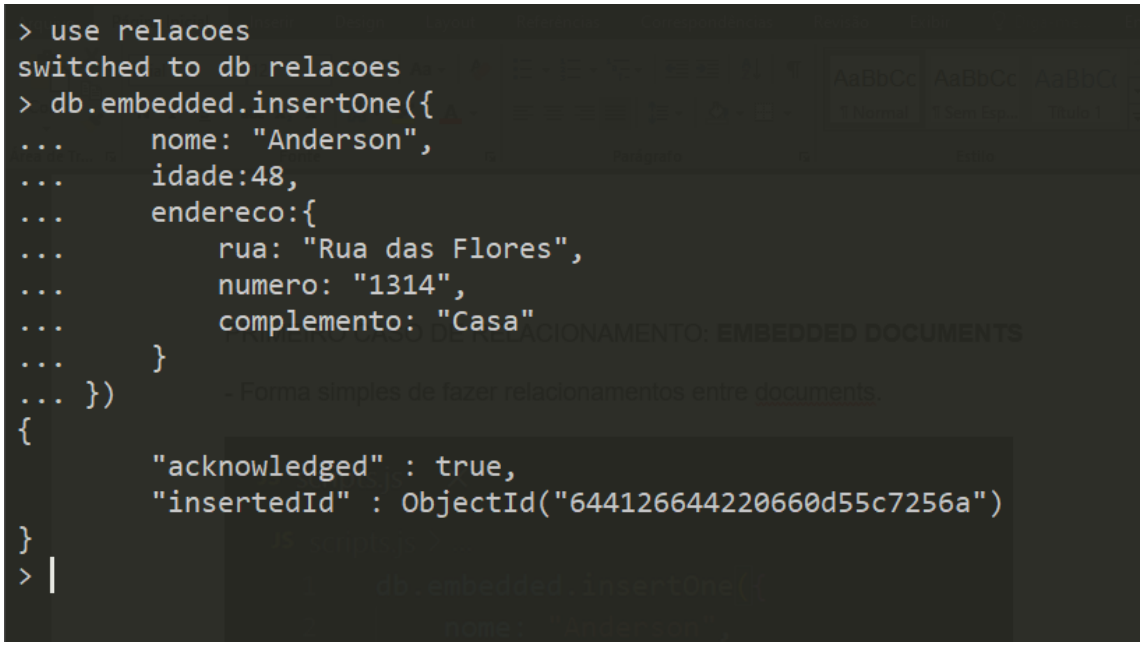
AULA 05 - RELACIONAMENTOS

EMBEDDED DOCUMENTS

- Forma simples de fazer relacionamentos entre documents.



Podemos transformar a consulta em uma variável específica:

```
> const anderson = db.embedded.findOne()
```

```
> anderson.endereco
{ "rua" : "Rua das Flores", "numero" : "1314", "complemento" : "Casa" }
```

```
> anderson.endereco.rua
Rua das Flores
> anderson.endereco.numero
1314
```

