


Desafio Pulse 1308

 <https://github.com/gm-pulse/test-pulse-1308/blob/master/avaliacao/prova-1308.pdf>

Domínios

Estoque:

- Integer id;
- Produto produto;
- Filial filial;
- Double preco;
- Double qtde;

Pedido:

- Long id;
- Usuario usuario;
- Filial filial;
- Cliente cliente;
- FormaPagamento formaPagamento;
- Date dataSaida;
- String Observacao;
- String tipo;
- Double totalPedido;

PedidoItem:

- Long id;
- Pedido pedido;
- Produto produto;
- Double quantidade;
- Double valor;
- Double total;
- String status;

Produto:

- Integer id;
- String descricao;
- String codigo;

FormaPagamento:

- Integer id;
- String descricao;

Filial:

- Integer id;
- String razao;
- String fantasia;

Usuario:

- Integer id;
- String nome;
- String username;
- String senha;
- Filial filial;

Cliente:

- Integer id;
- String nome;
- String cpf;
- String celular;
- String cep;
- String endereco;
- String complemento;
- String bairro;
- String cidade;
- String uf;

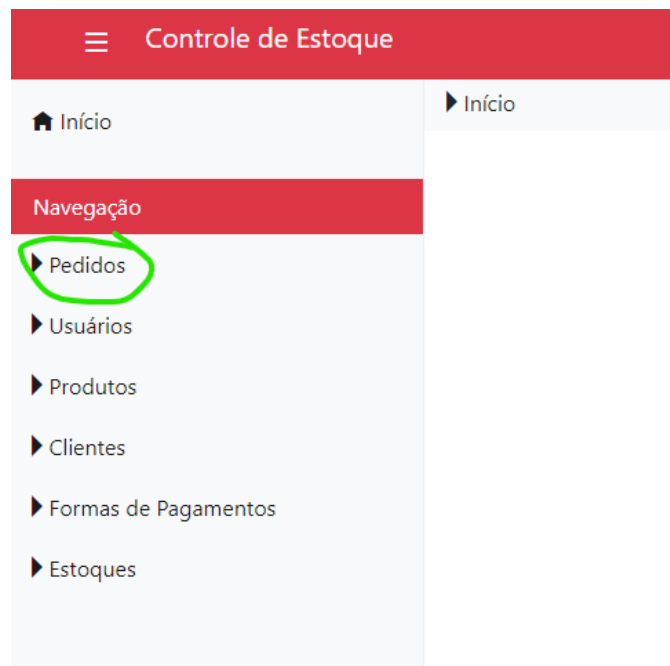
Configuração do Banco de Dados

adequações necessárias no arquivo `src/main/resources/application.properties`

```
spring.datasource.url=jdbc:mysql://localhost:3306/[NOME DO BANCO]
spring.datasource.username=[USUARIO DO BANCO]
spring.datasource.password=[SENHA DO BANCO]
spring.datasource.driver-class-name=com.mysql.jdbc.Driver
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect
spring.jpa.show-sql = true
spring.jpa.hibernate.ddl-auto=update
spring.jpa.properties.hibernate.check_nullability=true
```

