Health Clinic

SENAI . SP

TÉCNICO EM DESENVOLVIMENTO

Documentação

Lucas Bianchezzi Oliveira

Projeto finalização Sprint 2 - Banco de dados

SENAI

Sumário

1.	Resumo	3
2.	Descrição do projeto	3
3.	Banco de dados relacional	4
4.	Modelagem de dados	5
	Modelo Conceitual	5
	Modelo Lógico	6
	Modelo Físico	7
	Cronograma	7
5.	Tipos do SQL	8
	Data Definition Language (DDL)	8
	Data Manipulation Language (DML)	9
	Data Query Language (DQL)	10
	Trello	12

1. Resumo

Esse projeto foi criado com o principal intuito em colocar em prática as habilidades compreendidas na Sprint de banco de dados. Consiste em uma modelagem completa de um sistema de administração de uma clínica de saúde.

2. Descrição do projeto

Este projeto diz respeito a uma clínica de saúde denominada Health Clinic. Esta clínica buscava um sistema para facilitar o gerenciamento do sistema, que era realizado por meio de planilhas, mas que com o crescimento da clínica, passaram a ter problemas com o gerenciamento.

Com isso, foi criado uma modelagem completa do sistema por meio da ferramenta “draw.io”. Nele, foi inserido os modelos conceitual, lógico e físico. Os quais tiveram a aplicação do conceito de normalização para que haja uma base mais sólida e eficaz.

Além disso, para melhor organização do projeto, foi realizado o controle dos passos conseguintes por meio do Trello. Que permitiu o controle das tarefas pendentes e alterações que vieram a ocorrer durante o projeto, com uma divisão bem definida e organizada das áreas de cada função.

Por fim, no SQL SERVER foi realizado o DDL (Data Definition Language), DML (Data Manipulation Language) e DQL (Data Query Language). Com base nas solicitações do cliente por meio do storytelling.

3. Banco de dados relacional

Neste projeto utilizamos o banco de dados relacional SQL SERVER, que utiliza a linguagem SQL.

Um banco de dados relacional é aquele que possui uma gama de relações entre suas entidades. Nele, os dados são armazenados dentro de tabelas, que possuem um esquema pré-definido e com maior rigidez, garantindo maior organização e evitando problemas de inconsistências, garantindo que os dados sejam inseridos nos locais corretos. Além de garantirem o que chamamos de ACID: atomicidade, consistência, isolamento e durabilidade.

Um banco de dados é importantíssimo exatamente por ser a base de todo projeto que trabalha com armazenamento de dados. Ele permite que armazenamos informações e organizemo-os ou não em formatos definidos.

4. Modelagem de dados

A modelagem de dados consiste no planejamento e representação visual dos esquemas do sistema. Permitindo que os que tiverem acesso, possuam uma visão ampla, objetiva e unificada do fluxo dos dados, suas relações e cardinalidades .

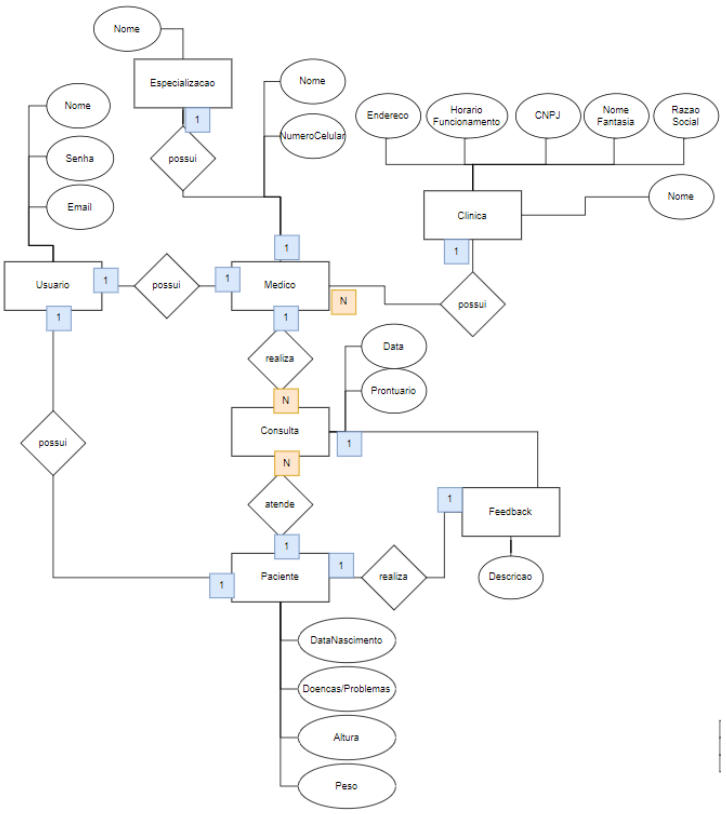
Neste projeto, foi utilizado o modelo MER (Modelo Entidade Relacionamento). Que define a estrutura da aplicação, divida em 3 modelos: Modelo conceitual, Modelo lógico e Modelo físico.

