

BANCO DE DADOS

Trabalho – Relatório

Curso:	Engenharia de Software
Aluno(a):	Iasmin Mara Kubis
RU:	2805428

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.

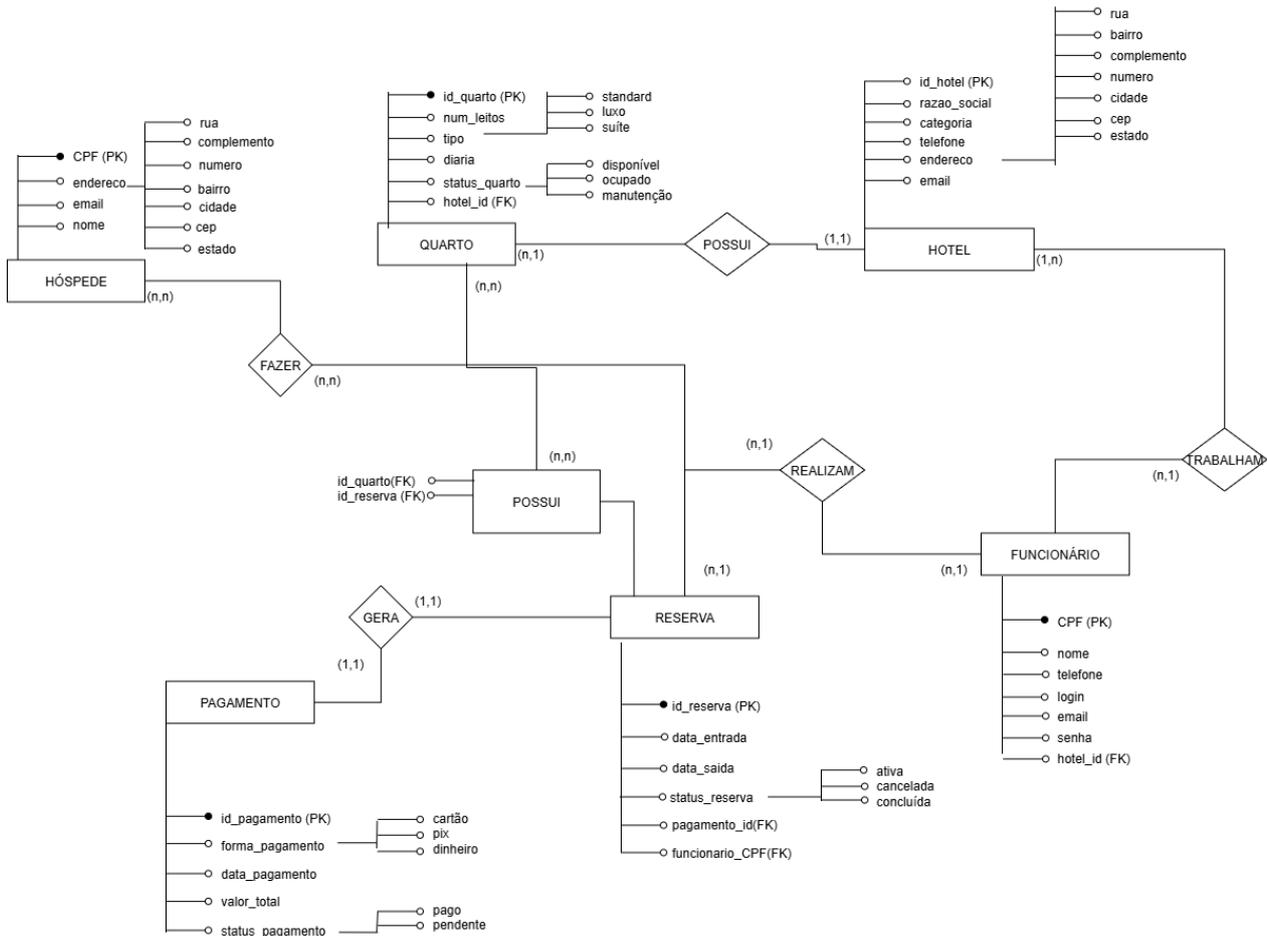
As regras de negócio são:

- 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.

