomadon

Não refletem a opinião ou mesmo alguma informação interna das diversas empresas que ele trabalhou ou está trabalhando !

O Contexto dos Bancos de Dados Relacionais



Limitações de Escalabilidade

Dificuldades com grandes volumes de dados

Propriedades ACID**

Foco em consistência transacional

Modelo Relacional

Predominância histórica

O Movimento NoSQL

Escalabilidade Horizontal

Distribuição eficiente em múltiplos nós

Alta Performance

Otimizado para operações específicas

Distribuição Geográfica

Suporte nativo a múltiplas regiões

Modelos Flexíveis

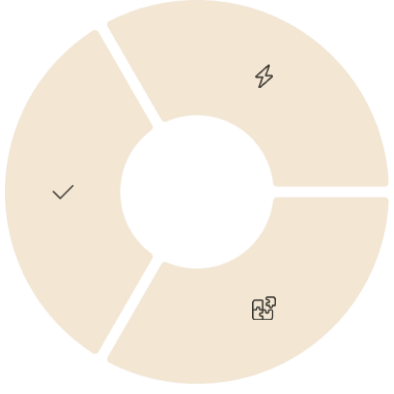
Esquemas adaptáveis às necessidades

O Teorema CAP

Consistência

Todos os nós veem os mesmos dados simultaneamente

- Leituras sempre retornam a versão mais recente
- Transações atômicas em todos os nós



Disponibilidade

O sistema permanece operacional mesmo com falhas

- Respostas garantidas para todas operações
- Sem tempo de inatividade

Tolerância a Partições

Funcionamento contínuo mesmo com falhas de rede

- Sistema distribuído resiliente
- Operação durante particionamentos

Principais Modelos NoSQL

Chave-Valor

Modelo mais simples, onde cada item é armazenado como um par chave-valor. Excelente para cache, sessões e dados simples com acesso por identificador único.

- Redis
- DynamoDB
- Riak

Documentos

Armazena documentos semi-estruturados (JSON, BSON, XML). Ideal para conteúdo heterogêneo com estrutura variável e consultas complexas.

- MongoDB
- CouchDB
- Firestore

Colunas

Armazena dados em famílias de colunas, otimizado para leituras/escritas em grandes volumes. Excelente para análises e big data.

- Cassandra
- HBase
- ScyllaDB

Grafos

Especializado em relações entre entidades. Ideal para dados altamente conectados e consultas de relacionamentos complexos.

- Neo4j
- JanusGraph
- Amazon Neptune

Cada modelo NoSQL foi desenvolvido para atender necessidades específicas, demonstrando que não existe uma solução única para todos os problemas. A escolha do modelo adequado depende das características dos dados, padrões de acesso e requisitos de consistência da aplicação.

Principais Modelos NoSQL

Bancos Relacionais vs. NoSQL

Bancos Relacionais

- Esquema fixo e estruturado
- Relacionamentos explícitos (chaves estrangeiras)
- Linguagem SQL padronizada
- Transações ACID
- Escalabilidade vertical predominante
- Normalização para evitar redundância

Bancos NoSQL

- Esquema flexível ou ausente
- Relacionamentos implícitos (desnormalização)
- APIs específicas por banco
- Relaxamento de propriedades ACID
- Escalabilidade horizontal nativa
- Redundância controlada para performance

Propriedades ACID

Atomicidade

Transações são executadas por completo ou nenhuma operação é realizada. Não há estados intermediários, garantindo que o sistema permaneça consistente mesmo após falhas.

Consistência

Transações levam o banco de dados de um estado válido para outro estado válido, mantendo todas as regras de integridade e restrições definidas no esquema.

Isolamento

Transações concorrentes são executadas como se fossem sequenciais, sem interferência mútua. Os efeitos de uma transação não são visíveis para outras até sua conclusão.

Durabilidade

Uma vez confirmada, a transação é persistente e sobrevive a falhas do sistema, normalmente através de logs de transação e técnicas de recuperação.