Este processo pode ser realizado por diversos sistemas de modelagem. Neste projeto foi utilizado o draw.io.

Modelo Conceitual

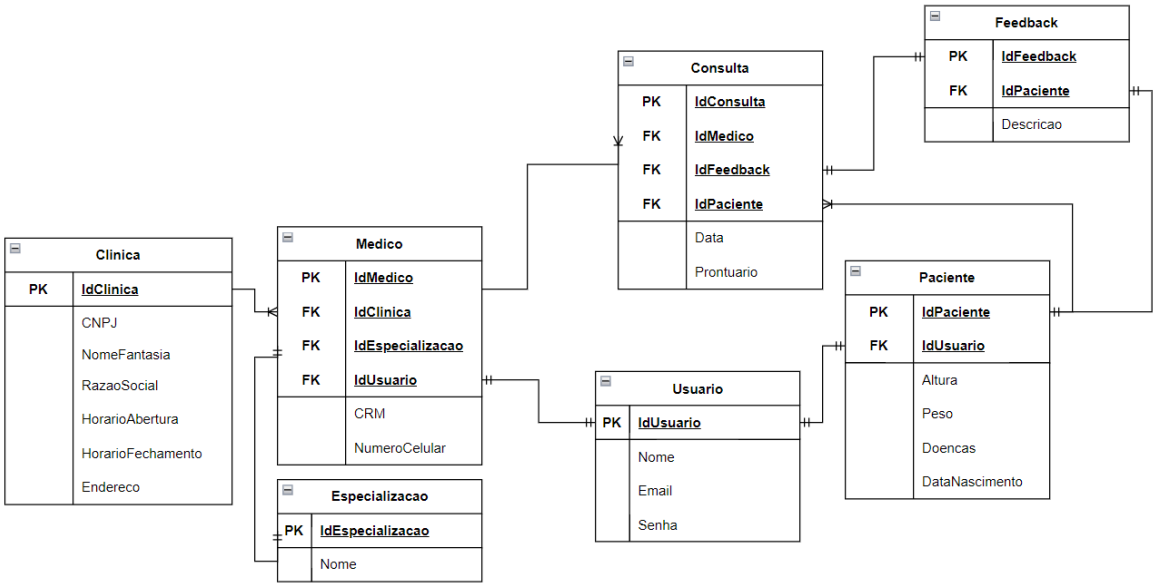
O modelo conceitual tem como principal objetivo a identificação de todas as entidades, relações e cardinalidades do projeto, com base nas regras de negócio. Servindo mais como um modelo conceitual para organização do modelo.

Primeiro foi realizado a identificação de cada entidade no Storytelling e inserido no draw.io. Em seguida, foi observado os relacionamentos entre as entidades, imposto as cardinalidades de cada e então, os atributos de cada entidade.



Modelo Lógico

No modelo lógico, o principal objetivo é a identificação da relação entre as entidades, por meio das chaves (primárias e estrangeiras).



Baseado no modelo conceitual, foi identificado as chaves primárias e estrangeiras de cada entidade e então, aplicado o modelo lógico.

