



IBD-016 – BANCO DE DADOS - NÃO RELACIONAL

Prof. Me. Anderson Vanin



O que são relacionamentos?

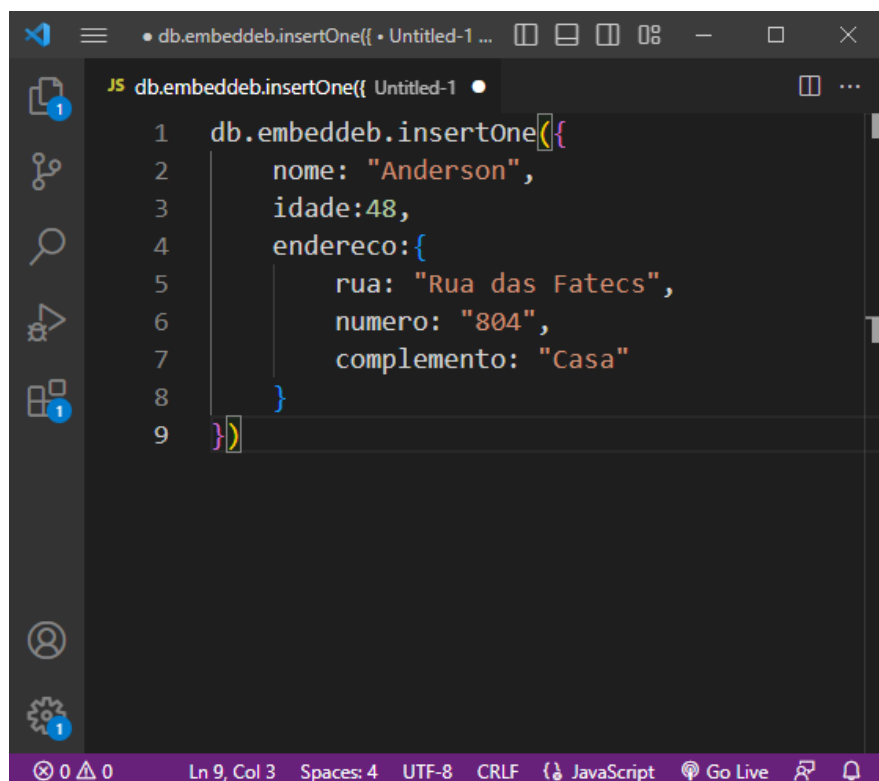
- São **registros que possuem ligações entre si**;
- Tipos de relação: **one to one, one to many e many to many**;
- Onde cada uma possui um método diferente de ser aplicado em MongoDB;
- Além de uma forma especial, graças a flexibilidade dos documents, que é a **embedded document**;
- Vamos ver cada uma delas!

Embedded Documents

- **Embedded documents** é uma forma simples de fazer relacionamento entre documents;
- A ideia é inserir um document dentro do registro principal;
- Este recurso **funciona bem para One to One e One to Many**, porém não para Many to Many;
- Vamos **ver** na prática!

One to One - Embedded

```
> use relacionamentos
switched to db relacionamentos
```



```
1 db.embeddeb.insertOne({
2   nome: "Anderson",
3   idade: 48,
4   endereco: {
5     rua: "Rua das Fatecs",
6     numero: "804",
7     complemento: "Casa"
8   }
9 })
```

```
> db.embeddeb.insertOne({
...   nome: "Anderson",
...   idade: 48,
...   endereco: {
...     rua: "Rua das Fatecs",
...     numero: "804",
...     complemento: "Casa"
...   }
... })
{
  "acknowledged" : true,
  "insertedId" : ObjectId("643da02dcc309ebaa62f3f6c")
}
```

```
> db.embeddeb.findOne()
{
  "_id" : ObjectId("643da02dcc309ebaa62f3f6c"),
  "nome" : "Anderson",
  "idade" : 48,
  "endereco" : {
    "rua" : "Rua das Fatecs",
    "numero" : "804",
    "complemento" : "Casa"
  }
}
```

One to One - Embedded

```
> const anderson = db.embedded.findOne()  
> anderson.endereco  
{ "rua" : "Rua das Fatecs", "numero" : "804", "complemento" : "Casa" }  
> anderson.endereco.rua  
Rua das Fatecs  
>
```


