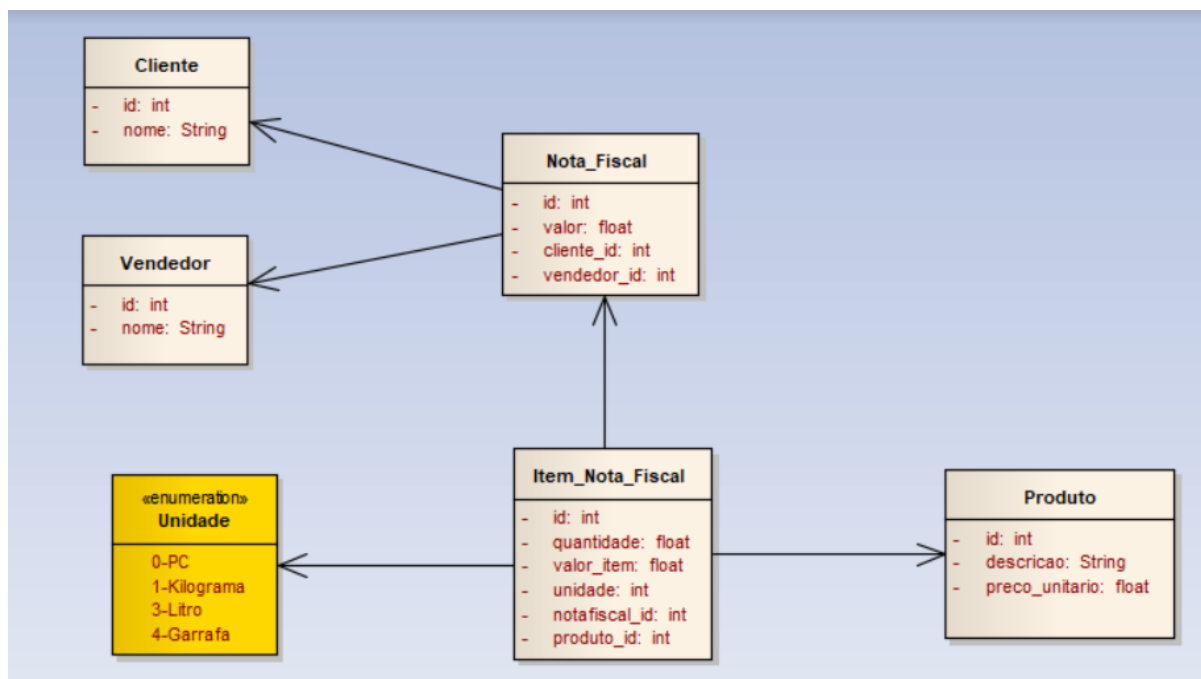


ATIVIDADE 07 - BANCO DE DADOS

Criar uma aplicação Node.js que usa Funcionalidades CRUD (insert, select, update, delete) usando REST API. A Aplicação que usa um Banco de Dados SQLite3 usará o Modelo Entidade Relacionamento anexo. Evidenciar o teste das chamadas das API usando o PostMan.



1. Tabela CLIENTES

GET

localhost:3000/clientes

Params

Authorization

Headers (9)

Body

Pre-request Script

Tests

Settings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

	Key	Value
<input type="checkbox"/>	NOME_CLI	Leticia
<input type="checkbox"/>	NOME_CLI	Julia
<input type="checkbox"/>	NOME_CLI	Sure
<input checked="" type="checkbox"/>	NOME_CLI	Taveira
<input type="checkbox"/>		
	Key	Value

Body

Cookies

Headers (7)

Test Results

Pretty

Raw

Preview

Visualize

JSON

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

"NOME_CLI": "Anderson"

}

{

"ID": 6,

"NOME_CLI": "Abner"

}

{

"ID": 7,

"NOME_CLI": "Camila"

}

{

"ID": 9,

"NOME_CLI": "Denise"

}

{

"ID": 10,

"NOME_CLI": "Daniel"

}

{

"ID": 11,

"NOME_CLI": "Taveira"

