

Histórico de Vendas

Avaliação D1

Nome: Matheus Henrique Elias Rosa

RA: 319141144

A. Valor: 20pts

B. Data de Entrega: 03 de maio de 2020 até 23:59


1. **Prazo Máximo, não serão aceitas entregas atrasadas!**

C. Entrega: ulife ou Google Sala de Aula

1. Se entregar via **ulife**, poste este arquivo no **formato PDF**; não serão aceitos outros formatos.

D. Seu código deverá estar muito bem formatado, indentado e estruturado;

E. **Não serão fornecidas respostas:** a interpretação e pesquisa das questões faz parte da avaliação, por isso, leia atentamente tudo antes de começar;

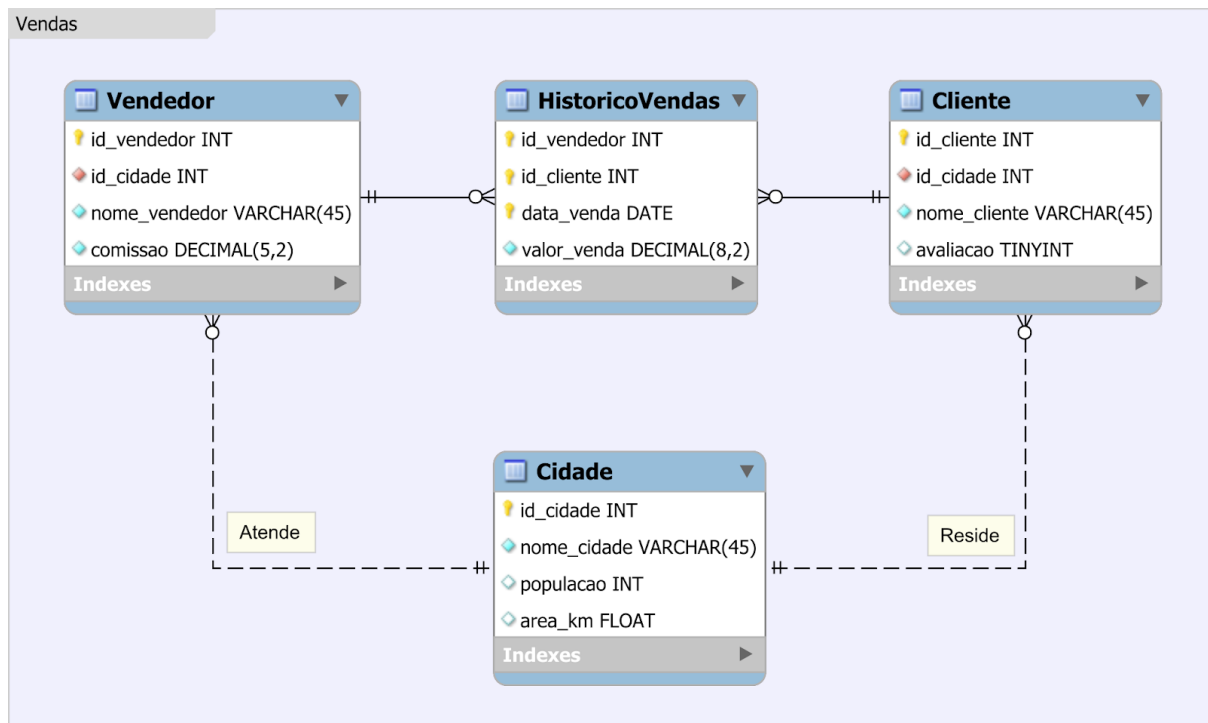
 **Importante • Não Serão Aceitas Cópias!**

Em hipótese alguma serão aceitas cópias de exercícios ou parte deles, caso seja identificada, será feita a correção de apenas uma das cópias e dividida a nota pela quantidade de cópias identificadas.

terça-feira, 28 de abril de 2020



1. Banco de Dados de Histórico de Vendas



Script de Criação

```

CREATE DATABASE vendas;
USE vendas;

CREATE TABLE Cidade(
    id_cidade INT NOT NULL,
    nome_cidade VARCHAR(45) NOT NULL,
    populacao INT DEFAULT 0,
    area_km FLOAT DEFAULT 0,
    PRIMARY KEY (id_cidade)
);

CREATE TABLE Vendedor(
    id_vendedor INT NOT NULL,
    id_cidade INT NOT NULL,