Apache Cassandra: Visão Geral

Origem

Desenvolvido pelo Facebook em 2008 para solucionar o problema da caixa de entrada de pesquisa. Tornou-se projeto Apache em 2009 e conquistou o status de projeto de alto nível em 2010.

Classificação

Banco de dados NoSQL orientado a colunas, projetado para gerenciar grandes quantidades de dados distribuídos em múltiplos servidores, proporcionando alta disponibilidade sem ponto único de falha.

Posicionamento CAP

Prioriza Disponibilidade e Tolerância a Partições, oferecendo consistência eventual. Com ajustes de configuração, pode ser otimizado para diferentes equilíbrios entre consistência e disponibilidade.

Características do Cassandra

Descentralizado

Arquitetura peer-to-peer sem nó mestre

Alta Disponibilidade

Sem pontos únicos de falha

Distribuído

Dados replicados em múltiplos nós

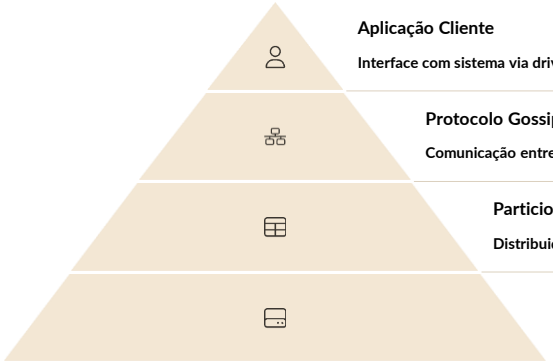
Escalabilidade Linear

Performance proporcional ao número de nós

Consistência Ajustável

Níveis configuráveis por operação

Arquitetura do Cassandra



Aplicação Cliente

Interface com sistema via drivers

Protocolo Gossip

Comunicação entre nós do cluster

Particionador

Distribuição de dados pelos nós

Sistema de Armazenamento

Persistência e recuperação de dados

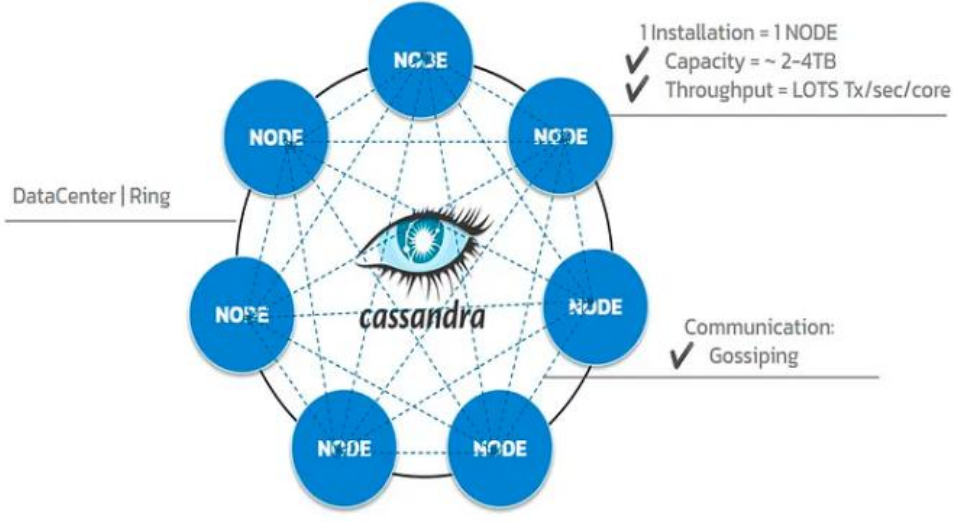
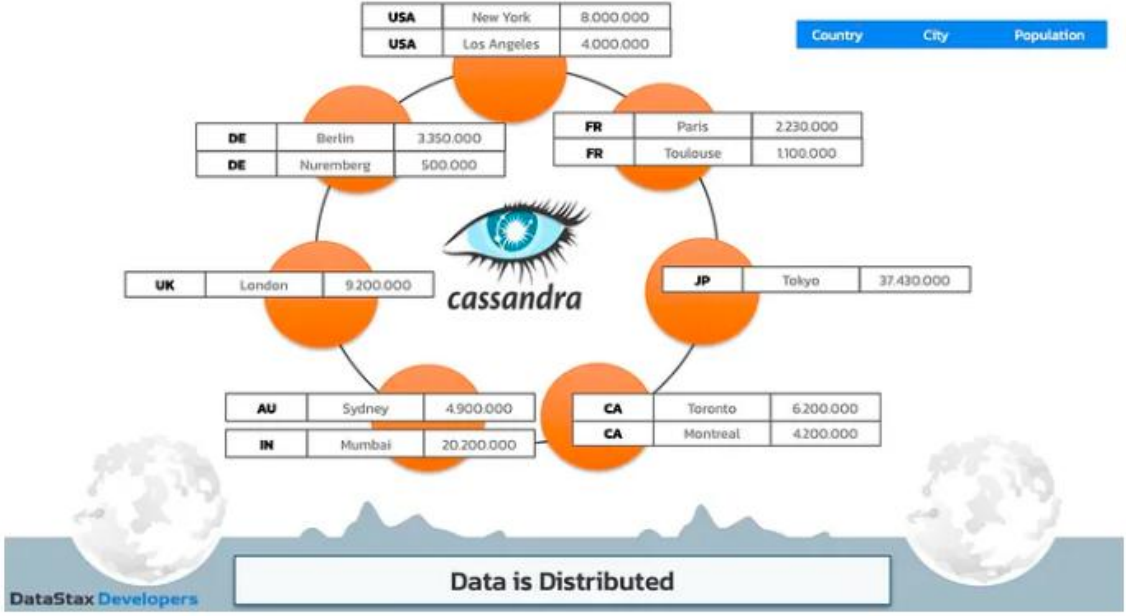