Modelo Físico

O modelo físico (ou teste de mesa) é literalmente um teste do banco de dados, onde é criado tabelas e inserido dados no formato em que deveria ficar no banco. Servindo de prova real para confirmação de que a modelagem do banco está correta.

Neste, há 2 formas de criação: por meio de tabelas ou criando o banco de dados e exportando-o.

Clinica

IdClinica	CNPJ	NomeFantasia	RazaoSocial	HorarioAbertura	HorarioFechamento	Endereco
1	91.861.423/0001-51	HelathClinic	Rede Health's Life	04:00	02:00	Rua engenheiro Garcia, 258 - SCS

Especializacao

IdEspecializacao	Nome
1	Cardiologista
2	Otorrinolaringologista

Usuario

IdUsuario	Email	Senha	Nome	DataNascimento
1	doutorjon@gmail.com	doutorjon123	Elton Jon	1989-02-03
2	giu@gmail.com	giu123	Giulia Santos	2005-05-26

Medico

IdMedico	IdClinica	IdEspecializacao	IdUsuario	CRM	NumeroCelular
1	1	2	1	932184/SP	+55 (11) 95684-5521

Paciente

IdPaciente	IdUsuario	Altura	Peso	Doencas
1	2	1,71	57	

Consulta

IdConsulta	IdMedico	IdPaciente	IdFeedback	Data	Horario	Prontuario
1	1	1	1	06/04/2013	10:05	Esse é um prontuário...
2	1	1		15/03/2023	15:20	Esse é um outro prontuário...

Feedback

IdFeedback	IdPaciente	Titulo	Descricao
1	1	Excelente!!	Médico extremamente profissional!

Cronograma

	Dia 1	Dia 2	Dia 3	Dia 4	Dia 5
Formalização dos conceitos	X				
Modelo Conceitual	X				
Modelo Lógico		X			
Modelo Físico		X			
DDL			X		
DML			X		
Normalização				X	
DQL Documentação e entrega				X	X

5. Tipos do SQL

Os comandos de SQL são classificados por função e divididos em DDL, DML e DQL.

Data Definition Language (DDL)

A DDL é onde insere-se os comandos de gerenciamento de estruturas do banco de dados. Nele criamos todas as entidades com seus atributos.

```
-- DDL: Data Definition Language

-- Criar bd
CREATE DATABASE [HealthClinic];

-- abrir bd
USE [HealthClinic];

-- Criar tabelas


CREATE TABLE Clinica
(
    IdClinica INT PRIMARY KEY IDENTITY,
    CPNJ VARCHAR(20) NOT NULL,
    NomeFantasia VARCHAR(50),
    RazaoSocial VARCHAR(60) NOT NULL,
    HorarioAbertura DATE NOT NULL,
    HorarioFechamento DATE NOT NULL,
    Endereco VARCHAR(150) NOT NULL
);
--SELECT * FROM Clinica;
/*  Inserir se houver escalabilidade
CREATE TABLE Endereco
(
    IdEndereco INT PRIMARY KEY IDENTITY,
    Endereco VARCHAR(50) NOT NULL,
    Bairro VARCHAR(25) NOT NULL,
    Numero INT NOT NULL
);*/


CREATE TABLE Especializacao
(
    IdEspecializacao INT PRIMARY KEY IDENTITY,
    Nome VARCHAR(40) NOT NULL
);
--SELECT * FROM Especializacao;


CREATE TABLE Usuario
(
    IdUsuario INT PRIMARY KEY IDENTITY,
    Email VARCHAR(50) NOT NULL,
    Senha VARCHAR(20) NOT NULL,
    Nome VARCHAR(60) NOT NULL,
    DataNascimento DATE NOT NULL
);
--SELECT * FROM Usuario;


CREATE TABLE Paciente
(
    IdPaciente INT PRIMARY KEY IDENTITY,
    IdUsuario INT FOREIGN KEY REFERENCES Usuario(IdUsuario) NOT NULL,
    Altura DECIMAL(3, 2),
    Peso DECIMAL(4, 2),
    Doencas VARCHAR(200)
);
--SELECT * FROM Paciente;


CREATE TABLE Medico
(
    IdMedico INT PRIMARY KEY IDENTITY,
    IdClinica INT FOREIGN KEY REFERENCES Clinica(IdClinica) NOT NULL,
    IdEspecializacao INT FOREIGN KEY REFERENCES Especializacao(IdEspecializacao) NOT NULL,
    IdUsuario INT FOREIGN KEY REFERENCES Usuario(IdUsuario) NOT NULL,
    CRM VARCHAR(11) UNIQUE NOT NULL,
    NumeroCelular VARCHAR(25) NOT NULL
);
--SELECT * FROM Medico;


CREATE TABLE Consulta
(
    IdConsulta INT PRIMARY KEY IDENTITY,
    IdMedico INT FOREIGN KEY REFERENCES Medico(IdMedico) NOT NULL,
    IdPaciente INT FOREIGN KEY REFERENCES Paciente(IdPaciente) NOT NULL,
    IdFeedback INT FOREIGN KEY REFERENCES Feedback(IdFeedback),
    Data DATE NOT NULL,
    Horario TIME NOT NULL,
    Prontuario VARCHAR(MAX) NOT NULL
);
--SELECT * FROM Consulta;


CREATE TABLE Feedback
(
    IdFeedback INT PRIMARY KEY IDENTITY,
    IdPaciente INT FOREIGN KEY REFERENCES Paciente(IdPaciente) NOT NULL,
    Titulo VARCHAR(150) NOT NULL,
    Descricao VARCHAR(400) NOT NULL
);
```

