Documentação Técnica - Sistema de Banco de Dados para Técnico de Enfermagem

1. Visão Geral do Projeto

Este projeto implementa um banco de dados para auxiliar técnicos de enfermagem na gestão de informações de pacientes, medicamentos administrados e salários recebidos. O sistema foi desenvolvido usando PostgreSQL, um banco de dados relacional robusto e eficiente, proporcionando uma maneira organizada e confiável de gerenciar dados sensíveis.

2. Estrutura do Banco de Dados

2.1. Diagrama ERD (Entidade-Relacionamento)

O diagrama ERD descreve as entidades e seus relacionamentos no sistema:

- **Paciente**: Armazena informações pessoais de cada paciente.
- **Medicamento**: Contém dados sobre medicamentos prescritos e sua administração.
- **Administração**: Registra o momento em que os medicamentos foram administrados a cada paciente.
- **Salário**: Armazena os pagamentos efetuados pelos pacientes ao técnico de enfermagem.

Modelo Lógico

- **Paciente** (id_paciente, nome, idade, endereco, telefone)
- **Medicamento** (id_medicamento, nome_medicamento, dosagem, frequencia, instrucoes)
- **Administração** (id_administracao, data_adm, horario, observacoes, id_paciente, id_medicamento)
- **Salário** (id_salario, mes, valor, observacoes, id_paciente)

3. Instalação e Configuração

3.1. Requisitos

- PostgreSQL (versão 9.5 ou superior).
- Ferramentas como pgAdmin ou psql para gerenciamento do banco de dados.

3.2. Como clonar e configurar o projeto

1. Clone este repositório:

```
git clone https://github.com/usuario/repo-banco-dados-enfermagem.git
```

2. Acesse o diretório do projeto:

```
cd repo-banco-dados-enfermagem
```

3. Execute o script SQL no PostgreSQL para criar as tabelas:

```
psql -U <usuario> -d <database> -f banco_de_dados.sql
```

Isso criará as tabelas **Paciente**, **Medicamento**, **Administração**, e **Salário** no seu banco de dados PostgreSQL.

4. Estrutura das Tabelas

4.1. Tabela Paciente

```
CREATE TABLE PACIENTE (
  id_paciente SERIAL PRIMARY KEY,
  nome VARCHAR(100) NOT NULL,
  idade INT,
  endereco VARCHAR(150),
  telefone VARCHAR(15)
);
```

4.2. Tabela Medicamento

```
CREATE TABLE MEDICAMENTO (
  id_medicamento SERIAL PRIMARY KEY,
  nome_medicamento VARCHAR(100) NOT NULL,
  dosagem VARCHAR(50),
```

```
  frequencia VARCHAR(50),
  instrucoes TEXT
);
```

4.3. Tabela Administração

```
CREATE TABLE ADMINISTRACAO (
  id_administracao SERIAL PRIMARY KEY,
  data_adm DATE,
  horario TIME,
  observacoes TEXT,
  id_paciente INT,
  id_medicamento INT,
  FOREIGN KEY (id_paciente) REFERENCES PACIENTE(id_paciente),
  FOREIGN KEY (id_medicamento) REFERENCES
MEDICAMENTO(id_medicamento)
);
```

4.4. Tabela Salário

```
CREATE TABLE SALARIO (
  id_salario SERIAL PRIMARY KEY,
  mes VARCHAR(20),
  valor DECIMAL(10,2),
  observacoes TEXT,
  id_paciente INT,
  FOREIGN KEY (id_paciente) REFERENCES PACIENTE(id_paciente)
);
```

5. Manual de Uso

5.1. Inserção de Dados

- Para adicionar um novo paciente:

```
INSERT INTO PACIENTE (nome, idade, endereco, telefone)
VALUES ('João Silva', 45, 'Rua das Flores, 123', '11-98765-4321');
```

- Para adicionar um novo medicamento:

```
INSERT INTO MEDICAMENTO (nome_medicamento, dosagem, frequencia,
instrucoes)
VALUES ('Paracetamol', '500mg', '8 em 8 horas', 'Tomar após as
refeições');
```

5.2. Consultas

- Para listar todos os pacientes:

```
SELECT * FROM PACIENTE;
```

- Para listar os medicamentos administrados a um paciente:

```
SELECT p.nome, m.nome_medicamento, a.data_adm, a.horario
FROM ADMINISTRACAO a
JOIN PACIENTE p ON a.id_paciente = p.id_paciente
JOIN MEDICAMENTO m ON a.id_medicamento = m.id_medicamento;
```

6. Atualizações e Manutenção

- Para atualizar o valor de um salário registrado:

```
UPDATE SALARIO
SET valor = 1500.00
WHERE id_salario = 1;
```

- Para remover um paciente do banco de dados:

```
DELETE FROM PACIENTE
WHERE id_paciente = 1;
```