

BANCO DE DADOS

Trabalho – Relatório

Curso:	BACHARELADO EM ENGENHARIA DE SOFTWARE - DISTÂNCIA
Aluno(a):	Andrey Ricardo Da Maia
RU:	5244478

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.

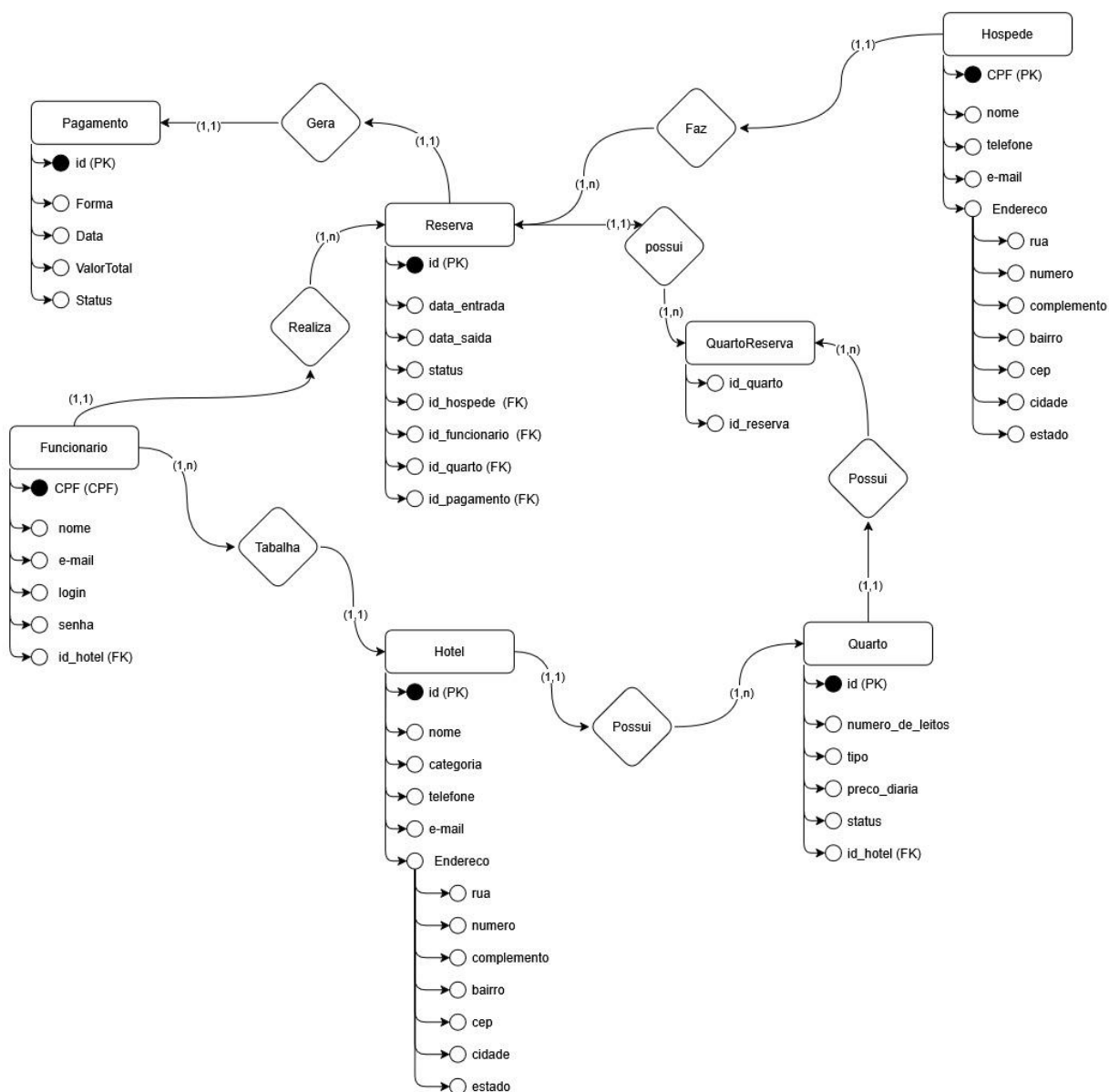
As regras de negócio são:

- 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.

