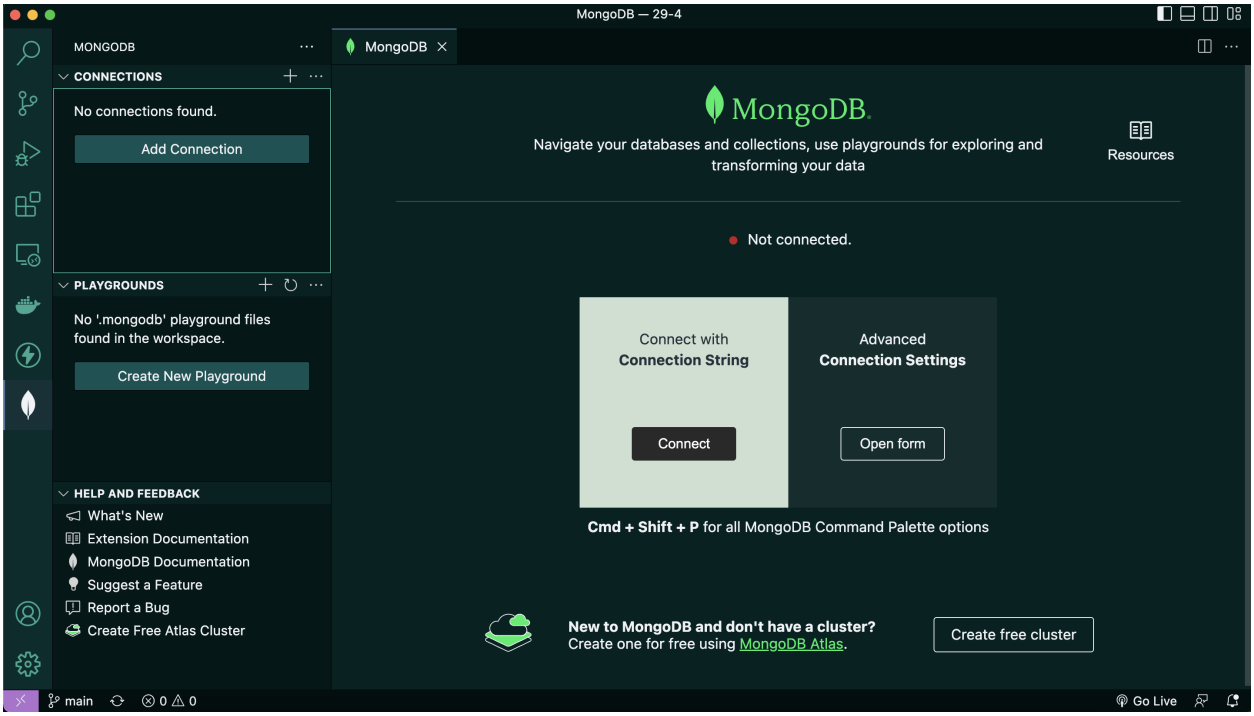
Intro 🗓️ (01/11/2022)

Iniciando nosso Projeto

Usaremos uma imagem do MongoDB em um container Docker e nos conectaremos a respectiva porta utilizando a extensão MongoDB for VSCode para utilizarmos o **Playground**

```
docker run -d -p 27017:27017 --name mongoDocker mongo:latest
```



Consultando nossos Databases e Coleções

```
show dbs;
show collections;
```

Realizando Inserções

O comando insertOne

```
use class;
db.inventory.insertOne({ item: "journal", qty: 25, size: { h: 14, w: 21, uom: "cm" }, status: "A" })
```

O comando insertMany

```
db.inventory.insertMany([
  { item: "paper", qty: 100, size: { h: 8.5, w: 11, uom: "in" }, status: "D" },
  { item: "planner", qty: 75, size: { h: 22.85, w: 30, uom: "cm" }, status: "D" },
```

```
  { item: "postcard", qty: 45, size: { h: 10, w: 15.25, uom: "cm" }, status: "A" }
]);
```

Consultando nossos dados

Docs Home → Develop Applications → MongoDB Manual

db.collection.find()

Definition

`db.collection.find(query, projection, options)`

```
// SELECT * FROM inventory;

db.inventory.find({})
db.inventory.find()
```

Como aplicar condições?

```
// SELECT * FROM inventory WHERE status = 'D';

db.inventory.find({ status: "D" });
```

