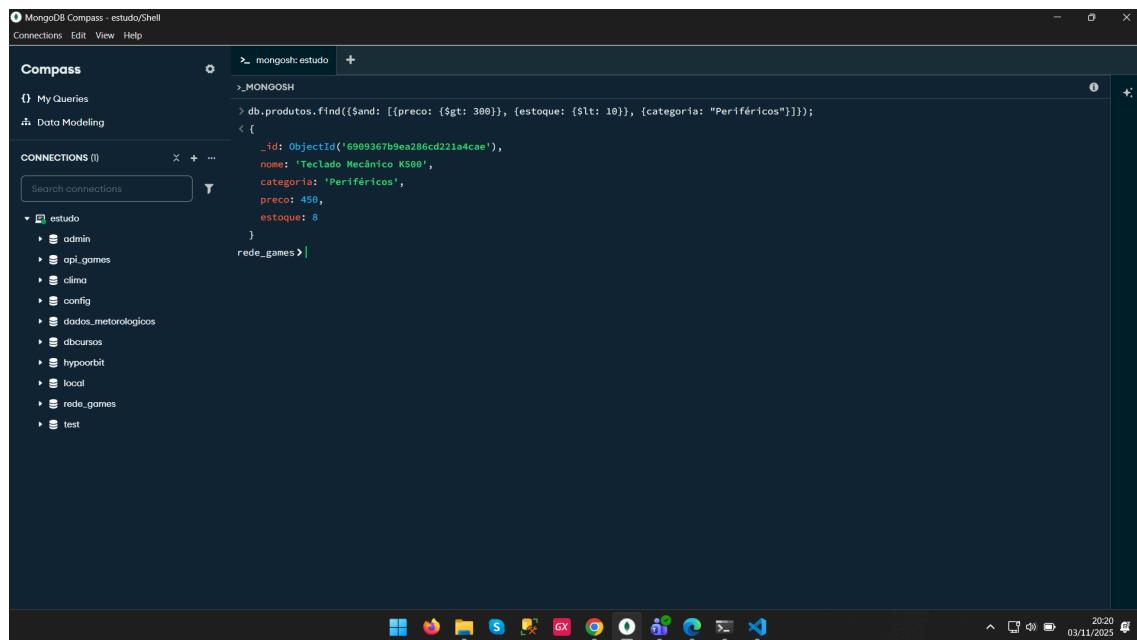


Atividade - Banco de Dados Não Relacional – DSM – Profa. Lucineide
Nome: Carlos Eduardo da Silva Magalhães
Tema: Atividade prática semanas 5 a 8

Exercício 01

```
db.produtos.find({$and: [{preco: {$gt: 300}}, {estoque: {$lt: 10}}, {categoria: "Periféricos"}]});
```



Exercício 02

```
db.produtos.updateMany({}, { $set: { desconto: 0.00 } });
db.produtos.updateOne({nome: "Monitor UltraWide"},{$set: {desconto: 10}});
db.produtos.find({}, { _id: 0, nome: 1, desconto: 1});
```

The screenshot shows the MongoDB Compass interface with a connection named 'estudo'. In the left sidebar, under 'CONNECTIONS', there is a list of databases: 'estudo', 'admin', 'api.games', 'clima', 'config', 'dados_meteorologicos', 'dbcursos', 'hypoorbit', 'local', 'rede_games', and 'test'. The main pane displays the results of a query in the 'mongosh: estudo' shell. The query is:

```
> db.produtos.updateMany({categoria: "Áudio"}, {$inc: {estoque: 3}});  
> db.produtos.find({}, {_id: 0, nome: 1, categoria:1, estoque: 1});
```

The output shows the updated document:

```
{  
  nome: 'Headset ProSound',  
  categoria: 'Áudio',  
  estoque: 15  
}
```

Exercício 03

```
db.produtos.updateMany({categoria: "Áudio"}, {$inc: {estoque: 3}});  
db.produtos.find({}, {_id: 0, nome: 1, categoria:1, estoque: 1});
```

The screenshot shows the MongoDB Compass interface with a connection named 'estudo'. In the left sidebar, under 'CONNECTIONS', there is a list of databases: 'estudo', 'admin', 'api.games', 'clima', 'config', 'dados_meteorologicos', 'dbcursos', 'hypoorbit', 'local', 'rede_games', and 'test'. The main pane displays the results of a query in the 'mongosh: estudo' shell. The query is:

```
> db.produtos.updateMany({categoria: "Áudio"}, {$inc: {estoque: 3}});  
> db.produtos.find({}, {_id: 0, nome: 1, categoria:1, estoque: 1});
```

