André Moura Lima
Enzo Felipe Prudencio Avelino Lima
Francisco Elias Da Silva Fernandes

PROJETO DE BANCO DE DADOS

1.0 JUSTIFICATIVA

O banco de dados proposto é projetado para gerenciar informações relacionadas a uma clínica médica. Ele permite armazenar dados de pacientes, médicos, funcionários, históricos de procedimentos, agendamentos, pagamentos e controle de insolventes. O problema que ele resolve é facilitar o gerenciamento eficiente dessas informações e automatizar certos processos relacionados à administração da clínica.

1.1 DEFINIÇÃO DO PROBLEMA

Em uma cidade existem várias Clínicas Médicas com uma alta demanda de pacientes, realizava-se os registros de todos os seus pacientes e funcionários de forma manual, utilizando o Excel, no entanto, com o passar do tempo tornou-se impossível realizar algumas consultas simples nos dados armazenados.

Cada Clínica tem um CNPJ, nome e endereço. Uma clínica pode ter vários funcionários associados a ela, mas um funcionário só pode trabalhar em uma clínica. Dos funcionários, deseja-se guardar um id, CPF, nome e salário. Sabe-se também que os funcionários podem ser médicos ou atendentes, dos médicos deseja-se salvar o CRM e a especialidade.

O atendente pode fazer um ou mais agendamentos de pacientes, mas um agendamento só pode ser feito por um e somente um atendente. Um paciente pode efetuar muitos agendamentos, porém um agendamento deve estar ligado a somente um paciente. Dos agendamentos deseja-se salvar um id, a data do agendamento (data atual), a data da consulta e uma descrição.

Dos pacientes é necessário salvar o CPF, nome, número telefônico, data de nascimento e sexo. Um paciente pode ser atendido por muitas clínicas e uma clínica pode atender nenhum ou muitos pacientes. Os pacientes possuem históricos que contém: id, procedimento e a data. Um histórico está vinculado a um e somente um paciente e um paciente pode ter vários históricos. O paciente faz o pagamento à clínica, no pagamento tem-se a data atual, uma identificação (id), data de vencimento, status (pago, pendente, atrasado) e o valor a ser pago pelo cliente.