Agora vamos trabalhar com múltiplos endereços

```
db.embedded.insertOne({
    nome:"João",
    idade:35,
    enderecos:{
        casa:{
            rua: "Ruas das Flores",
            numero: "1234",
            complemento: "Casa"
        },
        trabalho:{
            rua: "Rua das Árvores",
            numero: "123 C",
            complemento: "Galpão"
        }
    }
})
```

```
> db.embedded.insertOne({
...     nome:"João",
...     idade:35,
...     enderecos:{
...         casa:{
...             rua: "Ruas das Flores",
...             numero: "1234",
...             complemento: "Casa"
...         },
...         trabalho:{
...             rua: "Rua das Árvores",
...             numero: "123 C",
...             complemento: "Galpão"
...         }
...     }
... })
{
        "acknowledged" : true,
        "insertedId" : ObjectId("644128584220660d55c7256b")
}
```

```
> db.embedded.find()
{ "_id" : ObjectId("644126644220660d55c7256a"), "nome" : "Anderson", "idade" :
 48, "endereco" : { "rua" : "Rua das Flores", "numero" : "1314", "complemento"
 : "Casa" } }
{ "_id" : ObjectId("644128584220660d55c7256b"), "nome" : "João", "idade" : 35,
 "enderecos" : { "casa" : { "rua" : "Ruas das Flores", "numero" : "1234", "com
plemento" : "Casa" }, "trabalho" : { "rua" : "Rua das Árvores", "numero" : "12
3 C", "complemento" : "Galpão" } } }
```

```
> db.embedded.find().pretty()
{
        "_id" : ObjectId("644126644220660d55c7256a"),
        "nome" : "Anderson",
        "idade" : 48,
        "endereco" : {
                "rua" : "Rua das Flores",
                "numero" : "1314",
                "complemento" : "Casa"
        }
}
{
        "_id" : ObjectId("644128584220660d55c7256b"),
        "nome" : "João",
        "idade" : 35,
        "enderecos" : {
                "casa" : {
                        "rua" : "Ruas das Flores",
                        "numero" : "1234",
                        "complemento" : "Casa"
                },
                "trabalho" : {
                        "rua" : "Rua das Árvores",
                        "numero" : "123 C",
                        "complemento" : "Galpão"
                }
        }
}
```

Criando uma variável especifica para o João

```
> const joao = db.embedded.findOne({nome:"João"})
> joao
{
        "_id" : ObjectId("644128584220660d55c7256b"),
        "nome" : "João",
        "idade" : 35,
        "enderecos" : {
                "casa" : {
                        "rua" : "Ruas das Flores",
                        "numero" : "1234",
                        "complemento" : "Casa"
                },
                "trabalho" : {
                        "rua" : "Rua das Árvores",
                        "numero" : "123 C",
                        "complemento" : "Galpão"
                }
        }
}
```

Agora como acessar o endereço da casa do João?

```
> joao.enderecos.casa
{ "rua" : "Ruas das Flores", "numero" : "1234", "complemento" : "Casa" }
```

```
> joao.enderecos.casa
{ "rua" : "Ruas das Flores", "numero" : "1234", "complemento" : "Casa" }
> joao.enderecos.trabalho
{ "rua" : "Rua das Árvores", "numero" : "123 C", "complemento" : "Galpão" }
>
```

```
> db.embedded.find()
{ "_id" : ObjectId("644126644220660d55c7256a"), "nome" : "Anderson", "idade" :
 48, "endereco" : { "rua" : "Rua das Flores", "numero" : "1314", "complemento"
 : "Casa" } }
{ "_id" : ObjectId("644128584220660d55c7256b"), "nome" : "João", "idade" : 35,
 "enderecos" : { "casa" : { "rua" : "Ruas das Flores", "numero" : "1234", "com
plemento" : "Casa" }, "trabalho" : { "rua" : "Rua das Árvores", "numero" : "12
3 C", "complemento" : "Galpão" } } }
>
```

```
> db.embedded.find().pretty()
{
        "_id" : ObjectId("644126644220660d55c7256a"),
        "nome" : "Anderson",
        "idade" : 48,
        "endereco" : {
                "rua" : "Rua das Flores",
                "numero" : "1314",
                "complemento" : "Casa"
        }
}
{
        "_id" : ObjectId("644128584220660d55c7256b"),
        "nome" : "João",
        "idade" : 35,
        "enderecos" : {
                "casa" : {
                        "rua" : "Ruas das Flores",
                        "numero" : "1234",
                        "complemento" : "Casa"
                },
                "trabalho" : {
                        "rua" : "Rua das Árvores",
                        "numero" : "123 C",
                        "complemento" : "Galpão"
                }
        }
}
>
```

Embedded

One To One

Embedded

One To Many

ONE TO ONE

Exemplo: Nosso sistema permite o cadastro de um único endereço por usuário, então podemos dizer que o endereço é único para cada usuário.