```

```

    nome_vendedor VARCHAR(45) NOT NULL,
    comissao DECIMAL(5, 2) NOT NULL DEFAULT 0,
    PRIMARY KEY (id_vendedor),
    CONSTRAINT fk_vendedor_cidade
    FOREIGN KEY (id_cidade) REFERENCES Cidade(id_cidade)
);

CREATE TABLE Cliente(
    id_cliente INT NOT NULL,
    id_cidade INT NOT NULL,
    nome_cliente VARCHAR(45) NOT NULL,
    avaliacao TINYINT DEFAULT 0,
    PRIMARY KEY (id_cliente),
    CONSTRAINT fk_cliente_cidade
    FOREIGN KEY (id_cidade) REFERENCES Cidade(id_cidade)
);

CREATE TABLE HistoricoVendas(
    id_vendedor INT NOT NULL,
    id_cliente INT NOT NULL,
    data_venda DATE NOT NULL,
    valor_venda DECIMAL(8, 2) DEFAULT 0,
    PRIMARY KEY (id_vendedor, id_cliente, data_venda),
    CONSTRAINT fk_historico_vendas_vendedor
    FOREIGN KEY (id_vendedor) REFERENCES Vendedor(id_vendedor),
    CONSTRAINT fk_historico_vendas_cliente
    FOREIGN KEY (id_cliente) REFERENCES Cliente(id_cliente)
);

```

Dados

```

-- -----
-- Dados da tabela Cidade --
-- -----

INSERT INTO Cidade VALUES (1, 'Belo Horizonte', 2512070, 331.4);
INSERT INTO Cidade VALUES (2, 'Contagem', 663855, 195.3);
INSERT INTO Cidade VALUES (3, 'Betim', 439340, 343.7);

```

```
INSERT INTO Cidade VALUES (4, 'Ribeirão das Neves', 334858, 155.5);
INSERT INTO Cidade VALUES (5, 'Sete Lagoas', 239639, 537.6);
INSERT INTO Cidade VALUES (6, 'Santa Luzia', 219134, 235.3);
INSERT INTO Cidade VALUES (7, 'Ibirité', 180204, 72.6);
INSERT INTO Cidade VALUES (8, 'Sabará', 136344, 302.2);
INSERT INTO Cidade VALUES (9, 'Vespasiano', 127601, 71.2);
INSERT INTO Cidade VALUES (10, 'Nova Lima', 94889, 429.0);
```

```
-- -----
-- Dados da tabela Vendedor --
-- -----
```

```
INSERT INTO Vendedor VALUES (1, 6, 'Bernardo', 0.10);
INSERT INTO Vendedor VALUES (2, 1, 'Giovana', 0.25);
INSERT INTO Vendedor VALUES (3, 8, 'Nilton', 0.15);
INSERT INTO Vendedor VALUES (4, 1, 'Paulo', 0.25);
INSERT INTO Vendedor VALUES (5, 2, 'Vanessa', 0.20);
```

```
-- -----
-- Dados da tabela Cliente --
-- -----
```

```
INSERT INTO Cliente VALUES (1, 1, 'Danilo', 3);
INSERT INTO Cliente VALUES (2, 4, 'Elen', 2);
INSERT INTO Cliente VALUES (3, 2, 'Caetano', 4);
INSERT INTO Cliente VALUES (4, 1, 'Gerson', 5);
INSERT INTO Cliente VALUES (5, 2, 'Vinícius', 4);
INSERT INTO Cliente VALUES (6, 9, 'Lucas', 1);
INSERT INTO Cliente VALUES (7, 7, 'Clarisse', 3);
INSERT INTO Cliente VALUES (8, 1, 'Davi', 5);
INSERT INTO Cliente VALUES (9, 2, 'Rafaela', 4);
INSERT INTO Cliente VALUES (10, 3, 'José', 3);
```

```
-- -----
-- Dados da tabela Histórico Vendas --
-- -----
```

```
INSERT INTO HistoricoVendas VALUES (2, 1, '2019-08-15', 501.10);
INSERT INTO HistoricoVendas VALUES (2, 4, '2019-08-16', 645.30);
INSERT INTO HistoricoVendas VALUES (2, 1, '2019-09-02', 580.00);
INSERT INTO HistoricoVendas VALUES (2, 4, '2019-09-06', 600.25);
INSERT INTO HistoricoVendas VALUES (2, 1, '2019-09-15', 406.24);
```

```
INSERT INTO HistoricoVendas VALUES (2, 1, '2019-09-17', 560.00);
INSERT INTO HistoricoVendas VALUES (3, 7, '2019-10-09', 560.30);
INSERT INTO HistoricoVendas VALUES (3, 7, '2019-10-28', 250.45);
INSERT INTO HistoricoVendas VALUES (4, 1, '2019-10-16', 576.00);
INSERT INTO HistoricoVendas VALUES (4, 1, '2019-10-18', 343.86);
INSERT INTO HistoricoVendas VALUES (4, 8, '2019-10-27', 675.00);
INSERT INTO HistoricoVendas VALUES (5, 9, '2019-08-15', 600.00);
INSERT INTO HistoricoVendas VALUES (5, 9, '2019-08-22', 520.25);
INSERT INTO HistoricoVendas VALUES (5, 5, '2019-10-14', 505.10);
INSERT INTO HistoricoVendas VALUES (5, 5, '2019-10-19', 480.25);
```

