Pró-reitora Acadêmica
Escola de Educação, Tecnologia e Comunicação
Curso de Bacharelado em Engenharia de Software
Trabalho da Disciplina de Modelagem de Banco de Dados

Gerenciador de estoque

Autores: Felipe Farias Marinho
João Pedro Tavares Teixeira
João Marcos Azevedo Cruz
Paulo Eduardo Teixeira Barbosa
João Victor Martins Albernaz

Orientador: Prof. Ranyelson Neres Carvalho

Brasília - DF
2024

Felipe Farias Marinho
João Pedro Tavares Teixeira
João Marcos Azevedo Cruz
Paulo Eduardo Teixeira Barbosa
João Victor Martins Albernaz

Gerenciador de estoque

Documento apresentado ao Curso de graduação de Bacharelado em Engenharia de Software, da Universidade Católica de Brasília, como requisito parcial para obtenção da aprovação na disciplina de Laboratório Banco de Dados.

Orientador: Prof. Ranyelson Neres Carvalho

Brasília
2024

1 INTRODUÇÃO

O Gerenciador de Estoque é um sistema completo desenvolvido em Java com integração a um banco de dados relacional utilizando SQL, projetado para oferecer uma gestão eficiente e organizada de estoques em empresas de diferentes portes. O sistema permite o cadastro, edição, exclusão e consulta de produtos e categorias, possibilitando o controle detalhado de informações como preços, descrições e quantidade de estoque. Também é possível registrar entradas e saídas de mercadorias, mantendo um histórico preciso das movimentações.

Além disso, o sistema inclui relatórios analíticos que auxiliam na visualização de dados sobre vendas, lucros e produtos com baixo estoque, fornecendo informações essenciais para a tomada de decisões estratégicas. Com a utilização de procedures e triggers no banco de dados, o sistema automatiza operações críticas e assegura a integridade das informações. A combinação de uma interface intuitiva com recursos avançados torna este sistema uma solução moderna e confiável para atender às demandas de gestão empresarial.

2 BANCO DE DADOS

2.1 Tabelas

1. Tabela Categoria

- **Objetivo**: Armazena as categorias de produtos.
- **Atributos**:
  - id_categoria: Identificador único (chave primária).
  - nome: Nome da categoria.
  - descricao: Descrição detalhada da categoria.

2. Tabela Produto

- **Objetivo**: Armazena informações sobre os produtos.
- **Atributos**:
  - id_produto: Identificador único (chave primária).
  - nome: Nome do produto.
  - quantidade_estoque: Quantidade disponível no estoque.
  - preco_compra: Preço de compra.
  - preco_venda: Preço de venda.
  - descricao: Descrição do produto.
  - id_categoria: Chave estrangeira para a tabela Categoria.
- **Restrições**:
  - Não permite valores negativos para quantidade_estoque, preco_compra e preco_venda.
  - Índice na coluna id_categoria para otimizar buscas.

3. Tabela MovimentacaoEstoque

- **Objetivo**: Registra as movimentações de estoque (entradas e saídas).
- **Atributos**:
  - id_movimentacao: Identificador único (chave primária).
  - id_produto: Identificador da tabela Produto.
  - quantidade: Quantidade movimentada.
  - tipo: Tipo da movimentação (entrada ou saída).
  - data: Data e hora da movimentação.
- **Restrições**:
  - Índice na coluna id_produto para facilitar consultas.

2.2 Procedures

1. Cadastro e Edição de Produtos e Categorias

- **Funções**:
  - cadastrarProduto, atualizarProduto, deletarProduto, cadastrarCategoria, atualizarCategoria, deletarCategoria: Operações CRUD para produtos e categorias.

2. Movimentação de Estoque (Entrada e Saída)

- **Funções**:
    - registrar_entrada_estoque: Aumenta a quantidade do produto no estoque.
    - registrar_saida_estoque: Diminui a quantidade do produto no estoque e verifica se há quantidade suficiente.

