Database Modeling & SQL

Estudo de Caso

Aluno: Marco Antonio Monteiro Pedro

https://github.com/marcosamambaia/ModelagemSQL_Estudo_Caso

Requisitos do sistema:

Perguntas para Cliente

1. Quais tipos de produtos serão cadastrados no sistema?

"Queremos registrar produtos de diferentes categorias, como eletrônicos, móveis, alimentos e eletrodomésticos. Cada produto precisa ter um nome, código único, preço e estoque disponível."

2. Quais informações sobre os clientes devem ser armazenadas?

"Precisamos armazenar o nome, CPF e um histórico de compras, para entender melhor os hábitos de consumo."

3. Como deve ser registrado o endereço dos clientes?

"Cada cliente pode ter um único endereço, e precisamos registrar detalhes como rua, bairro, CEP, cidade, estado e país."

4. Quais dados sobre colaboradores são relevantes para o sistema?

"Devemos armazenar nome, CPF e setor onde o colaborador trabalha, para facilitar a gestão de vendas e funções administrativas."

5. Como as vendas serão registradas e quais detalhes devem ser incluídos?

"Cada venda precisa registrar a data e hora da compra, o cliente que realizou a compra, o colaborador que atendeu e a forma de pagamento utilizada."

6. O sistema deve permitir múltiplos produtos em uma única venda?

"Sim! Um cliente pode comprar vários produtos de uma vez, então precisamos que cada venda suporte múltiplos itens."

7. Como será o controle de estoque dos produtos após uma venda?

"Quando um produto for vendido, o sistema deve automaticamente atualizar o estoque, reduzindo a quantidade disponível."

8. Quais formas de contato dos clientes serão registradas?

"Precisamos armazenar e-mails e telefones dos clientes, pois às vezes precisamos contatá-los para suporte ou promoções."

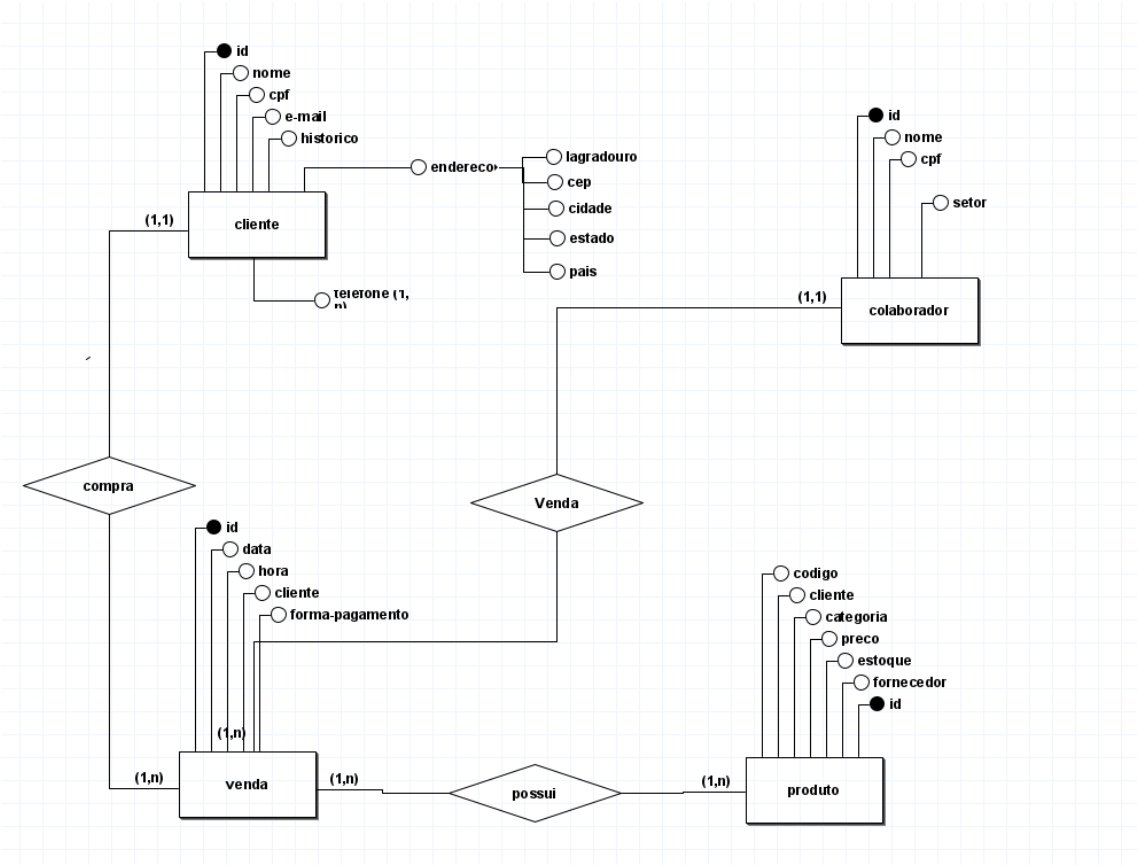
9. O sistema deve bloquear compras caso não haja estoque suficiente?

