





PARTE 1:

1) Criar no SQL Server uma database com a seguinte tabela e fazer uma procedure que permita fazer o insert, update e delete dos dados na tabela, a partir de um código de operação (I, U ou D):

Produto	
 Codigo	integer(10)
 Nome	varchar(30)
 Valor_Unitario	decimal(7, 2)
 Qtd_Estoque	integer(10)

2) Se não estiver adicionado, baixar o Tomcat 9.0 e adicionar ao Eclipse

3) Criar um novo Dynamic Web Project para fazer o CRUD de Produto

- Criar uma classe model Filme com os 5 atributos propostos, seus Getters e Setters e a sobrescrita do toString();
- Criar uma classe persistence GenericDao com a conexão com o SQLServer;
- Criar uma interface persistence IProdutoDao com as assinaturas dos métodos para insert, update, delete, selectOne e selectAll;
- Criar uma classe persistence ProdutoDao que implementa a interface IProdutoDao e implementa as operações com PreparedStatement e CallableStatement;
- Criar um controller Servlet ProdutoServlet que receba os parâmetros de um JSP e permita cadastrar, atualizar, excluir, consultar um filme e consultar a lista de filmes;
- Criar um view produto.jsp com os campos para cada atributo do filme e os botões para acessar cada operação. O JSP deve implementar o JSTL para receber os atributos que retornam do Servlet para exibi-los em tela, como saída, erros, objeto Produto e a lista de Produto;
- A database deve ser incrementada:
 - Fazer uma Scalar Function que verifique, na tabela produtos (codigo, nome, valor unitário e qtd estoque) quantos produtos estão com estoque abaixo de um valor de entrada (O valor mínimo deve ser parâmetro de entrada)
 - Fazer uma Multi Statement Table Function que liste o código, o nome e a quantidade dos produtos que estão com o estoque abaixo de um valor de entrada (O valor mínimo deve ser parâmetro de entrada)
- A interface IProdutoDao deve ter mais 2 assinaturas de métodos:
 - public int consultaQtdForaEstoque(int qtdMinima) throws SQLException;
 - public List<Produto> consultaListaProdutosForaEstoque (int qtdMinima) throws SQLException;
- A classe ProdutoDao deve implementar os novos métodos oferecendo os retornos
- Fazer um novo servlet EstoqueServlet com 2 atributos de retorno (Considerar retorno de erro também):
 - int qtdForaEstoque
 - List<Produto> listaProdutosForaEstoque
- O jsp produtoestoque.jsp deve ter uma div com um form e uma table:

produto.jsp

MENU

Código

Nome

Valor Unitário

Qtd Estoque

produtoestoque.jsp

MENU

Mínimo

Qtd

Lista

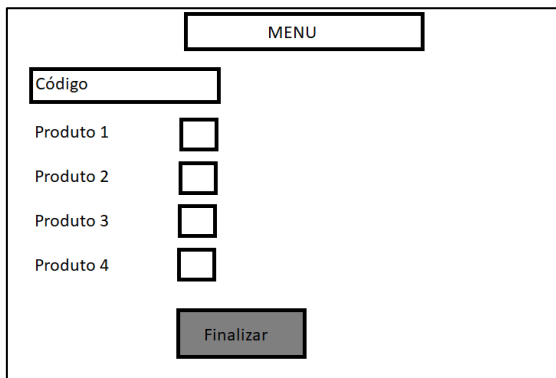
PARTE 2:

Criar a tabela venda

Venda	
 Codigo_Venda	integer(10)
 Data_Compra	date
 Codigo_Produto	integer(10)
 Quantidade	integer(10)

Criar uma trigger after que, se a quantidade em estoque do produto for igual a zero, não permitir o INSERT

Criar, no Java com Servlets, uma página venda.jsp que liste todos os produtos cadastrados e permita selecionar a quantidade de cada produto desejado e fazer o insert de cada produto na tabela venda.



A dica para montar o jsp é:

Receber uma Lista de produtos, fazer o foreach do JSTL colocando no input text o id e o nome da PK do produto:

```
<input type="number" id="${produto.codigo }" name="${produto.codigo }">
```

Para o servlet capturar cada produto, ao invés de definir o nome do parâmetro que será capturado, trabalharemos com HashMap

```
Map<String, String[]> parameters = request.getParameterMap();
```

A operação `parameters.keySet()` vai retornar as chaves (name) dos produtos que estavam na tela. Se for feito como definido acima, serão os códigos dos produtos

A operação `parameters.get(key)`, onde `key` é o código do produto que vem da operação acima, retorna o valor digitado no campo.

Nessa parte, ainda não é necessário ter uma tela para visualizar as vendas.

Exemplo de uso do HashMap:

```
Map<String, String[]> parameters = request.getParameterMap();
for(String key : parameters.keySet()) {
    String[] values = parameters.get(key);
    //mais programação
}
}
```