Where status = "D” and item = “paper”

```
// SELECT * FROM inventory WHERE status = 'D' AND item = 'paper';

db.inventory.find({ status: "D", item: "paper" });
```

“Relacionamentos”

1:1 (Um para Um)

O Pensamento Relacional (SQL)

```
// documento do usuário
{
  _id: "joe",
  name: "Joe Bookreader"
}

// documento do endereço
{
  user_id: "joe", // referencia o documento do usuário
  street: "123 Fake Street",
  city: "Faketon",
  state: "MA",
  zip: "12345"
}
```

O Pensamento NoSQL

```
{
  _id: "joe",
  name: "Joe Bookreader",
  address: {
    street: "123 Fake Street",
    city: "Faketon",
    state: "MA",
    zip: "12345"
  }
}
```

https://www.youtube.com/watch?v=FFj04Apz_BE

Incluindo um Documento e um subdocumento ou documento incorporado

```
db.users.insertOne({
  _id: "joe",
  name: "Joe Bookreader",
  address: {
    street: "123 Fake Street",
    city: "Faketon",
    state: "MA",
    zip: "12345"
  }
});
```

Recuperando os dados

```
db.users.find({ _id: "joe" }).pretty();
```

1:N (Um para muitos)

O Pensamento relacional (SQL)

```


// documento da usuária
{
  _id: "amanda",
  name: "Amanda Granger"
}

// documento do endereço
{
  user_id: "amanda", // referencia o documento da usuária
  street: "Sunset Boulevard",
  city: "Los Angeles",
  state: "CA",
  zip: "12345"
}

{
  user_id: "amanda",
  street: "1 Some Other Street",
  city: "New York",
  state: "NY",
  zip: "12345"
}
```

O Pensamento NoSQL

```
{
  _id: "amanda",
  name: "Amanda Granger",
  addresses: [
    {
      street: "Sunset Boulevard",
      city: "Los Angeles",
      state: "CA",
      zip: "12345"
    },
    {
      street: "1 Some Other Street",
      city: "New York",
      state: "NY",
      zip: "12345"
    }
  ]
 }
```

Inserindo e realizando a consulta

```
db.users.insertOne({
  _id: "amanda",
  name: "Amanda Granger",
  addresses: [
    {
      street: "Sunset Boulevard",
      city: "Los Angeles",
      state: "CA",
      zip: "12345"
    },
    {
      street: "1 Some Other Street",
      city: "New York",
      state: "NY",
      zip: "12345"
    }
  ]
 }
);

db.users.find({ _id: "amanda" }).pretty();
```

Acessando sub-documentos

```
db.users.find({ "addresses.state": "NY" }).pretty();
```

Projeções

Docs Home → Develop Applications → MongoDB Manual

db.collection.find()

Definition

`db.collection.find(query, projection, options)`

```
db.inventory.find({ status: "A" }, { item: 1, status: 1 });
db.inventory.find({ status: "A" }, { item: true, status: true });
db.inventory.find({ status: "A" }, { item: 1, status: 1, _id: 0 });
db.inventory.find({ status: "A" }, { item: true, _id: false });
```

Suprimindo apenas alguns campos

```
db.inventory.find({ status: "A" }, { status: 0, instock: 0 });
```

Skip and Limit

Paginação

```
db.inventory.find({}, { item: 1 }); // retorna todos os cinco documentos
db.inventory.find({}, { item: 1 }).skip(0).limit(2); // retorna o primeiro e segundo documentos
db.inventory.find({}, { item: 1 }).skip(1*2).limit(2); // retorna o terceiro e quarto documentos
db.inventory.find({}, { item: 1 }).skip(2*2).limit(2); // retorna o último elemento
db.inventory.find({}, { item: 1 }).skip(3*2).limit(2); // retorna nada
```

O que aprendemos?

- Realizar queries simples
- Inserir dados com find e findOne
- Mindset SQL vs NoSQL
- Documentos incorporados
- Projeções
- Skip e Limit

Realize os exercícios 🚀

> *🚀 Se liga nesse foguete!*
> *Os exercícios destacados com 🚀 são os fundamentais pra você ir bem no projeto!*
> *Todos os exercícios vão contribuir com sua formação, mas fique de olho nesses! 👀*