The output shows the updated document:

```
{  
  nome: 'Headset ProSound',  
  categoria: 'Áudio',  
  estoque: 15  
}
```

Exercício 04

```
db.produtos.replaceOne({ nome: "Console Zeta" }, { nome: "Console Zeta",  
preco: 3500.00, garantia: "2 anos", categoria: "Consoles", estoque: 3  
});
```

```
db.produtos.find({nome: "Console Zeta"});
```

The screenshot shows the MongoDB Compass interface. On the left, the 'Connections' sidebar lists databases: 'estudo', 'admin', 'api.games', 'clima', 'config', 'dados_meteorologicos', 'dbcursos', 'hypoorbit', 'local', 'rede_games', and 'test'. The main panel displays the results of a query in the 'mongosh: estudo' shell:

```
>_MONGOSH
> db.produtos.replaceOne({ nome: "Console Zeta" }, { nome: "Console Zeta", preco: 3500.00, garantia: "2 anos", categoria: "Consoles", estoque: 3 });
< {
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
> db.produtos.find({nome: "Console Zeta"})
< [
  {
    _id: ObjectId('6009367b0ea286cd221a4cb0'),
    nome: 'Console Zeta',
    preco: 3500,
    garantia: '2 anos',
    categoria: 'Consoles',
    estoque: 3
  }
]
rede_games >
```

The status bar at the bottom right shows the date as 03/11/2025 and the time as 20:45.

Exercício 05

```
db.produtos.deleteOne({estoque: {$lt: 5}});
db.produtos.find({estoque: {$lt: 5}});
```

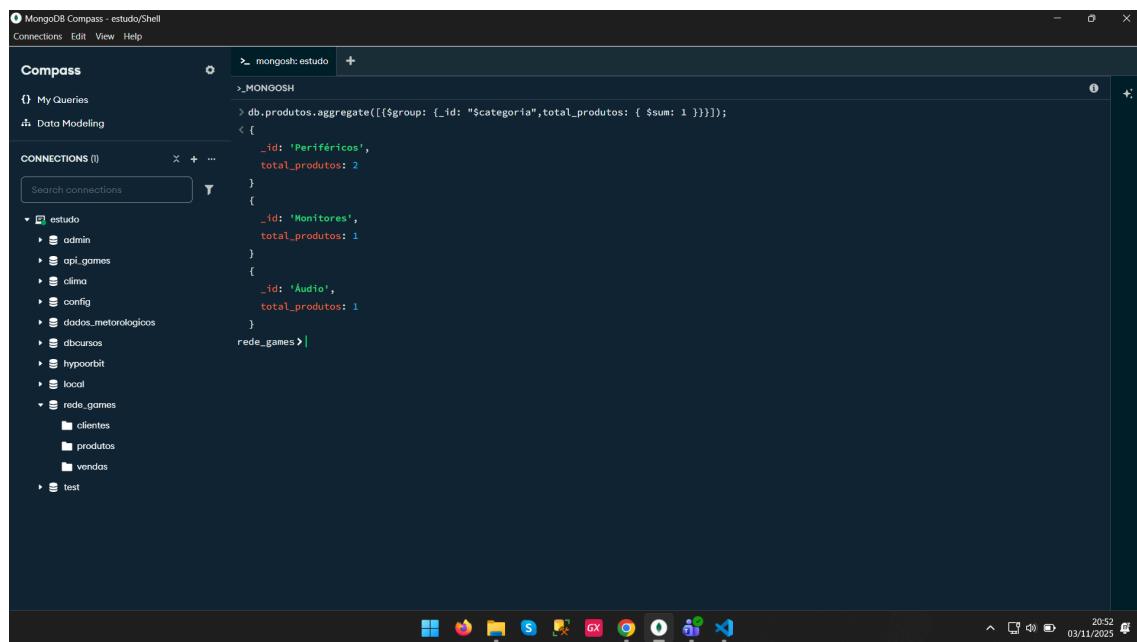
The screenshot shows the MongoDB Compass interface. The 'Connections' sidebar lists the same databases as before. The main panel shows the results of the following commands in the 'mongosh: estudo' shell:

```
>_MONGOSH
> db.produtos.deleteOne({estoque: {$lt: 5}});
< {
  acknowledged: true,
  deletedCount: 1
}
> db.produtos.find({estoque: {$lt: 5}})
<
rede_games >
```

The 'rede_games' database is expanded to show collections: 'clientes', 'produtos', and 'vendas'. The status bar at the bottom right shows the date as 03/11/2025 and the time as 20:50.

