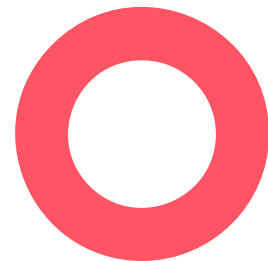


DESENVOLVIMENTO DE SISTEMAS

UC13

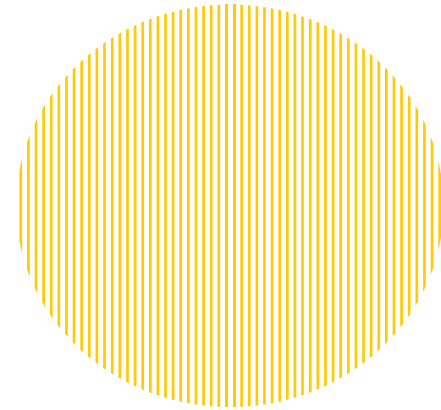
Prof. Viviane de Lima

viviane.lfrancelino@sp.senac.br

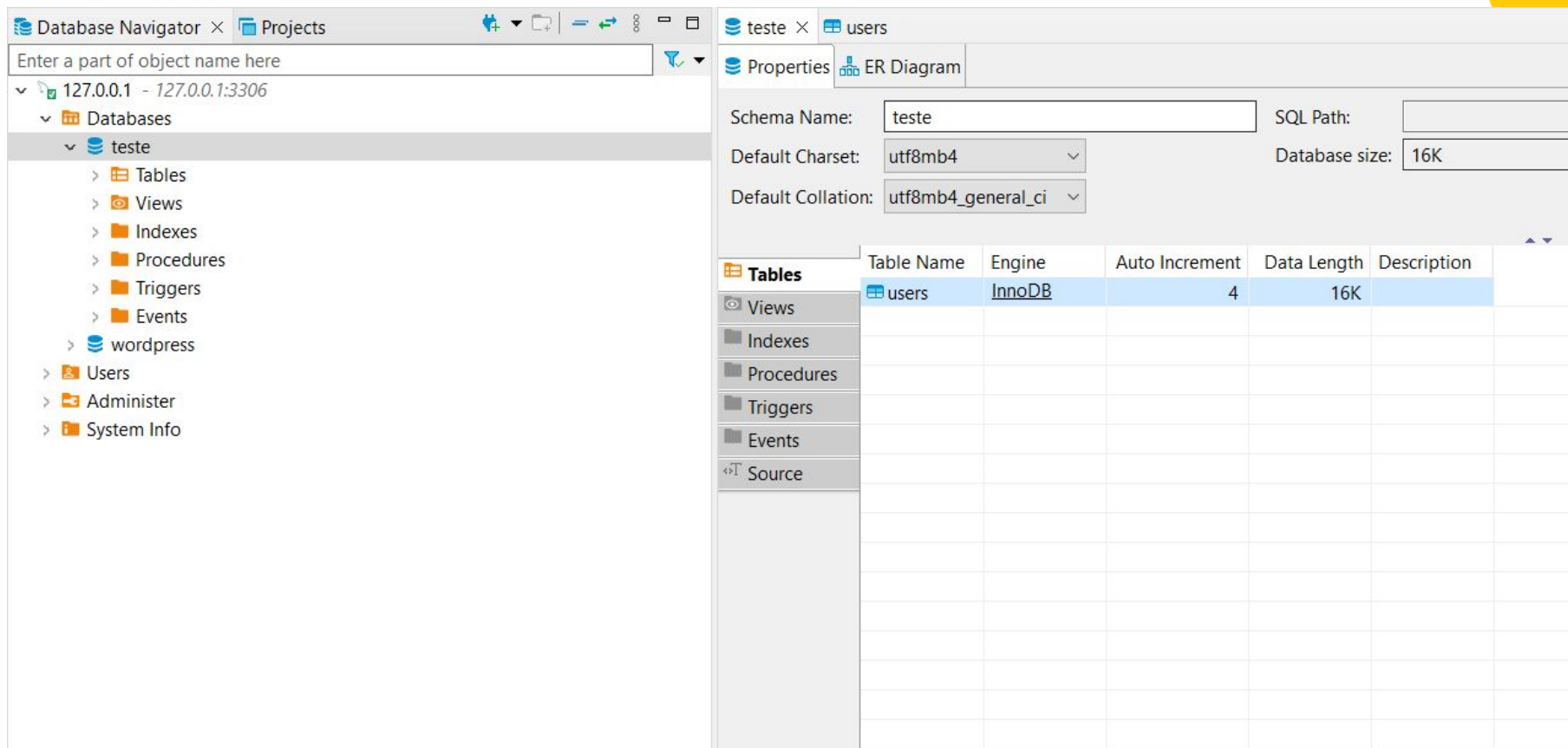


AULA 10

MODELS NO SEQUELIZE+ TYPESCRIPT



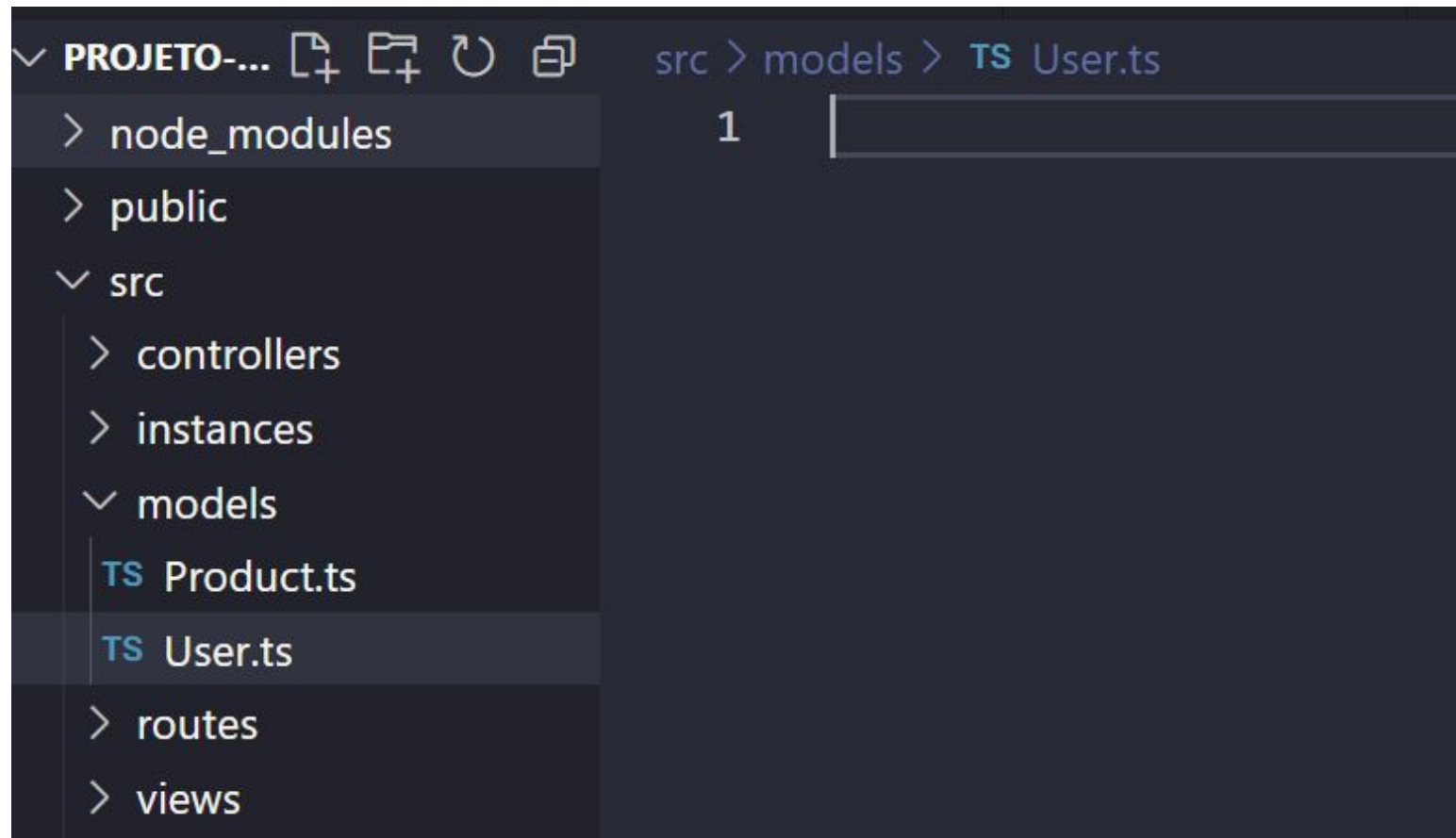
VAMOS CRIAR UM MODEL PARA A TABELA QUE TEMOS NO DEBEAVER



The screenshot displays the MySQL Database Navigator interface. On the left, the 'Database Navigator' pane shows the hierarchy of the 'teste' database, including Tables, Views, Indexes, Procedures, Triggers, Events, and other system-related items. The 'users' table is highlighted under the 'Tables' category. On the right, the 'Properties' pane for the 'users' table is visible, showing the Schema Name as 'teste', Default Charset as 'utf8mb4', and Default Collation as 'utf8mb4_general_ci'. Below the Properties pane, a table lists the database objects, with the 'users' table selected.

Table Name	Engine	Auto Increment	Data Length	Description
users	InnoDB	4	16K	

DENTRO DA PASTA MODELS CRIE USER.TS



CRIANDO A INSTANCIA DE USER

TS User.ts X

src > models > TS User.ts > UserInstance

```
1 //importar o Sequelize com o Model e DataTypes
2 import { Model , DataTypes } from "sequelize";
3
4 //importar a conexão com BD
5 import {sequelize} from '../instances/mysql'
6
7 //vamos criar um type para nossa instancia de User
8 export interface UserInstance extends Model {
9     id: number
10    name: string
11    age: number
12 }
```

