1 - A tabela "Cliente" não está na 1ª Forma Normal (1FN) porque ela contém campos multivalorados. Mais especificamente, o campo "telefone" da tabela "Cliente" pode conter vários números de telefone para um único cliente, o que viola o princípio fundamental da 1FN, que exige que cada valor em uma coluna seja atômico, ou seja, contenha apenas um único valor.Na 1FN, cada célula da tabela deve conter apenas dados simples, e não deve haver listas, conjuntos ou estruturas complexas dentro de uma única célula. Para corrigir isso e atender à 1FN, precisamos remover o campo multivalorado "telefone" da tabela "Cliente" e criar uma nova tabela para armazenar os números de telefone de cada cliente em registros separados, estabelecendo uma relação um-para-muitos entre os clientes e seus números de telefone. Essa nova tabela, geralmente chamada de "TelefoneCliente", terá um campo de chave estrangeira que se refere ao cliente específico ao qual cada número de telefone pertence, permitindo que vários números de telefone sejam associados a um único cliente de forma organizada e normalizada. Exemplo:

Tabela "Cliente":• id (INT, PK)• nome (VARCHAR 255)• cpf (VARCHAR 11)• endereco (VARCHAR 255)Tabela "TelefoneCliente":• id (INT, PK)• cliente\_id (INT, FK para a tabela "Cliente")• telefone (VARCHAR 45)

2 - A tabela "Item\_venda" não está na 2ª Forma Normal (2FN) devido à presença de uma dependência parcial em relação à sua chave primária. A chave primária é composta por "produto\_id" e "venda\_id", e o campo "nome\_produto" depende exclusivamente do "produto\_id" e não do "venda\_id".Para resolver esse problema e alcançar a 2FN, precisamos eliminar a dependência parcial. A solução proposta envolve a criação de uma nova tabela que conterá as informações específicas do produto e, em seguida, relacioná-la com a tabela "Item\_venda" através do atributo "produto\_id".Aqui está a proposta para normalização na 2FN:Tabela "Item\_venda":venda\_id (INT, PK, FK para a tabela "venda")produto\_id (INT, PK, FK para a tabela "produto")quantidade (INT)valor\_unitario (DECIMAL 9,2)subtotal (DECIMAL 9,2)Tabela "Produto":id (INT, PK)nome (VARCHAR 255)descricao (VARCHAR 255)estoque (INT)fabricante (VARCHAR 255)Com essa abordagem, movemos o campo "nome\_produto" para a tabela "Produto", onde podemos armazenar informações específicas sobre cada produto, como o nome e a descrição. A tabela "Item\_venda" agora possui apenas os atributos relacionados à venda específica, eliminando a dependência parcial e alcançando a 2ª Forma Normal.

3 - A tabela "venda" não está na 3ª Forma Normal (3FN) devido à presença de uma dependência transitiva em relação à sua chave primária. O campo "funcionario\_id" depende diretamente do atributo "funcionario\_id", mas também depende indiretamente do atributo "cargo\_id" através da tabela "funcionario".

Para alcançar a 3FN e remover a dependência transitiva, devemos criar uma nova tabela relacionando "funcionario\_id" ao "cargo\_id". Essa tabela intermediária irá eliminar a dependência direta entre "funcionario\_id" e "cargo\_id" na tabela "venda".

Aqui está a proposta para normalização na 3FN:

dividida em três partes distintas para evitar dependências transitivas:A tabela pagamento armazena informações relacionadas ao pagamento, com uma chave estrangeira venda\_id referenciando a tabela venda.A tabela venda\_cliente armazena a relação entre vendas e clientes, com a chave estrangeira venda\_id referenciando a tabela venda e a chave estrangeira cliente\_id referenciando a tabela cliente.A tabela venda agora contém apenas informações gerais da venda, sem dependências transitivas.Dessa forma, todas as dependências transitivas foram eliminadas, e a tabela venda está organizada de acordo com a Terceira Forma Normal (3FN). Isso garante a integridade dos dados e reduz a redundância, tornando o esquema do banco de dados mais eficiente.