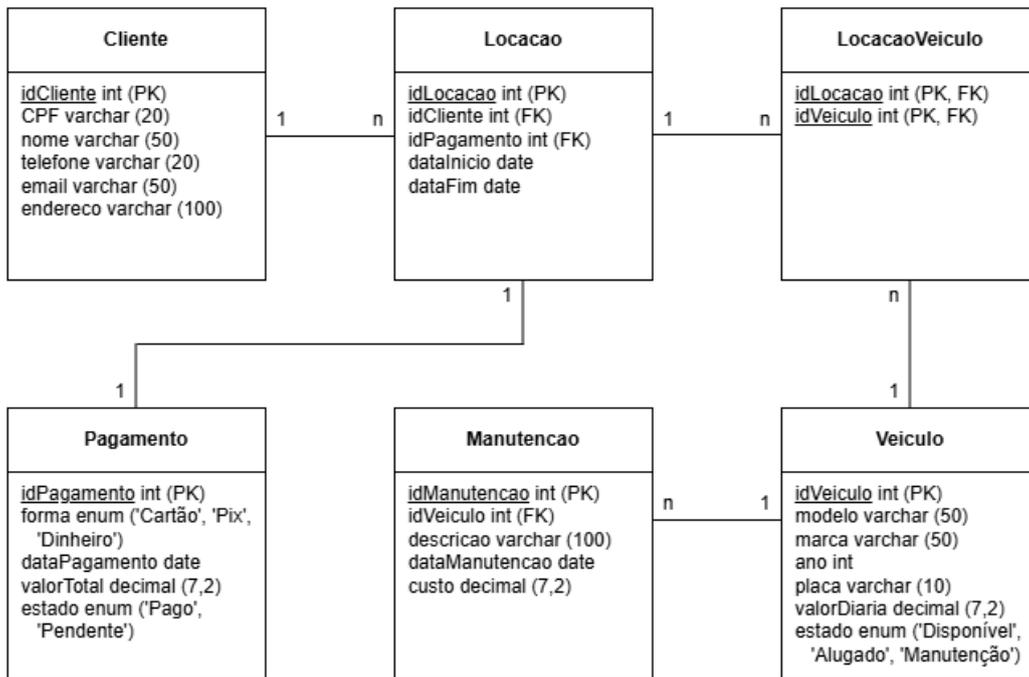
Importante:

- 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;
```

-- Tabela Cliente

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

-- Tabela Pagamento

```
CREATE TABLE Pagamento (  
    idPagamento INT PRIMARY KEY AUTO_INCREMENT,  
    forma ENUM('Cartao', 'Pix', 'Dinheiro') NOT NULL,  
    dataPagamento DATETIME NOT NULL,  
    valorTotal DECIMAL(7,2) NOT NULL,  
    estado ENUM('Pago', 'Pendente') NOT NULL  
);
```

-- Tabela Locacao

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

-- Tabela Veiculo

```
CREATE TABLE Veiculo (  
    idVeiculo INT PRIMARY KEY AUTO_INCREMENT,
```

```
modelo VARCHAR(50) NOT NULL,  
marca VARCHAR(50) NOT NULL,  
ano INT NOT NULL,  
placa VARCHAR(10) NOT NULL,  
valorDiaria DECIMAL(7,2) NOT NULL,  
estado ENUM('Disponivel', 'Alugado', 'Manutencao') NOT NULL  
);
```

-- Tabela LocacaoVeiculo (associativa N:N)

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

-- Tabela Manutencao