Exercício 06

```
db.produtos.aggregate([{$group: {_id: "$categoria", total_produtos: { $sum: 1 }}}]);
```



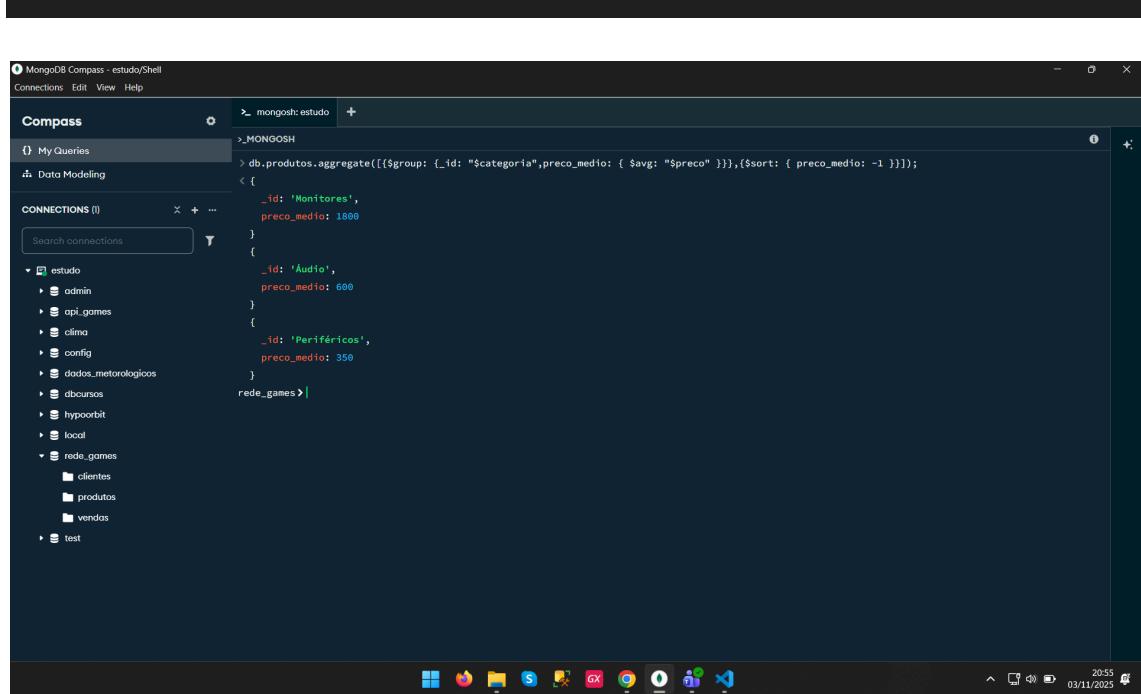
The screenshot shows the MongoDB Compass interface. On the left, the 'Compass' sidebar lists databases: 'estudo', 'admin', 'api.games', 'clima', 'config', 'dados_meteorologicos', 'dbcurso', 'hypoorbit', 'local', 'rede_games' (which contains 'clientes', 'produtos', and 'vendas'), and 'test'. The main panel displays the command line and its output:

```
>_MONGOSH
> db.produtos.aggregate([{$group: {_id: "$categoria", total_produtos: { $sum: 1 }}}]);
< [
  {
    _id: 'Periféricos',
    total_produtos: 2
  },
  {
    _id: 'Monitores',
    total_produtos: 1
  },
  {
    _id: 'Áudio',
    total_produtos: 1
  }
]
```

The status bar at the bottom right indicates the time as 20:52 and the date as 03/11/2025.