2. Tabela **VENDEDORES**

HTTP localhost:3000/vendedor

GET localhost:3000/vendedor

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

none form-data **x-www-form-urlencoded** raw binary GraphQL

	Key	Value	Description
<input checked="" type="checkbox"/>	NOME_VEND	Matias Ítalo Damasceno	
	Key	Value	Description

Body Cookies Headers (7) Test Results 200 OK

Pretty Raw Preview Visualize JSON

```
1  [
2    "vendedor": [
3      {
4        "ID": 2,
5        "NOME_VEND": "Luciano Arruda Cavalcante"
6      },
7      {
8        "ID": 3,
9        "NOME_VEND": "Joana Alves Pessoa"
10     },
11     {
12       "ID": 4,
13       "NOME_VEND": "Mercia Bessa Santos"
14     },
15     {
16       "ID": 5,
17       "NOME_VEND": "Antonio de Padua Lopes"
18     },
19     {
20       "ID": 6,
21       "NOME_VEND": "Matias Ítalo Damasceno"
22     }
23   ]
24 ]
```

3. Tabela **NOTA FISCAL**

GET Get Cliente

POST Post Cliente

POST Post NotaFiscal

POST Post Produto

GET Get Produto

localhost:3000/notafiscal

GET

localhost:3000/notafiscal

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

☒

CLIENTE_ID

Key

Body

Cookies

Headers (7)