Aqui vamos criar a nossa instancia para o User, então sempre quando usarmos o User ele vai vir junto com esses types que colocamos (id, name, age). Basicamente estamos detalhando nossa tabela User para o TypeScript

ENSINANDO O SEQUELIZE SOBRE AS INFORMAÇÕES DO NOSSO BANCO



TS User.ts X

src > models > TS User.ts > ...

```
1  import { Model , DataTypes } from "sequelize";
2  import {sequelize} from '../instances/mysql'
3
4
5  export interface UserInstance extends Model {
6    id: number
7    name: string
8    age: number
9  }
```

Estamos usando o Sequelize que vai receber o nosso UserInstance (para ele saber quais são os types que colocamos que são: id, name e age)

```
12 export const User = sequelize.define<UserInstance>("User",{
13   id:{
14     primaryKey:true,
15     type:DataTypes.INTEGER
16   },
17   name:{
18     type:DataTypes.STRING
19   },
20   age:{
21     type: DataTypes.INTEGER,
22     defaultValue: 18
23   }
24 })
```

Nome do nosso model

Agora vou detalhar para o sequelize quais são os campos do meu banco de dados

EM SEGUIDA PRECISAMOS COLOCAR PROPRIEDADES ESPECIFICAS PARA O SEQUELIZE

```
export interface UserInstance extends Model {
  id: number
  name: string
  age: number
}

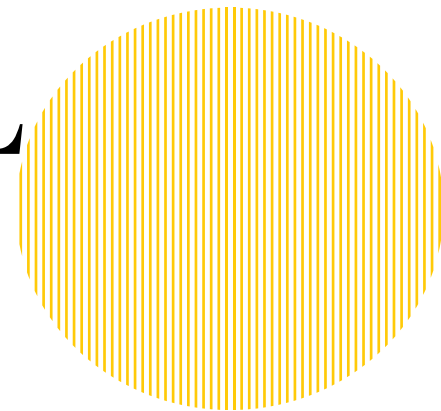
export const User = sequelize.define<UserInstance>("User", {
  id: {
    primaryKey: true,
    type: DataTypes.INTEGER
  },
  name: {
    type: DataTypes.STRING
  },
  age: {
    type: DataTypes.INTEGER,
    defaultValue: 18
  },
}, {
  tableName: 'users',
  timestamps: false
})
```

