**BANCO DE DADOS**

**Trabalho – Relatório**

|  |  |
| --- | --- |
| **Curso:** | CST ANÁLISE E DESENVOLVIMENTO DE SISTEMAS - DISTÂNCIA |
| **Aluno(a):** | Emanuel Rosa Zolet |
| **RU:** | 4557826 |

**1ª Etapa – Modelagem**

**Pontuação:** 25 pontos.

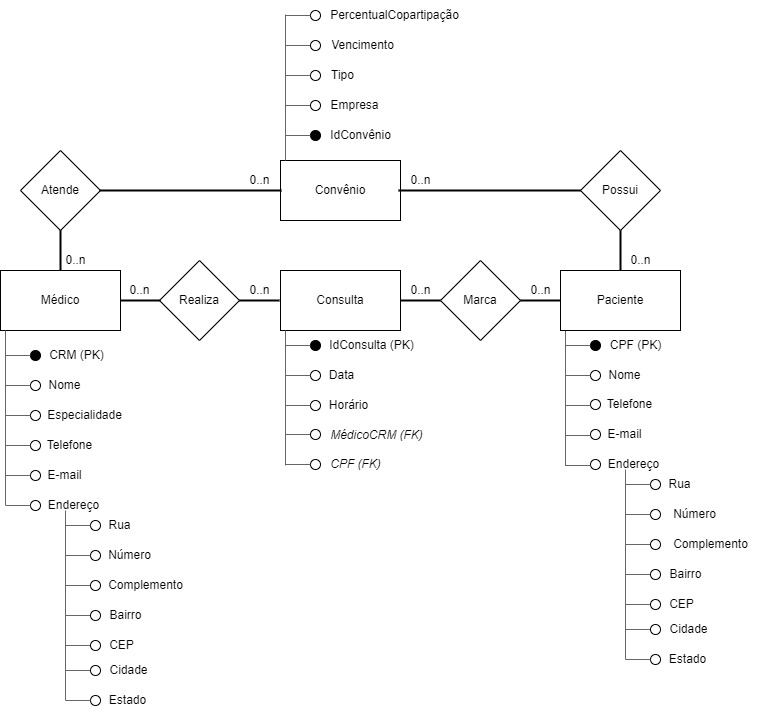
Dadas as regras de negócio abaixo listadas, referentes ao estudo de caso de uma Clínica Médica, elabore o Modelo Entidade-Relacionamento (MER), isto é, o modelo conceitual.

O Modelo Entidade-Relacionamento (MER) deve contemplar os seguintes itens:

* Entidades;
* Atributos;
* Relacionamentos;
* Cardinalidades;
* Chaves primárias;
* Chaves estrangeiras.

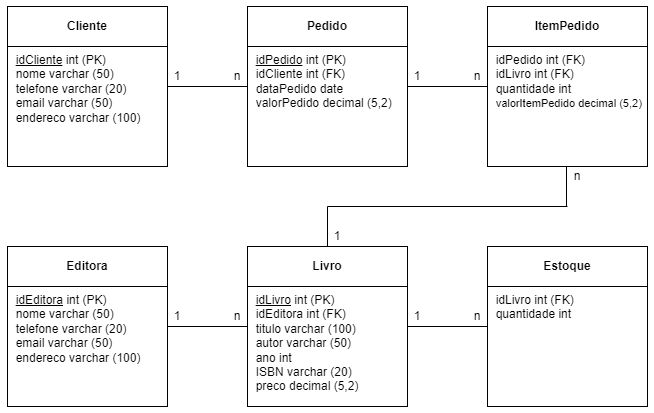
Uma Clínica Médica necessita controlar os dados das consultas realizadas. Para isso, contratou um profissional de Banco de Dados, a fim de modelar o Banco de Dados que armazenará os dados das consultas.

As regras de negócio são:

* Médico – Deverão ser armazenados os seguintes dados: CRM, especialidade, nome, telefone, e-mail e endereço, sendo o endereço composto por rua, número, complemento, bairro, CEP, cidade e estado;
* Consulta – Deverão ser armazenados os seguintes dados: identificação da consulta, data e horário;
* Paciente – Deverão ser armazenados os seguintes dados: CPF, nome, telefone, e-mail e endereço, sendo o endereço composto por rua, número, complemento, bairro, CEP, cidade e estado;
* Convênio – Deverão ser armazenados os seguintes dados: identificação do convênio, empresa, tipo, vencimento e percentual de coparticipação;
* Um médico pode realizar zero ou várias consultas, assim como zero ou várias consultas podem ser realizadas por um médico;
* Um médico pode atender zero ou vários convênios, assim como zero ou vários convênios podem ser atendidos por um médico;
* Um paciente pode marcar zero ou várias consultas, assim como zero ou várias consultas podem ser marcadas por um paciente;
* ****Um paciente pode possuir zero ou vários convênios, assim como zero ou vários convênios podem pertencer a um paciente.

**2ª Etapa – Implementação**

Considere o seguinte Modelo Relacional (modelo lógico), referente ao estudo de caso de uma Livraria:



Com base no Modelo Relacional dado e utilizando a *Structured Query Language* (SQL), no MySQL Workbench, implemente o que se pede.