Data Manipulation Language (DML)

No DML é onde é inserido os comandos de manipulação de dados (insert, update, delete). No caso abaixo, foi populado todas as entidades do banco de dados.

```sql
-- DML: Data Manipulation Language

 -- Entrar no banco de dados
USE HealthClinic;


 -- Inserir dados nas tabelas

 INSERT INTO Especializacao(Nome) VALUES('Cardiologista'), ('Otorrinolaringologista');

INSERT INTO Usuario(Email, Senha, Nome, DataNascimento) VALUES('doutorjon@gmail.com', 'doutorjon123', 'Elton Jon', '1989-02-03'),
 ('giu@gmail.com', 'giu123', 'Giulia Santos', '2007-06-10'),
 ('doutoramel@gmail.com', 'doutoramel123', 'Mel Eduarda', '04-02-2003'), ('carlos', 'carlos123', 'Carlos Costa', '1989-05-08');

INSERT INTO Clinica(CPNJ, NomeFantasia, RazaoSocial, HorarioAbertura, HorarioFechamento, Endereco) VALUES
 ('91.861.423/0001-51', 'HelathClinic', 'Rede Health''s Life', '04:00', '02:00', 'Rua engenheiro Garcia, 258 - SCS');

 INSERT INTO Paciente(IdUsuario, Altura, Peso) VALUES (1, 1.64, 63), (4, 1.70, 63);


INSERT INTO Medico(IdClinica, IdEspecializacao, IdUsuario, CRM, NumeroCelular) VALUES(1, 2, 1, '932184/SP', '+55 (11) 95684-5521'),
(1, 1, 3, '123456/SP', '+55 (11) 95684-5521');

INSERT INTO Feedback(IdPaciente, Titulo, Descricao) VALUES(1, 'Excelente!!', 'Médico extremamente profissional!');

INSERT INTO Consulta(IdMedico, IdFeedback, IdPaciente, Data, Horario, Prontuario) VALUES(1, 1, 1, '10-08-2023', '10:05:00', 'Este é um prontuário..');
INSERT INTO Consulta(IdMedico, IdPaciente, Data, Horario, Prontuario) VALUES(1, 1, '15-03-2023', '15:20:00', 'Este é um outro prontuário..');
```

Data Query Language (DQL)

No DQL é onde é realizada a consulta personalizada ou não de dados (select).

