**Relatório da aula 02**

**DDL (Data Definition Language)**

**Capitulo 03 da Apostila do Curso**

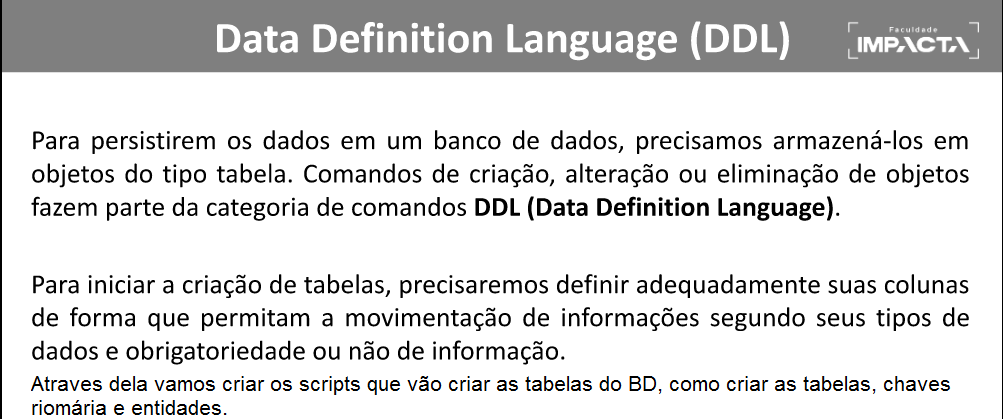
**Definição de DDL (Data Definition Linguage):**

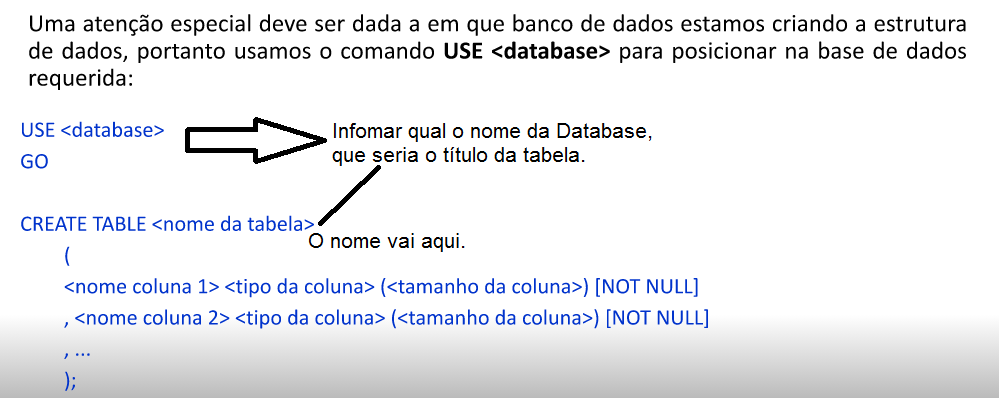
A Data Definition Language (DDL) é uma linguagem de programação que é usada para definir a estrutura, esquema e metadados de um banco de dados. A DDL é usada para criar, modificar e excluir objetos do banco de dados, como tabelas, índices, procedimentos armazenados, funções, visões e outros componentes do banco de dados.