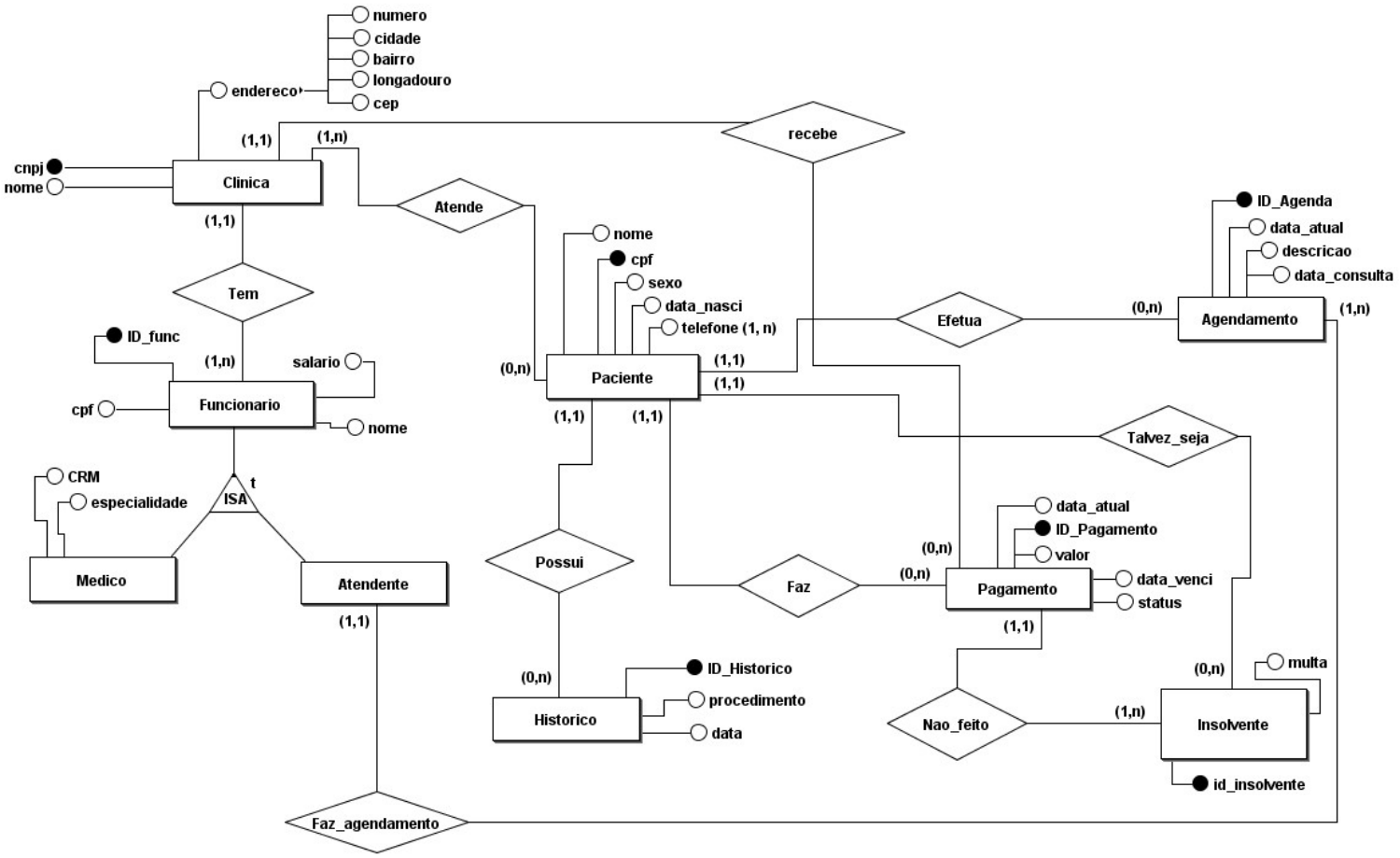
Um pagamento é feito por um e somente um paciente e um paciente pode fazer nenhum ou muitos pagamentos. Uma clínica recebe o pagamento de muitos pacientes.

Quando um ou mais pagamentos tem o status como atrasado o paciente é considerado insolvente e é aplicado uma multa 100 reais mais 10% do valor do pagamento atrasado. Uma insolvência está ligada a um e somente um pagamento com status atrasado. Um paciente pode ter várias insolvências, mas uma insolvência está ligada a um paciente.

2.0 MODELO CONCEITUAL

O modelo conceitual foi empregado para representar, definir e comunicar relações de alto nível entre as entidades.