3. Consultas e Relatórios

- **Funções**:
    - consultar_produtos,consultar_categorias,
      relatorio_movimentacao_estoque,relatorio_vendas_lucro,
      relatorio_produtos_baixo_estoque: Permitem consultas e geração de relatórios sobre movimentações, vendas, lucros e produtos com baixo estoque.

2.3 Triggers

- **verificar_estoque_baixo**: Acionado antes de uma atualização na tabela Produto, verifica se a quantidade ficará abaixo de 10 unidades. Caso contrário, a operação é bloqueada.

3 SISTEMA EM JAVA

3.1 Classes e Funcionalidades

1. Classe Conexao (Database.Conexao)

- **Objetivo**: Estabelece a conexão com o banco de dados MySQL.
- **Função principal**: conectar() , utiliza o DriverManager.getConnection() para conectar ao banco e retorna um objeto Connection. Em caso de erro, exibe uma mensagem.

2. Classe CategoriaDao (Gerenciador.CategoriaDao)

- **Objetivo**: Manipula categorias no banco de dados.
- **Funções**:
    - cadastrarCategoria: Insere uma nova categoria.
    - atualizarCategoria: Atualiza informações de uma categoria.
    - deletarCategoria: Exclui uma categoria.
    - consultarCategoria: Consulta todas as categorias cadastradas.

3. Classe MovimentacaoEstoqueDao (Gerenciador.MovimentacaoEstoqueDao)

- **Objetivo**: Gera relatórios sobre movimentações de estoque.
- **Funções**:
    - relatorioMovimentacaoEstoque: Exibe as movimentações de estoque.
    - relatorioVendasLucro: Gera relatórios detalhados de vendas e lucros.

4. Classe ProdutoDao (Gerenciador.ProdutoDao)

- **Objetivo**: Manipula produtos no banco de dados.
- **Funções**:
    - cadastrarProduto: Insere um novo produto.
    - atualizarProduto: Atualiza informações de um produto.
    - deletarProduto: Exclui um produto.
    - registrarEntradaEstoque: Registra a entrada de produtos no estoque.
    - registrarSaidaEstoque: Registra a saída de produtos.
    - consultarProdutos: Consulta produtos com filtros como nome, categoria ou quantidade.

5. Classe Main (main.Main)

- **Objetivo**: Controla o fluxo do sistema e interage com o usuário através de um menu.
- **Funções**:
    - Apresenta um menu para cadastrar, atualizar, excluir e consultar produtos e categorias.
    - Permite registrar movimentações e gerar relatórios.
    - Utiliza o Scanner para capturar a entrada do usuário.

6. Classe Categoria (Models.Categoria)

- **Objetivo:** Representa uma categoria de produtos no sistema.
- **Atributos:**

  - id: Identificador único.
  - nome: Nome da categoria.
  - descricao: Descrição da categoria.

- **Métodos:** Getters e setters para acessar e modificar os atributos.

7. Classe MovimentacaoEstoque (Models.MovimentacaoEstoque)

- **Objetivo:** Representa as movimentações de estoque.
- **Atributos:**
  - idMovimentacao: Identificador único da movimentação.
  - produto: Produto afetado.
  - quantidade: Quantidade movimentada.
  - tipo: Tipo da movimentação (entrada ou saída).
  - data: Data e hora da movimentação.

- **Métodos:** Getters e setters para acessar e modificar os atributos.

8. Classe Produto (Models.produto)

- **Objetivo:** Representa os produtos no sistema.
- **Atributos:**
  - id: Identificador único do produto.
  - nome: Nome do produto. ▪ descricao: Descrição do produto.
  - quantidade: Quantidade disponível.
  - precoCompra: Preço de compra.
  - precoVenda: Preço de venda.
  - categoria: Categoria à qual o produto pertence.
- **Métodos:** Getters e setters para acessar e modificar os atributos.