Dados da Tabela Cidade

id_cidade	nome_cidade	populacao	area_km
1	Belo Horizonte	2512070	331.4
2	Contagem	663855	195.3
3	Betim	439340	343.7
4	Ribeirão das Neves	334858	155.5
5	Sete Lagoas	239639	537.6
6	Santa Luzia	219134	235.3
7	Ibirité	180204	72.6
8	Sabará	136344	302.2
9	Vespasiano	127601	71.2
10	Nova Lima	94889	429

Dados da Tabela Vendedor

id_vendedor	id_cidade	nome_vendedor	comissao
1	6	Bernardo	0.10

2	1	Giovana	0.25
3	7	Nilton	0.15
4	1	Paulo	0.25
5	2	Vanessa	0.20

Dados da Tabela Cliente

id_cliente	id_cidade	nome_cliente	avaliacao
1	1	Danilo	3
2	4	Elen	2
3	2	Caetano	4
4	1	Gerson	5
5	2	Vinícius	4
6	9	Lucas	1
7	7	Clarisse	3
8	1	Davi	5
9	2	Rafaela	4
10	3	José	3

Dados da Tabela Histórico de Vendas

id_vendedor	id_cliente	data_venda	valor_venda
2	1	2019-08-15	501.10
2	4	2019-08-16	645.30
2	1	2019-09-02	580.00

2	4	2019-09-06	600.25
2	1	2019-09-15	406.24
2	1	2019-09-17	560.00
3	7	2019-10-09	560.30
3	7	2019-10-28	250.45
4	1	2019-10-16	576.00
4	1	2019-10-18	343.86
4	8	2019-10-27	675.00
5	9	2019-08-15	600.00
5	9	2019-08-22	520.25
5	5	2019-10-14	505.10

2. Consultas

1. Selecione todas as informações dos vendedores. <0,8 pts>

```
SELECT * FROM vendedor;
```

2. Selecione todos os clientes com avaliação acima de 3 estrelas em ordem decrescente por avaliação. <0,8 pts>

```
SELECT nome_cliente, avaliacao  
FROM cliente  
WHERE avaliacao > 3  
ORDER BY avaliacao desc;
```

3. Selecione em ordem alfabética o **id_vendedor**, **nome_vendedor** e a **comissão** dos vendedores que possuem comissão acima de 20%. <0,8 pts>

```
SELECT id_vendedor, nome_vendedor, comissao  
FROM vendedor  
WHERE comissao > 0.20  
ORDER BY nome_vendedor ASC;
```


4. Selecione em ordem alfabética todos os clientes que residem em cidades que começam com a letra "S". <0,8 pts>

```
SELECT nome_cliente, nome_cidade
FROM cliente
JOIN cidade ON cidade.id_cidade = cliente.id_cidade
WHERE nome_cidade
LIKE 'S%'
ORDER BY nome_cliente asc;
```

5. Selecione em ordem alfabética todos os vendedores das cidades de "Belo Horizonte" e "Contagem". <0,8 pts>

```
SELECT nome_vendedor, nome_cidade
FROM vendedor
JOIN cidade ON cidade.id_cidade = vendedor.id_cidade
WHERE cidade.nome_cidade = 'Belo Horizonte' || cidade.nome_cidade =
'Contagem'
ORDER BY nome_vendedor asc;
```