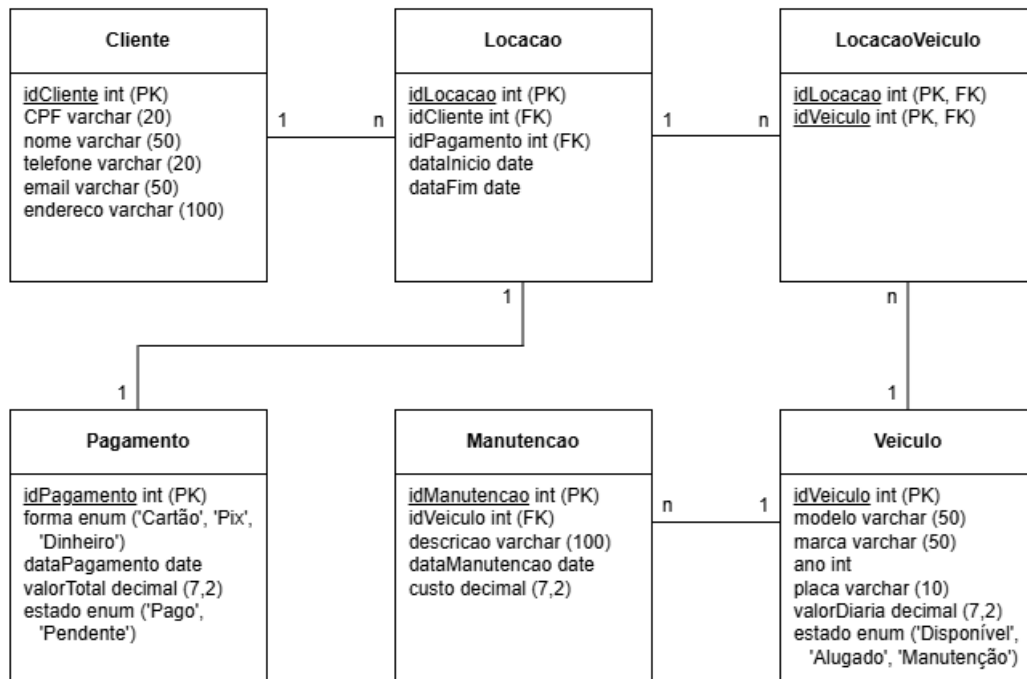
Importante:

- 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 IF NOT EXISTS LocadoraVeiculos;
USE LocadoraVeiculos;
```

```
CREATE TABLE IF NOT EXISTS Cliente (
    idCliente INT AUTO_INCREMENT PRIMARY KEY,
    CPF VARCHAR(20),
    nome VARCHAR(50),
    telefone VARCHAR(20),
    email VARCHAR(50),
    endereco VARCHAR(100)
```

);

```
CREATE TABLE IF NOT EXISTS Pagamento (  
    idPagamento INT AUTO_INCREMENT PRIMARY KEY,  
    forma ENUM('Cartão', 'Pix', 'Dinheiro'),  
    dataPagamento DATE,  
    valorTotal DECIMAL(7, 2),  
    estado ENUM('Pago', 'Pendente')  
);
```

```
CREATE TABLE IF NOT EXISTS Locacao (  
    idLocacao INT AUTO_INCREMENT PRIMARY KEY,  
    idCliente INT,  
    idPagamento INT,  
    dataInicio DATE,  
    dataFim DATE,  
    FOREIGN KEY (idCliente) REFERENCES Cliente(idCliente),  
    FOREIGN KEY (idPagamento) REFERENCES Pagamento(idPagamento)  
);
```

```
CREATE TABLE IF NOT EXISTS Veiculo (  
    idVeiculo INT AUTO_INCREMENT PRIMARY KEY,  
    modelo VARCHAR(50),  
    marca VARCHAR(50),  
    ano INT,  
    placa VARCHAR(10),  
    valorDiaria DECIMAL(7, 2),  
    estado ENUM('Disponível', 'Alugado', 'Manutenção')  
);
```

```
CREATE TABLE IF NOT EXISTS LocacaoVeiculo (  
    idLocacao INT,  
    idVeiculo INT,  
    FOREIGN KEY (idLocacao) REFERENCES Locacao(idLocacao),  
    FOREIGN KEY (idVeiculo) REFERENCES Veiculo(idVeiculo)  
);
```

```
CREATE TABLE IF NOT EXISTS Manutencao (  
    idManutencao INT AUTO_INCREMENT PRIMARY KEY,  
    idVeiculo INT,  
    descricao VARCHAR(100),  
    dataManutencao DATE,  
    custo DECIMAL(7, 2),  
    FOREIGN KEY (idVeiculo) REFERENCES Veiculo(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 m.descricao, m.dataManutencao, m.custo
from Manutencao m
```

Query 1 x

Limit to 1000 rows

1 • select m.descricao, m.dataManutencao, m.custo

2 from Manutencao m

	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

Result Grid

Form Editor

Field Types

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'
```

Query 1 x

Limit to 1000 rows

1 • select sum(valorTotal)