```
CREATE TABLE Manutencao (  
    idManutencao INT PRIMARY KEY AUTO_INCREMENT,  
    idVeiculo INT NOT NULL,  
    descricao VARCHAR(100) NOT NULL,  
    dataManutencao DATETIME NOT NULL,  
    custo DECIMAL(7,2) NOT NULL,  
    FOREIGN KEY (idVeiculo) REFERENCES Veiculo(idVeiculo)  
);
```

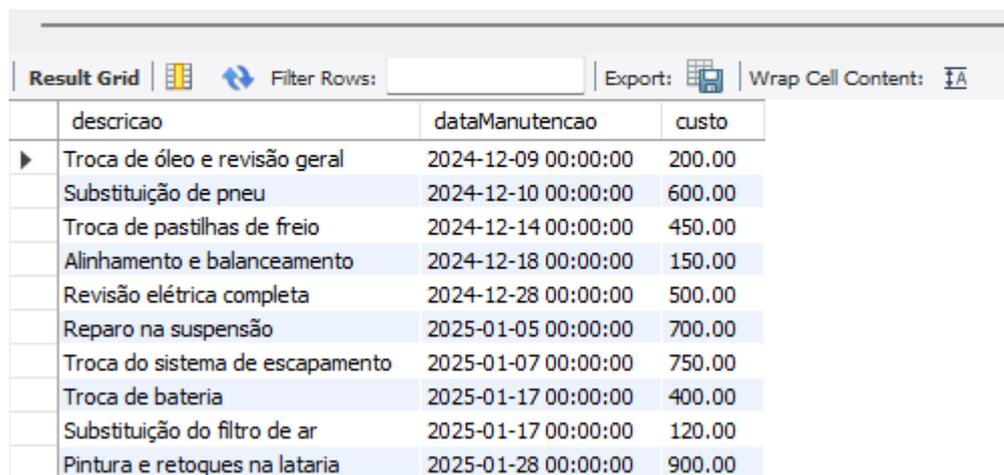
8	20:11:58	insert into Cliente (idCliente, CPF, nome, telefone, email, endereco) values (1, '045.159.785-1...	10 row(s) affected	Records: 10	Duplicates: 0	Warnings: 0	0.032 sec
9	20:12:08	insert into Veiculo (idVeiculo, modelo, marca, ano, placa, valorDiaria, estado) values (1, 'Onix'...	10 row(s) affected	Records: 10	Duplicates: 0	Warnings: 0	0.063 sec
10	20:12:15	insert into Manutencao (idManutencao, idVeiculo, descricao, dataManutencao, custo) values ...	10 row(s) affected	Records: 10	Duplicates: 0	Warnings: 0	0.062 sec
11	20:12:29	insert into Pagamento (idPagamento, forma, dataPagamento, valorTotal, estado) values (1, '...	20 row(s) affected	Records: 20	Duplicates: 0	Warnings: 0	0.047 sec
12	20:12:45	insert into Locacao (idLocacao, idCliente, idPagamento, dataInicio, dataFim) values (1, 1, 1, '...	20 row(s) affected	Records: 20	Duplicates: 0	Warnings: 0	0.047 sec
13	20:12:54	insert into LocacaoVeiculo (idLocacao, idVeiculo) values (1, 2), (2, 3), (3, 5), (4, 8), ...	20 row(s) affected	Records: 20	Duplicates: 0	Warnings: 0	0.031 sec

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;
```

```
1  
2 • SELECT descricao, dataManutencao, custo FROM manutencao;
```