"Sim! Se um cliente tentar comprar mais unidades do que temos no estoque, a venda deve ser impedida e uma mensagem de erro exibida."

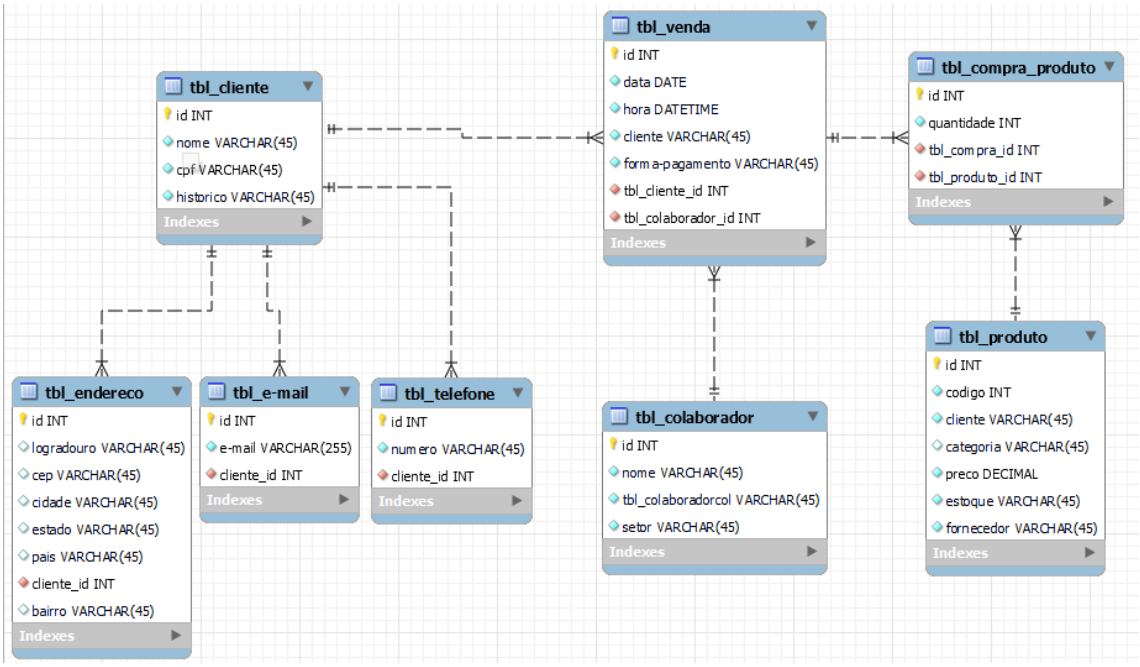
10. Quais relatórios ou consultas devem ser gerados para facilitar a gestão do sistema?

"Queremos relatórios de vendas, estoque e clientes. Por exemplo, uma listagem de clientes mais ativos e produtos mais vendidos."

Modelo Conceitual:



Modelo Logico:



Código:

```
/*********************************************************************************************
*********************************************************************************************
*********************************************************************************************
**********************************ESTUDO DE caso ***********************************************
*********************************************************************************************
*********************************************************************************************/


create database db_estudo_de_caso;
use db_estudo_de_caso;


create table tbl_produto (
  id int not null primary key auto_increment,
```

```
    codigo int not null unique,
    nome varchar(100) not null,
    categoria varchar(45),
    preco float not null,
    estoque int not null,
    fornecedor varchar(100)
);

create table tbl_colaborador (
    id int not null primary key auto_increment,
    nome varchar(100) not null,
    cpf varchar(45),
    setor varchar(45)
);

create table tbl_cliente (
    id int not null primary key auto_increment,
    nome varchar(100) not null,
    cpf varchar(45),
    historico text
);

create table tbl_endereco (
    id int not null primary key auto_increment,
    logradouro varchar(45),  -- Corrigido
    bairro varchar(45),
    cep varchar(45),
    cidade varchar(45),
    estado varchar(45),
    pais varchar(45),
    id_cliente int not null,

    constraint FK_Cliente_Endereco
```

```
    foreign key (id_cliente)
    references tbl_cliente (id)
);

create table tbl_email (
    id int not null primary key auto_increment,
    email varchar(255) not null,
    id_cliente int not null,

    constraint FK_Cliente_Email
    foreign key (id_cliente)
    references tbl_cliente (id)
);

create table tbl_telefone (
    id int not null primary key auto_increment,
    numero varchar(15) not null,
    id_cliente int not null,

    constraint FK_Cliente_Telefone
    foreign key (id_cliente)
    references tbl_cliente (id)
);

create table tbl_venda (
    id int not null primary key auto_increment,
    data_Compra date not null,
    hora datetime,
    forma_pagamento varchar(45),
    id_cliente int not null,
    id_colaborador int not null,

    constraint FK_Cliente_Venda
```

```
    foreign key (id_cliente)
    references tbl_cliente (id),

    constraint FK_Colaborador_Venda
    foreign key (id_colaborador)
    references tbl_colaborador (id)
);

create table tbl_compra_produto (
    id int not null primary key auto_increment,
    quantidade int not null,
    id_venda int not null,
    id_produto int not null,

    constraint FK_Venda_Compra_Produto
    foreign key (id_venda)
    references tbl_venda (id),

    constraint FK_Produto_Compra_Produto
    foreign key (id_produto)
    references tbl_produto (id)
);

-- definir um delimitador para agrupar comandos no trigger
delimiter $$

-- trigger que reduz o estoque após uma venda
create trigger trg_update_estoque
after insert on tbl_compra_produto -- o trigger será ativado após uma inserção em
tbl_compra_produto
for each row -- executa a ação para cada nova linha inserida
begin
    -- atualiza o estoque subtraindo a quantidade vendida
```

```
    update tbl_produto
    set estoque = estoque - new.quantidade
    where id = new.id_produto;
end $$

-- resetar o delimitador padrão
delimiter ;

-- definir um novo delimitador para agrupar o próximo trigger
delimiter $$

-- trigger que impede venda se não houver estoque suficiente
create trigger trg_verifica_estoque
before insert on tbl_compra_produto -- o trigger será ativado antes de uma inserção em tbl_compra_produto
for each row -- executa a ação para cada nova linha inserida
begin
    declare estoque_atual int; -- declara uma variável para armazenar o estoque atual

    -- obtém o estoque do produto antes da venda
    select estoque into estoque_atual from tbl_produto where id = new.id_produto;

    -- se o estoque for menor que a quantidade desejada, bloqueia a inserção
    if estoque_atual < new.quantidade then
        signal sqlstate '45000' -- gera um erro personalizado
        set message_text = 'erro: estoque insuficiente para esta venda!';
    end if;
end $$

-- resetar o delimitador padrão
delimiter ;

-- parte de testes e simulações
```

```
-- inserir produtos
insert into tbl_produto (codigo, nome, categoria, preco, estoque, fornecedor) values
(101, 'notebook dell inspiron', 'eletrônicos', 3500.00, 10, 'dell'),
(102, 'smartphone samsung galaxy s22', 'eletrônicos', 4200.00, 15, 'samsung'),
(103, 'geladeira brastemp frost free', 'eletrodomésticos', 3200.00, 8, 'brastemp'),
(104, 'cadeira gamer xtreme', 'móveis', 850.00, 20, 'xtreme comfort'),
(105, 'arroz branco tipo 1', 'alimentos', 25.00, 50, 'camil'),
(106, 'feijão preto', 'alimentos', 12.00, 40, 'kikaldo'),
(107, 'óleo de soja', 'alimentos', 9.00, 60, 'soya'),
(108, 'leite integral 1l', 'laticínios', 6.00, 80, 'italac');

-- inserir colaboradores
insert into tbl_colaborador (nome, cpf, setor) values
('carlos silva', '12345678900', 'vendas'),
('fernanda souza', '98765432100', 'administração');

-- inserir clientes
insert into tbl_cliente (nome, cpf, historico) values
('joão pereira', '11122233344', 'cliente frequente, já realizou 5 compras.'),
('maria oliveira', '55566677788', 'primeira compra no sistema.');

select * from tbl_produto;
select * from tbl_colaborador;
select * from tbl_cliente;

-- cliente comprando
insert into tbl_venda (data_compra, hora, forma_pagamento, id_cliente, id_colaborador)
values ('2025-06-02', now(), 'pix', 2, 2);

-- registrar compra do produto
insert into tbl_compra_produto (quantidade, id_venda, id_produto)
values (5, 2, 8); -- cliente comprou 5 unidades do leite (id_produto = 8)
```

```
-- verificar o estoque atualizado

select nome, estoque from tbl_produto where id = 8;

/*********************************************************************************************

**********************************  Testes OK!!!  ******************************************/
```