Test Results

Pretty

Raw

Preview

Visualize

JSON

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

"notafiscal": [

{

"ID": 1,

"VALOR": 15,

"CLIENTE_ID": 1,

"VENDEDOR_ID": 2

},

{

"ID": 2,

"VALOR": 150,

"CLIENTE_ID": 4,

"VENDEDOR_ID": 3

},

{

"ID": 3,

"VALOR": 75,

"CLIENTE_ID": 3,

"VENDEDOR_ID": 4

},

{

"ID": 4,

"VALOR": 98,

"CLIENTE_ID": 5,

"VENDEDOR_ID": 6

},

{

"ID": 5,

"VALOR": 1800,

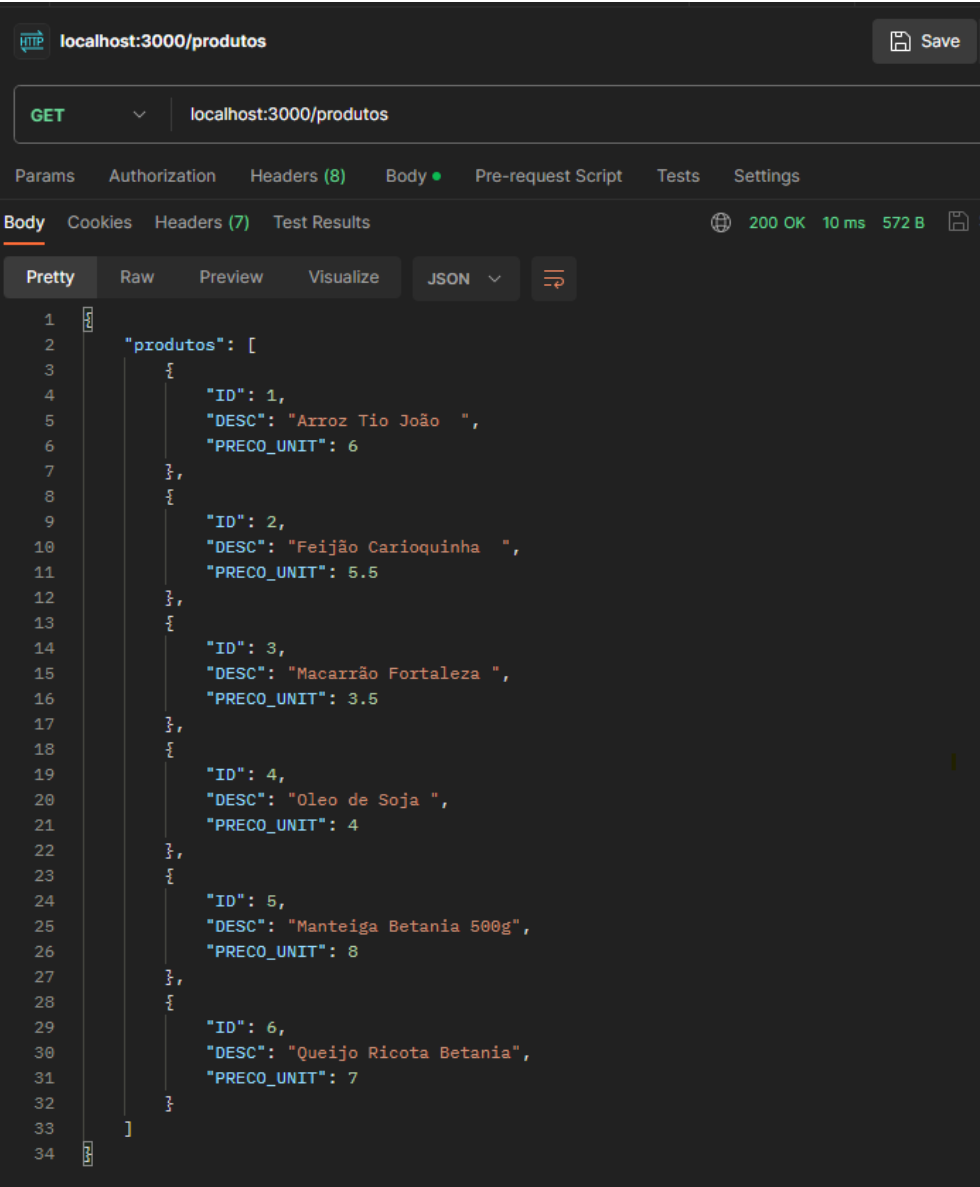
"CLIENTE_ID": 4,

"VENDEDOR_ID": 6

}

]

