

## BANCO DE DADOS

### Trabalho – Relatório

<b>Curso:</b>	Análise e Desenvolvimento de Sistemas
<b>Aluno(a):</b>	Ana Eliza Carvalho Palu
<b>RU:</b>	5355412

#### 1. 1ª Etapa – Modelagem

**Pontuação:** 30 pontos.

Dadas as regras de negócio abaixo listadas, referentes ao estudo de caso de uma Rede de Hotéis, 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 Rede de Hotéis necessita controlar os dados dos funcionários, das unidades, dos quartos, dos hóspedes, das reservas e dos pagamentos. Para isso, contratou um profissional de Banco de Dados, a fim de modelar o Banco de Dados que armazenará todos os dados.

As regras de negócio são:

- Funcionário – Deverão ser armazenados os seguintes dados: CPF, nome, telefone, e-mail, login e senha;

- Hotel – Deverão ser armazenados os seguintes dados: identificação do hotel, nome, categoria, telefone, e-mail e endereço, sendo o endereço composto por rua, número, complemento, bairro, CEP, cidade e estado;
- Quarto – Deverão ser armazenados os seguintes dados: identificação do quarto, número de leitos, tipo (*standard*, luxo ou suíte), preço da diária e *status* (disponível, ocupado ou manutenção);
- Hóspede – 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;
- Reserva – Deverão ser armazenados os seguintes dados: identificação da reserva, data de entrada, data de saída e *status* (ativa, cancelada ou concluída);
- Pagamento – Deverão ser armazenados os seguintes dados: identificação do pagamento, forma de pagamento (cartão, pix ou dinheiro), data do pagamento, valor total e *status* (pago ou pendente);
- Um hotel possui um ou vários quartos;
- Um ou vários funcionários trabalham em um hotel;
- Um funcionário realiza uma ou várias reservas;
- Um ou vários quartos fazem parte de uma ou várias reservas;
- Um hóspede pode fazer uma ou várias reservas;
- Uma reserva gera um pagamento.

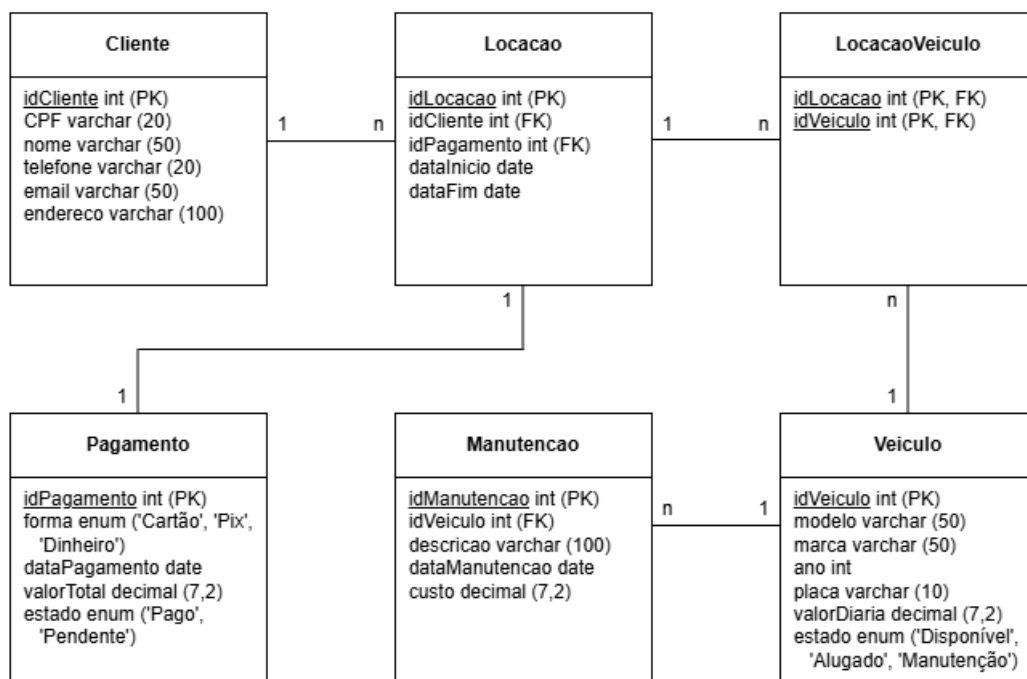
**Importante:**

- O Modelo Entidade-Relacionamento (MER) deve considerar somente as regras de negócio dadas, não podendo ser criada nenhuma outra entidade ou atributo que não estejam nas regras de negócio;
- Em caso de haver entidade associativa, a mesma deve ser representada pela “Representação 1” (texto da Aula 1 – Fundamentos de Banco de Dados, Figura 25);
- Em caso de haver cardinalidade (1,1), a chave estrangeira deve fazer parte da entidade que possui o maior número de chaves estrangeiras.