2 from Pagamento

3 where estado = 'Pago'

	sum(valorTotal)
▶	14700.00

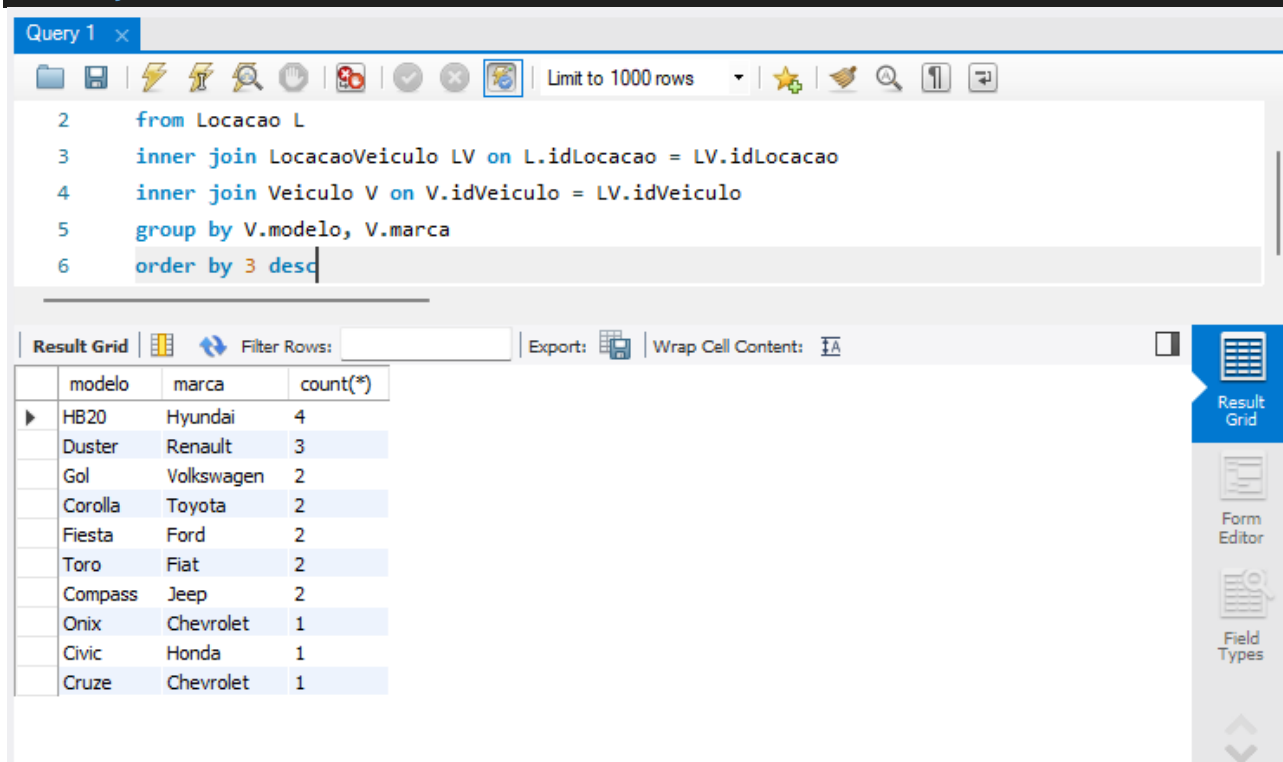
Result Grid

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 aluguéis.

Dica: Utilize a cláusula *group by*.

```
select V.modelo, V.marca, count(*)  
from Locacao L  
inner join LocacaoVeiculo LV on L.idLocacao = LV.idLocacao  
inner join Veiculo V on V.idVeiculo = LV.idVeiculo  
group by V.modelo, V.marca  
order by 3 desc
```



The screenshot shows a database query tool interface. The top toolbar includes icons for file operations, execution, and settings. The query editor displays the following SQL code:

```
2 from Locacao L  
3 inner join LocacaoVeiculo LV on L.idLocacao = LV.idLocacao  
4 inner join Veiculo V on V.idVeiculo = LV.idVeiculo  
5 group by V.modelo, V.marca  
6 order by 3 desc
```

Below the query editor, the 'Result Grid' tab is active, showing the results of the query in a table:

modelo	marca	count(*)
HB20	Hyundai	4
Duster	Renault	3
Gol	Volkswagen	2
Corolla	Toyota	2
Fiesta	Ford	2
Toro	Fiat	2
Compass	Jeep	2
Onix	Chevrolet	1
Civic	Honda	1
Cruze	Chevrolet	1

On the right side of the interface, there are buttons for 'Result Grid', 'Form Editor', and 'Field Types'.

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*.

```
select C.nome, sum(P.valorTotal)  
from Pagamento P  
inner join Locacao L on L.idPagamento = P.idPagamento  
inner join Cliente C on C.idCliente = L.idCliente  
where P.estado = 'Pendente'  
group by C.nome
```

order by C.nome asc

Query 1 x

Limit to 1000 rows

```
3 inner join Locacao L on L.idPagamento = P.idPagamento
4 inner join Cliente C on C.idCliente = L.idCliente
5 where P.estado = 'Pendente'
6 group by C.nome
7 order by C.nome asc
```

Result Grid

Filter Rows:

Export:

Wrap Cell Content:

	nome	sum(P.valorTotal)
▶	João da Silva	880.00
	Lucas Martins	2220.00
	Pedro dos Santos	280.00

Result Grid