BANCO DE DADOS
Trabalho – Relatório

Curso:	BACHARELADO EM ENGENHARIA DE SOFTWARE - DISTÂNCIA (6270)
Aluno(a):	Guilherme Darjan Froggel Kozechen
RU:	5257850

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;

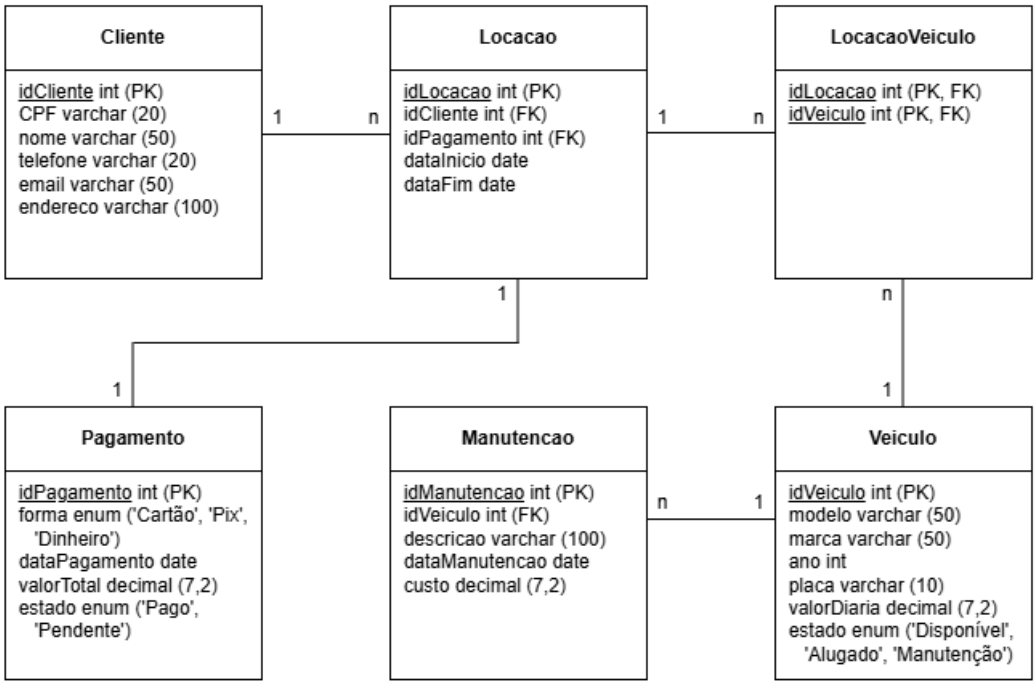
- Hotel – Deverão ser armazenados os seguintes dados: identificação do hotel, nome, categoria, telefone, e-mail e endereço, sendo o endereço composto por rua, número, complemento, bairro, CEP, cidade e estado;
- Quarto – Deverão ser armazenados os seguintes dados: identificação do quarto, número de leitos, 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*).

```
-- Criação do Banco de Dados
create database LocadoraVeiculos;
use LocadoraVeiculos;

 -- Tabela dos Possiveis Clientes:
```

```
create table Cliente(
idCliente int primary key not null,
   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 Dos Pagamentos nela estão todos os pagamentos
create table Pagamento(
idPagamento int primary key not null,
   forma enum ('Cartão','Pix','Dinheiro') not null,
   dataPagamento date not null,
   valorTotal decimal (7,2) not null,
   estado enum ('Pago','Pendente') not null
);

 -- Tabela Da Locação dos Veiculos
create table Locacao(
idLocacao int primary key not null,
   idCliente int not null,
   constraint fkLocaCliente foreign key (idCliente) references Cliente(idCliente),
   idPagamento int not null,
   constraint fkLocaPagamento foreign key (idPagamento) references
Pagamento(idPagamento),
   dataInicio date not null,
   dataFim date not null
);

 -- Tabela dos Veiculos nela estão contidos todos os veiculos que podem ser
-- Alugados
create table Veiculo(
idVeiculo int primary key not null,
```

```
    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','Manutenção') not null
);

 -- Tabela das Locações
create table locacaoVeiculo(
    idLocacao int not null,
    idVeiculo int not null,
    constraint pk_LocacaoVeiculo primary key (idLocacao, idVeiculo),
    constraint fkLocacao foreign key (idLocacao) references Locacao(idLocacao),
    constraint fkVeiculo foreign key (idVeiculo) references Veiculo(idVeiculo)
);

-- Tabela dos Veiculos que foram para manutenção
create table Manutencao(
idManutencao int primary key not null,
    idVeiculo int,
    constraint fkVehicle foreign key (idVeiculo) references Veiculo(idVeiculo),
    descricao varchar(100) not null,
    dataManutencao date not null,
    custo decimal (7,2) not null
);
```

Pontuação: 10 pontos.