## 2. 2ª Etapa – Implementação

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



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

**Importante:** 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:** 30 pontos.

1. Implemente um Banco de Dados chamado “LocadoraVeiculos”. 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*).

```
create database LocadoraVeiculos;
```

```
use LocadoraVeiculos;
```

```
CREATE TABLE Cliente (
```

```
    idCliente INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
```

```
    CPF VARCHAR(20) NULL,
```

```
Nome VARCHAR(50) NULL,  
Telefone VARCHAR(20) NULL,  
Email VARCHAR(50) NULL,  
Endereco VARCHAR(100) NULL,  
PRIMARY KEY(idCliente)  
);
```

```
CREATE TABLE Locacao (  
    idLocacao INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,  
    idCliente INTEGER UNSIGNED NOT NULL,  
    idPagamento INTEGER UNSIGNED NOT NULL,  
    dataInicio DATE NULL,  
    dataFim DATE NULL,  
    PRIMARY KEY(idLocacao),  
    INDEX Locação_FKIndex1(idCliente)  
);
```

```
CREATE TABLE LocacaoVeiculo (  
    idLocacao INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,  
    idVeiculo INTEGER UNSIGNED NOT NULL,  
  
    PRIMARY KEY(idLocacao),  
    INDEX LocacaoVeiculo_FKIndex1(idLocacao),  
    INDEX LocacaoVeiculo_FKIndex2(idVeiculo)  
);
```

```
CREATE TABLE Manutencao (  
    idManutencao INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,  
    idVeiculo INTEGER UNSIGNED NOT NULL,  
    Descricao VARCHAR(100) NULL,  
    DataManutencao DATE NULL,  
    Custo DECIMAL(7,2) NULL,  
    PRIMARY KEY(idManutencao),
```

```
INDEX Manutencao_FKIndex1(idVeiculo)
);
```

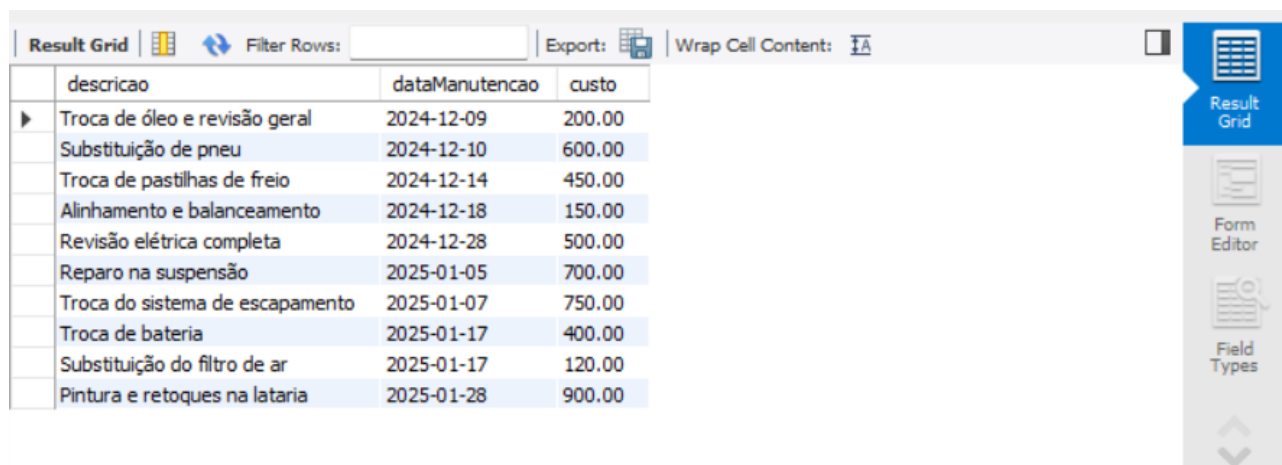
```
CREATE TABLE Pagamento (
  idPagamento INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  Forma ENUM('Cartão','Pix','Dinheiro') NULL,
  dataPagamento DATE NULL,
  valorTotal DECIMAL(7,2) NULL,
  estado ENUM('Pago','Pendente') NULL,
  PRIMARY KEY(idPagamento)
);
```

```
CREATE TABLE Veiculo (
  idVeiculo INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
  Modelo VARCHAR(50) NULL,
  Marca VARCHAR(50) NULL,
  Ano INTEGER UNSIGNED NULL,
  Placa VARCHAR(10) NULL,
  ValorDiaria DECIMAL(7,2) NULL,
  estado ENUM('Disponivel','Alugado','Manutenção') NULL,
  PRIMARY KEY(idVeiculo)
);
```

