Criação de bancos para exercícios

Aqui está a criação das tabelas e bancos de dados para serem utilizados em cada um dos exercícios mencionados, com foco em Triggers e Stored Procedures. O código a seguir é compatível com MySQL:

1. Banco de Dados para Trigger de Atualização de Salários (Exercício 1)

```sql
CREATE DATABASE Empresa;
USE Empresa;

-- Tabela de Funcionários
CREATE TABLE Funcionarios (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(100),
    salario DECIMAL(10, 2)
);

-- Tabela de Histórico de Salários
CREATE TABLE Historico_Salarios (
    id INT AUTO_INCREMENT PRIMARY KEY,
    funcionario_id INT,
    salario_antigo DECIMAL(10, 2),
    salario_novo DECIMAL(10, 2),
    data_alteracao TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (funcionario_id) REFERENCES Funcionarios(id)
);
```

2. Banco de Dados para Inserção de Produtos (Exercício 2)

```sql
CREATE DATABASE Loja;
USE Loja;

-- Tabela de Produtos
CREATE TABLE Produtos (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(100),
    preco DECIMAL(10, 2),
    categoria VARCHAR(50)
);
```

3. Banco de Dados para Registro de Exclusões de Clientes (Exercício 3)

```sql
CREATE DATABASE Vendas;
USE Vendas;

-- Tabela de Clientes
CREATE TABLE Clientes (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(100),
    email VARCHAR(100),
    telefone VARCHAR(15)
);

-- Tabela de Clientes Excluídos
CREATE TABLE Clientes_Excluidos (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(100),
    email VARCHAR(100),
    telefone VARCHAR(15),
    data_exclusao TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

4. Banco de Dados para Cálculo de Média (Exercício 4)

```sql
CREATE DATABASE Escola;
USE Escola;

-- Tabela de Alunos
CREATE TABLE Alunos (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(100)
);

-- Tabela de Notas
CREATE TABLE Notas (
    id INT AUTO_INCREMENT PRIMARY KEY,
    aluno_id INT,
    nota DECIMAL(5, 2),
    FOREIGN KEY (aluno_id) REFERENCES Alunos(id)
);
```

```sql
CREATE DATABASE Estoque;
USE Estoque;

-- Tabela de Produtos no Estoque
CREATE TABLE Estoque (
    id INT AUTO_INCREMENT PRIMARY KEY,
    produto_id INT,
    quantidade INT
);

-- Tabela de Vendas
CREATE TABLE Vendas (
    id INT AUTO_INCREMENT PRIMARY KEY,
    produto_id INT,
    quantidade INT,
    data_venda TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (produto_id) REFERENCES Estoque(produto_id)
);
```

6. Banco de Dados para Aplicação de Desconto (Exercício 6)

```sql
CREATE DATABASE LojaDesconto;
USE LojaDesconto;

-- Tabela de Produtos
CREATE TABLE Produtos (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(100),
    preco DECIMAL(10, 2),
    categoria VARCHAR(50)
);
```

7. Banco de Dados para Bloqueio de Alterações em Contratos (Exercício 7)

```sql
CREATE DATABASE EmpresaContratos;
USE EmpresaContratos;

-- Tabela de Contratos
CREATE TABLE Contratos (
    id INT AUTO_INCREMENT PRIMARY KEY,
    descricao VARCHAR(255),
    valor DECIMAL(10, 2),
    status ENUM('Aberto', 'Fechado')
);
```

8. Banco de Dados para Geração de Relatórios de Vendas (Exercício 8)

```sql
CREATE DATABASE RelatorioVendas;
USE RelatorioVendas;

-- Tabela de Vendas
CREATE TABLE Vendas (
    id INT AUTO_INCREMENT PRIMARY KEY,
    produto_id INT,
    quantidade INT,
    valor_total DECIMAL(10, 2),
    data_venda DATE
);
```

9. Banco de Dados para Auditoria de Alterações de Usuários (Exercício 9)