Figura 2. Estrutura do Apache Cassandra.

Como o dado é distribuído?



Modelo de Dados do Cassandra

Keyspace

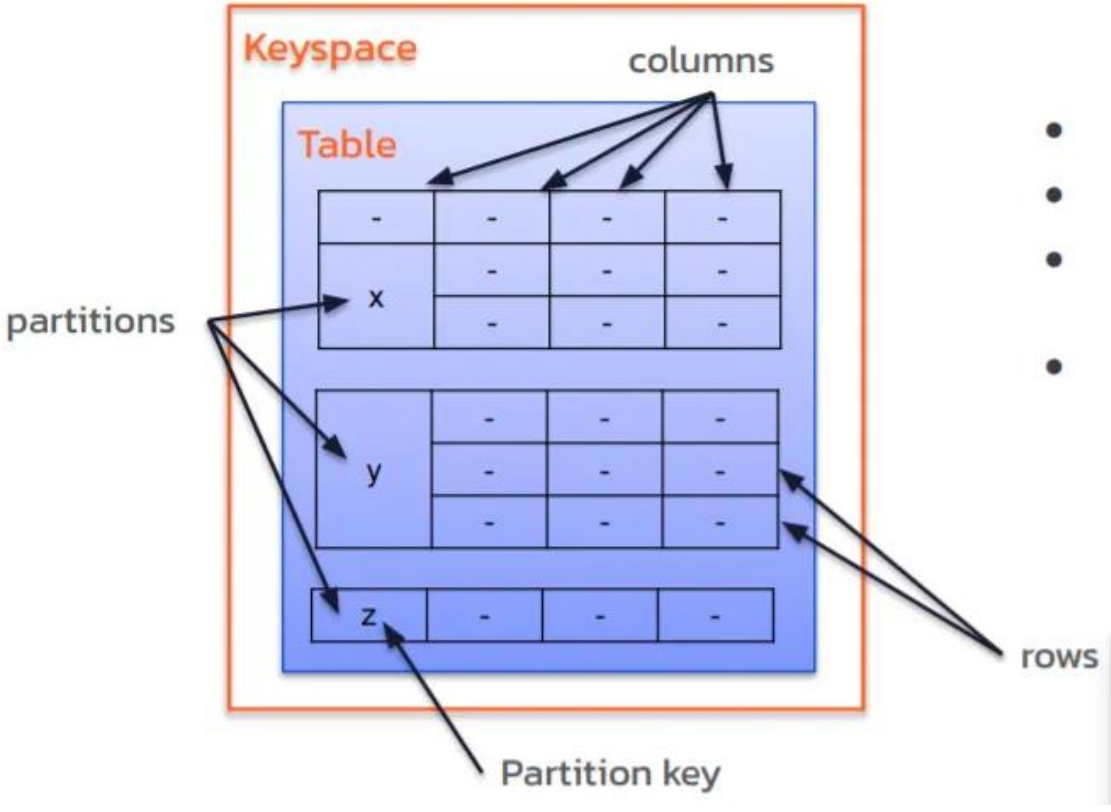
Equivalente a um banco de dados em RDBMS. Define como os dados são replicados no cluster, incluindo a estratégia de replicação e o fator de replicação. Um cluster Cassandra pode conter múltiplos keyspaces.