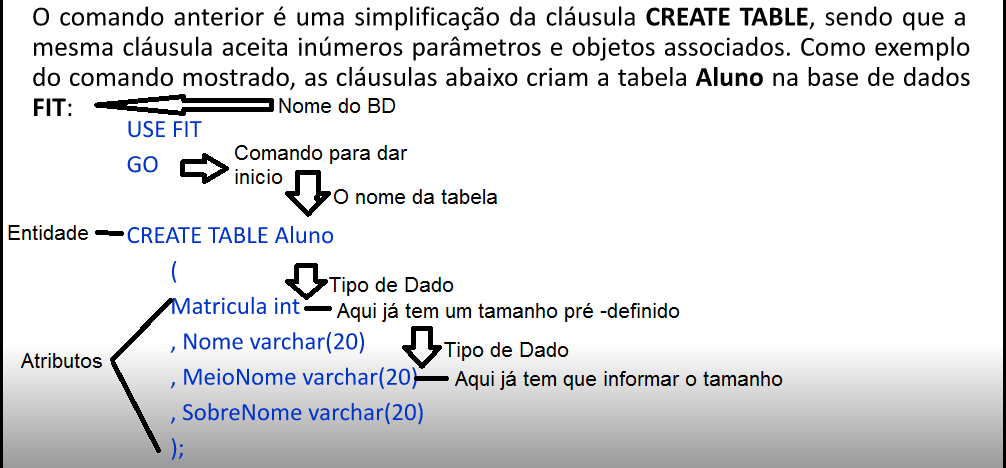
A DDL é usada para definir a estrutura de um banco de dados, incluindo seus campos, tipos de dados, restrições de integridade, chaves primárias e estrangeiras, índices e outras propriedades. As instruções DDL são usadas por desenvolvedores de banco de dados e administradores de banco de dados para criar e modificar o esquema de banco de dados.

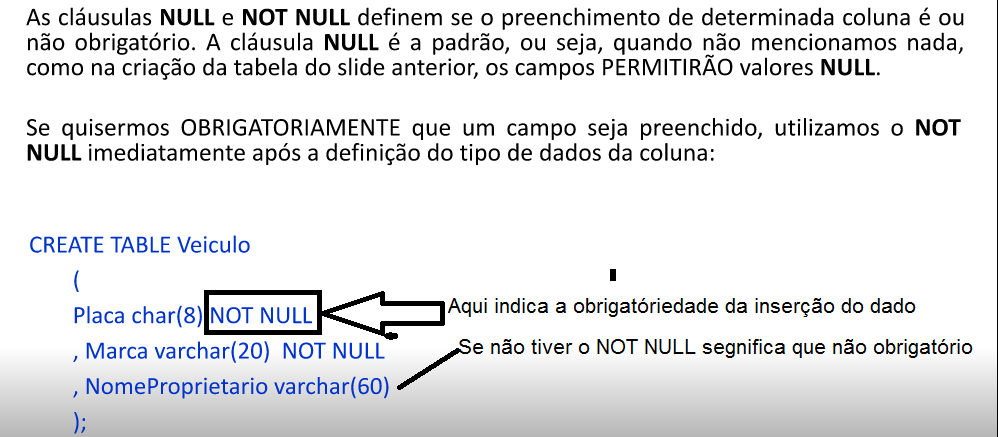
As instruções DDL incluem CREATE (criação), ALTER (alteração) e DROP (exclusão). Essas instruções são executadas pelo SGBD (Sistema Gerenciador de Banco de Dados) para criar e modificar o banco de dados e seus objetos.