Exercício 07

```
db.produtos.aggregate([{$group: {_id: "$categoria", preco_medio: { $avg: "$preco" }}}, {$sort: { preco_medio: -1 }}]);
```



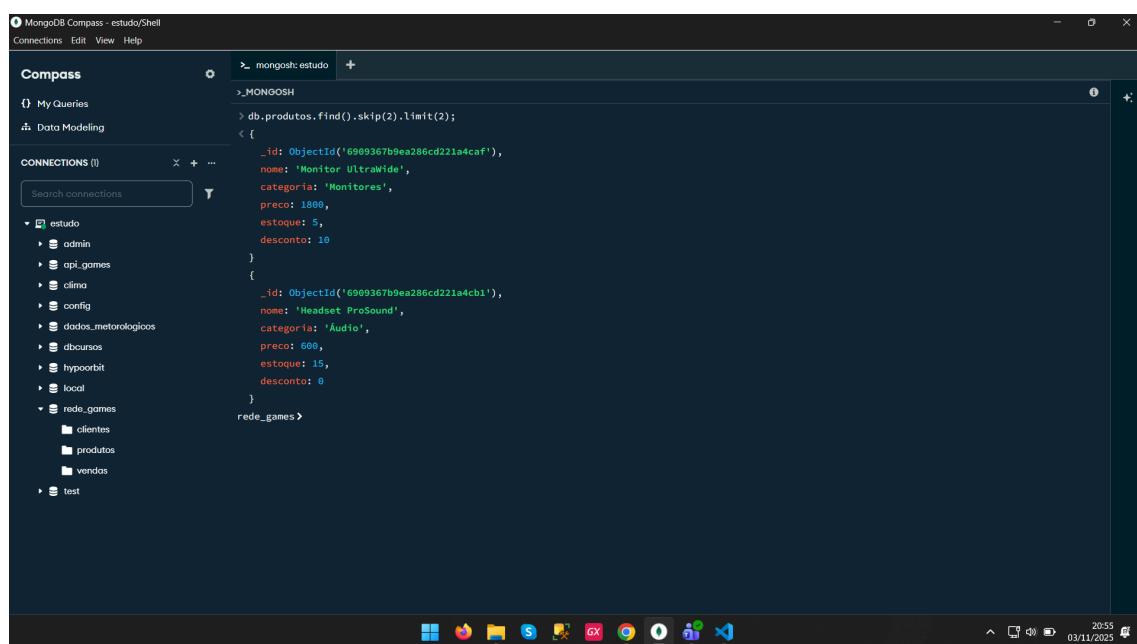
The screenshot shows the MongoDB Compass interface. The 'Compass' sidebar lists the same databases as the previous screenshot. The main panel displays the command line and its output:

```
>_MONGOSH
> db.produtos.aggregate([{$group: {_id: "$categoria", preco_medio: { $avg: "$preco" }}}, {$sort: { preco_medio: -1 }}]);
< [
  {
    _id: 'Monitores',
    preco_medio: 1800
  },
  {
    _id: 'Áudio',
    preco_medio: 600
  },
  {
    _id: 'Periféricos',
    preco_medio: 350
  }
]
```

The status bar at the bottom right indicates the time as 20:55 and the date as 03/11/2025.