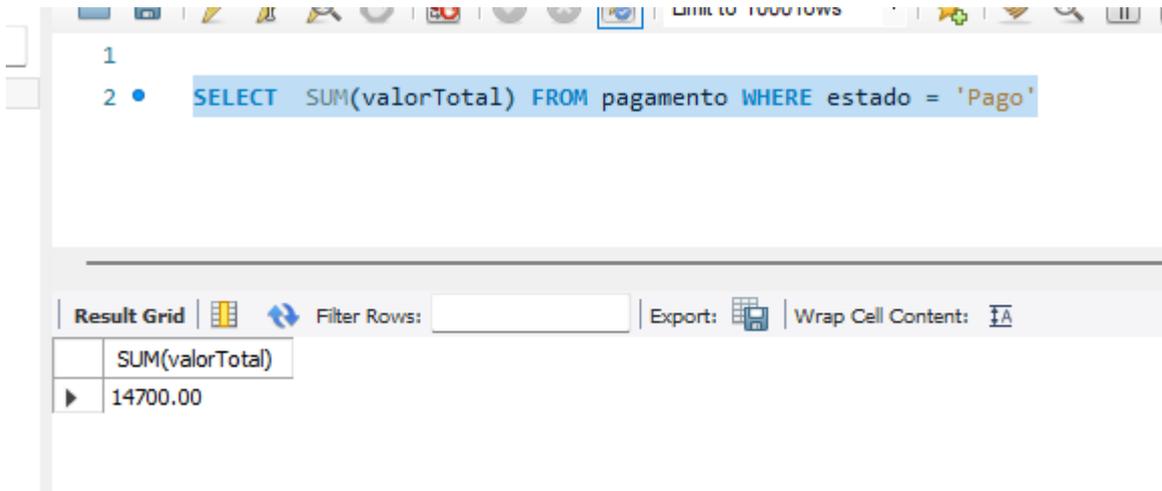


	descricao	dataManutencao	custo
▶	Troca de óleo e revisão geral	2024-12-09 00:00:00	200.00
	Substituição de pneu	2024-12-10 00:00:00	600.00
	Troca de pastilhas de freio	2024-12-14 00:00:00	450.00
	Alinhamento e balanceamento	2024-12-18 00:00:00	150.00
	Revisão elétrica completa	2024-12-28 00:00:00	500.00
	Reparo na suspensão	2025-01-05 00:00:00	700.00
	Troca do sistema de escapamento	2025-01-07 00:00:00	750.00
	Troca de bateria	2025-01-17 00:00:00	400.00
	Substituição do filtro de ar	2025-01-17 00:00:00	120.00
	Pintura e retoques na lataria	2025-01-28 00:00:00	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'
```

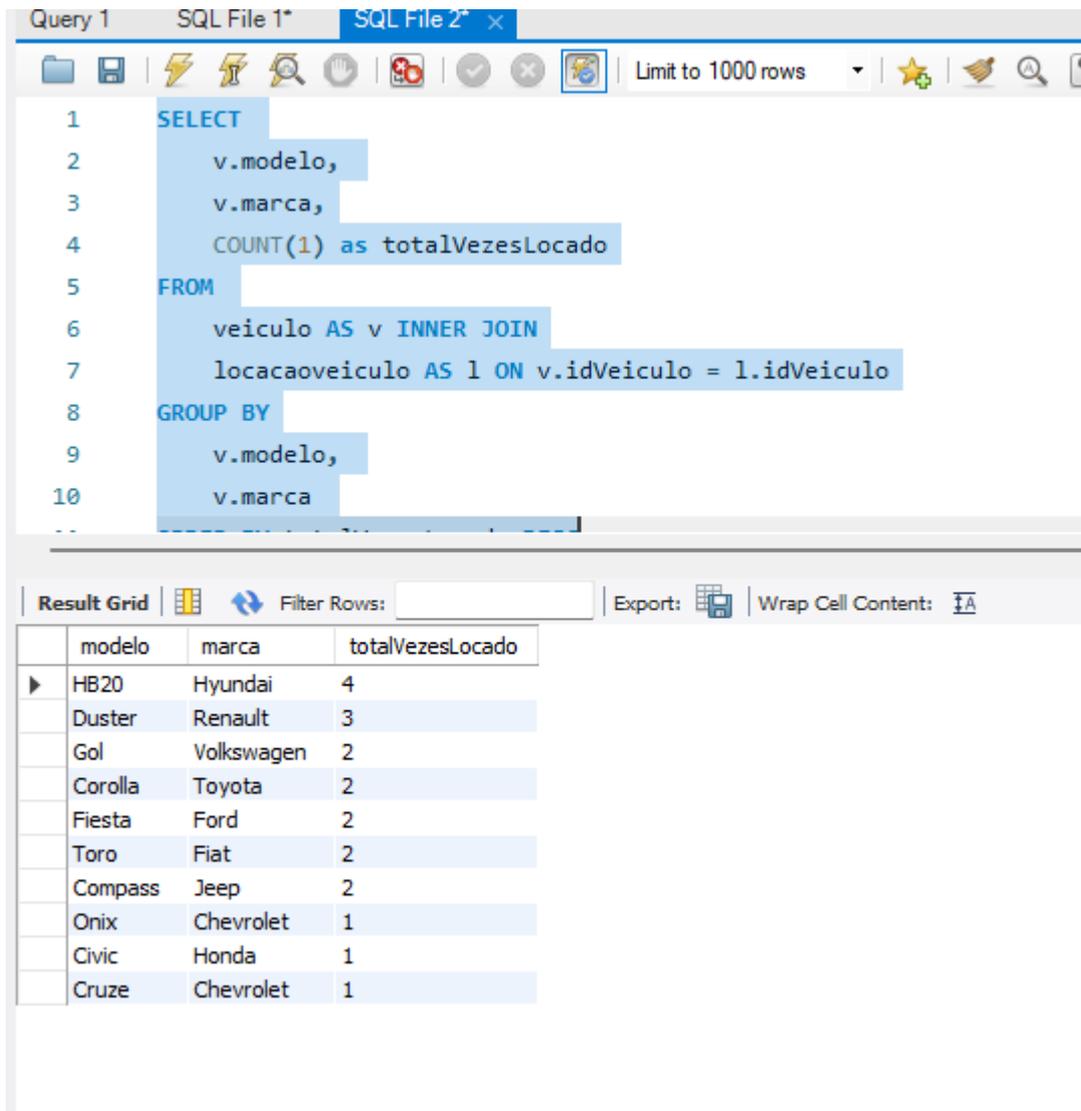


Pontuação: 10 pontos.

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(1) as totalVezesLocado
FROM
    veiculo AS v INNER JOIN
    locacaoveiculo AS l ON v.idVeiculo = l.idVeiculo
GROUP BY
    v.modelo,
    v.marca
ORDER BY totalVezesLocado DESC
```



```
1 SELECT
2     v.modelo,
3     v.marca,
4     COUNT(1) as totalVezesLocado
5 FROM
6     veiculo AS v INNER JOIN
7     locacaoveiculo AS l ON v.idVeiculo = l.idVeiculo
8 GROUP BY
9     v.modelo,
10    v.marca
```