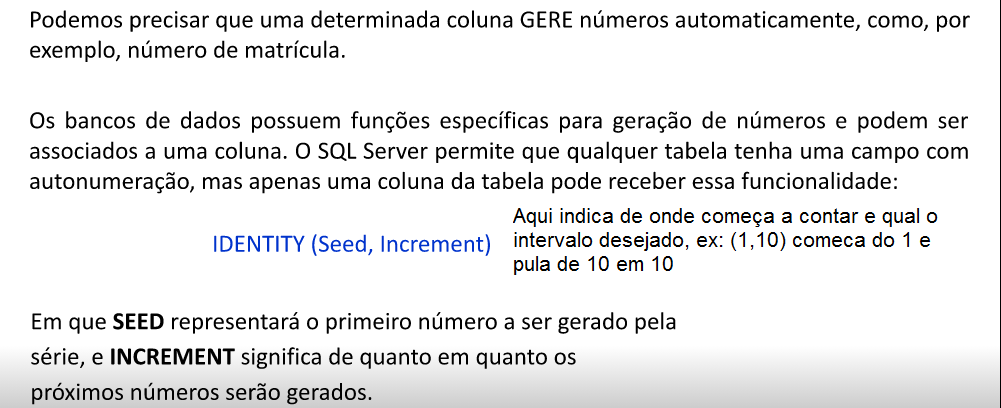
**Fonte: ChatGPT**

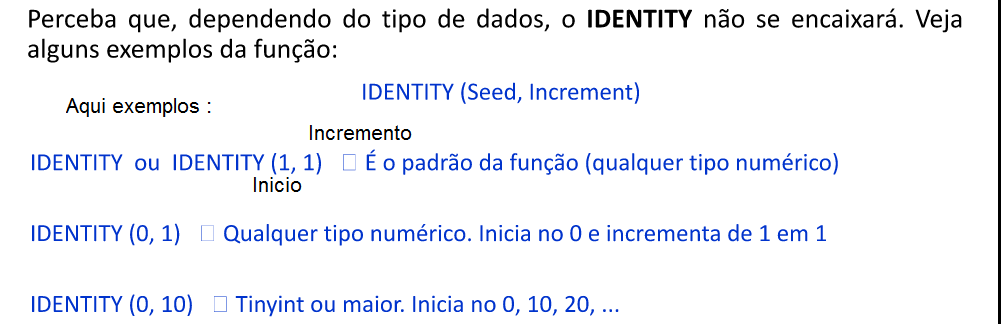
****

****

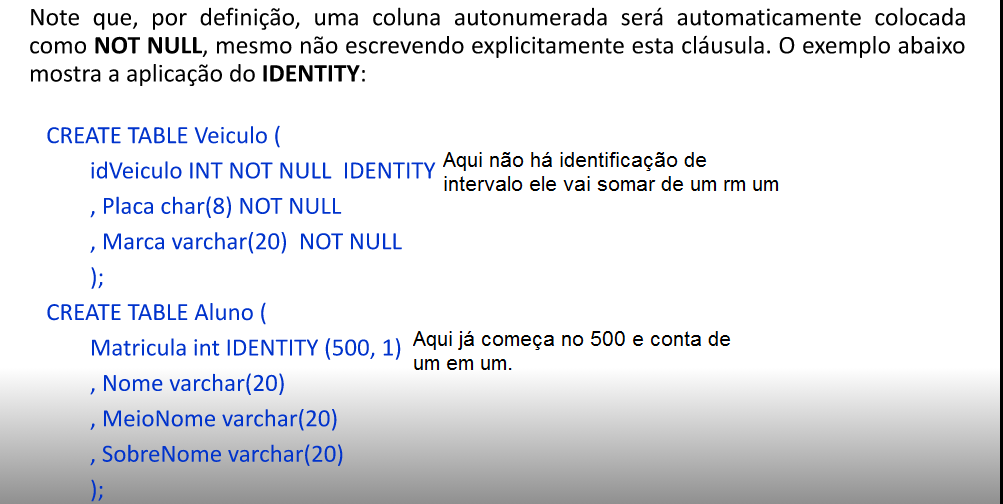
****

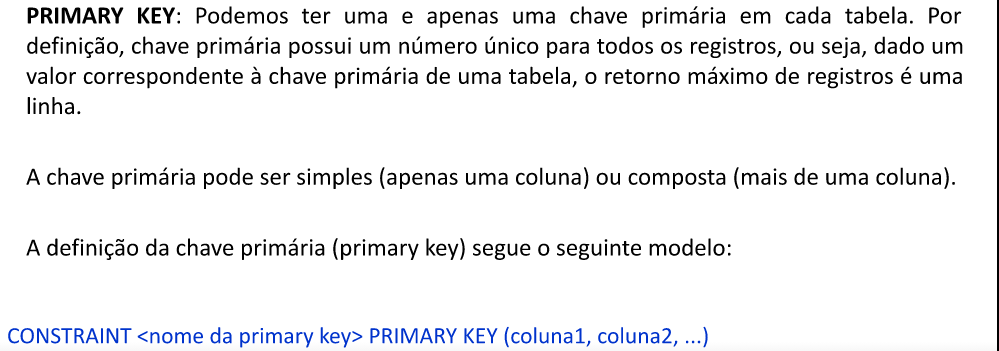
****

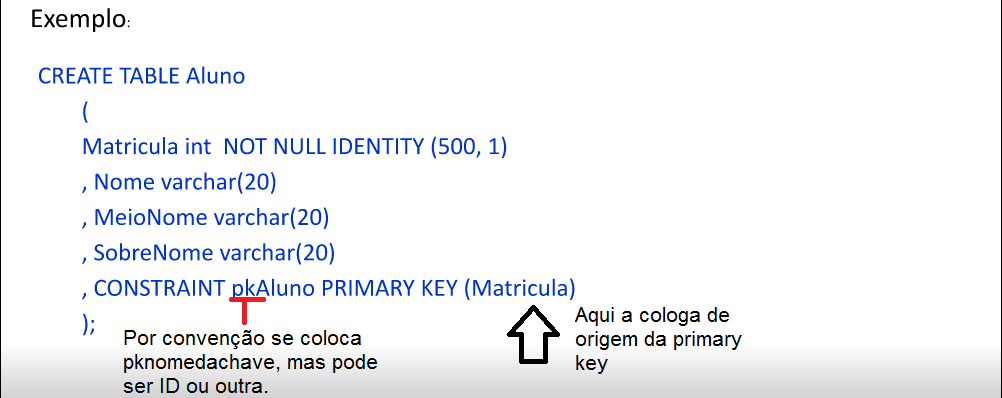