Figura 1: Modelo Conceitual do Problema Proposto

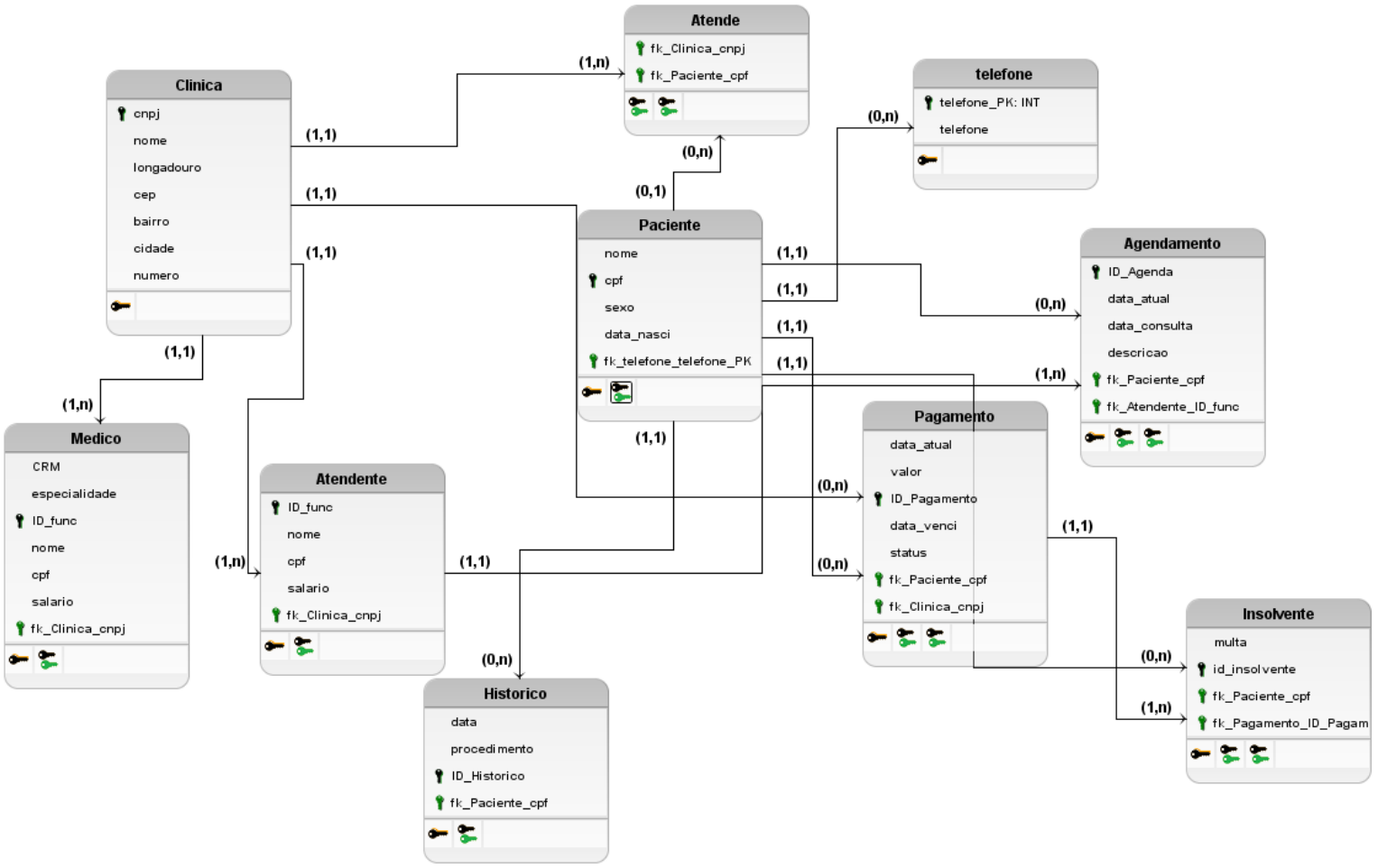


Fonte: Autores.

3.0 MODELO LÓGICO

O modelo lógico, serve para estabelecer a estrutura dos elementos de dados e os relacionamentos entre eles. Pode-se inferir então, que tal modelo irá detalhar como os dados serão implementados no banco físico.

Figura 2: Modelo lógico do Problema Proposto



Fonte: Autores.

4.0 MODELO FÍSICO

Pode ser descrito como um uma representação concreta e específica de como os dados são armazenados. Para construção do modelo proposto neste trabalho foi utilizado a linguagem SQL e como SGBD o PostgreSQL.

