

Engenharia de Software - Banco de Dados 2 - Solução MAPA

RA 1650466-5 - Guionardo Furlan

Solução registrada em <https://github.com/guionardo/engsoft/tree/master/Banco%20de%20Dados/MAPA%20BD2>

Script para testes no banco de dados: [solve.sql](#)

a) Crie o schema e as tabelas conforme o diagrama apresentado

```
create database esoft_bd2;  
use esoft_bd2;
```

a.1) Criando tabela clientes

```
create table clientes (  
    id int not null auto_increment,  
    nome varchar(100),  
    telefone varchar(20),  
    celular varchar(20),  
    email varchar(100),  
    endereco varchar(100),  
    numero int,  
    complemento varchar(50),  
    bairro varchar(100),  
    cidade varchar(100),  
    cep varchar(10),  
    primary key (id)  
);
```

a.2) Criando tabela vendedores

```
create table vendedores (  
    id int not null auto_increment,  
    nome varchar(100),  
    celular varchar(20),  
    email varchar(100),  
    perc_comissao decimal(4,2),  
    primary key (id)  
);
```

a.3) Criando tabela produtos

```
create table produtos (  
    id int not null auto_increment,  
    nome varchar(100),  
    descricao text,  
    valor_venda decimal(12,2),  
    primary key (id)  
);
```

a.4) Criando tabela vendas

```
create table vendas (  
    id int not null auto_increment,  
    data_venda date,  
    id_cliente int,
```

```

        id_vendedor int,
        primary key (id),
        CONSTRAINT vendas_clientes_fk FOREIGN KEY (id_cliente) REFERENCES clientes (id),
        CONSTRAINT vendas_vendedores_fk FOREIGN KEY (id_vendedor) REFERENCES vendedores (id)
    );

```

a.5) Criando tabela vendas_itens

```

create table vendas_itens(
    id int not null auto_increment,
    id_vendas int,
    id_produtos int,
    quantidade decimal(12,2),
    valor_unitario decimal(12,2),
    primary key (id),
    CONSTRAINT itens_vendas_fk FOREIGN KEY (id_vendas) REFERENCES vendas (id),
    CONSTRAINT itens_produtos_fk FOREIGN KEY (id_produtos) REFERENCES produtos (id)
);

```

a.6) Populando tabelas para teste

```

INSERT INTO clientes (nome) VALUES
    ('José'),
    ('Maria'),
    ('Antônio');

INSERT INTO vendedores (nome, perc_comissao) VALUES
    ('Guionardo', 10.0),
    ('Marines', 25.0),
    ('João', 15.0),
    ('Benjamin', 12.5);

INSERT INTO produtos (nome, valor_venda) VALUES
    ('BOLACHA', 1.50),
    ('PASTA DE DENTE', 2.25),
    ('PASTEL', 5.00),
    ('IATE', 1500000.00),
    ('RINOCERONTE', 0.89);

INSERT INTO vendas (data_venda, id_cliente, id_vendedor) VALUES
    ('2019-05-13', 1, 1),
    ('2019-05-13', 1, 2),
    ('2019-05-13', 2, 3),
    ('2019-05-13', 2, 4);

INSERT INTO vendas_itens (id_vendas, id_produtos, quantidade, valor_unitario) VALUES
    (1, 1, 2, 1.5),
    (1, 2, 1, 2.25),
    (2, 2, 1, 2.25),
    (3, 3, 4, 5.00),
    (4, 4, 1, 1500000.00),
    (4, 5, 3, 0.89);

```

a.7) Dados das tabelas

```

SELECT id,nome FROM clientes;

```

id	nome
1	José

id	nome
2	Maria
3	Antônio

```
SELECT id,nome,perc_comissao FROM vendedores;
```

id	nome	perc_comissao
1	Guionardo	10.00
2	Marines	25.00
3	João	15.00
4	Benjamin	12.50

```
SELECT id,nome,valor_venda FROM produtos;
```

id	nome	valor_venda
1	BOLACHA	1.50
2	PASTA DE DENTE	2.25
3	PASTEL	5.00
4	IATE	1500000.00
5	RINOCERONTE	0.89

```
SELECT * FROM vendas;
```

id	data_venda	id_cliente	id_vendedor
1	2019-05-13	1	1
2	2019-05-13	1	2
3	2019-05-13	2	3
4	2019-05-13	2	4

```
SELECT * FROM vendas_items;
```

d	id_vendas	id_produtos	quantidade	valor_unitario
1	1	1	2.00	1.50
2	1	2	1.00	2.25
3	2	2	1.00	2.25
4	3	3	4.00	5.00
5	4	4	1.00	1500000.00
6	4	5	3.00	0.89

b) Criando função valor comissão

```

DELIMITER $$
CREATE FUNCTION comissao_vendedor(valor_venda_p decimal(12,2), perc_comissao_p decimal(4,2))
RETURNS decimal(12,2)
begin
    return valor_venda_p * perc_comissao_p / 100.0;
end$$
DELIMITER ;