Colocando o nome da nossa tabela do banco de dados

Se deixarmos o timestamps como true, automaticamente ele vai assumir que a nossa tabela tem mais dois campos, que são **CreatedAt** e **UpdatedAt** (como não temos essas colunas no banco, precisamos por como false)



**CONSULTANDO
DADOS NO MODEL**



VÁ PARA homecontroller.ts E REMOVA OS CÓDIGOS ABAIXO

TS User.ts

TS homeController.ts X

src > controllers > TS homeController.ts > [🔍] home

```
1  import { Request, Response } from 'express';
2  import { Product } from '../models/Product';
3
4  import { sequelize } from '../instances/mysql';
5
6  export const home = async (req: Request, res: Response) => {
7    try {
8      await sequelize.authenticate()
9      console.log("Conexão estabelecida com sucesso!")
10    } catch (error) {
11      console.log("Deu problema: ", error)
12    }
13
14    let age: number = 90;
15    let showOld: boolean = false;
16
17    if (age > 50) {
18      showOld = true;
19    }
20
21    let list = Product.getAll();
22    let expensiveList = Product.getFromPriceAfter(12);
```

USANDO O MODEL

TS User.ts X TS homeController.ts X

src > controllers > TS homeController.ts > [🔍] home

```
1  import { Request, Response } from 'express';
2  import { Product } from '../models/Product';
3
4  //precisamos importar users
5  import { User } from '../models/User';
6
7  export const home = async (req: Request, res: Response)=>{
8
9      //puxar os usuários que estão no meu banco de dados
10     let users = await User.findAll()
11
12     //exibir os usuários no console
13     console.log("USUÁRIOS: ",JSON.stringify(users))
14
15 }
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

Atualize a sua página no localhost:2000 e veja
se seus usuários do banco apareceram

[nodemon] starting `ts-node src/server.ts`

Executing (default): SELECT `id`, `name`, `age` FROM `users` AS `User`,`

USUÁRIOS: [{"id":1,"name":"Naruto","age":30},{ "id":2,"name":"Viviane","age":26},{ "id":3,"name":"Kira","age":17}]

Executing (default): SELECT `id`, `name`, `age` FROM `users` AS `User`,`

USUÁRIOS: [{"id":1,"name":"Naruto","age":30},{ "id":2,"name":"Viviane","age":26},{ "id":3,"name":"Kira","age":17}]


PARA EXIBIR OS USUÁRIOS NA TELA PRECISAMOS IR NA NOSSA VIEW>HOME.MUSTACHE

TS User.ts TS homeController.ts X home.mustache X

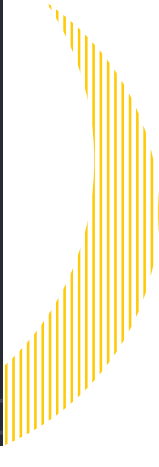
src > views > pages > home.mustache

```
1  {{>partials/header}}
2
3  Opa {{name}} {{lastName}}, tudo bem?
4  {{#showOld}}
5  |   Como vai a vida?
6  {{/showOld}}
7  {{^showOld}}
8  |   Beleza?
9  {{/showOld}}
10
11 <hr>
12
13 <h2>Usuários do banco </h2>
14
15 {{#users}}
16 |   <li>{{name}} - Idade: {{age}}</li>
17 {{/users}}
18
```

EM SEGUIDA VÁ EM homeController.ts E COLOQUE USER PARA RENDERIZAR NA TELA



```
TS User.ts    TS homeController.ts X    home.mustache
src > controllers > TS homeController.ts > ...
  export const home = async (req: Request, res: Response): Promise<void> => {
8
9     let users = await User.findAll()
10    console.log("USUÁRIOS: ", JSON.stringify(users))
11
12    let age: number = 90;
13    let showOld: boolean = false;
14
15    if(age > 50) {
16        showOld = true;
17    }
18
19    let list = Product.getAll();
20    let expensiveList = Product.getFromPriceAfter(12);
21
22    res.render('pages/home', {
23        name: 'Viviane',
24        lastName: 'de Lima',
25        showOld,
26        products: list,
27        expansives: expensiveList,
28        frasesDoDia: [],
29        users
30    });
31  };
```





Novo Título

Opa Viviane de Lima, tudo bem? Como vai a vida?

Usuários do banco

- Naruto - Idade: 30
- Viviane - Idade: 26
- Kira - Idade: 17

Produtos

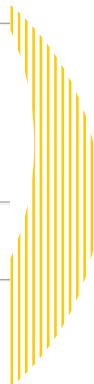
- Produto X - R\$ 10
- Produto Y - R\$ 15
- Produto W - R\$ 20
- Produto G - R\$ 5

Produtos a partir de R\$ 12

- Produto Y - R\$ 15
- Produto W - R\$ 20

Não há frases motivacionais hoje.

Todos os direitos reservados



CRIE UM NOVO USUÁRIO NA TABELA USER NO DBEAVER E VEJA SE ATUALIZA AUTOMATICAMENTE

Properties Data ER Diagram			
users Enter a SQL expression to filter results (use Ctrl+Space)			
Grid	123 id	ABC name	123 age
	1	Naruto	30
Text	2	Viviane	26
	3	Kira	17
	4	Ikki	28

Novo Título

Opa Viviane de Lima, tudo bem? Como vai a vida?

Usuários do banco

- Naruto - Idade: 30
- Viviane - Idade: 26
- Kira - Idade: 17
- Ikki - Idade: 28

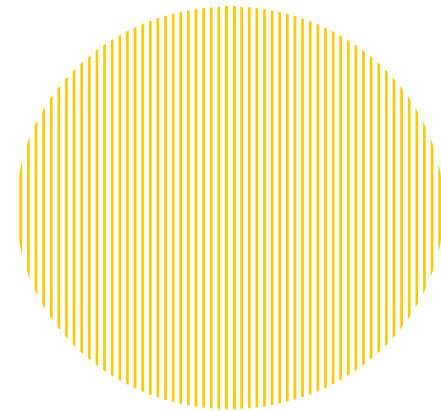
Produtos

- Produto X - R\$ 10
- Produto Y - R\$ 15
- Produto W - R\$ 20
- Produto G - R\$ 5

Produtos a partir de R\$ 12



TIPOS DE CONSULTA



VAMOS PEGAR DADOS ESPECÍFICOS DO USUÁRIO