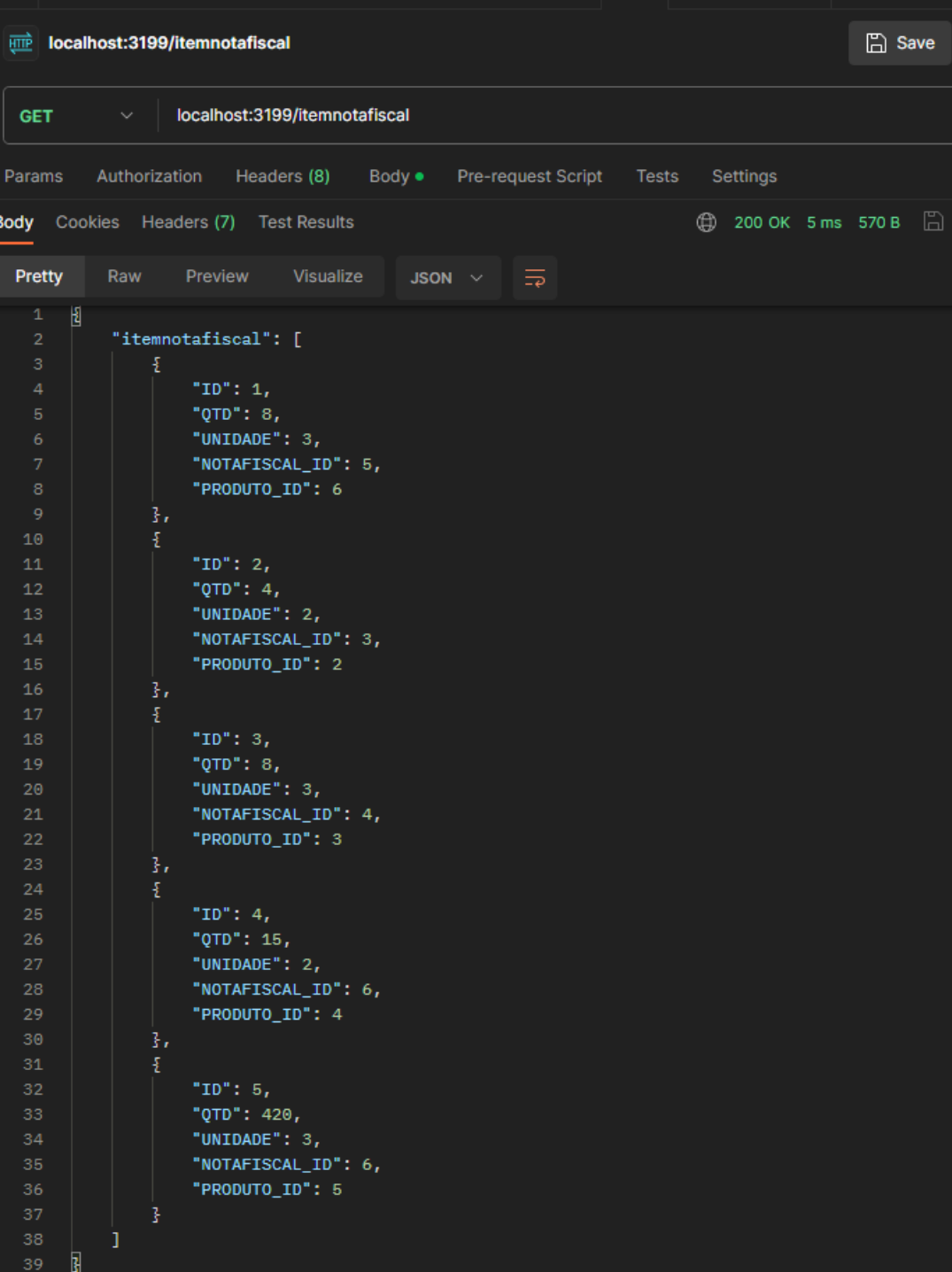
4. Tabela **PRODUTOS**



The screenshot displays a REST client interface for a GET request to `localhost:3000/produtos`. The response is a JSON array of product data, showing 6 items. The interface includes tabs for Params, Authorization, Headers (8), Body, Pre-request Script, Tests, and Settings. The Body tab is selected, showing the JSON response in a Pretty format. The status bar indicates a 200 OK response with a 10 ms latency and 572 B of data.

```
1  [
2    "produtos": [
3      {
4        "ID": 1,
5        "DESC": "Arroz Tio João ",
6        "PRECO_UNIT": 6
7      },
8      {
9        "ID": 2,
10       "DESC": "Feijão Cariquinha ",
11       "PRECO_UNIT": 5.5
12     },
13     {
14       "ID": 3,
15       "DESC": "Macarrão Fortaleza ",
16       "PRECO_UNIT": 3.5
17     },
18     {
19       "ID": 4,
20       "DESC": "Oleo de Soja ",
21       "PRECO_UNIT": 4
22     },
23     {
24       "ID": 5,
25       "DESC": "Manteiga Betania 500g",
26       "PRECO_UNIT": 8
27     },
28     {
29       "ID": 6,
30       "DESC": "Queijo Ricota Betania",
31       "PRECO_UNIT": 7
32     }
33   ]
34 ]
```

5. Tabela ITEM NOTA FISCAL



The screenshot displays a REST client interface with a GET request to `localhost:3199/itemnotafiscal`. The response is a JSON array of 5 items, each with fields `ID`, `QTD`, `UNIDADE`, `NOTAFISCAL_ID`, and `PRODUTO_ID`.

```
1  [
2    "itemnotafiscal": [
3      {
4        "ID": 1,
5        "QTD": 8,
6        "UNIDADE": 3,
7        "NOTAFISCAL_ID": 5,
8        "PRODUTO_ID": 6
9      },
10     {
11       "ID": 2,
12       "QTD": 4,
13       "UNIDADE": 2,
14       "NOTAFISCAL_ID": 3,
15       "PRODUTO_ID": 2
16     },
17     {
18       "ID": 3,
19       "QTD": 8,
20       "UNIDADE": 3,
21       "NOTAFISCAL_ID": 4,
22       "PRODUTO_ID": 3
23     },
24     {
25       "ID": 4,
26       "QTD": 15,
27       "UNIDADE": 2,
28       "NOTAFISCAL_ID": 6,
29       "PRODUTO_ID": 4
30     },
31     {
32       "ID": 5,
33       "QTD": 420,
34       "UNIDADE": 3,
35       "NOTAFISCAL_ID": 6,
36       "PRODUTO_ID": 5
37     }
38   ]
39 ]
```