Projeto de Banco de Dados: Sistema de Fábrica de Eletrônicos

Sumário

---

1. Introdução:

Este projeto modela um banco de dados completo para uma fábrica de eletrônicos, abrangendo todos os aspectos do negócio: desde o cadastro de funcionários, fornecedores e clientes, até o controle de estoque, gestão de pedidos, compras e produção. O sistema utiliza recursos avançados de SQL como views, stored procedures, triggers e functions para automatizar processos e manter a integridade dos dados.

2. Estrutura do Banco de Dados:

- **Banco**: FabricaEletronicosDB
- **Tabelas Principais**:
    - Funcionarios;
    - Fornecedores;
    - CategoriasProdutos;
    - Produtos;
    - Estoque;
    - ProdutosFornecedores;
    - Clientes;
    - Pedidos;
    - ItensPedido;
    - HistoricoPedidos;
    - Compras;
    - ItensCompra;
    - MovimentacaoEstoque;
    - Producao;
    - MateriaisProducao;

3. Tabelas Principais:

Funcionarios:

Armazena dados dos colaboradores da empresa.

- id (PK);
- nome;
- cargo;
- data_admissao;
- email (UNIQUE);
- telefone;
- ativo;

Fornecedores:

Cadastro de empresas que fornecem matérias-primas e componentes.

- id (PK);
- nome;
- cnpj (UNIQUE);
- endereco;
- cidade, estado, cep;
- telefone, email;
- pessoa_contato;
- data_cadastro;
- ativo;

Produtos:

Registro de todos os produtos fabricados ou comercializados.

- id (PK);
- codigo (UNIQUE);
- nome;
- descricao;
- preco_custo;
- preco_venda;
- categoria_id (FK);
- unidade_medida;
- data_cadastro;

Estoque:

Controle de quantidade disponível de cada produto.

- id (PK);
- produto_id (FK);
- quantidade;
- quantidade_minima;
- quantidade_maxima;
- localizacao;

- ultima_atualizacao;

Pedidos:

Registro de pedidos feitos pelos clientes.

- id (PK);
- cliente_id (FK);
- funcionario_id (FK);
- data_pedido;
- data_entrega_prevista;
- data_entrega_real;
- status (enum);
- forma_pagamento (enum);
- observacoes;
- valor_total;

Producao:

Controle dos processos produtivos da fábrica.

- id (PK);
- produto_id (FK);
- quantidade_planejada;
- quantidade_produzida;
- data_inicio;
- data_fim_prevista;
- data_fim_real;
- status (enum);
- funcionario_responsavel_id (FK);
- observacoes;

4. Objetos do Banco (View, Function, Procedure, Trigger):

A. VIEWs

View_EstoqueDetalhado

Fornece informações detalhadas do estoque, incluindo nome do produto, categoria, quantidade, status e valor.

```sql
CREATE VIEW View_EstoqueDetalhado AS
SELECT
    e.id AS estoque_id,
    p.id AS produto_id,
    p.codigo AS codigo_produto,
    p.nome AS nome_produto,
    c.nome AS categoria,
    e.quantidade,
```

```
    e.quantidade_minima,
    e.quantidade_maxima,
    p.preco_custo,
    p.preco_venda,
    (p.preco_venda * e.quantidade) AS valor_total_estoque,
    CASE
        WHEN e.quantidade <= e.quantidade_minima THEN 'CRÍTICO'
        WHEN e.quantidade <= (e.quantidade_minima * 1.5) THEN 'BAIXO'
        WHEN e.quantidade >= e.quantidade_maxima THEN 'EXCESSO'
        ELSE 'NORMAL'
    END AS status_estoque,
    e.localizacao,
    e.ultima_atualizacao
FROM
    Estoque e
JOIN
    Produtos p ON e.produto_id = p.id
LEFT JOIN
    CategoriasProdutos c ON p.categoria_id = c.id;
```

View_PedidosPendentes
Mostra pedidos que ainda não foram confirmados, com status de estoque.

View_EstoqueCritico
Lista produtos com estoque abaixo do mínimo, incluindo informações de fornecedores.

View_PedidosPorCliente
Análise de pedidos por cliente, incluindo total, valor, frequência e cancelamentos.

