Banco de Dados
Não Relacional

mongoDB

Desenvolvimento de
Sistemas Multiplataformas

Updates - Atualizações

Além do insertOne e insertMany utilizados para inserir dados, podemos atualizar documentos já existentes, por exemplo usando o comando replaceOne e derivados. O jeito mais simples de atualizar um documento é chamando a função replaceOne na coleção. Esta função possui dois parâmetros obrigatórios e um opcional:

- filtro para saber qual(is) documento(s) será(ão) atualizado(s);
- novo documento que substituirá o antigo;
- opções de atualização;

Como em:

Busca do documento a ser alterado

```
db.clientes.replaceOne({nome:'Luiz'}, {nome:'Luiz Antonio', idade:38, Estado:'RS'})
```

Alterações a serem realizadas

Updates - Atualizações

Neste comando estou dizendo para substituir (atualizar completamente) somente o primeiro documento cujo nome seja (literalmente) Luiz pelo objeto presente no segundo parâmetro.

Como resultado você deve ter um **modifiedCount** igual a 1, mostrando quantos documentos foram atualizados.

```
acknowledged: true,
insertedId: null,
matchedCount: 1,
modifiedCount: 1,
upsertedCount: 0
```

E aqui vão alguns cuidados que você deve ter... Primeiro, esta função de update substitui completamente o documento filtrado com o JSON passado como segundo argumento, ou seja, ela exige que você passe o documento completo a ser atualizado no segundo parâmetro, pois ele substituirá o original!

Então vamos a algumas dicas para um "**update inteligente**".

Updates - Atualizações

- **Primeira regra do update inteligente**:
  - Se você quer atualizar um documento apenas, comece usando a função **updateOne** ao invés de **replaceOne.**
  - O **updateOne** vai te obrigar a usar operadores de atualização ao invés de um documento inteiro para a atualização, o que é muito mais seguro.

- **Segunda regra do update inteligente**:
  - Sempre que possível, use a chave primária (_id) como filtro da atualização, pois ela é sempre única dentro da coleção.

- **Terceira regra do update inteligente**: evite usar **updateMany** sempre que possível.

Updates Operators – Operadores de Atualizações

Assim como o **find** possui operadores de filtro, o **updateOne**, que é a função mais recomendada de update, possui operadores de atualização.

Se eu quero, por exemplo, mudar apenas o nome de um **cliente**, eu não preciso enviar todo o documento do respectivo **cliente** com o nome alterado, mas sim apenas a expressão de alteração do nome, como abaixo (já usando o _id como filtro, que é mais seguro):

```
db.clientes.updateOne({ _id:ObjectId('65ef360a8d3a816e920cc02a')}, { $set: { idade: 28 } })
```

Antes

```
_id: ObjectId('65ef360a8d3a816e920cc02a'),
nome: 'Luiz Antonio Santos',
idade: 38,
Cidade: 'Porto Alegre',
Estado: 'RS'
```

```
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Depois

```
_id: ObjectId('65ef360a8d3a816e920cc02a'),
nome: 'Luiz Antonio Santos',
idade: 28,
Cidade: 'Porto Alegre',
Estado: 'RS'
```

Updates Operators – Operadores de Atualizações

Nota: para saber o _id correto do seu update, faça um **find** primeiro e não tente copiar o meu exemplo de _id pois não vai repetir.

Esta função vai alterar (operador $set) a idade para o valor 28 do documento cujo _id seja "59ab46e433959e2724be2cbd" (note que usei uma função ObjectId para converter, pois esse valor não é uma string).

Nota: você pode usar null se quiser "limpar" um campo. O operador $set recebe um documento contendo todos os campos que devem ser alterados e seus respectivos novos valores. Qualquer campo do documento original que não seja indicado no set continuará com os valores originais.

Updates Operators – Operadores de Atualizações

Não importa o valor que ela tenha antes, o operador **$set** vai sobrescrevê-lo.

Agora, se o valor anterior importa, como quando queremos incrementar o valor de um campo, não se usa o operador **$set**, mas sim outros operadores.

A lista dos operadores de update mais comuns estão abaixo:

- **$unset**: remove o respectivo campo do documento;
- **$inc**: incrementa o valor original do campo com o valor especificado;
- **$mul**: multiplica o valor original do campo com o valor especificado;
- **$rename**: muda o nome do campo para o nome especificado;

Updates Operators – Operadores de Atualizações

Atenção: os operadores **$rename**, **$unset** e potencialmente o operador **$set**, podem ser usados para alterar a estrutura original de um documento (renomeando, removendo ou adicionando campos, respectivamente).

No entanto, como não há um schema compartilhando entre todos os documentos de uma coleção, eles alteram apenas o(s) documento(s) englobados no filtro de update utilizado.

```
db.clientes.updateOne({nome:'Luiz Santos'},{$set:{nome:'Luiz Santos',Estado:'SP'}},{upsert:true})
```

Um **upsert** é um update como qualquer outro, ou seja, vai atualizar o documento que atender ao filtro passado como primeiro parâmetro, porém, se não existir nenhum documento com o respectivo filtro, ele será inserido, como se fosse um **insert**. **upsert = "update ou insert".**

```
_id: ObjectId('65ef43876d0a6ae98f84c3e6'),
nome: 'Luiz Santos',
Estado: 'SP'
```

Delete

Existe uma função **deleteOne** e uma **deleteMany**, o que a essa altura do campeonato você já deve imaginar a diferença. Também existe uma função remove, mas não aconselho utilizá-la.

Além disso, assim como o **find e o update**, o primeiro parâmetro dos deletes é o **filtro** que vai definir quais documentos serão deletados e todos os filtros normais do **find** são aplicáveis.

Sendo assim, de maneira bem direta:

```
db.clientes.deleteMany({nome:'Luiz Antonio Santos'})
```

Vai excluir todos os clientes cujo nome seja igual a "Luiz“.

Delete

```
db.clientes.deleteOne({_id:ObjectId('65e51285334e07bb97397fb7')})
```

Note que para que o **deleteOne** seja mais preciso, recomenda-se que o filtro utilizado seja um índice único, como o _id, por exemplo.

Outras Funções CRUD

Existem outras funções de CRUD além das já citadas nas seções anteriores.

A maioria delas é apenas **syntax-sugar**, ou seja, apenas uma forma diferente de fazer a mesma coisa que outras funções já fazem, geralmente englobando um ou mais comportamentos com uma lógica em cima do documento passado por parâmetro.

De qualquer forma, é interessante conhecê-las, mesmo que de maneira superficial como você verá a seguir:

- **save**: age como se fosse um insertOne, mas se o documento passado por parâmetro possuir um campo _id, age como se fosse um updateOne sobre o documento original com aquele _id.

- **findAndModify**: busca os documentos com o filtro especificado e faz um update deles, retornando a versão original deles ao término da operação. Opcionalmente você pode passar um parâmetro para retornar a versão nova dos documentos;

Outras Funções CRUD

- **findOneAndDelete**: busca um documento com o filtro especificado e faz um delete nele, retornando a versão original ao término da operação;

- **findOneAndUpdate**: assim como o findAndModify, mas sobre apenas um documento;

Obrigado!!!!