**Pontuação:** 10 pontos.

2. Implemente uma consulta para listar a descrição, a data e o custo de todas as manutenções realizadas nos veículos.

```
SELECT descricao, dataManutencao, custo FROM Manutencao;
```

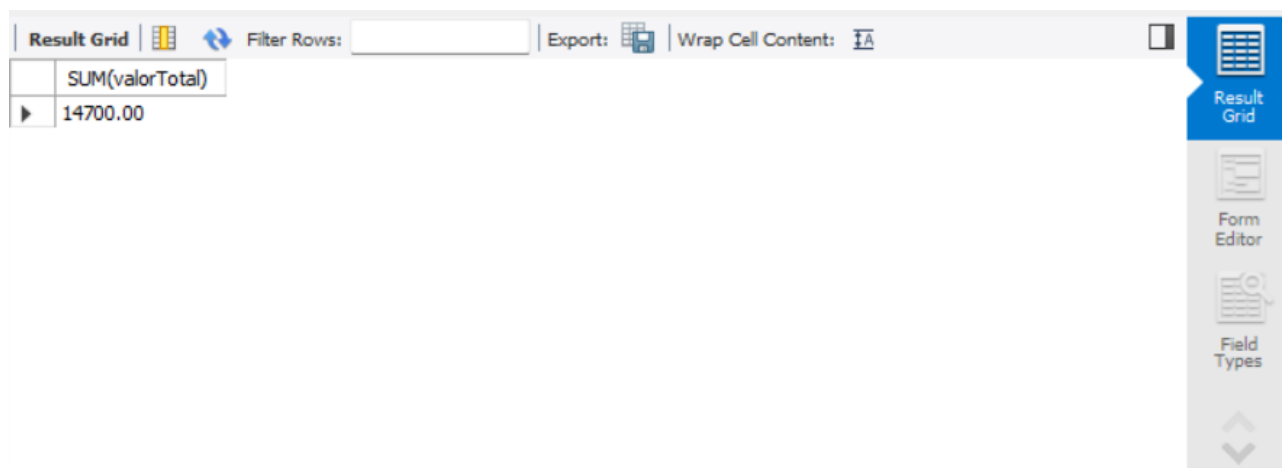


	descricao	dataManutencao	custo
▶	Troca de óleo e revisão geral	2024-12-09	200.00
	Substituição de pneu	2024-12-10	600.00
	Troca de pastilhas de freio	2024-12-14	450.00
	Alinhamento e balanceamento	2024-12-18	150.00
	Revisão elétrica completa	2024-12-28	500.00
	Reparo na suspensão	2025-01-05	700.00
	Troca do sistema de escapamento	2025-01-07	750.00
	Troca de bateria	2025-01-17	400.00
	Substituição do filtro de ar	2025-01-17	120.00
	Pintura e retoques na lataria	2025-01-28	900.00

**Pontuação:** 10 pontos.

3. Implemente uma consulta para listar o valor total arrecadado pela locadora.  
Lembre-se que pagamentos “pendentes” não fazem parte da soma.

```
SELECT SUM(valorTotal)
FROM Pagamento
WHERE estado = 'Pago';
```