```

c) Crie um comando de consulta em SQL que retorne a comissão dos vendedores por produto:

- Id do vendedor.
- Nome do vendedor.
- Id do produto.
- Nome do produto.
- Quantidade vendida.
- Valor vendido.
- Percentual de comissão.
- Valor de comissão.

```

select
    v.id_vendedor,
    ve.nome nm_vendedor,
    p.id id_produto,
    p.nome nm_produto,
    sum(vi.quantidade) qtd_vendida,
    sum(vi.quantidade * vi.valor_unitario) vlr_vendido,
    ve.perc_comissao,
    sum(vi.quantidade * vi.valor_unitario * ve.perc_comissao / 100) vlr_comissao
from
    produtos p
left join
    vendas_itens vi on vi.id_produtos=p.id
left join
    vendas v on v.id=vi.id_vendas
left join
    vendedores ve on ve.id=v.id_vendedor
group by p.id, v.id_vendedor;

```

id_vendedor	nm_vendedor	id_produto	nm_produto	qtd_vendida	vlr_vendido	perc_comissao	vlr_comissao
1	Guionardo	1	BOLACHA	2.00	3.0000	10.00	0.3000000000
1	Guionardo	2	PASTA DE DENTE	1.00	2.2500	10.00	0.2250000000
2	Marines	2	PASTA DE DENTE	1.00	2.2500	25.00	0.5625000000
3	João	3	PASTEL	4.00	20.0000	15.00	3.0000000000
4	Benjamin	4	IATE	1.00	1500000.0000	12.50	187500.0000000000
4	Benjamin	5	RINOCERONTE	3.00	2.6700	12.50	0.3337500000

d) Alterar tabela vendas_itens

```

alter table vendas_itens
    add column valor_total decimal(12,2),
    add column perc_comissao decimal(4,2),
    add column valor_comissao decimal(12,2);

```

e) Criar trigger para vendas_itens

```

DELIMITER $$

CREATE TRIGGER vendas_itens_bi

```

```

        BEFORE INSERT ON vendas_itens
        FOR EACH ROW
begin
    set new.valor_total = new.quantidade * new.valor_unitario;
    set new.perc_comissao = ifnull((select ifnull(ve.perc_comissao,0) from vendedores ve, vendas
v where ve.id=v.id_vendedor and v.id=new.id_vendas),0);
    set new.valor_comissao = comissao_vendedor(new.valor_total,new.perc_comissao);
end$$

DELIMITER ;

INSERT INTO vendas_itens (id_vendas, id_produtos, quantidade, valor_unitario) VALUES
    (4, 4, 2, 1.5),
    (4, 5, 1, 0.90);

SELECT * FROM vendas_itens;

```

id	id_vendas	id_produtos	quantidade	valor_unitario	valor_total	perc_comissao	valor_comissao
1	1	1	2.00	1.50			
2	1	2	1.00	2.25			
3	2	2	1.00	2.25			
4	3	3	4.00	5.00			
5	4	4	1.00	1500000.00			
6	4	5	3.00	0.89			
7	4	4	2.00	1.50	3.00	12.50	0.38
8	4	5	1.00	0.90	0.90	12.50	0.11

f) Crie 1 usuário coordenador com permissão de leitura, gravação e exclusão nas tabelas clientes, produtos e vendedores. Este usuário pode apenas visualizar as vendas emitidas.

```

CREATE USER 'coordenador'@'localhost' IDENTIFIED BY '1234';

GRANT SELECT, INSERT, UPDATE, delete, trigger on esoft_bd2.clientes to coordenador@localhost;
GRANT SELECT, INSERT, UPDATE, delete, trigger on esoft_bd2.produtos to coordenador@localhost;
GRANT SELECT, INSERT, UPDATE, delete, trigger on esoft_bd2.vendedores to coordenador@localhost;
GRANT SELECT on esoft_bd2.vendas to coordenador@localhost;
FLUSH PRIVILEGES;

show grants for coordenador@localhost;

```

Grants for coordenador@localhost

```

GRANT USAGE ON . TO 'coordenador'@'localhost' IDENTIFIED BY PASSWORD '*A4B6157319038724E3560894F7F932C8886EBFCF'
GRANT SELECT, INSERT, UPDATE, DELETE, TRIGGER ON esoft_bd2.clientes TO 'coordenador'@'localhost'
GRANT SELECT, INSERT, UPDATE, DELETE, TRIGGER ON esoft_bd2.produtos TO 'coordenador'@'localhost'
GRANT SELECT ON esoft_bd2.vendas TO 'coordenador'@'localhost'
GRANT SELECT, INSERT, UPDATE, DELETE, TRIGGER ON esoft_bd2.vendedores TO 'coordenador'@'localhost'

```