Estratégias de modelagem no MongoDB

Termos de Uso

Propriedade Growdev

Todo o conteúdo deste documento é propriedade da Growdev. O mesmo pode ser utilizado livremente para estudo pessoal.

É proibida qualquer utilização desse material que não se enquadre nas condições acima sem o prévio consentimento formal, por escrito, da Growdev. O uso indevido está sujeito às medidas legais cabíveis.

Introdução

MongoDB é um banco de dados NoSQL baseado em documentos que oferece flexibilidade para modelar dados de maneira diferente dos bancos relacionais tradicionais.

Ao contrário do modelo rígido de tabelas, linhas e colunas de um banco SQL, o MongoDB utiliza coleções de documentos JSON-like, permitindo uma estrutura dinâmica e adaptável.

Modelagem SQL vs MongoDB

Modelagem em bancos relacionais

Estrutura Rígida: Requer esquemas definidos com antecedência, incluindo tipos de dados e relacionamentos.

Normalização: Dados são frequentemente separados em várias tabelas para evitar redundâncias, o que resulta em consultas baseadas em joins.

Relacional: Relacionamentos são explícitos e definidos por chaves primárias e estrangeiras.

Modelagem no MongoDB

Flexibilidade: Não requer um esquema fixo, permitindo que documentos de uma mesma coleção tenham campos diferentes.

Desnormalização: Dados podem ser embutidos em documentos para otimizar a leitura e reduzir operações complexas de join.

Baseada em Documentos: Relacionamentos podem ser embutidos ou referenciados, dependendo do caso de uso.

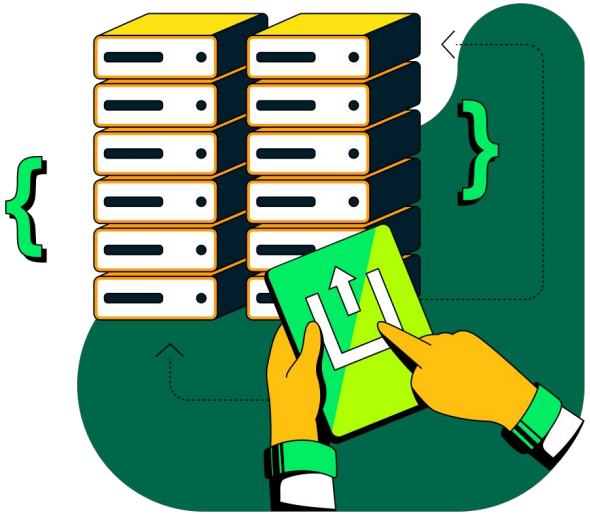
Estratégias

Relacionamentos

A modelagem de relacionamentos no MongoDB se difere bastante da modelagem em bancos relacionais, pois tem como objetivo organizar os dados no ponto de vista das consultas.

Dessa forma, documentos não precisam ser normalizados e podem – muitas vezes devem – estar juntos.

Essa característica se aplica principalmente se as entidades co-existem em um relacionamento forte, onde uma existe por causa da outra.

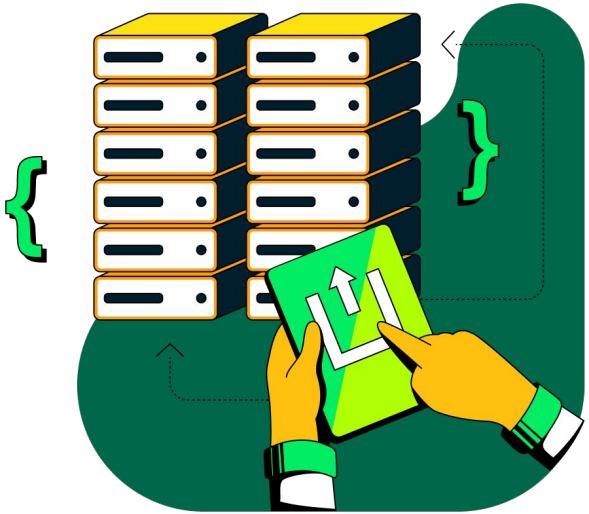


Incorporação

A estratégia de incorporação (Embebbed Documents) ocorre quando as duas entidades estão intimamente relacionadas e são frequentemente acessadas juntas.

Nessa estratégia, os dois elementos do relacionamento estão juntos dentro do mesmo documento.

Dessa forma, para acessar ambos os recursos não é necessário juntar dois documentos diferentes. No SQL isso seria feito através de um JOIN.

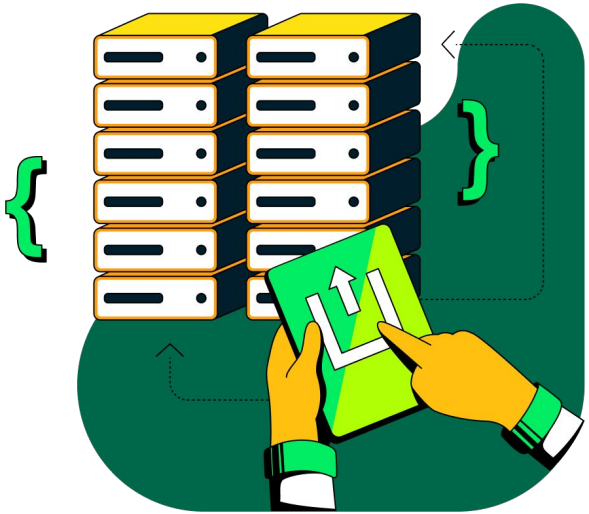


Referenciamento

A estratégia de referenciamento ocorre os dados precisam ser acessados separadamente.