****

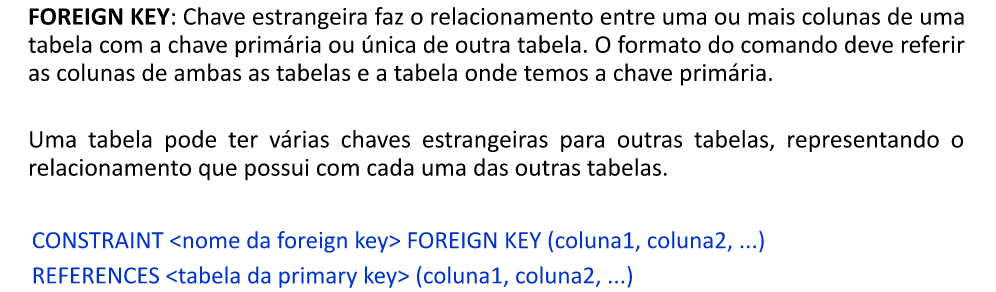
****

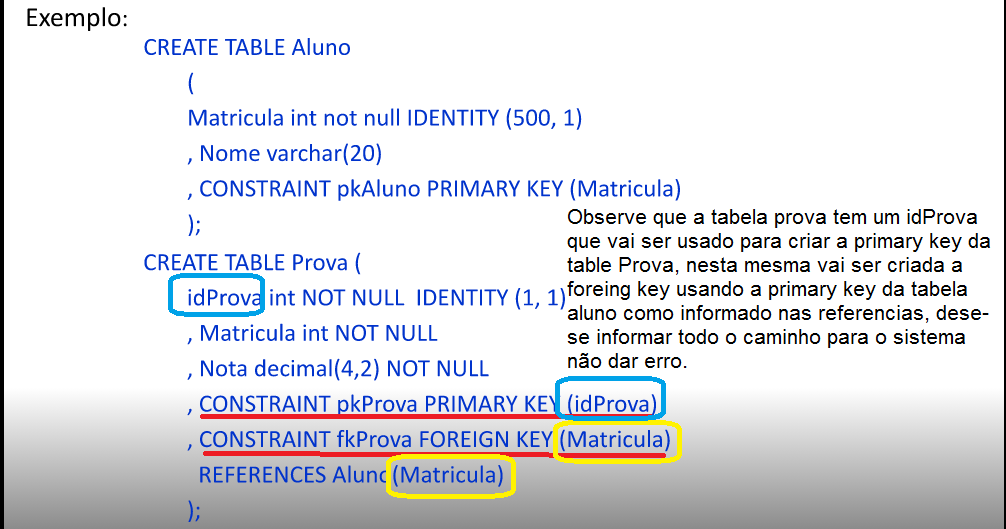
Obs: IDENTITY ele é recomendado a ser usado quando o valor inicial for diferente de 0 ou 1 e o intervalor for diferente de 1, se não usar o incrementy.