Columns

Similar a uma tabela em bancos relacionais. Contém colunas e linhas de dados, porém com esquema flexível que pode variar entre as linhas. É otimizada para armazenar e recuperar dados por chave de linha.

Coluna

A unidade básica de armazenamento, consistindo em um nome, um valor e um timestamp. As colunas podem ser adicionadas dinamicamente a qualquer linha sem afetar outras linhas, permitindo esquemas heterogêneos.


Keyspace
columns
Table
partitions
x
y
z
rows
Partition key

Operações de Leitura e Escrita

Operação de Escrita

1. **Cliente envia operação para qualquer nó**
2. **Nó coordenador identifica réplicas responsáveis**
3. **Dados são gravados no commit log para durabilidade**
4. **Dados são adicionados à memtable em memória**
5. **Confirmação é enviada ao cliente (baseada no nível de consistência)**
6. **Periodicamente, memtables são persistidas em SSTable**

Operação de Leitura

1. **Cliente solicita leitura a qualquer nó**
2. **Nó coordenador identifica réplicas relevantes**
3. **Solicita dados às réplicas conforme nível de consistência**
4. **Verifica dados em memtable e SSTables**
5. **Realiza reconciliação se necessário (compara timestamps)**
6. **Retorna resultado consolidado ao cliente**
7. **Executa read repair em segundo plano se detectar inconsistências**

O que são Dados Abertos?

Disponibilidade e Acesso

Dados disponíveis integralmente, preferencialmente via download pela internet, em formato conveniente e modificável

Reutilização e Redistribuição

Dados fornecidos sob termos que permitam reutilização e redistribuição, inclusive com combinação com outros conjuntos de dados

Participação Universal

Ausência de restrições tecnológicas, comerciais ou discriminatórias quanto ao uso dos dados por pessoas ou grupos

Classificação de Dados Abertos (5 Estrelas)

Dados Abertos no Brasil

2011

Lei de Acesso à Informação (LAI) - Lei nº 12.527, estabelecendo mecanismos para garantir acesso a informações públicas

2012

Lançamento do Portal Brasileiro de Dados Abertos (dados.gov.br) como ponto central de acesso aos dados governamentais

2012

Brasil torna-se membro fundador da Open Government Partnership (OGP), iniciativa internacional para promover transparência

2016

Decreto nº 8.777 - Institui a Política de Dados Abertos do Poder Executivo Federal

Princípios dos Dados Governamentais Abertos

Três Leis

1. **Se não pode ser encontrado e indexado na Web, não existe**
2. **Se não estiver aberto e disponível em formato compreensível por máquina, não pode ser reaproveitado**
3. **Se não for de livre uso, reaproveitamento e distribuição, não é útil**

Oito Princípios

1. **Completos**
2. **Primários**
3. **Atuais**
4. **Acessíveis**
5. **Processáveis por máquina**
6. **Acesso não discriminatório**
7. **Formatos não proprietários**
8. **Livres de licenças**

Estudo de Caso: Programa Bolsa Família

O Programa

Programa de transferência direta de renda do governo federal brasileiro, destinado a famílias em situação de pobreza e extrema pobreza. Atende milhões de famílias em todo o território nacional.