Vamos trabalhar com 2 collections.

```
> db.pessoas.insertOne({
...     nome: "Godofredo",
...     idade: 30,
...     profissao: "Programador"
... })
{
        "acknowledged" : true,
        "insertedId" : ObjectId("64412dc64220660d55c7256c")
}
>
```

```
> const p = db.pessoas.findOne()
> p.nome
Godofredo
> p._id
ObjectId("64412dc64220660d55c7256c")
>
```

O dado do id é o que vou precisar usar na outra collection para fazer referência a essa collection chamada pessoa.

```
> db.enderecos.insertOne({
...     rua: "Ruas das Flores",
...     numero:"1212",
...     complemento: "Casa",
...     pessoa_id: p._id
... })
{
        "acknowledged" : true,
        "insertedId" : ObjectId("64412ec94220660d55c7256d")
}
>
```

```
> db.pessoas.find()
{ "_id" : ObjectId("64412dc64220660d55c7256c"), "nome" : "Godofredo", "idade"
: 30, "profissao" : "Programador" }
>
```

```
> db.enderecos.find().pretty()
{
        "_id" : ObjectId("64412ec94220660d55c7256d"),
        "rua" : "Ruas das Flores",
        "numero" : "1212",
        "complemento" : "Casa",
        "pessoa_id" : ObjectId("64412dc64220660d55c7256c")
}
>
```

Selecionando o endereço de Godofredo

```
> db.enderecos.find({ pessoa_id : p._id })
{ "_id" : ObjectId("64412ec94220660d55c7256d"), "rua" : "Ruas das Flores", "nu
mero" : "1212", "complemento" : "Casa", "pessoa_id" : ObjectId("64412dc6422066
0d55c7256c") }
>
```

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.

```
> db.pessoas.insertOne({
...     nome: "Gustavo",
...     idade: 29,
...     profissao:"Gerente"
... })
{
        "acknowledged" : true,
        "insertedId" : ObjectId("644131db4220660d55c7256e")
}
> show collections
embedded
enderecos
pessoas
>
```

```
> db.pessoas.find()
{ "_id" : ObjectId("64412dc64220660d55c7256c"), "nome" : "Godofredo", "idade"
: 30, "profissao" : "Programador" }
{ "_id" : ObjectId("644131db4220660d55c7256e"), "nome" : "Gustavo", "idade" :
29, "profissao" : "Gerente" }
>
```

Vamos definir uma variável para Gustavo e pegar o id dele.

```
> const gustavo = db.pessoas.findOne({ nome: "Gustavo" })
> gustavoid = gustavo._id
ObjectId("644131db4220660d55c7256e")
> gustavo
{
        "_id" : ObjectId("644131db4220660d55c7256e"),
        "nome" : "Gustavo",
        "idade" : 29,
        "profissao" : "Gerente"
}
>
```

Vamos fazer a mesma coisa com Godofredo (criar uma variável para Godofredo e recuperar o id dele)

```
> const godofredo = db.pessoas.findOne({ nome: "Godofredo" })
> const godofredoid = godofredo._id
> godofredoid
ObjectId("64412dc64220660d55c7256c")
>
```

Agora vamos criar a collections de compras

```
> db.compras.insertMany([
...     { produtos: ["Livro", "Celular"], pessoa_id: godofredoid },
...     { produtos: ["Mouse", "Teclado"], pessoa_id: godofredoid },
...     { produtos: ["Agenda"], pessoa_id: gustavoid },
...     { produtos: ["Barbeador", "Suporte TV"], pessoa_id: gustavoid }
... ])
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("644134824220660d55c7256f"),
                ObjectId("644134824220660d55c72570"),
                ObjectId("644134824220660d55c72571"),
                ObjectId("644134824220660d55c72572")
        ]
}
>
```

```
> db.compras.find()
{ "_id" : ObjectId("644134824220660d55c7256f"), "produtos" : [ "Livro", "Celular" ], "pessoa_id" : ObjectId("64412dc64220660d55c7256c") }
{ "_id" : ObjectId("644134824220660d55c72570"), "produtos" : [ "Mouse", "Teclado" ], "pessoa_id" : ObjectId("64412dc64220660d55c7256c") }
{ "_id" : ObjectId("644134824220660d55c72571"), "produtos" : [ "Agenda" ], "pessoa_id" : ObjectId("644131db4220660d55c7256e") }
{ "_id" : ObjectId("644134824220660d55c72572"), "produtos" : [ "Barbeador", "Suporte TV" ], "pessoa_id" : ObjectId("644131db4220660d55c7256e") }
>
```