4.1 CRIAÇÃO DAS TABELAS

```
-- Identificadores Unicos Universais
CREATE EXTENSION IF NOT EXISTS "uuid-ossp";

CREATE TABLE Clinica(
        nome VARCHAR(120),
        cnpj VARCHAR(14) PRIMARY KEY,
        cep VARCHAR(50),
        logradouro VARCHAR(150),
        cidade VARCHAR(120),
        numero VARCHAR(10)
);



CREATE TABLE Medico(
        id_func UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
        cpf VARCHAR(11) UNIQUE,
        nome VARCHAR(100),
        salario NUMERIC(10, 2),
        CRM VARCHAR(20) UNIQUE,
        especialidade VARCHAR(100),
        fk_cnpj VARCHAR(14) REFERENCES Clinica(cnpj)
);



CREATE TABLE Atendente(
        id_func UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
```

```
    cpf VARCHAR(11) UNIQUE,
    nome VARCHAR(100),
    salario NUMERIC(10, 2),
    fk_cnpj VARCHAR(14) REFERENCES Clinica(cnpj)
);


CREATE TABLE Paciente(
    cpf VARCHAR(11) PRIMARY KEY,
    nome VARCHAR(100),
    sexo VARCHAR(40),
    data_nasci DATE
);


CREATE TABLE Atende(
   fk_cnpj VARCHAR(14) REFERENCES Clinica(cnpj),
   FK_cpf VARCHAR(11) REFERENCES Paciente(cpf),
   PRIMARY KEY(fk_cnpj, FK_cpf)
);


CREATE TABLE Telefone(
    telefone VARCHAR(20),
    FK_cpf VARCHAR(11) REFERENCES Paciente(cpf),
    PRIMARY KEY(telefone, FK_cpf)
);


CREATE TABLE Historico(
    ID_Historico UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    data DATE,
    procedimento VARCHAR,
    FK_cpf VARCHAR(11) REFERENCES Paciente(cpf)
```

```
);


CREATE TABLE Agendamento(
        ID_Agenda UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
        data_atual DATE,
        data_consulta DATE,
        descricao VARCHAR,
        FK_cpf VARCHAR(11) REFERENCES Paciente(cpf),
        fk_id_func UUID REFERENCES Atendente(id_func)
);


CREATE TABLE Pagamento (
        Id_Pagamento UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
        valor NUMERIC(10, 2),
        status VARCHAR(35),
        data_venci DATE,
        data_atual DATE,
        fk_cnpj VARCHAR(14) REFERENCES Clinica(cnpj),
        FK_cpf VARCHAR(11) REFERENCES Paciente(cpf)
);


CREATE TABLE Insolvente (
        id_insolvente UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
        multa NUMERIC(10, 2),
        FK_cpf VARCHAR(11) REFERENCES Paciente(cpf),
        fk_Id_Pagamento UUID REFERENCES Pagamento(Id_Pagamento)

);
```