```
TS homeController.ts X home.mustache

src > controllers > TS homeController.ts > [⌕] home > [⌕] expensiveList

1  import { Request, Response } from 'express';
2  import { Product } from '../models/Product';
3
4  //precisamos importar users
5  import { User } from '../models/User';
6
7  export const home = async (req: Request, res: Response)=>{
8
9      let users = await User.findAll({
10         //pegando apenas o nome do usuário
11         attributes: ['name']
12     })
13
14
15     let age: number = 90;
16     let showOld: boolean = false;
17
18     if(age > 50) {
19         showOld = true;
20     }
21
```


EXCLUINDO UM DADO ESPECÍFICO DO USUÁRIO

```
TS homeController.ts X home.mustache

src > controllers > TS homeController.ts > [e] home

1  import { Request, Response } from 'express';
2  import { Product } from '../models/Product';
3
4  //precisamos importar users
5  import { User } from '../models/User';
6
7  export const home = async (req: Request, res: Response)=>{
8
9      let users = await User.findAll({
10         //excluindo uma informação específica
11         attributes: {exclude: ['name']}
12     })
13
14
15     let age: number = 90;
16     let showOld: boolean = false;
17
```

FILTRAR RESULTADOS ESPECÍFICOS

TS homeController.ts X

home.mustache

src > controllers > TS homeController.ts > [🔍] home

```
1  import { Request, Response } from 'express';
2  import { Product } from '../models/Product';
3  import { User } from '../models/User';
4
5  export const home = async (req: Request, res: Response)=>{
6
7      //encontrando nome Viviane no meu banco
8      let users = await User.findAll({
9          where: {name: 'Viviane'}
10     })
11
12
13     let age: number = 90;
14     let showOld: boolean = false;
15
16     if(age > 50) {
17         showOld = true;
18     }
```

FILTRAR RESULTADOS COM CONDIÇÕES

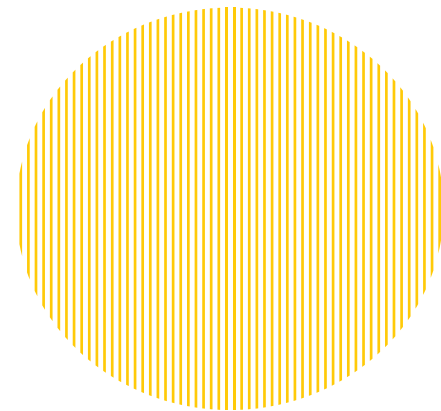
TS homeController.ts X home.mustache

src > controllers > TS homeController.ts > [🔍] home

```
1  import { Request, Response } from 'express';
2  import { Product } from '../models/Product';
3  import { User } from '../models/User';
4
5  //importando o OP (para fazer operações) do sequelize
6  import {Op} from 'sequelize'
7
8  export const home = async (req: Request, res: Response)=>{
9
10     let users = await User.findAll({
11         //quero saber no meu banco quem tem 30 ou 17 anos
12         where: {
13             [Op.or]: [
14                 {age: 30},
15                 {age: 17}
16             ]
17         }
18     })
19
```



TIPOS DE CONSULTA



FILTRANDO USUÁRIOS POR IDADE

TS homeController.ts ✕

home.mustache

src > controllers > TS homeController.ts > [🔗] home

```
4
5 //importando o OP (para fazer operações) do sequelize
6 import {Op} from 'sequelize'
7
8 export const home = async (req: Request, res: Response)=>{
9
10     let users = await User.findAll({
11         where: {
12             age:{
13                 [Op.gt]: 20, // > 20 (GT significa greater than)
14                 [Op.gte]:20, //>= 20 (GT significa greater than or equal)
15             }
16         }
17     })
18
19     let age: number = 90;
20     let showOld: boolean = false;
21
22     if(age > 50) {
23         showOld = true;
```

FILTRANDO USUÁRIOS POR IDADE

TS homeController.ts X

home.mustache

src > controllers > TS homeController.ts > [🔍] home > [🔍] users > 🔑 where > 🔑 age