Como encontrar as compras de Godofredo?

```
> db.compras.find({ pessoa_id: godofredoid })
{ "_id" : ObjectId("644134824220660d55c7256f"), "produtos" : [ "Livro", "Celular" ], "pessoa_id
" : ObjectId("64412dc64220660d55c7256c") }
{ "_id" : ObjectId("644134824220660d55c72570"), "produtos" : [ "Mouse", "Teclado" ], "pessoa_id
" : ObjectId("64412dc64220660d55c7256c") }
>
```

Encontrar compras de Gustavo

```
> db.compras.find({ pessoa_id: gustavoid })
{ "_id" : ObjectId("644134824220660d55c72571"), "produtos" : [ "Agenda" ], "pessoa_id" : Object
Id("644131db4220660d55c7256e") }
{ "_id" : ObjectId("644134824220660d55c72572"), "produtos" : [ "Barbeador", "Suporte TV" ], "pe
ssoa_id" : ObjectId("644131db4220660d55c7256e") }
>
```

Vamos inserir outra pessoa

```
> db.pessoas.insertOne({
...     nome: "Pedro",
...     idade: 44,
...     profissao: "Motorista"
... })
{
        "acknowledged" : true,
        "insertedId" : ObjectId("644136294220660d55c72573")
}
>
```

Pegando diretamente o id de Pedro

```
> const pedroid = db.pessoas.findOne({ nome:"Pedro" })._id
> pedroid
ObjectId("644136294220660d55c72573")
>
```

Consultando as compras de Pedro

```
> db.compras.find({ pessoa_id: pedroid })
>
```

Veja que não é retornado nada, pois na collection compras, Pedro não realizou nenhuma compra!

```
> db.compras.find({ pessoa_id: pedroid }).count()
0
>
```

MANY TO MANY

Acontece quando os registros das duas collections possuem mais de uma relação entre si

Exemplo: Temos alunos e cursos, um curso pode ter vários alunos matriculados e um aluno pode estar fazendo vários cursos.

Normalmente se cria uma estrutura intermediaria, ou seja, uma collection.

Esta collection contém apenas os ids de cursos e alunos.

Vamos criar uma collection de Cursos

```
> db.cursos.insertMany([
...     {nome:"PHP Avançado"},
...     {nome:"Javascript Básico"},
...     {nome:"Banco de Dados NoSQL"}
... ])
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("644138974220660d55c72574"),
                ObjectId("644138974220660d55c72575"),
                ObjectId("644138974220660d55c72576")
        ]
}
>
```

```
> db.cursos.find()
{ "_id" : ObjectId("644138974220660d55c72574"), "nome" : "PHP Avançado" }
{ "_id" : ObjectId("644138974220660d55c72575"), "nome" : "Javascript Básico" }
{ "_id" : ObjectId("644138974220660d55c72576"), "nome" : "Banco de Dados NoSQL" }
>
```

Vamos aproveitar a collection pessoas para este exemplo. Verifique as variáveis criadas anteriormente.

```
> gustavo
{
        "_id" : ObjectId("644131db4220660d55c7256e"),
        "nome" : "Gustavo",
        "idade" : 29,
        "profissao" : "Gerente"
}
> godofredo
{
        "_id" : ObjectId("64412dc64220660d55c7256c"),
        "nome" : "Godofredo",
        "idade" : 30,
        "profissao" : "Programador"
}
>
```