One to Many - Embedded

```
db.embeddeb.insertOne({
  nome: "João",
  idade: 40,
  enderecos: {
    casa: {
      rua: "Rua das Fatecs",
      numero: "804",
      complemento: "Casa"
    },
    trabalho: {
      rua: "Rua das Arvores",
      numero: "1000",
      complemento: "Galpão"
    }
  }
})
```

```
> db.embeddeb.find()
{ "_id" : ObjectId("643da02dcc309ebaa62f3f6c"), "nome" : "Anderson", "idade" : 48, "endereco" : { "rua" : "Rua das Fatecs", "numero" : "804", "complemento" : "Casa" } }
{ "_id" : ObjectId("643da209d75eb89bb4e1eefb"), "nome" : "João", "idade" : 40, "enderecos" : { "casa" : { "rua" : "Rua das Fatecs", "numero" : "804", "complemento" : "Casa" }, "trabalho" : { "rua" : "Rua das Arvores", "numero" : "1000", "complemento" : "Galpão" } } }
>
```

One to Many - Embedded

```
> const joao = db.embedded.findOne({ nome: "João" })
> joao
{
  "_id" : ObjectId("643da209d75eb89bb4e1eefb"),
  "nome" : "João",
  "idade" : 40,
  "enderecos" : {
    "casa" : {
      "rua" : "Rua das Fatecs",
      "numero" : "804",
      "complemento" : "Casa"
    },
    "trabalho" : {
      "rua" : "Rua das Arvores",
      "numero" : "1000",
      "complemento" : "Galpão"
    }
  }
}
```

```
> joao.enderecos.casa
{ "rua" : "Rua das Fatecs", "numero" : "804", "complemento" : "Casa" }
> joao.enderecos.trabalho
{ "rua" : "Rua das Arvores", "numero" : "1000", "complemento" : "Galpão" }
>
```

One to One

- A relação One to One é quando **um registro possui uma ligação única com outro**, e o inverso também é verdadeiro;
- **Exemplo:** Nosso sistema permite o cadastro de um único endereço por usuário, então podemos dizer que o endereço é único para cada usuário;
- E agora vamos trabalhar com **duas collections**;
- Precisamos inserir uma informação que faça referência ao registro, como o **id**, vamos ver na prática!

One to One

```
> db.pessoas.insertOne({
...   nome: "Anderson",
...   idade: 48,
...   profissao: "Professor"
... })
{
  "acknowledged" : true,
  "insertedId" : ObjectId("643da3fad75eb89bb4e1eefc")
}
>
```

```
> const p = db.pessoas.findOne()
> p.nome
Anderson
> p._id
ObjectId("643da3fad75eb89bb4e1eefc")
>
```

```
> db.enderecos.insertOne({
...   rua: "Rua das Flores",
...   numero: "1414",
...   complemento: "Casa",
...   pessoa_id: p._id
... })
{
  "acknowledged" : true,
  "insertedId" : ObjectId("643da4f3d75eb89bb4e1eefd")
}
```

One to One

```
> db.pessoas.find()
{ "_id" : ObjectId("643da3fad75eb89bb4e1eefc"), "nome" : "Anderson", "idade" : 48, "profissao" : "Professor" }
> db.enderecos.find()
{ "_id" : ObjectId("643da4f3d75eb89bb4e1eefd"), "rua" : "Rua das Flores", "numero" : "1414", "complemento" : "Casa", "pessoa_id" : ObjectId("643da3fad75eb89bb4e1eefc") }
> db.enderecos.find().pretty()
{
  "_id" : ObjectId("643da4f3d75eb89bb4e1eefd"),
  "rua" : "Rua das Flores",
  "numero" : "1414",
  "complemento" : "Casa",
  "pessoa_id" : ObjectId("643da3fad75eb89bb4e1eefc")
}
```

```
> db.enderecos.find({pessoa_id: p._id})
{ "_id" : ObjectId("643da4f3d75eb89bb4e1eefd"), "rua" : "Rua das Flores", "numero" : "1414", "complemento" : "Casa", "pessoa_id" : ObjectId("643da3fad75eb89bb4e1eefc") }
>
```

One to Many

- A relação **One to Many** é quando um **registro pode possuir mais vínculos com uma outra collection**, porém o inverso é falso;
- **Exemplo:** Um usuário pode fazer várias compras, mas uma compra pertence a apenas um usuário;
- Desta maneira a collection de compras contém em cada compra uma referência ao usuário;
- Que será o **_id**;