```
4
5 //importando o OP (para fazer operações) do sequelize
6 import {Op} from 'sequelize'
7
8 export const home = async (req: Request, res: Response)=>{
9
10     let users = await User.findAll({
11         where: {
12             age: {
13                 [Op.gt]: 20,
14                 [Op.gte]: 20,
15                 [Op.lt]: 20, // <20
16                 [Op.lte]: 20 // <=20
17             }
18         }
19     })
20
21     let age: number = 90;
22     let showOld: boolean = false;
23
24     if (age > 50) {
```

FILTRANDO USUÁRIOS ENTRE 10 ANOS E 30 ANOS

TS homeController.ts X

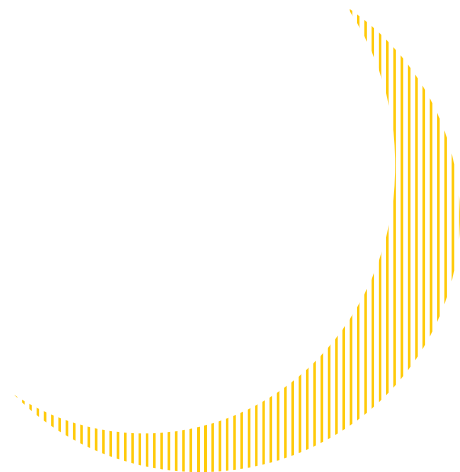
home.mustache

src > controllers > TS homeController.ts > [x] home > [x] users > 🔑 where > 🔑 age > 🔑 [Op.between]

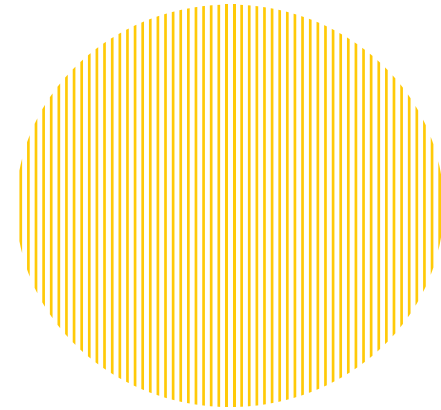
```
4
5 //importando o OP (para fazer operações) do sequelize
6 import {Op} from 'sequelize'
7
8 export const home = async (req: Request, res: Response)=>{
9
10     let users = await User.findAll({
11         where: {
12             age:{
13                 [Op.between]: [10,30]
14             }
15         }
16     })
17
18     let age: number = 90;
19     let showOld: boolean = false;
20
21     if(age > 50) {
22         showOld = true;
23     }
```



[Op.in]
[Op.notIn]
[Op.like] : '%a'



INSERINDO DADOS COM SEQUELIZE



APAGUE O TRECHO DE CÓDIGO DE USERS


```
TS homeController.ts X TS userController.ts home.mustache
src > controllers > TS homeController.ts > [🔍] home
1  import { Request, Response } from 'express';
2  import { Product } from '../models/Product';
3  import { User } from '../models/User';
4
5  //importando o OP (para fazer operações) do sequelize
6  import { Op } from 'sequelize'
7
8  export const home = async (req: Request, res: Response) => {
9
10   ⚡ let users = await User.findAll({
11     ... where: {
12       ... age: {
13         ... [Op.between]: [10, 30]
14       }
15     }
16   })
17
18   let age: number = 90;
19   let showOld: boolean = false;
20
21   if (age > 50) {
22     showOld = true;
23   }
```

DELETE O USERS TAMBÉM

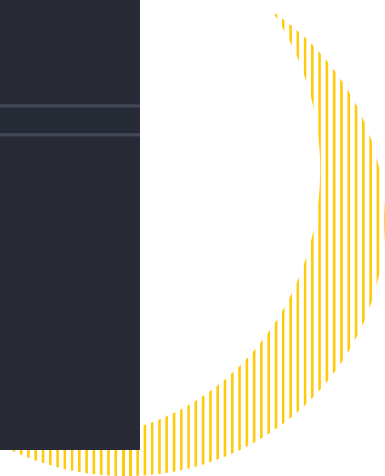
```
TS homeController.ts 1 X TS userController.ts home.mustache
src > controllers > TS homeController.ts > [e] home
1  import { Request, Response } from 'express';
2  import { Product } from '../models/Product';
3  import { User } from '../models/User';
4
5  //importando o OP (para fazer operações) do sequelize
6  import { Op } from 'sequelize'
7
8  export const home = async (req: Request, res: Response) => {
9
10     let age: number = 90;
11     let showOld: boolean = false;
12
13     if (age > 50) {
14         showOld = true;
15     }
16
17     let list = Product.getAll();
18     let expensiveList = Product.getFromPriceAfter(12);
19
20     res.render('pages/home', {
21         name: 'Viviane',
22         lastName: 'de Lima',
23         showOld,
24         products: list,
25         expansives: expensiveList,
26         frasesDoDia: [],
27         users
28     });
29 }
```

EXISTEM DUAS FORMAS DE INSERIR DADOS USANDO O SEQUELIZE

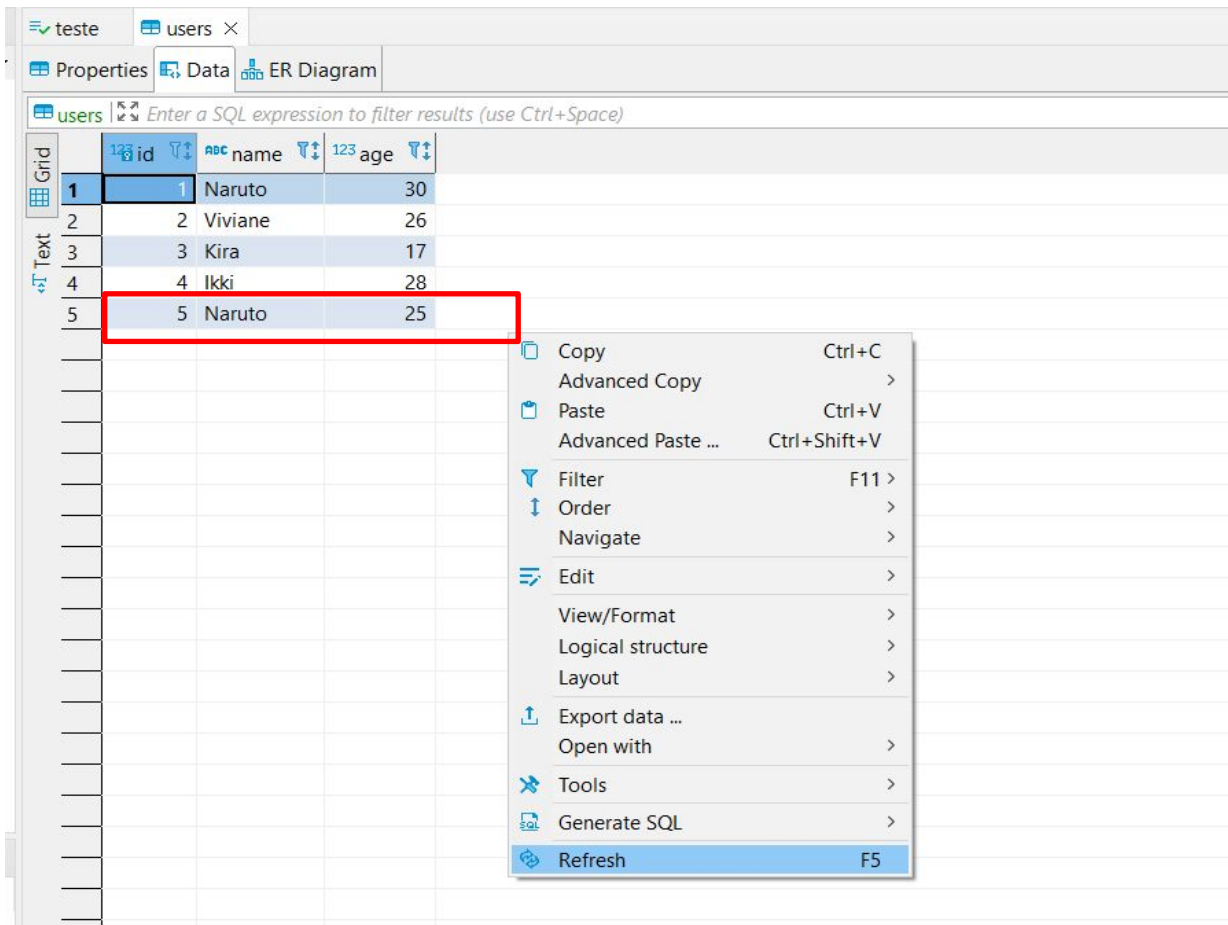
1 FORMA: BUILD + SAVE



