Enunciado do Projeto

Objetivo: Desenvolver um sistema de gerenciamento para uma clínica médica que organiza e gerencia agendamentos, consultas, informações de pacientes e médicos, histórico de consultas e notificações. Esse sistema integrará banco de dados relacional e recursos avançados, como *functions* e *triggers*, para garantir a integridade dos dados e automatizar operações importantes.

1. Descrição Geral do Sistema

A clínica médica precisa de uma solução que facilite a administração de suas operações diárias, permitindo um controle eficiente de agendamentos de consultas, cadastro de pacientes e médicos, gerenciamento de especialidades e monitoramento de consultas realizadas. O sistema deve fornecer recursos para:

- Agendar consultas e impedir conflitos de horário.
- Registrar informações detalhadas de médicos, especialidades e pacientes.
- Monitorar e armazenar dados históricos de consultas.
- Enviar notificações de lembrete e confirmação de consultas (simuladas por meio de tabelas de notificação).
- Registrar o motivo e dados de cancelamento de consultas, se necessário.
- Controlar o limite de consultas por paciente a cada mês.

2. Funcionalidades e Tabelas do Sistema

Tabelas principais:

- **Pacientes**: Cadastro de pacientes, com informações pessoais básicas.
- **Medicos**: Cadastro dos médicos, incluindo suas especialidades.
- **Especialidades**: Lista de especialidades médicas disponíveis.
- **Agendamentos**: Registro de consultas agendadas, incluindo informações de data, hora, paciente e médico.
- **ConsultasRealizadas**: Armazena o histórico das consultas realizadas, com dados de diagnóstico, prescrição e observações.
- **HorariosFuncionamento**: Horário de funcionamento da clínica, para evitar agendamentos fora do horário.
- **Salas**: Cadastro de salas para as consultas, evitando conflitos de agendamento para uma mesma sala e horário.

Funcionalidades adicionais:

- **Notificacoes**: Armazena informações de notificações de lembretes de consultas e confirmações para simular envio de notificações por e-mail ou SMS.
- **Cancelamentos**: Armazena dados sobre consultas canceladas e o motivo do cancelamento.
- **PlanosSaude**: Cadastro de planos de saúde vinculados aos pacientes.

3. Regras e Restrições do Sistema

- **Horário de Funcionamento:** Agendamentos só podem ocorrer dentro do horário de funcionamento definido na tabela HorariosFuncionamento.
- **Limite de Consultas Mensais**: Cada paciente só pode agendar até 5 consultas por mês.
- **Conflito de Sala:** Um mesmo horário e sala não podem ser reservados para mais de um paciente.
- **Histórico de Consultas**: Ao concluir uma consulta, ela é registrada na tabela ConsultasRealizadas e removida da tabela Agendamentos.
- **Notificações de Agendamento**: Ao agendar uma consulta, uma notificação de confirmação deve ser registrada na tabela Notificacoes.
- **Cancelamento de Consultas**: Ao cancelar uma consulta, o registro e o motivo do cancelamento devem ser salvos na tabela Cancelamentos.

4. Exigências Técnicas: Functions e Triggers

Para atender aos requisitos do sistema, devem ser criadas functions e triggers no banco de dados:

Functions:

- **consultas_no_mes**: Função para verificar o número de consultas de um paciente em um mês específico, para aplicar a restrição de limite mensal.

Triggers:

- **verificar_horario_funcionamento:** Trigger BEFORE INSERT em Agendamentos para impedir agendamentos fora do horário de funcionamento.
- **concluir_consulta**: Trigger AFTER UPDATE em Agendamentos para transferir uma consulta concluída para a tabela ConsultasRealizadas.
- **limitar_consultas_mensais:** Trigger BEFORE INSERT em Agendamentos para impedir que um paciente ultrapasse o limite de 5 consultas no mês.
- **evitar_conflito_sala:** Trigger BEFORE INSERT em Agendamentos para impedir a duplicidade de reserva de sala no mesmo horário.
- **notificar_agendamento:** Trigger AFTER INSERT em Agendamentos para registrar uma notificação de confirmação na tabela Notificacoes.
- **registrar_cancelamento:** Trigger BEFORE DELETE em Agendamentos para registrar o cancelamento e motivo na tabela Cancelamentos.

5. Desenvolvimento e Testes

Para desenvolver o sistema, cada aluno deverá:

5.1 Criar o Esquema do Banco de Dados: Implementar todas as tabelas descritas no projeto com as devidas relações e restrições.

5.2 Implementar Functions e Triggers: Criar as functions e triggers necessárias para manter as regras de negócio e a integridade dos dados.

5.3 Executar Testes: Validar as funções e triggers desenvolvidas através de cenários de teste. Exemplos de cenários a serem testados incluem:

- Tentar agendar uma consulta fora do horário de funcionamento.
- Agendar uma consulta em um horário já reservado por outro paciente na mesma sala.
- Verificar a notificação de confirmação ao registrar um agendamento.
- Exceder o limite de consultas para um paciente em um único mês.

5.4 Documentar o Projeto: Explicar a lógica de cada function e trigger, descrevendo seu funcionamento e propósito.

6. Exemplo Prático

Para ajudar no desenvolvimento, um exemplo completo de consulta SQL para criar a tabela Pacientes com uma estrutura básica será fornecido, além de uma trigger de exemplo para verificar conflitos de sala.

```
-- Exemplo: Criação da tabela Pacientes

CREATE TABLE Pacientes (

    id_paciente INT PRIMARY KEY AUTO_INCREMENT,

    nome VARCHAR(100) NOT NULL,

    data_nascimento DATE NOT NULL,

    telefone VARCHAR(15),

    endereco TEXT

);

-- Exemplo: Trigger para evitar conflito de sala

DELIMITER //

CREATE TRIGGER evitar_conflito_sala

BEFORE INSERT ON Agendamentos

FOR EACH ROW

BEGIN

    DECLARE conflito INT;

    SELECT COUNT(*) INTO conflito

    FROM Agendamentos

    WHERE id_sala = NEW.id_sala

    AND data_agendamento = NEW.data_agendamento
```

```
    AND hora_agendamento = NEW.hora_agendamento;

    IF conflito > 0 THEN

        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Sala já ocupada para este
horário.';

    END IF;

END //

DELIMITER ;
```

7. Entrega do Projeto

Cada aluno deverá entregar:

- Scripts SQL contendo todas as instruções para criar as tabelas, functions e triggers.
- Documentação do Projeto, explicando a funcionalidade de cada componente implementado.
- Relatório de Testes, com os resultados dos principais cenários testados.

8. Critérios de Avaliação

O projeto será avaliado com base nos seguintes critérios:

- Correção e funcionalidade das tabelas, functions e triggers.
- Clareza e objetividade na documentação.
- Validade e completude dos cenários de teste.
- Estrutura e organização dos scripts SQL.