4.2 INSERÇÃO DE DADOS

```
INSERT INTO Paciente( cpf, nome, sexo, data_nasci)
VALUES ('12345678901', 'João Silva', 'Masculino', '1990-01-01');
INSERT INTO Telefone(telefone, FK_cpf)
VALUES ('(11) 99999-9999', 12345678901);

INSERT INTO Paciente(cpf, nome, sexo, data_nasci)
VALUES ('98765432101', 'Maria Souza', 'Feminino', '1985-05-10');
INSERT INTO telefone(telefone, FK_cpf)
VALUES ('(11) 99356-9856', '98765432101');

INSERT INTO Paciente(cpf, nome, sexo, data_nasci)
VALUES ('98775432106', 'Pedro Souza', 'Masculino', '1985-06-11');
INSERT INTO telefone(telefone, FK_cpf)
VALUES ('(11) 99920-9910', '98775432106');

INSERT INTO Paciente(cpf, nome, sexo, data_nasci)
VALUES ('99706432109', 'Cleo Souza', 'Feminino', '2005-07-15');
INSERT INTO telefone(telefone, FK_cpf)
VALUES ('(11) 98520-9810', '99706432109');

INSERT INTO Paciente(cpf, nome, sexo, data_nasci)
VALUES ('82421795060', 'Breno Costa', 'Masculino', '2000-07-17');
INSERT INTO telefone(telefone, FK_cpf)
VALUES ('(11) 98550-5050', '82421795060');

INSERT INTO Paciente(cpf, nome, sexo, data_nasci)
VALUES ('37393484021', 'Rui Barros', 'Masculino', '2001-08-19');
INSERT INTO telefone(telefone, FK_cpf)
VALUES ('(11) 98825-5015', '37393484021');

INSERT INTO Paciente(cpf, nome, sexo, data_nasci)
VALUES ('37393456826', 'Mariana Barros', 'Feminino', '2011-01-18');
```

```
INSERT INTO telefone(telefone, FK_cpf)
VALUES ('(98) 99725-3782', '37393456826');

INSERT INTO Paciente(cpf, nome, sexo, data_nasci)
VALUES ('66666666666', 'Marilza Rodrigues', 'Feminino', '2001-06-15');
INSERT INTO telefone(telefone, FK_cpf)
VALUES ('(98) 99825-6666', '66666666666');

INSERT INTO Paciente(cpf, nome, sexo, data_nasci)
VALUES ('98562345231', 'Ze Da Manga', 'Masculino', '1999-10-19');
INSERT INTO telefone(telefone, FK_cpf)
VALUES ('(98) 98845-9867', '98562345231');


INSERT INTO Clinica(nome , cnpj, cep, logradouro, cidade, numero)
VALUES('Med Center', '65635489000113', '65520000', 'Avenida das Flores','Cidade X', '10');

INSERT INTO Clinica(nome , cnpj, cep, logradouro, cidade, numero)
VALUES('Clinica Master', '71341818000116', '65520000', 'Avenida das Flores','Cidade X', '11');

INSERT INTO Clinica(nome , cnpj, cep, logradouro, cidade, numero)
VALUES('Clinica Y', '46606962000190', '65520000', 'Avenida das Flores','Cidade X', '16');


INSERT INTO Medico(cpf, nome, salario, CRM, especialidade, fk_cnpj)
VALUES('94580406044','Cleython Rasta', 30000.00, 'SP59013', 'Cardiologista', '71341818000116');

INSERT INTO Medico(cpf, nome, salario, CRM, especialidade, fk_cnpj)
VALUES('47023389008','Rosana Silva', 15000.00, 'SP59015', 'Dermatologista', '65635489000113');
```

```
INSERT INTO Medico(cpf, nome, salario, CRM, especialidade, fk_cnpj)
VALUES('94497245047','Jubileu Pessoa', 18000.00,'SP59915', 'Psiquitra','65635489000113');

INSERT INTO Medico(cpf, nome, salario, CRM, especialidade, fk_cnpj)
VALUES('76568155039','Ana Pessoa', 25000.00, 'SP10015', 'Neurologista', '46606962000190');

INSERT INTO Medico(cpf, nome, salario, CRM, especialidade, fk_cnpj)
VALUES('68375192040','Manoel Costa', 23000.00, 'SP10150', 'Neurologista', '71341818000116');

INSERT INTO Medico(cpf, nome, salario, CRM, especialidade, fk_cnpj)
VALUES('17247446072', 'Telma Maria', 10000.00, 'SP10265', 'Pediatra', '46606962000190');

INSERT INTO Medico(cpf, nome, salario, CRM, especialidade, fk_cnpj)
VALUES('65635489000', 'Paula Tejano', 24000.00, 'SP30205', 'Proctologista', '71341818000116');

INSERT INTO Medico(cpf, nome, salario, CRM, especialidade, fk_cnpj)