```
TS homeController.ts X .env TS userController.ts home.mustache
src > controllers > TS homeController.ts > [e] home > [e] user
1  import { Request, Response } from 'express';
2  import { Product } from '../models/Product';
3  import { User } from '../models/User';
4
5  //importando o OP (para fazer operações) do sequelize
6  import { Op } from 'sequelize'
7
8  export const home = async (req: Request, res: Response) => {
9
10     /* 1 FORMA BUILD+SAVE
11     criamos uma constante user e colocamos User que é o nosso model
12     dentro de build colocamos as informações que eu quero inserir */
13     const user = User.build({
14         //não vamos inserir o ID pois ele é inserido automaticamente
15         name: 'Naruto',
16         age: 25
17     });
18     //agora precisamos salvar essas informações com await
19     await user.save();
20
21     let age: number = 90;
22     let showOld: boolean = false;
23
24     if (age > 50) {
25         showOld = true;
26     }
27
```



FEITO ISSO, ABRA O NAVEGADOR E ATUALIZE A PÁGINA E EM SEGUIDA ATUALIZE O DBEAVER E VEJA SE O USUÁRIO APARECE NA TABELA USERS



teste users ×

Properties Data ER Diagram



users Enter a SQL expression to filter results (use Ctrl+Space)

	id	name	age
1	1	Naruto	30
2	2	Viviane	26
3	3	Kira	17
4	4	Ikki	28
5	5	Naruto	25

- Copy Ctrl+C
- Advanced Copy >
- Paste Ctrl+V
- Advanced Paste ... Ctrl+Shift+V
- Filter F11 >
- Order >
- Navigate >
- Edit >
- View/Format >
- Logical structure >
- Layout >
- Export data ... >
- Open with >
- Tools >
- Generate SQL >
- Refresh F5

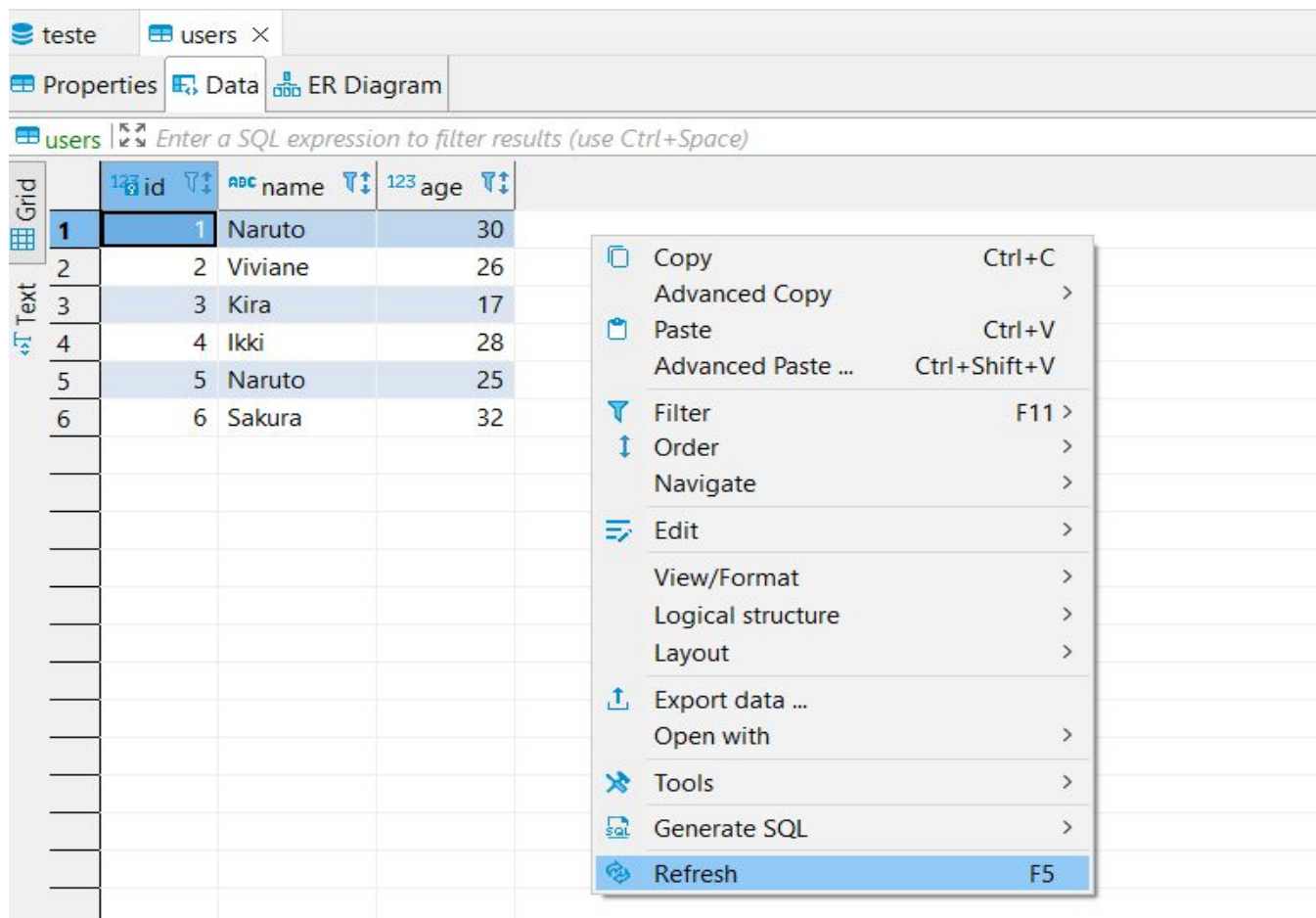
EXISTEM DUAS FORMAS DE INSERIR DADOS USANDO O SEQUELIZE

2 FORMA:CREATE