Script :

use teste;

-- Parte 1 inicio --

-- Primeir passo pra correção é remover a coluna telefone da tabela usuario:

ALTER TABLE cliente DROP COLUMN telefone;

-- Segundo passo é fazer a criação de uma nova tabela que vai armazenar os numeros de telefone com base no id do cliente:

CREATE TABLE telefone\_cliente (

id INT PRIMARY KEY,

cliente\_id INT NOT NULL,

telefone VARCHAR(45) NOT NULL,

FOREIGN KEY (cliente\_id) REFERENCES cliente(id)

);

-- Parte 1 Fim --

-- Parte 2 inicio --

-- Primeira parte é remover a coluna 'nome\_produto' e 'valor\_unitario':

ALTER TABLE item\_venda

DROP COLUMN nome\_produto;

ALTER TABLE item\_venda

DROP COLUMN valor\_unitario;

-- Segunda parte criamos uma nova tabela que contem as informações dos porudutos vendidos

CREATE TABLE item\_venda\_produto (

id INT PRIMARY KEY,

nome\_produto VARCHAR(255),

valor\_unitario DECIMAL (9,2),

produto\_id INT,

FOREIGN KEY (produto\_id) REFERENCES produto(id)

);

-- terceira parte é criar uma coluna que contem o id da nova tabela criada para ter a relação dos produtos comprados e seus dados.

ALTER TABLE item\_venda

ADD COLUMN item\_venda\_produto\_id INT,

ADD FOREIGN KEY (item\_venda\_produto\_id) REFERENCES item\_venda\_produto(id);

-- Parte 2 fim --

-- Parte 3 inicio --

-- Primeira parte, criamos uma nova tabela onde serão armazenados os dados referentes a pagamentos

CREATE TABLE pagamento(

id INT NOT NULL AUTO\_INCREMENT,

venda\_id INT NOT NULL,

tipo\_pagamento CHAR(1) NOT NULL,

numero\_cartao\_pagamento VARCHAR(16) NULL,

numero\_parcelas\_pagamento INT NULL,

data\_pagamento DATETIME NULL,

PRIMARY KEY (id),

FOREIGN KEY (venda\_id) REFERENCES venda (id) ON DELETE NO ACTION ON UPDATE NO ACTION

);

-- Segunda parte, retiramos da tabela venda todos os dados referentes a pagamento pois ja passamos eles pra uma nova tabela

ALTER TABLE venda

DROP COLUMN tipo\_pagamento,

DROP COLUMN numero\_cartao\_pagamento,

DROP COLUMN numero\_parcelas\_pagamento,

DROP COLUMN data\_pagamento;

-- Terceiro passo, criamos uma tabela que relacione a veenda com o cliente

CREATE TABLE venda\_cliente(

venda\_id INT NOT NULL,

cliente\_id INT NOT NULL,

PRIMARY KEY (venda\_id),

FOREIGN KEY (venda\_id) REFERENCES venda (id) ON DELETE NO ACTION ON UPDATE NO ACTION,

FOREIGN KEY (cliente\_id) REFERENCES cliente (id) ON DELETE NO ACTION ON UPDATE NO ACTION

);

-- Quarto passo, deletamos a foreingnKey que não é mais utel na tabela venda

ALTER TABLE venda

DROP FOREIGN KEY fk\_cliente;

-- Quinto passo, removemos o id do usuario da venda, pois ele agora esta na nova tabela

ALTER TABLE venda

DROP COLUMN cliente\_id;

-- Sexto passo, criamos um nova ForgeignKey com o mesmo principio da que apagamos anteriormente, so que na nova tabela

ALTER TABLE venda\_cliente

ADD CONSTRAINT fk\_cliente

FOREIGN KEY (cliente\_id) REFERENCES cliente (id) ON DELETE NO ACTION ON UPDATE NO ACTION;

-- Parte 3 Fim --