4 FLUXO DO SISTEMA

1. Conexão com o Banco de Dados (Classe Conexao)

- A classe Conexao estabelece a conexão com o banco de dados MySQL no início do sistema. Caso a conexão falhe, uma mensagem de erro é exibida.

2. Manipulando atributos de tabelas (Classes Categoria, MovimentacaoEstoque e Produto)

- Essas classes permitem que seja possível acessar e modificar os atributos das tabelas presentes no banco de dados com os getters e setters.

3. Interação do Usuário (Classe Main)

- O sistema apresenta um menu de opções para o usuário interagir via console (com o uso de Scanner). O sistema executa a operação escolhida através dos métodos apropriados das classes CategoriaDao, ProdutoDao ou MovimentacaoEstoqueDao.

4. Gerenciamento de Categorias (Classe CategoriaDao)

- O usuário pode cadastrar, atualizar, excluir e consultar categorias no sistema. As operações chamam as stored procedures correspondentes no banco.

5. Gerenciamento de Produtos (Classe ProdutoDao)

- O usuário pode cadastrar, atualizar, excluir e consultar produtos. O sistema registra entradas e saídas de produtos no estoque, ajustando as quantidades disponíveis.

6. Movimentações de Estoque (Classe MovimentacaoEstoqueDao)

- O sistema gera relatórios detalhados sobre as movimentações de estoque e os lucros provenientes das vendas.

5 PRINCIPAIS FUNCIONALIDADES

1. **CRUD de Categorias:** Cadastro, atualização, exclusão e consulta de categorias de produtos.

2. **CRUD de Produtos:** Cadastro, atualização, exclusão e consulta de produtos com filtros.

3. **Movimentações de Estoque:** Registro de entradas e saídas de produtos e relatórios de todas as movimentações no estoque.

4. **Relatórios de Vendas e Lucro:** Análise do desempenho financeiro de cada produto, com relatórios de vendas e lucros.

5. **Relatório de Baixo Estoque:** Relatório de produtos com estoque abaixo de um determinado limite, permitindo identificar itens a serem repostos.

6 COMO USAR O SISTEMA

1. Configurar o Banco de Dados

Antes de executar o sistema, é necessário criar o banco de dados no MySQL. Siga as etapas abaixo:

1. Certifique-se de que o MySQL esteja instalado e configurado no seu computador.
2. Abra o MySQL Workbench ou o terminal do MySQL.
3. Localize o arquivo de script SQL fornecido no projeto (**Tables.sql e Procedures_triggers.sql**).
4. Execute o script no MySQL para criar o banco de dados.

2. Configurar a Conexão com o Banco de Dados

O arquivo **Conexao.java** gerencia a conexão do sistema com o banco de dados. Você precisa inserir suas credenciais do MySQL nesse arquivo.

1. Abra o arquivo Conexao.java no seu editor de código.
2. Localize as constantes USUARIO e SENHA.
3. Substitua os valores das constantes pelas suas credenciais no MySQL:

    1. Altere seu usuário, caso não seja root, e insira entre as aspas duplas:
    ```
    private static final String USUARIO = "root";
    ```
    2. Altere sua senha, e insira entre as aspas duplas:
    ```
    private static final String SENHA = "SuaSenha";
    ```

3. Execução do Sistema

1. Verifique se o Java e suas extensões estão instaladas corretamente no seu computador e IDE.
2. Execute o arquivo **Main.java** para que o programa comece a funcionar.

7 CONCLUSÃO

O Gerenciador de Estoque demonstra a aplicação prática de conceitos essenciais de programação e banco de dados em um contexto real, oferecendo uma solução robusta para a gestão de estoques. Combinando a linguagem Java para o desenvolvimento da aplicação e SQL para o gerenciamento de dados, o sistema garante eficiência, segurança e facilidade de uso. O projeto alcançou o objetivo de centralizar e automatizar operações relacionadas ao controle de produtos e categorias, movimentações de estoque e geração de relatórios analíticos.