VALUES('50220073040', 'Elisa Moura', 50000.00, 'MA50649', 'Cardiologista', '65635489000113');

INSERT INTO Medico(cpf, nome, salario, CRM, especialidade, fk_cnpj)
VALUES('22800052040', 'Keila Costa', 27000.00, 'SC56823', 'Ortopedista', '46606962000190');



INSERT INTO Atendente(cpf, nome, salario, fk_cnpj)
VALUES('79656891031', 'Miguel Santos', 3000.00, '46606962000190');

INSERT INTO Atendente(cpf, nome, salario, fk_cnpj)
VALUES('65536680005', 'Kiara Silva', 5000.00, '71341818000116');
```

```
INSERT INTO Atendente(cpf, nome, salario, fk_cnpj)
VALUES('08408576089', 'Lorena Carbonari', 4500.90, '65635489000113');


INSERT INTO Historico(data, procedimento, FK_cpf)
VALUES ('2023-07-10', 'Procedimento X', '98765432101');

INSERT INTO Historico (data, procedimento, FK_cpf)
VALUES ('2023-07-15', 'Procedimento Y', '98765432101');

INSERT INTO Historico (data, procedimento, FK_cpf)
VALUES ('2023-07-20', 'Procedimento Z', '98765432101');


INSERT INTO Agendamento(data_atual, data_consulta, descricao, FK_cpf, fk_id_func)
VALUES('2023-07-10', '2023-07-15', 'Descricao 1', '98562345231', (SELECT id_func FROM
Atendente WHERE cpf = '08408576089'));

INSERT INTO Agendamento(data_atual, data_consulta, descricao, FK_cpf, fk_id_func)
VALUES ('2023-07-10', '2023-07-15', 'Descricao 2', '66666666666', (SELECT id_func
FROM Atendente WHERE cpf = '79656891031'));

INSERT INTO Agendamento(data_atual, data_consulta, descricao, FK_cpf, fk_id_func)
VALUES ('2023-07-10', '2023-07-15', 'Descricao 3', '37393456826', (SELECT id_func
FROM Atendente WHERE cpf = '79656891031'));

INSERT INTO Agendamento(data_atual, data_consulta, descricao, FK_cpf, fk_id_func)
VALUES ('2023-07-10', '2023-07-15', 'Descricao 4', '99706432109', (SELECT id_func
FROM Atendente WHERE cpf = '65536680005'));
```

```
INSERT INTO Pagamento(valor, status, data_venci, data_atual, fk_cnpj, FK_cpf)
VALUES(5000, 'pendente', '2023-07-31', '2023-07-20', '71341818000116', '98765432101');

INSERT INTO Pagamento(valor, status, data_venci, data_atual, fk_cnpj, FK_cpf)
VALUES(10000, 'pendente', '2023-07-31', '2023-07-20', '71341818000116', '98765432101');

INSERT INTO Pagamento(valor, status, data_venci, data_atual, fk_cnpj, FK_cpf)
VALUES(15000, 'pendente', '2023-07-31', '2023-07-20', '71341818000116', '98765432101');

INSERT INTO Pagamento(valor, status, data_venci, data_atual, fk_cnpj, FK_cpf)
VALUES(20000, 'pendente', '2023-08-10', '2023-07-27', '65635489000113', '98562345231');



INSERT INTO Atende(fk_cnpj, FK_cpf)
VALUES
   ('65635489000113', '12345678901'),
        ('65635489000113', '98765432101'),
   ('65635489000113', '98775432106'),
   ('65635489000113', '99706432109'),
   ('65635489000113', '82421795060'),
   ('65635489000113', '37393484021'),
   ('65635489000113', '37393456826'),
   ('65635489000113', '66666666666'),
   ('65635489000113', '98562345231'),

   ('71341818000116', '12345678901'),
   ('71341818000116', '98765432101'),
   ('71341818000116', '98775432106'),
   ('71341818000116', '99706432109'),
   ('71341818000116', '82421795060'),
   ('71341818000116', '37393484021'),
   ('71341818000116', '37393456826'),
```

```
    ('71341818000116', '66666666666'),
    ('71341818000116', '98562345231'),

    ('46606962000190', '12345678901'),
    ('46606962000190', '98765432101'),
    ('46606962000190', '98775432106'),
    ('46606962000190', '99706432109'),
    ('46606962000190', '82421795060'),
    ('46606962000190', '37393484021'),
    ('46606962000190', '37393456826'),
    ('46606962000190', '66666666666'),
    ('46606962000190', '98562345231');
```