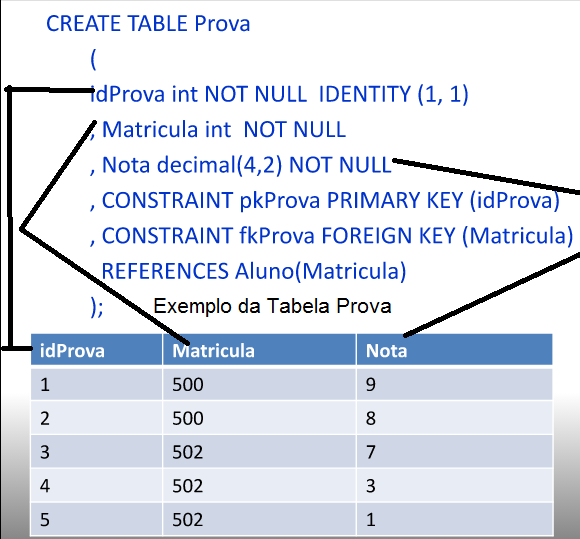
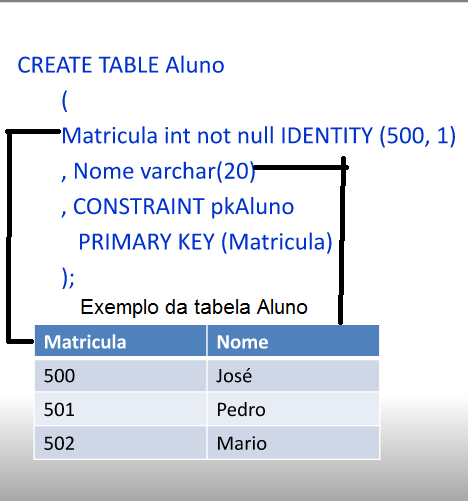
****

****

****

****

****

****

Unicidade

Chave única é semelhante a chave primária, fazendo com que a coluna envolvida seja única na tabela. Podemos ter várias chaves únicas em uma tabela, diferentemente da chave primária, onde só podemos ter uma.

Por exemplo, numa tabela de Cliente, podemos ter um campo que é o número do cliente, CPF e RG. Todos os três campos não permitem repetição na tabela. Podemos eleger qualquer um desses campos como chave primária, por exemplo, número do cliente. Neste caso o CPF e o RG poderiam ter chaves únicas, já que a tabela só permite uma chave primária. Comando deve referir às colunas de ambas as tabelas e a tabela onde temos a chave primária. A sintaxe básica da restrição de unicidade é mostrada abaixo:

CONSTRAINT <nome da unique key> UNIQUE (coluna1, coluna2, ...)