Volume de Dados

Informações sobre mais de 14 milhões de famílias beneficiárias, incluindo dados cadastrais, valores de benefícios e histórico de pagamentos, representando um volume significativo de dados.

Características dos Dados

Dados estruturados com atualizações frequentes e necessidade de consultas rápidas para transparência pública e gestão do programa, ideais para testar a escalabilidade do Cassandra.

Modelagem dos Dados no Cassandra

Keyspace	**bolsa_familia**
Família de Colunas	**pagamentos**
Chave Primária	**(nis_favorecido, mes_referencia)**
Colunas	**nome_favorecido, valor_parcela, uf, codigo_municipio, nome_municipio**
Índices Secundários	**uf, codigo_municipio**
Estratégia de Replicação	**SimpleStrategy com fator de replicação 3**

Arquitetura do Ambiente de Testes

Configuração do Cluster

- **Testes com 1, 2 e 3 nós**
- **Máquinas virtuais com recursos equivalentes**
- **8GB de RAM por nó**
- **4 vCPUs por nó**
- **100GB de armazenamento SSD**
- **Rede de 1Gbps entre os nós**

Software

- **Apache Cassandra 3.11.4**
- **Ubuntu Server 18.04 LTS**
- **OpenJDK 8**
- **Driver Java DataStax 3.7.1**
- **Aplicação de teste desenvolvida em Java**
- **JMeter para simulação de carga**

O ambiente de testes foi projetado para avaliar o desempenho do Cassandra com diferentes tamanhos de cluster, permitindo analisar o impacto da escalabilidade horizontal na performance de operações de leitura e escrita. Todas as máquinas virtuais foram configuradas com recursos idênticos para garantir comparações justas.

A configuração do Cassandra foi mantida com os valores padrão em todos os nós, com exceção dos parâmetros de memória, que foram ajustados para otimizar o uso dos recursos disponíveis. O fator de replicação foi mantido em 3 em todos os testes, garantindo que cada dado fosse armazenado em todos os nós quando o cluster estava completo.

Desenvolvimento da Aplicação

Importação de Dados

Módulo para processamento de arquivos CSV do Portal da Transparência e inserção eficiente no Cassandra utilizando operações em lote (batch) para otimizar a performance.

Consultas

Implementação de diferentes tipos de consultas para avaliar a performance em cenários variados, incluindo buscas por beneficiário, região e período.

Benchmarking

Componente para execução automática de testes de performance, medindo latência e throughput em diferentes configurações de cluster.

Processo de Carga de Dados

Extração

Download e processamento dos arquivos CSV do Portal da Transparência, contendo dados de pagamentos do Bolsa Família