4.3 CONSULTAS COM EXISTS

```
-- Verifica se existe um Medico que ganhe 24000.00 reais
SELECT EXISTS (
    SELECT *
    FROM MEDICO
    WHERE salario = 24000.00
);

-- Existencia de pacientes que sao insolventes
SELECT P.nome AS Nome_Insolventes FROM Paciente P
WHERE EXISTS (
    SELECT * FROM Insolvente I
    WHERE I.FK_cpf = P.cpf
);
```

4.4 CONSULTA COM O HAVING

```
-- Retornar a especialidade dos medicos e a media dos salarios dos funcionarios que sao medicos
-- e exibi apenas as especialidades cuja media salarial seja maior que 20000.
SELECT M.especialidade, AVG(M.salario) AS Media_Salario
FROM Medico M
GROUP BY M.especialidade
HAVING AVG(M.salario) > 20000;
```

4.5 FUNÇÃO

```
-- Funcao que retorna o medico com o maior salario
CREATE OR REPLACE FUNCTION MaiorSalario()
RETURNS TABLE (nome VARCHAR(100), salario NUMERIC(10, 2)) AS $$
BEGIN
   RETURN QUERY
   SELECT M.nome, M.salario
   FROM Medico M
   WHERE M.salario = (SELECT MAX(M2.salario) FROM Medico M2);
END;
$$ LANGUAGE plpgsql;
```

4.6 TRIGGER

```
-- Serve para adicionar pacientes na tabela Insolvente  (FUNCIONA COM INSERT E UPDATE)
CREATE OR REPLACE FUNCTION adicionarPacientesInsolvente()
RETURNS TRIGGER AS $$
BEGIN
   IF TG_OP = 'INSERT' THEN
      -- Dada insercao, verifica se o status e 'atrasado'
      IF NEW.status = 'atrasado' THEN
         -- Calcular a multa como 100 + 10% do valor do pagamento
         INSERT INTO Insolvente (id_insolvente, FK_cpf, fk_id_pagamento, multa)
         VALUES (uuid_generate_v4(), NEW.FK_cpf, NEW.id_pagamento, 100 + (NEW.valor * 0.1));
      END IF;
   ELSIF TG_OP = 'UPDATE' THEN
      -- Dada atualizacao, verifica se o status foi alterado para 'atrasado'
      IF NEW.status = 'atrasado' AND OLD.status != 'atrasado' THEN
         -- Calculo da multa como 100 + 10% do valor do pagamento + gerar o id_insolvente
         INSERT INTO Insolvente (id_insolvente, FK_cpf, fk_id_pagamento, multa)
         VALUES (uuid_generate_v4(), NEW.FK_cpf, NEW.id_pagamento, 100 + (NEW.valor * 0.1));
      END IF;
   END IF;

   RETURN NEW;
END;
$$ LANGUAGE plpgsql;

-- Usando a função anterior em um gatilho AFTER INSERT OU UPDATE
CREATE OR REPLACE TRIGGER trigger_adicionarPacientesInsolvente
AFTER INSERT OR UPDATE ON Pagamento
FOR EACH ROW
EXECUTE FUNCTION adicionarPacientesInsolvente();
```

5.0 OBSERVAÇÕES

- Os pacientes só serão adicionados a tabela insolvente quando a TRIGGER estiver funcionando.
- Observe que os pagamentos foram todos inseridos com status = ‘pendente’ para testar o funcionamento da Trigger, pode ser inserido um pagamento já com o status como atrasado ou utilizando o update para modificar o status.

Atraso direto no insert:

```
INSERT INTO Pagamento(valor, status, data_venci, data_atual, fk_cnpj, FK_cpf)
VALUES(3000, 'atrasado', '2023-08-10', '2023-07-27', '65635489000113', '37393456826');
```

Atraso usando o UPDATE:

```
UPDATE Pagamento
SET status = 'atrasado'
WHERE Id_Pagamento = '';
```

Como o Id_Pagamento é uma chave longa e aleatória pode ser usada a seguinte variação no update:

```
UPDATE Pagamento
SET status = 'atrasado'
WHERE Id_Pagamento = (SELECT Id_Pagamento FROM Pagamento WHERE FK_cpf = '98562345231');
```

- O link do drive contém as imagens dos modelos conceitual e lógico, um arquivo .sql contendo todo o banco e suas consultas.
  DRIVE:
  https://drive.google.com/drive/folders/1rULZKc22KWe_DGF3pAD85HIaPixfOZUc?usp=sharing