No projeto realizamos o dql conforme o pedido no classroom: https://classroom.google.com/u/1/c/NjE2NDA5NDEyODM4/m/NjE3MjM2NzIzNTA2/details

Criar script que exiba os seguintes dados:

- Id Consulta
- Data da Consulta
- Horário da Consulta
- Nome da Clínica
- Nome do Paciente
- Nome do Médico
- Especialidade do Médico
- CRM
- Prontuário ou Descrição
- FeedBack(Comentário da consulta)

```
-- DQL: Data Query Language

-- Entrar no banco de dados
USE HealthClinic;


-- Pesquisas personalizadas

SELECT
    Consulta.IdConsulta AS 'Id da Consulta',
    Consulta.Data 'Dia da Consulta',
    Consulta.Horario 'Horário da Consulta',
    Clinica.NomeFantasia AS 'Nome da Clínica',
    [Nome Paciente].Nome AS 'Nome do usuário',
    [Nome Medico].Nome AS 'Nome do médico',
    Especializacao.Nome AS 'Especialidade médica',
    Medico.CRM AS 'CRM do médico',
    Consulta.Prontuario AS 'Prontuário',
    Feedback.Descricao AS 'Comentários'

FROM
    Medico
INNER JOIN Clinica
ON Medico.IdClinica = Clinica.IdClinica
INNER JOIN Consulta
ON Medico.IdMedico = Consulta.IdMedico
INNER JOIN Paciente
ON Consulta.IdPaciente = Paciente.IdPaciente
INNER JOIN Usuario AS [Nome Paciente]
ON Paciente.IdUsuario = [Nome Paciente].IdUsuario
INNER JOIN Usuario AS [Nome Medico]
ON Medico.IdUsuario = [Nome Medico].IdUsuario
INNER JOIN Especializacao
ON Medico.IdEspecializacao = Especializacao.IdEspecializacao
LEFT JOIN Feedback -- retorna os que tiverem ou não comentários
ON Consulta.IdFeedback = Feedback.IdFeedback;
```

Criar função para retornar os médicos de uma determinada especialidade

```sql
-- Criar função para retornar os médicos de uma determinada especialidade

CREATE FUNCTION FiltrarMedicosPorEspecialidade
(
    @especialidade VARCHAR(80)
)
RETURNS TABLE
AS
RETURN
(
    SELECT
        Usuario.Nome AS 'Nome do Médico',
        Especializacao.Nome AS 'Especialização'
    FROM
        Especializacao
    INNER JOIN Medico
    ON Medico.IdEspecializacao = Especializacao.IdEspecializacao
    INNER JOIN Usuario
    ON Usuario.IdUsuario = Medico.IdUsuario
    WHERE Especializacao.Nome = @especialidade
);

-- Execução do método de Busca de Médicos por especialidade: FiltrarMedicosPorEspecialidade
SELECT *
FROM FiltrarMedicosPorEspecialidade('cardiologista');
```

Criar procedure para retornar a idade de um determinado usuário específico

```sql
-- Criar procedure para retornar a idade de um determinado usuário específico
CREATE PROCEDURE BuscarIdadeUsuario
@nomeUsuario VARCHAR(50)
AS
BEGIN
    -- Declaração da variável que receberá a data de nascimento do usuário pesquisado
    DECLARE @ano INT

    -- Busca a data de nascimento do usuário pesquisado e armazena na variável @ano
    SELECT @ano = YEAR(Usuario.DataNascimento)
    FROM Usuario
    WHERE Usuario.Nome = @nomeUsuario

    -- Variável que recebe o ano atual
    DECLARE @anoAtual INT = YEAR(GETDATE());

    -- Calcula a idade do usuário
    DECLARE @idadePesquisada INT = @anoAtual - @ano
    PRINT 'A idade do usuário ' + CAST(@nomeUsuario AS VARCHAR(50)) + ' é: ' + CAST(@idadePesquisada AS VARCHAR(50));
END;

EXEC BuscarIdadeUsuario 'Giulia Santos';



SELECT * FROM Clinica;
SELECT * FROM Consulta;
SELECT * FROM Especializacao;
SELECT * FROM Medico;
SELECT * FROM Paciente;
SELECT * FROM Usuario;
SELECT * FROM Feedback;
```

Trello

Link do quadro Trello: https://trello.com/b/PIsABqia/health-clinic