```
TS homeController.ts X .env TS userController.ts home.mustache
src > controllers > TS homeController.ts > [e] home
1  import { Request, Response } from 'express';
2  import { Product } from '../models/Product';
3  import { User } from '../models/User';
4
5  //importando o OP (para fazer operações) do sequelize
6  import { Op } from 'sequelize'
7
8  export const home = async (req: Request, res: Response)=>{
9
10     /* 1 FORMA BUILD+SAVE
11     const user = User.build({
12
13         name:'Naruto',
14         age: 25
15     })
16     */
17
18     //deixe o await comentando para ele não ficar criando usuário toda hora
19     //await user.save()
20
21     /* 2 FORMA CREATE*
22     essa forma é mais simples, apenas com esse comando, sem o await.user.save ele já
23     insere o usuário no banco de dados */
24     const user = await User.create({
25         name:'Sakura',
26         age:32
27     })
28 }
```

FEITO ISSO, ABRA O NAVEGADOR E ATUALIZE A PÁGINA E EM SEGUIDA ATUALIZE O DBEAVER E VEJA SE O USUÁRIO APARECE NA TABELA USERS



The screenshot shows the DBeaver interface with the 'users' table selected. The table has columns 'id', 'name', and 'age'. The first row (id=1, name=Naruto, age=30) is highlighted. A context menu is open over this row, showing various actions. The 'Refresh' option at the bottom is highlighted in blue.

	id	name	age
1	1	Naruto	30
2	2	Viviane	26
3	3	Kira	17
4	4	Ikki	28
5	5	Naruto	25
6	6	Sakura	32

- Copy Ctrl+C
- Advanced Copy >
- Paste Ctrl+V
- Advanced Paste ... Ctrl+Shift+V
- Filter F11 >
- Order >
- Navigate >
- Edit >
- View/Format >
- Logical structure >
- Layout >
- Export data ... >
- Open with >
- Tools >
- Generate SQL >
- Refresh F5**

EXERCÍCIO: CRIAR USUÁRIOS NOVOS



- Para o exercício e acesse o repositório e faça download:

github.com/zennom/exercicio-nodejs

- Dicas:
 1. Crie o formulário em home.mustache
 2. Em homeController.ts pegue o nome e a idade
 3. Crie a rota dessa requisição em index.ts



EXERCÍCIO: CRIAR USUÁRIOS NOVOS



Novo Título

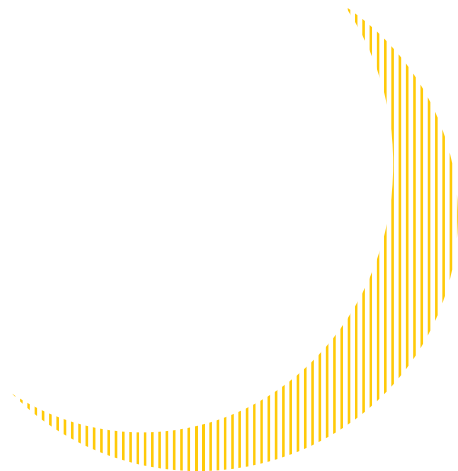
Opa Naruto Uzumaki, tudo bem? Como vai a vida?

Novo Usuário

Crie um formulário funcional, quando o usuário clicar no botão inserir usuário, deve aparecer o usuário abaixo do formulário

Usuários do banco

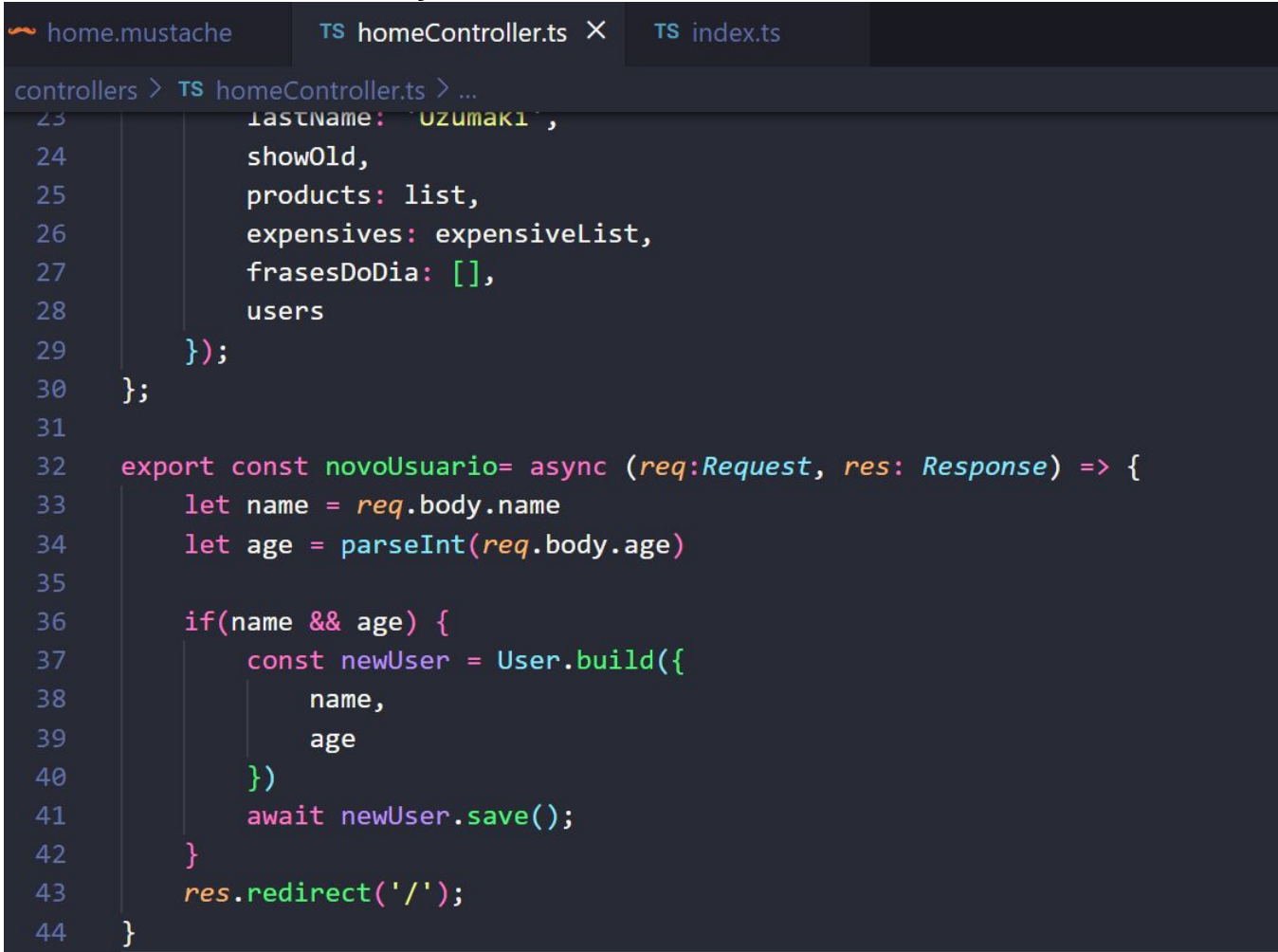
- Naruto - Idade: 30
- Viviane - Idade: 26
- Kira - Idade: 17
- Ikki - Idade: 28
- Kakashi - Idade: 25
- Sakura - Idade: 32
- Izuku Midoriya - Idade: 16



RESOLUÇÃO 1. FORMULÁRIO