**Observação:** Para testar o Banco de Dados após a implementação, utilize os comandos contidos no arquivo “Trabalho – Populando o Banco de Dados” para popular as tabelas. Tal arquivo contém todos os comandos de inserção dos dados (fictícios) necessários para a realização dos testes.

**Pontuação:** 25 pontos.

1. Implemente um Banco de Dados chamado “Livraria”. Após, implemente as tabelas, conforme o Modelo Relacional dado, observando as chaves primárias e as chaves estrangeiras. Todos os campos, de todas as tabelas, não podem ser nulos (*not null*).

-- CRIAÇÂO DO BANCO

CREATE DATABASE Livraria;

USE Livraria;

-- Criação das tabelas

CREATE TABLE Editora(

idEditora int PRIMARY KEY NOT NULL UNIQUE,

nome varchar(50) NOT NULL,

telefone varchar(20) NOT NULL,

email varchar(50) NOT NULL,

endereco varchar(100) NOT NULL

);

CREATE TABLE Cliente (

idCliente int PRIMARY KEY NOT NULL,

nome varchar(50) NOT NULL,

telefone varchar(20) NOT NULL,

email varchar(50) NOT NULL,

endereco varchar(100) NOT NULL

);

CREATE TABLE Pedido (

idPedido int PRIMARY KEY NOT NULL UNIQUE,

idCliente int NOT NULL,

dataPedido date NOT NULL,

valorPedido decimal(5,2) NOT NULL,

CONSTRAINT idCliente\_FK FOREIGN KEY (idCliente) REFERENCES Cliente(idCliente)

);

CREATE TABLE Livro (

idLivro int PRIMARY KEY NOT NULL UNIQUE,

idEditora int NOT NULL,

titulo varchar(100) NOT NULL,

autor varchar(50) NOT NULL,

ano int NOT NULL,

ISBN varchar(20) NOT NULL,

preco decimal(5,2) NOT NULL,

CONSTRAINT idEditora\_FK FOREIGN KEY (idEditora) REFERENCES Editora(idEditora)

);

CREATE TABLE Estoque (

idLivro int NOT NULL,

quantidade int NOT NULL,

CONSTRAINT idLivro\_FK FOREIGN KEY (idLivro) REFERENCES Livro(idLivro)

);

CREATE TABLE ItemPedido (

idPedido int NOT NULL,

idLivro int NOT NULL,

quantidade int NOT NULL,

valorItemPedido decimal(5,2) NOT NULL,

CONSTRAINT itemIdPedido\_FK FOREIGN KEY (idPedido) REFERENCES Pedido(idPedido),

CONSTRAINT itemIdLivro\_FK FOREIGN KEY (idLivro) REFERENCES Livro(idLivro)

);

**Pontuação:** 10 pontos.

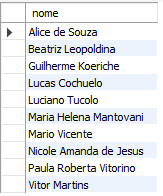
1. Implemente uma consulta para listar o quantitativo de livros cadastrados, independentemente da editora.

SELECT COUNT(idLivro) AS totalLivrosCadastrados

FROM Livro;

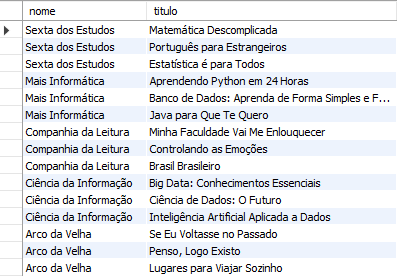
**Pontuação:** 10 pontos.

1. Implemente uma consulta para listar o nome dos clientes cadastrados. A listagem deve ser mostrada em ordem crescente.

SELECT nome FROM Cliente ORDER BY nome ASC;

**Pontuação:** 10 pontos.

1. Implemente uma consulta para listar o nome de todas as editoras e os títulos de seus respectivos livros. A listagem deve ser mostrada em ordem decrescente pelo nome das editoras.

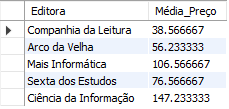
SELECT E.nome, L.titulo FROM Editora E INNER JOIN Livro L ON E.idEditora = L.idEditora ORDER BY E.nome DESC;

**Pontuação:** 10 pontos.

1. Implemente uma consulta para listar o nome das editoras e a média de preço de seus respectivos livros. Para isso, utilize o comando *group by*.

SELECT E.nome AS Editora, AVG(L.preco) AS Média\_Preço FROM Editora E

INNER JOIN Livro L ON E.idEditora = L.idEditora

GROUP BY E.nome;

**Pontuação:** 10 pontos.

1. Implemente uma consulta para listar o nome de todos os clientes e a quantidade de títulos de livros comprados pelos mesmos. Para isso, utilize o comando *group by*.

SELECT C.nome, COUNT(ip.idLivro) AS Titulos\_Comprados

FROM Cliente C INNER JOIN Pedido P ON C.idCliente = P.idCliente

INNER JOIN ItemPedido IP ON P.idPedido = IP.idPedido

GROUP BY C.nome;