```sql
CREATE DATABASE AuditoriaUsuarios;
USE AuditoriaUsuarios;

-- Tabela de Usuários
CREATE TABLE Usuarios (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(100),
    email VARCHAR(100),
    senha VARCHAR(100)
);

-- Tabela de Auditoria de Usuários
CREATE TABLE Usuarios_Auditoria (
    id INT AUTO_INCREMENT PRIMARY KEY,
    usuario_id INT,
    tipo_alteracao ENUM('INSERT', 'UPDATE', 'DELETE'),
    data_alteracao TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (usuario_id) REFERENCES Usuarios(id)
);
```

10. Banco de Dados para Verificação de Créditos de Clientes (Exercício 10)

```sql
CREATE DATABASE LojaCreditos;
USE LojaCreditos;

-- Tabela de Clientes
CREATE TABLE Clientes (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(100),
    saldo_creditos DECIMAL(10, 2)
);

-- Tabela de Vendas
CREATE TABLE Vendas (
    id INT AUTO_INCREMENT PRIMARY KEY,
    cliente_id INT,
    valor_venda DECIMAL(10, 2),
    data_venda TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (cliente_id) REFERENCES Clientes(id)
);
```

Exercícios

1. Criar uma Trigger para Atualizar um Registro

- **Descrição:** Crie uma trigger que seja executada automaticamente após a atualização de uma tabela de funcionários, de forma que sempre que o salário de um funcionário for alterado, a trigger atualize uma tabela de histórico de salários.

2. Criar uma Stored Procedure para Inserir Registros

- **Descrição:** Escreva uma stored procedure que insira um novo produto em uma tabela Produtos, recebendo parâmetros como nome, preço, e categoria. A procedure deve verificar se o nome já existe antes de inserir.

3. Criar uma Trigger para Registro de Exclusões

- **Descrição:** Crie uma trigger que, ao deletar um registro da tabela Clientes, mova as informações excluídas para uma tabela de backup chamada Clientes_Excluidos, juntamente com a data da exclusão.

4. Stored Procedure para Cálculo de Média

- **Descrição:** Crie uma stored procedure que receba como parâmetro o ID de um aluno e retorne a média das suas notas, considerando uma tabela Notas. A média deve ser calculada dinamicamente na procedure.

5. Criar uma Trigger para Atualizar Estoque

- **Descrição:** Crie uma trigger que, após uma venda ser registrada na tabela Vendas, atualize a quantidade de produtos disponíveis no estoque na tabela Estoque.

6. Stored Procedure para Aplicar Desconto

- **Descrição:** Crie uma stored procedure que aplique um desconto em todos os produtos de uma categoria específica. A procedure deve receber o nome da categoria e o percentual de desconto como parâmetros.

7. Criar uma Trigger para Bloquear Alterações

- **Descrição:** Crie uma trigger que impeça qualquer alteração em uma tabela Contratos se o campo Status estiver marcado como "Fechado". A trigger deve cancelar a operação e exibir uma mensagem de erro.

8. Stored Procedure para Geração de Relatórios

- **Descrição:** Escreva uma stored procedure que gere um relatório com a quantidade de vendas realizadas em cada mês. A procedure deve retornar o mês, a quantidade de vendas e o valor total vendido.

9. Criar uma Trigger para Auditoria

- **Descrição:** Crie uma trigger que grave todas as alterações realizadas na tabela Usuarios em uma tabela de auditoria chamada Usuarios_Auditoria. A trigger deve registrar o usuário que fez a alteração, o tipo de alteração (INSERT, UPDATE ou DELETE), e a data/hora.

10. Stored Procedure para Verificação de Créditos

- **Descrição:** Escreva uma stored procedure que verifique o saldo de créditos de um cliente em uma tabela Clientes antes de permitir que uma venda seja registrada. Se o saldo for inferior ao valor da compra, a procedure deve retornar uma mensagem de erro.