6. Selecione o **id_vendedor** e o **nome_vendedor** de todos os vendedores listados no Histórico de Vendas, sem repetições em ordem alfabética. <0,8 pts>

```
SELECT DISTINCT vendedor.id_vendedor, nome_vendedor
FROM vendedor
INNER JOIN HistoricoVendas ON vendedor.id_vendedor =
HistoricoVendas.id_vendedor
ORDER BY nome_vendedor asc;
```

7. Selecione a **data** e o **valor da venda** de todos os pedidos do mês de agosto em ordem crescente. <0,8 pts>

```
SELECT data_venda, valor_venda
FROM HistoricoVendas
WHERE data_venda < '2019-08-31'
ORDER BY data_venda asc;
```

8. Selecione em ordem alfabética o **nome** e **avaliação** de todos os clientes **5** estrelas da cidade de "**Belo Horizonte**". <0,8 pts>

```
SELECT nome_cliente, avaliacao, nome_cidade
FROM cliente
JOIN cidade ON cliente.id_cidade = cidade.id_cidade
WHERE avaliacao >= 5
ORDER BY nome_cliente asc;
```

9. Escreva uma consulta que exiba o valor total de todas as vendas no histórico de vendas. <0,8 pts>

```
SELECT SUM(valor_venda)
FROM HistoricoVendas;
```

10. Qual é o **valor médio** do valor das vendas no histórico de vendas?

```
SELECT AVG(valor_venda)
FROM HistoricoVendas;
```

11. Sem contar duplicadas, quantos vendedores estão listados no histórico de vendas?

⟨0,8 pts⟩

```
SELECT COUNT(DISTINCT id_vendedor)
FROM HistoricoVendas;
```

12. Qual é a quantidade de vendas durante o mês de outubro? ⟨0,8 pts⟩

```
SELECT data_venda, valor_venda
FROM HistoricoVendas
WHERE data_venda >= '2019-10-01' AND data_venda <= '2019-10-31'
ORDER BY data_venda asc;
```

13. Em ordem decrescente, qual a quantidade de vendas feita por cada vendedor? Sua consulta deverá exibir o nome de cada vendedor, bem como o total de vendas feitas por ele. ⟨0,8 pts⟩

```
SELECT DISTINCT vendedor.id_vendedor, nome_vendedor, SUM(valor_venda)
FROM vendedor
INNER JOIN HistoricoVendas ON vendedor.id_vendedor =
HistoricoVendas.id_vendedor
GROUP BY nome_vendedor
ORDER BY nome_vendedor desc;
```

14. Mostre em ordem decrescente por total, o nome de cada cidade e a quantidade de clientes que moram em cada uma delas. No resultado de sua consulta, cidades que

ainda não têm clientes deverão ter valor zero no resultado das contagens. <0,8 pts>

```
SELECT nome_cidade, COUNT(*)
FROM cliente
JOIN cidade ON cliente.id_cidade = cidade.id_cidade
GROUP BY nome_cidade
ORDER BY COUNT(*) desc;
```

15. Selecione o nome do vendedor, sua comissão, valor da venda e lucro (**comissão * valor venda**) de cada uma de suas vendas. <0,8 pts>

```
SELECT vendedor.id_vendedor, nome_vendedor, SUM(comissao * valor_venda)
FROM vendedor
INNER JOIN HistoricoVendas ON vendedor.id_vendedor =
HistoricoVendas.id_vendedor
GROUP BY nome_vendedor
ORDER BY nome_vendedor desc;
```

16. Selecione o id e nome dos clientes que foram atendidos por vendedores com comissão igual a 25%? <0,8 pts>

```
SELECT DISTINCT cliente.id_cliente, cliente.nome_cliente,
nome_vendedor, comissao
FROM vendedor
INNER JOIN cliente ON vendedor.id_cidade = cliente.id_cidade
WHERE comissao = 0.25
ORDER BY cliente.nome_cliente;
```

17. Liste todos os vendedores que fizeram ou não alguma venda; sua consulta deverá retornar os seguintes dados: **id_vendedor**, **nome_vendedor**, **comissao**, **data_venda**, **valor_venda** e **nome_cliente**. <0,8 pts>

```
SELECT vendedor.id_vendedor, nome_vendedor, comissao, data_venda,
valor_venda, nome_cliente
FROM vendedor
INNER JOIN HistoricoVendas ON vendedor.id_vendedor =
HistoricoVendas.id_vendedor
INNER JOIN cliente ON vendedor.id_cidade = cliente.id_cidade;
```