Result Grid

	modelo	marca	totalVezesLocado
▶	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

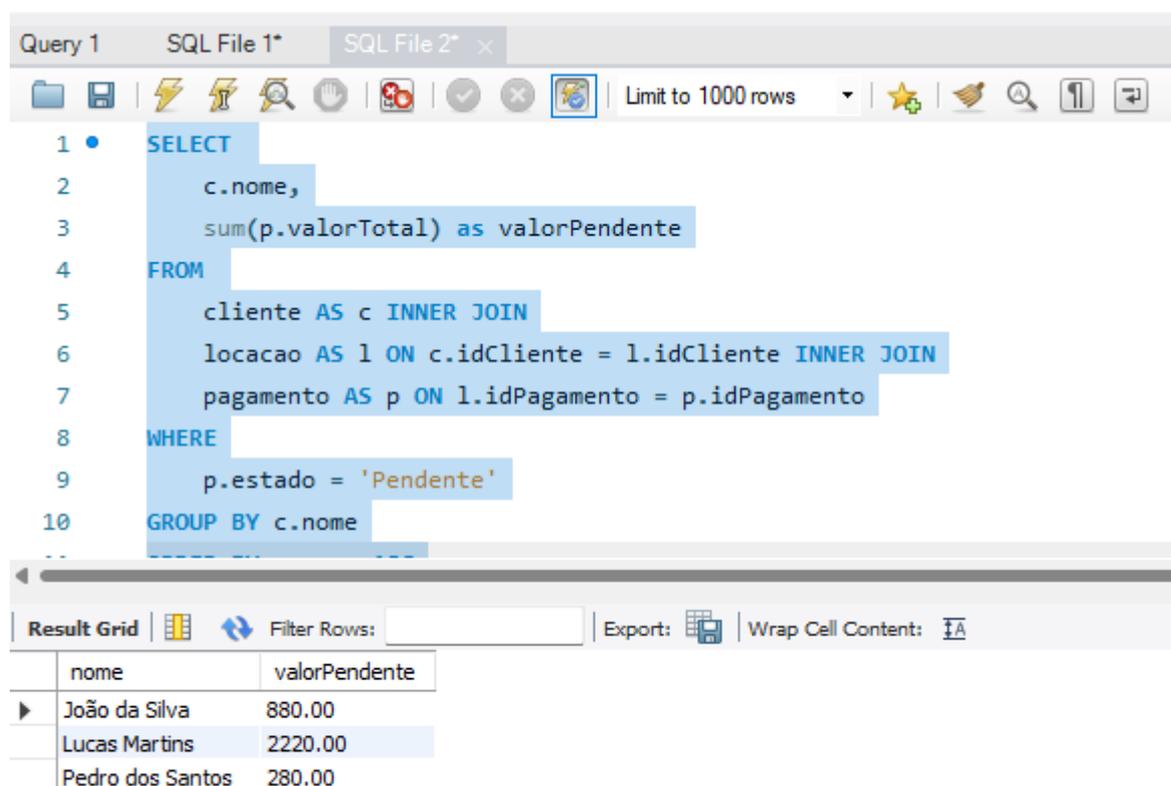
Pontuação: 10 pontos.

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) as valorPendente
FROM
    cliente AS c INNER JOIN
    locacao AS l ON c.idCliente = l.idCliente INNER JOIN
```

```
pagamento AS p ON l.idPagamento = p.idPagamento
WHERE
    p.estado = 'Pendente'
GROUP BY c.nome
ORDER BY c.nome ASC;
```



The screenshot shows a SQL IDE interface with a query editor and a result grid. The query editor contains the following SQL code:

```
1 • SELECT
2     c.nome,
3     sum(p.valorTotal) as valorPendente
4 FROM
5     cliente AS c INNER JOIN
6     locacao AS l ON c.idCliente = l.idCliente INNER JOIN
7     pagamento AS p ON l.idPagamento = p.idPagamento
8 WHERE
9     p.estado = 'Pendente'
10 GROUP BY c.nome
```

The result grid below the query editor displays the following data:

nome	valorPendente
João da Silva	880.00
Lucas Martins	2220.00
Pedro dos Santos	280.00