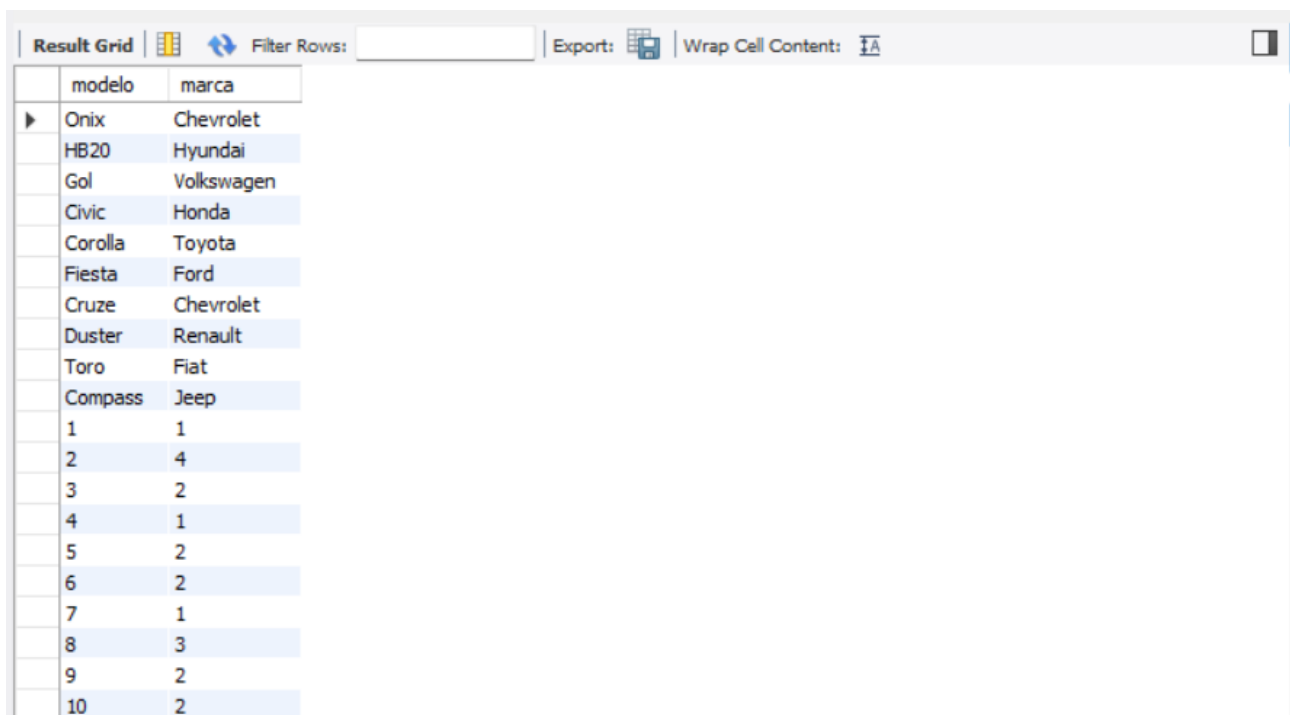


	SUM(valorTotal)
▶	14700.00

**Pontuação:** 10 pontos.

4. Implemente uma consulta para listar o modelo e a marca dos veículos, bem como o número de vezes que cada um foi locado. A listagem deve ser mostrada em ordem decrescente pelo número de alugueis.  
Dica: Utilize a cláusula *group by*.

```
SELECT modelo, marca FROM Veiculo
UNION
SELECT
    idVeiculo,
    COUNT(*) AS quantidade_locacoes
FROM LocacaoVeiculo GROUP BY idVeiculo;
```



	modelo	marca
▶	Onix	Chevrolet
	HB20	Hyundai
	Gol	Volkswagen
	Civic	Honda
	Corolla	Toyota
	Fiesta	Ford
	Cruze	Chevrolet
	Duster	Renault
	Toro	Fiat
	Compass	Jeep
	1	1
	2	4
	3	2
	4	1
	5	2
	6	2
	7	1
	8	3
	9	2
	10	2

**Pontuação:** 10 pontos.

5. Implemente uma consulta para listar o nome dos clientes que possuem pagamento “pendente”, bem como o valor devido por eles. A listagem deve ser mostrada em ordem alfabética crescente pelo nome dos clientes.

Dica: Utilize a cláusula *group by*.

**Cole o código e o *print* resultante da consulta aqui.**