18. Selecione o **nome** e a **população** de todas as cidades que não possuem clientes. <0,8 pts>

```
SELECT cidade.id_cidade, nome_cidade, populacao
FROM cidade
INNER JOIN cliente ON cidade.id_cidade = cliente.id_cidade
WHERE nome_cidade IS NULL;
```

19. Liste todos os clientes que fizeram ou não alguma compra; sua consulta deverá retornar os seguintes dados: **id_cliente**, **nome_cliente**, **avaliacao**, **data_venda**, **valor_venda**, **nome_vendedor**, ordenados de maneira crescente por data da venda e decrescente por valor da venda. <0,8 pts>

```
SELECT cliente.id_cliente, cliente.nome_cliente, cliente.avaliacao,
data_venda, vendedor.nome_vendedor
FROM cliente
LEFT JOIN HistoricoVendas ON cliente.id_cliente =
HistoricoVendas.id_cliente
LEFT JOIN vendedor ON vendedor.id_vendedor =
HistoricoVendas.id_vendedor
ORDER BY data_venda IS NOT NULL DESC, data_venda ASC, data_venda DESC;
```

20. Qual é a cidade com maior população no banco de dados? <0,8 pts>

```
SELECT nome_cidade, MAX(populacao) FROM cidade;
```

21. Qual é a média de avaliação dos clientes? <0,8 pts>

```
SELECT AVG(avaliacao)
FROM cliente;
```

22. Selecione o vendedor que realizou a venda mais cara no banco de dados. Sua consulta deverá retornar os seguintes dados: **id_vendedor**, **nome_vendedor**, **comissao**, **data_venda**, **valor_venda** e **data_venda**. <0,8 pts>

```
SELECT vendedor.id_vendedor, nome_vendedor, comissao, data_venda,
MAX(valor_venda)
FROM vendedor
INNER JOIN HistoricoVendas ON vendedor.id_vendedor =
HistoricoVendas.id_vendedor
WHERE valor_venda > 650
GROUP BY nome_vendedor
ORDER BY valor_venda desc;
```

23. Qual é o valor médio de vendas inferior à **R\$ 500,00** de cada vendedor durante o mês de outubro? Sua consulta deverá trazer os seguintes dados de cada vendedor, ordenados decrescente pela média: **id_vendedor**, **nome_vendedor** e **média**. <0,8 pts>

```
SELECT DISTINCT
vendedor.id_vendedor,nome_vendedor,AVG(valor_venda),data_venda
FROM HistoricoVendas
INNER JOIN vendedor ON HistoricoVendas.id_vendedor =
vendedor.id_vendedor
WHERE data_venda >= '2019-10-01' AND data_venda <= '2019-10-31'
GROUP BY vendedor.id_vendedor
HAVING AVG(valor_venda) <= 500;
```

24. Selecione o id do vendedor, nome do vendedor, nome da cidade que atende, data da venda, valor da venda, nome do cliente, avaliação e cidade que ele reside. **0,8 pts**

```
SELECT vendedor.id_vendedor, vendedor.nome_vendedor,
cidade.nome_cidade, HistoricoVendas.data_venda,
HistoricoVendas.valor_venda, cliente.nome_cliente, cliente.avaliacao,
cidade.nome_cidade
FROM vendedor
INNER JOIN cidade ON cidade.id_cidade = vendedor.id_cidade
INNER JOIN HistoricoVendas ON HistoricoVendas.id_vendedor =
vendedor.id_vendedor
INNER JOIN cliente ON cliente.id_cliente = HistoricoVendas.id_cliente;
```

25. Usando a junção cruzada (**CROSS JOIN**), gere um relatório de planejamento de visitas para os vendedores que irão atender os clientes que residem na mesma região de atuação deles. Sua consulta deverá exibir os seguintes dados: nome da cidade em que o vendedor atua, nome do vendedor, nome do cliente, avaliação do cliente; ordenados de maneira crescente por nome da cidade, nome vendedor, e, decrescente por avaliação do cliente. <0,8 pts>

```
SELECT nome_cidade, nome_vendedor, nome_cliente, avaliacao
FROM cidade
INNER JOIN vendedor ON vendedor.id_cidade = cidade.id_cidade
CROSS JOIN cliente ON cliente.id_cidade = vendedor.id_cidade
ORDER BY nome_cidade ASC, nome_vendedor ASC, avaliacao DESC;
```