Vamos encapsular também os cursos em variáveis

```
> const php = db.cursos.findOne({ nome: "PHP Avançado"})
> const js = db.cursos.findOne({ nome: "Javascript Básico"})
> const bd = db.cursos.findOne({ nome: "Banco de Dados NoSQL"})
```

```
> php
{ "_id" : ObjectId("644138974220660d55c72574"), "nome" : "PHP Avançado" }
> js
{
        "_id" : ObjectId("644138974220660d55c72575"),
        "nome" : "Javascript Básico"
}
> bd
{
        "_id" : ObjectId("644138974220660d55c72576"),
        "nome" : "Banco de Dados NoSQL"
}
>
```

Agora vamos criar a collection de relação entre Pessoas e Cursos

```
> db.curso_pessoa.insertMany([
...     { curso_id: php._id, pessoa_id: gustavo._id },
...     { curso_id: js._id, pessoa_id: gustavo._id },
...     { curso_id: php._id, pessoa_id: godofredo._id }
... ])
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("64413b5d4220660d55c72577"),
                ObjectId("64413b5d4220660d55c72578"),
                ObjectId("64413b5d4220660d55c72579")
        ]
}
>
```

```
> db.curso_pessoa.find()
{ "_id" : ObjectId("64413b5d4220660d55c72577"), "curso_id" : ObjectId("644138974220660d55c72574"), "pessoa_id" : ObjectId("644
131db4220660d55c7256e") }
{ "_id" : ObjectId("64413b5d4220660d55c72578"), "curso_id" : ObjectId("644138974220660d55c72575"), "pessoa_id" : ObjectId("644
131db4220660d55c7256e") }
{ "_id" : ObjectId("64413b5d4220660d55c72579"), "curso_id" : ObjectId("644138974220660d55c72574"), "pessoa_id" : ObjectId("644
12dc64220660d55c7256c") }
>
```

```
> db.curso_pessoa.find().pretty()
{
        "_id" : ObjectId("64413b5d4220660d55c72577"),
        "curso_id" : ObjectId("644138974220660d55c72574"),
        "pessoa_id" : ObjectId("644131db4220660d55c7256e")
}
{
        "_id" : ObjectId("64413b5d4220660d55c72578"),
        "curso_id" : ObjectId("644138974220660d55c72575"),
        "pessoa_id" : ObjectId("644131db4220660d55c7256e")
}
{
        "_id" : ObjectId("64413b5d4220660d55c72579"),
        "curso_id" : ObjectId("644138974220660d55c72574"),
        "pessoa_id" : ObjectId("64412dc64220660d55c7256c")
}
>
```

Criando um loop para relacionar todos os alunos que fazem o curso de Javascript

```
> db.curso_pessoa.find({ curso_id: js._id }).forEach(function(aluno){
...     idsAlunosJs.push(aluno.pessoa_id);
... });
>
```

```
> idsAlunosJs
[ ObjectId("644131db4220660d55c7256e") ]
>
```

Para PHP:

```
> const idsAlunosPhp = [];
> db.curso_pessoa.find({ curso_id: php._id }).forEach(function(aluno){
...     idsAlunosPhp.push(aluno.pessoa_id);
... });
> idsAlunosPhp
[
        ObjectId("644131db4220660d55c7256e"),
        ObjectId("64412dc64220660d55c7256c")
]
>
```

Para Banco de Dados:

```
> const idsAlunosBd = [];
> db.curso_pessoa.find({ curso_id: bd._id }).forEach(function(aluno){
...     idsAlunosBd.push(aluno.pessoa_id);
... });
>
```

```
> idsAlunosPhp
[
        ObjectId("644131db4220660d55c7256e"),
        ObjectId("64412dc64220660d55c7256c")
]
> idsAlunosBd
[ ]
>
```