Exercício 08

```
db.produtos.find().skip(2).limit(2);
```



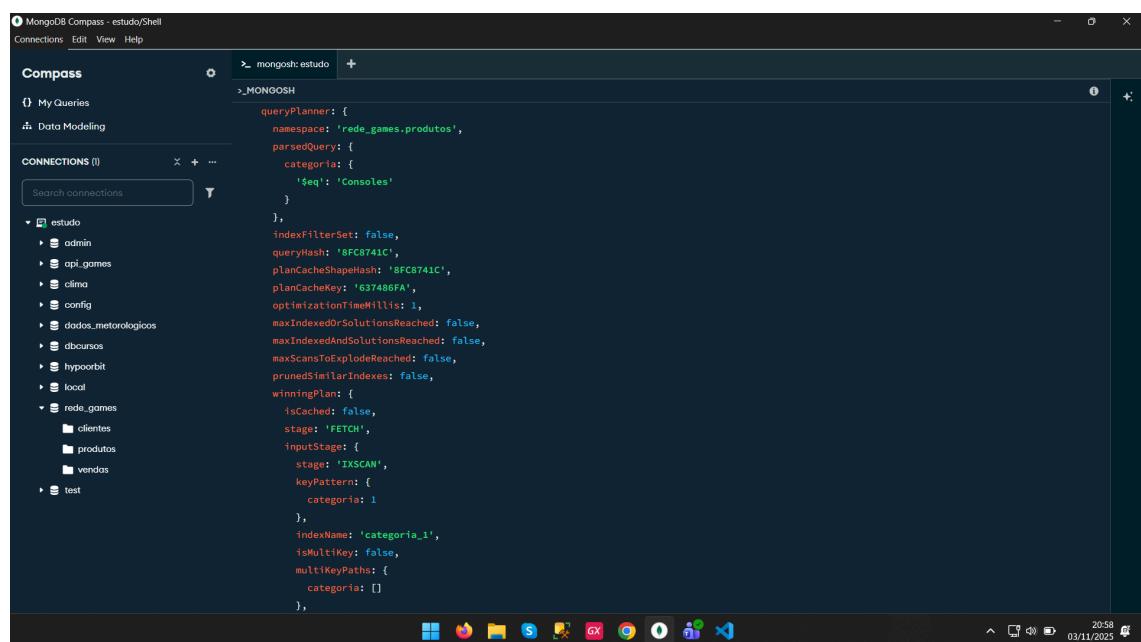
```
>_MONGOSH
> db.produtos.find().skip(2).limit(2);
< [
  {
    "_id": ObjectId("69e9367b9ea286cd221a4caf"),
    "name": "Monitor UltraWide",
    "categoria": "Monitores",
    "preco": 1800,
    "estoque": 5,
    "desconto": 10
  },
  {
    "_id": ObjectId("69e9367b9ea286cd221a4cb1"),
    "name": "Headset ProSound",
    "categoria": "Audio",
    "preco": 600,
    "estoque": 15,
    "desconto": 0
  }
]
```

Exercício 09

```
db.produtos.createIndex({ categoria: 1 });

db.produtos.getIndexes();

db.produtos.find({ categoria: "Consoles" }).explain("executionStats");
```



```
>_MONGOSH
> db.produtos.createIndex({ categoria: 1 });

> db.produtos.getIndexes();

> db.produtos.find({ categoria: "Consoles" }).explain("executionStats");
{
  "queryPlanner": {
    "namespace": "rede_games.produtos",
    "parsedQuery": {
      "categoria": {
        "$eq": "Consoles"
      }
    },
    "indexFilterSet": false,
    "queryHash": "8FC8741C",
    "planCacheShapeHash": "8FC8741C",
    "planCacheKey": "63746FA",
    "optimizationTimeMillis": 1,
    "maxIndexedOrSolutionsReached": false,
    "maxIndexedAndSolutionsReached": false,
    "maxScansToExplodeReached": false,
    "prunedSimilarIndexes": false,
    "winningPlan": {
      "isCached": false,
      "stage": "FETCH",
      "inputStage": {
        "stage": "IXSCAN",
        "keyPattern": {
          "categoria": 1
        },
        "indexName": "categoria_1",
        "isMultiKey": false,
        "multiKeyPaths": {
          "categoria": []
        }
      }
    }
  }
}
```

Exercício 10

```
import mongoose, {Document, Schema} from "mongoose";
```

```

export interface IProdutos extends Document {
  nome: string;
  categoria: string;
  preco: number;
  estoque: number;
}

const ProdutoSchema: Schema = new Schema({
  nome: { type: String, required: true },
  categoria: { type: String, required: true },
  preco: { type: Number, required: true },
  estoque: { type: Number, required: true }
});

export default mongoose.model<IProdutos>('Produto', ProdutoSchema);

```

```

import express, {Request, Response} from "express";
import mongoose from "mongoose";
import path from "path";
import Produto from "./models/produtos";

const app = express();
const PORT = 3000;
const MONGO_URI = 'mongodb://localhost:27017/rede_games';

app.use(express.static(path.join(__dirname, "../view")));
app.use(express.json());

mongoose.connect(MONGO_URI)
  .then(() => console.log('MongoDB conectado'))
  .catch(err => console.log('Erro ao conectar ao MongoDB: ', err));

app.get("/api/produtos", async (req: Request, res: Response) => {
  try{
    const produtos = await Produto.find({}, {_id: 0, nome: 1,
preco: 1}).limit(5);
    res.json(produtos);
  }catch(err){
    res.status(500).json({message: 'Erro ao buscar produtos'});
  }
});

```

```
app.listen(PORT, () => {
  console.log(`Servidor rodando em http://localhost:${PORT}`);
  console.log(`API de produtos em
http://localhost:${PORT}/api/produtos`);
});
```

