

Módulo 11: Projetos Práticos e Avaliações

Módulo 11: Projetos Práticos e Avaliações

O Projeto Final deste módulo tem como objetivo integrar e aplicar os conhecimentos adquiridos ao longo do curso. Você será desafiado a desenvolver um sistema prático que simule um cenário real, abrangendo todos os principais tópicos estudados, desde a modelagem de banco de dados até a otimização de consultas.

Projeto Final

Desenvolvimento de um Sistema Prático

Você deverá criar um sistema que gerencie pedidos de venda para uma empresa fictícia. O sistema deve incluir funcionalidades para gerenciar clientes, produtos, pedidos e o estoque. A seguir, estão as etapas que deverão ser seguidas no desenvolvimento do projeto:

1. Modelagem do Banco de Dados

Crie o modelo relacional do banco de dados com as seguintes tabelas:

1. Tabela **clientes**:
 - Colunas: **id_cliente** (PK), **nome**, **email**, **telefone**.
 - Descrição: Armazena informações dos clientes.
2. Tabela **produtos**:
 - Colunas: **id_produto** (PK), **nome**, **preco**, **estoque**.
 - Descrição: Contém informações dos produtos.
3. Tabela **pedidos**:
 - Colunas: **id_pedido** (PK), **id_cliente** (FK), **data_pedido**.
 - Descrição: Armazena os pedidos realizados.
4. Tabela **pedido_produto**:
 - Colunas: **id_pedido** (FK), **id_produto** (FK), **quantidade**.
 - Descrição: Associa produtos aos pedidos (tabela de junção para relacionamento N:N).

Ferramenta Sugerida


Utilize ferramentas de modelagem como MySQL Workbench para criar o diagrama EER (Entidade-Relacionamento Estendido).

2. Criação de Relacionamentos

Defina as chaves primárias e chaves estrangeiras para garantir a integridade referencial entre as tabelas.

Exemplo de Script

```
CREATE TABLE clientes (  
    id_cliente INT AUTO_INCREMENT PRIMARY KEY,  
    nome VARCHAR(100),  
    email VARCHAR(100),  
    telefone VARCHAR(15)  
);  
  
CREATE TABLE produtos (  
    id_produto INT AUTO_INCREMENT PRIMARY KEY,  
    nome VARCHAR(100),  
    preco DECIMAL(10, 2),  
    estoque INT  
);  
  
CREATE TABLE pedidos (  
    id_pedido INT AUTO_INCREMENT PRIMARY KEY,  
    id_cliente INT,  
    data_pedido DATE,  
    FOREIGN KEY (id_cliente) REFERENCES clientes(id_cliente)  
);  
  
CREATE TABLE pedido_produto (  
    id_pedido INT,  
    id_produto INT,  
    quantidade INT,  
    PRIMARY KEY (id_pedido, id_produto),  
    FOREIGN KEY (id_pedido) REFERENCES pedidos(id_pedido),  
    FOREIGN KEY (id_produto) REFERENCES produtos(id_produto)  
);
```



3. Utilização de JOINS e Subconsultas

Implemente consultas para extrair informações úteis.

Exemplo de Consultas

1. **JOIN:** Listar todos os pedidos com os nomes dos clientes e produtos.

```
SELECT p.id_pedido, c.nome AS cliente, pr.nome AS produto, pp.quantidade
FROM pedidos p
JOIN clientes c ON p.id_cliente = c.id_cliente
JOIN pedido_produto pp ON p.id_pedido = pp.id_pedido
JOIN produtos pr ON pp.id_produto = pr.id_produto;
```

2. **Subconsulta:** Encontrar os produtos mais vendidos.

```
SELECT nome
FROM produtos
WHERE id_produto = (
    SELECT id_produto
    FROM pedido_produto
    GROUP BY id_produto
    ORDER BY SUM(quantidade) DESC
    LIMIT 1
);
```

4. Implementação de Procedimentos Armazenados

Crie **procedimentos armazenados** para automatizar operações comuns.

Exemplo de Procedimento

Registrar um novo pedido e atualizar o estoque:

```
CREATE PROCEDURE registrar_pedido(  
  IN cliente_id INT  
  IN produto_id INT  
  IN qtd INT  
  
BEGIN  
  -- Inserir o pedido  
  INSERT INTO pedidos (id_cliente, data_pedido) VALUES (cliente_id, CURDATE());  
  SET @id_pedido = LAST_INSERT_ID();  
  
  -- Inserir produto no pedido  
  INSERT INTO pedido_produto (id_pedido, id_produto, quantidade) VALUES  
  (@id_pedido, produto_id, qtd);  
  
  -- Atualizar estoque  
  UPDATE produtos SET estoque = estoque - qtd WHERE id_produto = produto_id;  
END
```

Executar o procedimento:

```
CALL registrar_pedido(1, 2, 3);
```

5. Otimização de Consultas

Aplique boas práticas para otimizar as consultas:

1. Criar Índices:

```
CREATE INDEX idx_cliente ON pedidos(id_cliente);  
CREATE INDEX idx_produto ON pedido_produto(id_produto);
```

2. Usar **EXPLAIN** para analisar consultas:

```
EXPLAIN SELECT * FROM pedidos WHERE id_cliente = 1;
```

3. Evitar Operações que Impedem o Uso de Índices:

- **Evite:** `WHERE UPPER(nome) = 'JOÃO'`
- **Use:** `WHERE nome = 'João'`

Objetivos de Avaliação

Critérios de Avaliação

- 1 . Modelagem do Banco de Dados:
 - Estrutura clara e relacionamentos bem definidos.
 - 2 . Funcionalidade do Sistema:
 - Consultas que atendem aos requisitos do projeto.
 - Procedimentos que realizam operações automatizadas.
 - 3 . Otimização:
 - Índices criados e usados corretamente.
 - Análise de performance utilizando EXPLAIN.
 - 4 . Clareza e Organização:
 - Código bem estruturado e documentado.
-

Aplicabilidade

- 1 . Sistemas de Vendas:
 - Gerenciamento de clientes, pedidos e produtos.
 - Relatórios detalhados de vendas e estoque.
- 2 . Sistemas Acadêmicos:
 - Gestão de alunos, cursos e matrículas.
- 3 . E-commerce:
 - Controle de inventário, carrinho de compras e processamento de pedidos.

Este projeto oferece uma visão prática e integrada de como projetar, implementar e otimizar um sistema de banco de dados real, preparando você para desafios do mercado.