```
home.mustache X TS homeController.ts TS index.ts
views > pages > home.mustache
4  {{#showOld}}
5      Como vai a vida?
6  {{/showOld}}
7  {{^showOld}}
8      Beleza?
9  {{/showOld}}
10
11  <hr/>
12
13  <fieldset>
14      <legend>Novo Usuário</legend>
15      <form method="POST" action="/novo_usuario">
16          <input type="text" name="name" placeholder="Digite um nome" /><br/><br/>
17          <input type="text" name="age" placeholder="Digite uma idade" /><br/><br/>
18          <input type="submit" value="Inserir Usuário" />
19      </form>
20  </fieldset>
21
22  <h2>Usuários do banco</h2>
23  <ul>
```

RESOLUÇÃO 2.HOMECONTROLLER



The image shows a VS Code editor window with the file 'homeController.ts' open. The editor has a dark theme. The code is written in TypeScript and defines a controller for a home page. It includes a route handler for GET requests and an async function for creating a new user. The code is as follows:

```
home.mustache TS homeController.ts X TS index.ts
controllers > TS homeController.ts > ...
23     lastName: 'Uzumaki',
24     showOld,
25     products: list,
26     expensives: expensiveList,
27     frasesDoDia: [],
28     users
29   });
30 };
31
32 export const novoUsuario= async (req:Request, res: Response) => {
33   let name = req.body.name
34   let age = parseInt(req.body.age)
35
36   if(name && age) {
37     const newUser = User.build({
38       name,
39       age
40     })
41     await newUser.save();
42   }
43   res.redirect('/');
44 }
```

RESOLUÇÃO 3.INDEX.TS

home.mustache

TS homeController.ts

TS index.ts



routes > TS index.ts > ...

```
1  import { Router } from 'express';
2
3  import * as HomeController from '../controllers/homeController';
4  import * as InfoController from '../controllers/infoController';
5  import * as UserController from '../controllers/userController';
6
7  const router = Router();
8
9  router.get('/', HomeController.home);
10
11  // criar a rota para novoUsuario
12  router.post('/novoUsuario', HomeController.novoUsuario);
13
14  router.get('/contato', InfoController.contato);
15  router.get('/sobre', InfoController.sobre);
16
17  router.get('/nome', UserController.nome);
18  router.get('/idade', UserController.idadeForm);
19  router.post('/idade-resultado', UserController.idadeAction);
20
21  export default router;
```