Tela para operações básicas

Selecionar no menu lateral o item `Pedidos`



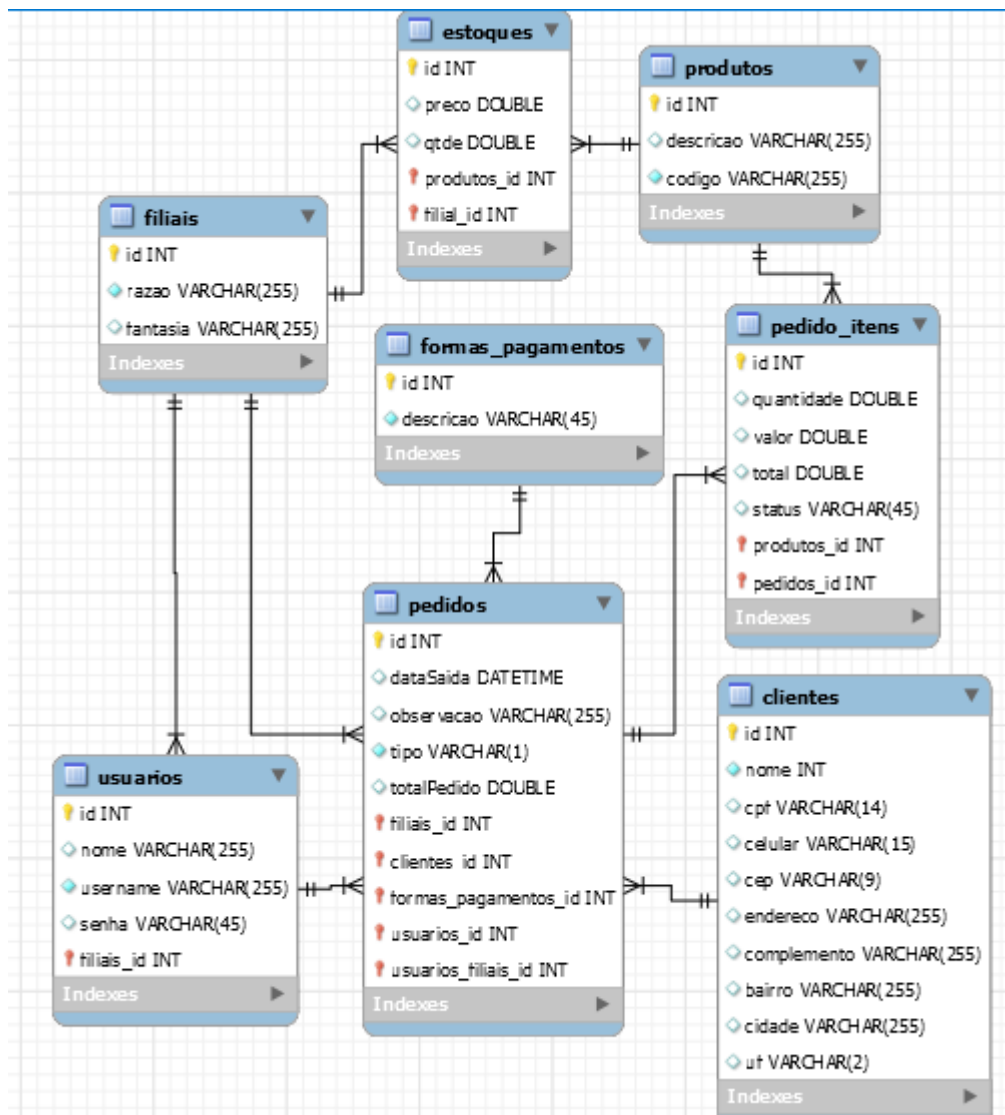
Registrar um pedido na filial desejada

Lista de Filiais					+ Cadastrar Filial
#	Razão Social	Nome Fantasia	Endereço	Opções	
1	Filial 1	Filial 1		  	
2	Filial 2	Filial 2		  	
3	Filial 3	Filial 3		  	

Obs: já existe um Seeder inicial para entrada de testes e operações básicas

Competências de bancos de dados

- Diagrama Entidade Relacionamento(DER)



- Escrever consulta que retorne todos os produtos com quantidade maior ou igual a 100

```

select
    produtos.descricao as produto,
    filiais.razao as filial,
    estoques.qtde
from estoques
    inner join produtos on estoques.produto_id = produtos.id
    inner join filiais on filiais.id = estoques.filial_id
where estoques.qtde >= 100;
  
```

- Escrever consulta que traga todos os produtos que têm estoque para a filial de código 60

```
select
  produtos.descricao as produto,
  filiais.razao as filial,
  estoques.qtde
from estoques
  inner join produtos on estoques.produto_id = produtos.id
  inner join filiais on filiais.id = estoques.filial_id
where filiais.id=60
and estoques.qtde > 0;
```

- Escrever uma consulta que liste todos os campos para o domínio PedidoEstoque e ItensPedido filtrando apenas o produto de código 7993.

```
select p.*, pi.* from pedido_estoque p
inner join itens_pedido pi on p.id = pi.pedido_id
where pi.produto_id = 7993;
```

- Escrever um Consulta que liste os pedidos com suas respectivas formas de pagamento

```
select * from pedidos p
inner join formas_pagamentos f on f.id = p.forma_pagamento_id;
```

- Escreva uma consulta para sumarizar e bater os valores da capa do pedido com os valores dos itens de pedidos

	id	tipo	valor_total_capa	valor_total_itens
	34	E	1475	1475
	38	S	29	29

```
select
  p.id,
  p.tipo,
  p.total_pedido valor_total_capa
```

```
,SUM(pi.total) valor_total_itens  
from pedidos p  
inner join pedido_itens pi on pi.pedido_id = p.id  
group by p.id;
```

- Escreva uma consulta para sumarizar o total dos itens por pedido e que filtre apenas os pedidos no qual a soma total da quantidade de itens de pedido seja maior que 10

```
select  
  p.id, p.tipo ,SUM(pi.quantidade) quantidade_itens  
from pedidos p  
inner join pedido_itens pi on pi.pedido_id = p.id  
group by p.id  
having SUM(pi.quantidade) > 10;
```