Transformação

Limpeza, normalização e estruturação dos dados conforme o modelo do Cassandra

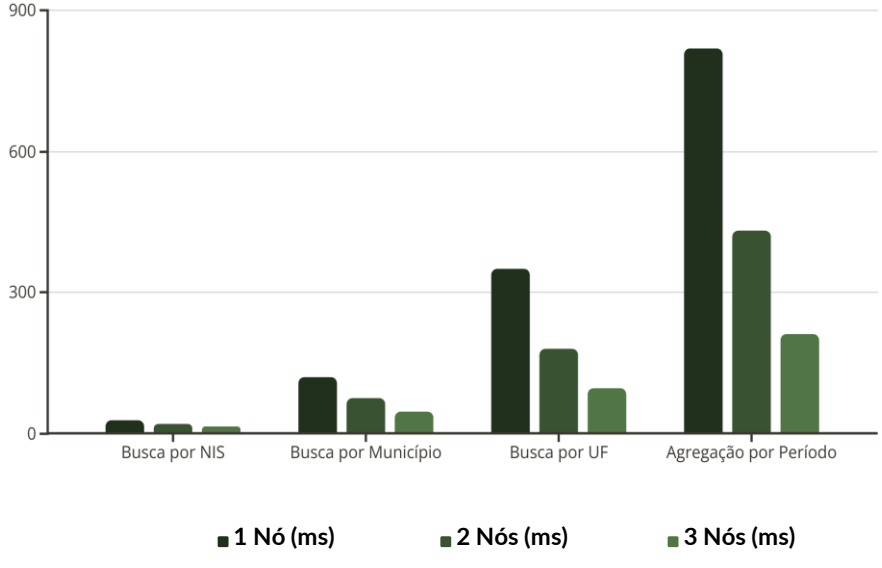
Carga

Inserção dos dados no cluster Cassandra utilizando operações em lote otimizadas

Validação

Verificação da integridade e completude dos dados carregados

Consultas e Resultados



Comparação dos Ambientes

98%

Disponibilidade

Tempo de atividade do cluster com 3 nós durante teste de resiliência com falhas simuladas

2.1x

Ganho de Performance

Aumento médio de velocidade nas consultas ao escalar de 1 para 3 nós

8.5K

Throughput

Operações por segundo no cluster de 3 nós sob carga intensa

22ms

Latência Média

Tempo de resposta para consultas pontuais no cluster otimizado

Conclusões e Recomendações

Benefícios Comprovados

O Apache Cassandra demonstrou ser uma solução altamente eficaz para o gerenciamento de grandes volumes de dados abertos governamentais, oferecendo escalabilidade horizontal, alta disponibilidade e desempenho consistente.

A melhoria de performance próxima do linear com a adição de novos nós confirma o potencial da arquitetura para crescer conforme a demanda, sem redesenho significativo da aplicação.

Considerações de Implementação

A modelagem de dados orientada a consultas é fundamental para obter o máximo desempenho do Cassandra. É essencial identificar os padrões de acesso antes de definir a estrutura dos dados.

Para ambientes de produção, recomenda-se distribuição geográfica em múltiplos data centers para maximizar a resiliência e disponibilidade, especialmente para dados críticos de programas sociais.

Trabalhos Futuros

Explorar a integração com tecnologias de processamento analítico como Apache Spark para possibilitar análises complexas sobre os dados armazenados no Cassandra.

Desenvolver APIs públicas baseadas nesta arquitetura para facilitar o acesso aos dados abertos por desenvolvedores e pesquisadores, promovendo a criação de aplicações inovadoras com dados governamentais.

Bibliografia da disciplina!

Bibliografia básica: BOAGLIO, Fernando. MongoDB: Construa novas aplicações com novas tecnologias. São Paulo: Casa do Código, 2015. ELMASRI, R.; NAVATHE, S. B. Sistemas de Banco de Dados: Fundamentos e Aplicações. 7ed. São Paulo: Pearson, 2019. SADALAGE, P.; FOWLER, M. Nosql Essencial: Um Guia Conciso Para o Mundo Emergente da Persistência Poliglota. São Paulo: Novatec, 2013. SINGH, Harry. Data Warehouse: conceitos, tecnologias, implementação e gerenciamento. São Paulo: Makron Books, 2001.

Bibliografia Complementar: FAROULT, Stephane. Refatorando Aplicativos SQL. Rio de Janeiro: Alta Books, 2009. PANIZ, D. NoSQL: Como armazenar os dados de uma aplicação moderna. Casa do Código, 2016. SOUZA, M. Desvendando o Mongodb. Rio de Janeiro: Ciência Moderna, 2015.

Armazenamento de dados abertos no SQL: Acesso em 19 de Maio de 2025,

Disponível em: https://bdm.unb.br/bitstream/10483/19381/1/2017_JorgeLuizAndrade.pdf