B. FUNCTIONs

CalcularValorEstoque
Calcula o valor total do estoque baseado nos preços de custo.

```sql
CREATE FUNCTION CalcularValorEstoque()
RETURNS DECIMAL(15,2)
DETERMINISTIC
BEGIN
    DECLARE total DECIMAL(15,2);

    SELECT SUM(p.preco_custo * e.quantidade)
    INTO total
    FROM Produtos p
    JOIN Estoque e ON p.id = e.produto_id;
```

```
    RETURN IFNULL(total, 0.00);
END;
```

VerificarEstoqueSuficiente
Verifica se há quantidade suficiente de um produto em estoque.

CalcularValorPedido
Calcula o valor total de um pedido somando os subtotais dos itens.

C. PROCEDUREs

CriarPedido
Cria um novo pedido com verificações de cliente e status.

```
sql
CREATE PROCEDURE CriarPedido(
    IN p_cliente_id INT,
    IN p_funcionario_id INT,
    IN p_data_entrega_prevista DATE,
    IN p_forma_pagamento VARCHAR(10),
    IN p_observacoes TEXT,
    OUT p_pedido_id INT,
    OUT p_msg VARCHAR(255)
)
BEGIN
    -- Verificações e criação do pedido
    -- [corpo da procedure]
END;
```

AdicionarItemPedido
Adiciona um item ao pedido com verificação de estoque.

ConfirmarPedido
Confirma um pedido e reduz o estoque dos produtos.

VerificarBaixoEstoque
Verifica produtos com estoque abaixo do mínimo.

CriarOrdemCompraReposicao
Cria automaticamente ordens de compra para produtos com estoque baixo.

D. TRIGGERs

Trigger_ImpedirExclusaoProdutosComEstoque
Impede a exclusão de produtos que ainda têm estoque.

```
sql
CREATE TRIGGER Trigger_ImpedirExclusaoProdutosComEstoque
BEFORE DELETE ON Produtos
FOR EACH ROW
BEGIN
```

```
    -- Verificações e bloqueio
    -- [corpo do trigger]
END;
```

Trigger_AtualizarEstoqueAposRecebimento
Atualiza o estoque após o recebimento de uma compra.

Trigger_CalcularValorTotalPedido
Calcula automaticamente o valor total do pedido após inserção de itens.

Trigger_AtualizarEstoqueAposCancelamentoPedido
Devolve itens ao estoque quando um pedido é cancelado.

5. Dados de Teste (INSERTs):

O sistema inclui dados de teste para todas as tabelas principais:

- **CategoriasProdutos**: 6 categorias (Capacitores, Resistores, etc.)
- **Funcionarios**: 5 funcionários com diferentes cargos
- **Fornecedores**: 5 fornecedores de componentes eletrônicos
- **Produtos**: 10 produtos eletrônicos diversos
- **Estoque**: Quantidades iniciais para os 10 produtos
- **ProdutosFornecedores**: Associações entre produtos e fornecedores
- **Clientes**: 5 clientes (4 PJ, 1 PF)
- **Pedidos**: 3 pedidos de teste através das procedures

6. Explicações Técnicas:

Controle de Transações

O sistema utiliza transações (START TRANSACTION, COMMIT, ROLLBACK) para garantir a integridade dos dados durante operações complexas como confirmação de pedidos e recebimentos de compras.

Hierarquia de Triggers

Os triggers são projetados para manter a coerência entre tabelas relacionadas:

- Atualização automática de estoque após movimentações
- Cálculo automático de valores totais
- Prevenção de exclusões que comprometam a integridade referencial

Cursores e Handlers

Várias procedures utilizam cursores para processar conjuntos de dados, como itens de pedidos e produtos com estoque baixo, com handlers para tratar exceções.

Validações e Regras de Negócio

O sistema implementa regras de negócio complexas:

- Verificação de estoque antes de confirmar pedidos
- Controle de status de pedidos e compras
- Registro de histórico de alterações

- Tratamento de situações críticas de estoque
-

7. Conclusão:

Este projeto implementa um sistema completo e robusto para gestão de uma fábrica de eletrônicos, com foco especial no controle de estoque e produção. Utilize-se de recursos avançados de SQL para automatizar processos e garantir a integridade dos dados.

O banco de dados é adequado para pequenas e médias fábricas do setor eletrônico, fornecendo todas as ferramentas necessárias para controle de fornecedores, clientes, estoque, vendas e produção. A estrutura permite fácil expansão para incluir novos módulos e funcionalidades conforme a necessidade.