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

```
-- Selecionamos apenas os dados orientados da tabela manutencao
select descricao, dataManutencao, custo from Manutencao;
```

-- Temos assim uma tabela com o que foi feito e seu valor

```
60      -- Selecionamos apenas os dados orientados da tabela manutencao
61  •   select descricao, dataManutencao, custo from Manutencao;
62      -- Temos assim uma tabela com o que foi feito e seu valor
63
64
65
66
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

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.

```
-- Selecionamos a soma do valor total
-- Alteramos seu nome para ficar esteticamente mais bonito
--  “A soma total dos Pagamentos onde o status dele é Pago”
select sum(valorTotal) as Arrecadação_Total
from Pagamento where estado = 'Pago';
```

```
60      -- Selecionamos a soma do valor total
61      -- Alteramos seu nome para ficar esteticamente mais bonito
62      --  "A soma total dos Pagamentos onde o status dele é Pago"
63  •   select sum(valorTotal) as Arrecadação_Total
64      from Pagamento where estado = 'Pago';
65
66
67
```

esult Grid | Filter Rows: | Export: | Wrap Cell Content:

Arrecadação_Total
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 aluguéis.
   Dica: Utilize a cláusula *group by*.

```
-- Selecionamos o modelo e a marca do veiculo

select Veiculo.modelo, Veiculo.marca,

-- Fazemos a contagem das vezes que foi locado e damos um nome a isso

count(locacaoVeiculo.idLocacao) as VezesQueFoiLocado

-- Agora ligamos as tabelas Veiculo e locação interligando os id dos veiculos

from Veiculo, locacaoVeiculo where Veiculo.idVeiculo = locacaoVeiculo.idVeiculo

-- Para finalizar agrupo os veiculos pelo modelo marca e as vezes que foi locado

group by Veiculo.modelo, Veiculo.idVeiculo, Veiculo.marca order by VezesQueFoiLocado desc;
```

```
70    -- Selecionamos o modelo e a marca do veiculo
71    select Veiculo.modelo, Veiculo.marca,
72    -- Fazemos a contagem das vezes que foi locado e damos um nome a isso
73    count(locacaoVeiculo.idLocacao) as VezesQueFoiLocado
74    -- Agora ligamos as tabelas Veiculo e locação interligando os id dos veiculos
75    from Veiculo, locacaoVeiculo where Veiculo.idVeiculo = locacaoVeiculo.idVeiculo
76    -- Para finalizar agrupo os veiculos pelo modelo marca e as vezes que foi locado
77    group by Veiculo.modelo, Veiculo.idVeiculo, Veiculo.marca order by VezesQueFoiLocado desc;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

modelo	marca	VezesQueFoiLocado
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*.

```
-- Selecionamos o nome do cliente e o valor total dos pagamentos
select Cliente.nome, Pagamento.valorTotal
-- Definimos de quais tabelas serão os valores
from Cliente, Locacao, Pagamento
-- Ligamos os ids das tabelas as suas devidas chaves
where Cliente.idCliente = Locacao.idCliente
-- Adicionamos o filtro de pagamento pendente
and Locacao.idPagamento = Pagamento.idPagamento and Pagamento.estado =
'Pendente'
-- Por fim definimos que deve ser mostrado em ordem alfabética
order by Cliente.nome asc;
```

```
67    -- Selecionamos o nome do cliente e o valor total dos pagamentos
68    select Cliente.nome, Pagamento.valorTotal
69    -- Definimos de quais tabelas serão os valores
70    from Cliente, Locacao, Pagamento
71    -- Ligamos os ids das tabelas as suas devidas chaves
72    where Cliente.idCliente = Locacao.idCliente
73    -- Adicionamos o filtro de pagamento pendente
74    and Locacao.idPagamento = Pagamento.idPagamento and Pagamento.estado = 'Pendente'
75    -- Por fim definimos que deve ser mostrado em ordem alfabética
76    order by Cliente.nome asc;
77
78
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

nome	valorTotal
João da Silva	880.00
Lucas Martins	1680.00
Lucas Martins	540.00
Pedro dos Santos	280.00