Agora vamos utilizar o operador $in para fazer a seleção

```
> db.pessoas.find({
...     _id:{ $in: idsAlunosJs }
... })
{ "_id" : ObjectId("644131db4220660d55c7256e"), "nome" : "Gustavo", "idade" : 29, "profissao" : "Gerente" }
>
```

```
> db.pessoas.find({
...     _id:{ $in: idsAlunosJs }
... })
{ "_id" : ObjectId("644131db4220660d55c7256e"), "nome" : "Gustavo", "idade" : 29, "profissao" : "Gerente" }
> db.pessoas.find({
...     _id:{ $in: idsAlunosPhp }
... })
{ "_id" : ObjectId("64412dc64220660d55c7256c"), "nome" : "Godofredo", "idade" : 30, "profissao" : "Programador" }
{ "_id" : ObjectId("644131db4220660d55c7256e"), "nome" : "Gustavo", "idade" : 29, "profissao" : "Gerente" }
> db.pessoas.find({
...     _id:{ $in: idsAlunosBd }
... })
>
```

Verificando os dados de Pessoas

```
> db.pessoas.find()
{ "_id" : ObjectId("64412dc64220660d55c7256c"), "nome" : "Godofredo", "idade" : 30, "profissao" : "Programador" }
{ "_id" : ObjectId("644131db4220660d55c7256e"), "nome" : "Gustavo", "idade" : 29, "profissao" : "Gerente" }
{ "_id" : ObjectId("644136294220660d55c72573"), "nome" : "Pedro", "idade" : 44, "profissao" : "Motorista" }
>
```

Repare que Pedro não está em nenhum curso. Vamos colocá-lo em um curso.

```
> const pedro = db.pessoas.findOne({nome:"Pedro"})
> pedro
{
        "_id" : ObjectId("644136294220660d55c72573"),
        "nome" : "Pedro",
        "idade" : 44,
        "profissao" : "Motorista"
}
> db.curso_pessoa.insertOne({curso_id: bd._id, pessoa_id: pedro._id})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("644140ea4220660d55c7257a")
}
>
```

```
> db.curso_pessoa.find().pretty()
{
        "_id" : ObjectId("64413b5d4220660d55c72577"),
        "curso_id" : ObjectId("644138974220660d55c72574"),
        "pessoa_id" : ObjectId("644131db4220660d55c7256e")
}
{
        "_id" : ObjectId("64413b5d4220660d55c72578"),
        "curso_id" : ObjectId("644138974220660d55c72575"),
        "pessoa_id" : ObjectId("644131db4220660d55c7256e")
}
{
        "_id" : ObjectId("64413b5d4220660d55c72579"),
        "curso_id" : ObjectId("644138974220660d55c72574"),
        "pessoa_id" : ObjectId("64412dc64220660d55c7256c")
}
{
        "_id" : ObjectId("644140ea4220660d55c7257a"),
        "curso_id" : ObjectId("644138974220660d55c72576"),
        "pessoa_id" : ObjectId("644136294220660d55c72573")
}
>
```

Se você usar o comando anterior para pesquisar os alunos que estão em Banco de Dados terá um retorno vazio.

```
> idsAlunosBd
[ ]
>
```

É necessário rodar o comando (com outro nome de variável) para adicionar esse novo registro.

```
> const idsAlunosBd2 = [];
> db.curso_pessoa.find({ curso_id: bd._id }).forEach(function(aluno){
...     idsAlunosBd2.push(aluno.pessoa_id);
... });
> idsAlunosBd2
[ ObjectId("644136294220660d55c72573") ]
>
```

```
> db.pessoas.find({     _id:{ $in: idsAlunosBd2 } })
{ "_id" : ObjectId("644136294220660d55c72573"), "nome" : "Pedro", "idade" : 44, "profissao"
: "Motorista" }
>
```

CONCLUSÃO

Por que não fazer tudo com embedded documents?

- Há um limite de 16 mb por document.

- Em projetos grandes isso se tornará um problema

- Dividir em collections trará mais benefícios a longo prazo

- Lidar com consultas pode ser mais trabalhoso!