Nessa estratégia, os dois elementos do relacionamento estarão em documentos específicos ligados por uma referência.

Essa estratégia se assemelha a técnica de normalização aplicada pelos bancos de dados relacionais.

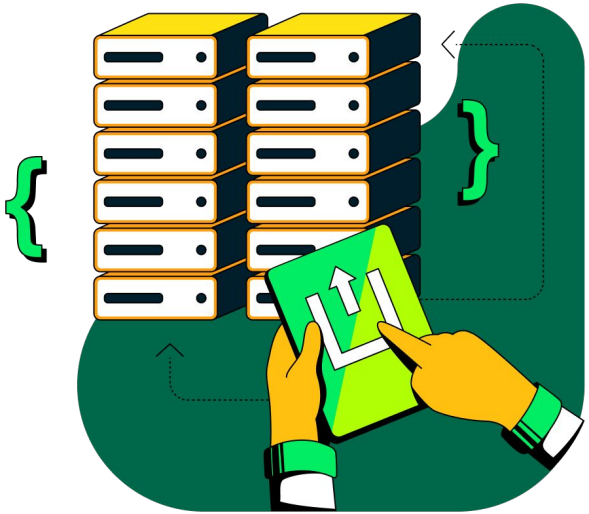


Bi-direcional

O MongoDB possui variações bi-direcionais para a incorporação e para o referenciamento, usadas quando o relacionamento precisa ser consultado de ambos os lados.

O grande desafio dessa estratégia é manter a sincronia entre os documentos associados. Se um lado muda, o outro deve refletir essa mudança.

Geralmente usado em relacionamentos N-N onde o acesso aos dados é frequente nos dois lados do relacionamento.



Tipos de relacionamento

Relacionamento 1-1

O relacionamento 1-1 ocorre quando uma entidade está associada a exatamente outra entidade.

Incorporação

```
{
    "_id": ObjectId(1),
    "nome": "Daphne",
    "email": "daphne@dog.com",
    "endereco": {
        "_id": ObjectId(2),
        "rua": "Rua dos Dogs",
        "numero": 20,
        "cidade": "Porto Alegre"
    }
}
```

Referenciamento

```
{
    "_id": ObjectId(1),
    "nome": "Daphne",
    "email": "daphne@dog.com",
    "endereco": ObjectId(2)

},
{
    "_id": ObjectId(2),
    "rua": "Rua dos Dogs",
    "numero": 20,
    "cidade": "Porto Alegre"
}
```

Relacionamento 1-N

O relacionamento 1-N ocorre quando uma entidade pode estar relacionada a várias outras entidades.

Incorporação

```
{
    "_id": ObjectId(1),
    "nome": "Daphne",
    "email": "daphne@dog.com",
    "enderecos": [
        {
            "_id": ObjectId(2),
            "rua": "Rua dos Dogs",
            "numero": 20,
            "cidade": "Porto Alegre"
        },
        {
            "_id": ObjectId(3),
            "rua": "Rua dos Pets",
            "numero": 99,
            "cidade": "Canoas"
        },
    ]
}
```

Referenciamento

```
{
    "_id": ObjectId(1),
    "nome": "Daphne",
    "email": "daphne@dog.com",
    "enderecos": [
        ObjectId(2),
        ObjectId(2)
    ]
},
{
    "_id": ObjectId(2),
    "rua": "Rua dos Dogs",
    "numero": 20,
    "cidade": "Porto Alegre"
},
{
    "_id": ObjectId(3),
    "rua": "Rua dos Pets",
    "numero": 99,
    "cidade": "Canoas"
},
```

Relacionamento N-N

O relacionamento N-N ocorre quando várias entidades de um tipo podem estar relacionadas a várias entidades de outro tipo.

Nesse caso, a **incorporação bi-direcional** é útil se os **dados são simples.**

Incorporação

```
{
    "_id": ObjectId(1),
    "nome": "Daphne",
    "email": "daphne@dog.com",
    "skills": ["NoSQL", "MongoDB"]
},
{
    "_id": ObjectId(2),
    "nome": "Toby",
    "email": "toby@pet.com",
    "skills": ["MongoDB", "Neo4j"]
}
```

Relacionamento N-N

O relacionamento N-N ocorre quando várias entidades de um tipo podem estar relacionadas a várias entidades de outro tipo.

Nesse caso, o **referenciamento bi-direcional** é útil se os **dados são mais complexos.**

Referenciamento (lado 1)

```
{
    "_id": ObjectId(1),
    "nome": "Daphne",
    "email": "daphne@dog.com",
    "skillIds": [
        ObjectId(4),
    ]
},
{
    "_id": ObjectId(2),
    "nome": "Toby",
    "email": "toby@pet.com",
    "skillIds": [
        ObjectId(3),
        ObjectId(4),
    ]
},
```

Referenciamento (lado 2)

```
{
    "_id": ObjectId(3),
    "valor": "NoSQL",
    "userIds": [
        ObjectId(1)
    ]
},
{
    "_id": ObjectId(4),
    "valor": "MongoDB",
    "userIds": [
        ObjectId(1),
        ObjectId(2)
    ]
}
```

Fixando ...

A modelagem no MongoDB é altamente flexível e permite escolher a abordagem ideal dependendo do caso de uso e das demandas de leitura/escrita.

Incorporação é excelente para otimizar leituras, enquanto referenciamento é preferido para dados independentes e altamente relacionais.

A escolha correta entre as estratégias pode melhorar o desempenho e simplificar o gerenciamento dos dados no MongoDB.

Parabéns!
Nos vemos na próxima etapa!