Um exemplo da restrição de unicidade é mostrado abaixo:

CREATE TABLE Cliente

(

NumCliente int not null IDENTITY (1, 1)

, CPF int NOT NULL

, RG int NOT NULL

, CONSTRAINT pkCliente PRIMARY KEY (NumCliente)

, CONSTRAINT **uqClienteCPF UNIQUE (CPF)**

, CONSTRAINT **uqClienteRG UNIQUE (RG)**

**);**

3.7. Valores Padrão

Ao definir uma coluna podemos definir um valor padrão (DEFAULT) que será usado quando não for passado um valor para essa coluna na inserção de dados. Não podem fazer referência à outra coluna da tabela, ou a outras tabelas, exibições ou procedimentos armazenados. As definições DEFAULT serão removidas quando a tabela for descartada.

A sintaxe básica da restrição é mostrada abaixo:

<nome da coluna> <tipo de dados> CONSTRAINT <nome do default>

DEFAULT (<valor, texto, data, função escalar>);

Exemplos:

MBAExterior VARCHAR(100) CONSTRAINT dfTextoNA DEFAULT ‘Não’;

Desconto DECIMAL(9, 2) CONSTRAINT dfDesconto DEFAULT 0;

DataVenda DATE NOT NULL CONSTRAINT dfDataVenda DEFAULT(getdate());

O exemplo abaixo ilustra a criação de uma tabela Venda usando a restrição de valores padrão:

CREATE TABLE Venda

(

DataVenda date not null CONSTRAINT dfDataVenda DEFAULT (getdate())

, Quantidade smallint not null CONSTRAINT dfQtd DEFAULT (1)

, NumeroCliente int not null

, CONSTRAINT pkVenda PRIMARY KEY (DataVenda)

, CONSTRAINT fkVenda FOREIGN KEY (NumeroCliente)

REFERENCES Cliente(idCliente));

Verificação de Valores

Os tipos de dados incluem uma restrição ao preenchimento dos dados em uma coluna, assim, quando definimos uma coluna como TINYINT, sabemos que os valores permitidos irão de 0 a 255.

No mesmo exemplo anterior poderíamos querer que os valores permitidos, além de serem numéricos, pudessem assumir somente os valores de 18 a 90. Esse tipo de restrição pode ser assegurada aplicando regras, que é o significado da cláusula CHECK.

Uma coluna pode ter qualquer número de restrições CHECK e os critérios podem incluir diversas expressões lógicas combinadas com AND e OR. Várias restrições CHECK são validadas na ordem de criação.

A avaliação do critério de pesquisa deve usar uma expressão Booleana (true/false) como base e não pode fazer referência a outra tabela. A restrição CHECK no nível de coluna pode fazer referência somente à coluna restrita. Restrições CHECK oferecem a mesma função de validação dos dados durante instruções INSERT e UPDATE. Se existirem uma ou mais restrições CHECK para uma coluna, todas as restrições serão avaliadas. A forma geral da restrição de verificação é mostrada abaixo:

CONSTRAINT <nome da regra> CHECK (<coluna com expressão booleana>)

Alguns exemplos são dados abaixo:

CONSTRAINT ckIdade CHECK (Idade <= 100)

CONSTRAINT ckTaxa CHECK (Taxa >= 1 and Taxa <= 5)

CONSTRAINT CK\_emp\_id CHECK (emp\_id LIKE ‘[A-Z][A-Z][A-Z][1-9][0-9][0-9][0-9]

[0-9][FM]’ OR emp\_id LIKE ‘[A-Z]-[A-Z][1-9][0-9][0-9][0-9][0-9][FM]’)

CONSTRAINT CK\_emp\_id CHECK (emp\_id IN (‘1389’, ‘0736’, ‘0877’, ‘1622’, ‘1756’)

OR emp\_id LIKE ‘99[0-9][0-9]’)

Um exemplo da restrição dentro do comando de criação de tabelas é mostrado a seguir:

CREATE TABLE Cliente

(

idCliente smallint identity(-32767, 1)

, Telefone VARCHAR(14)

, DataEntrada datetime

, Idade tinyint not null,

, constraint ckIdade CHECK (Idade between 18 and 90)

, constraint ckTelefone CHECK

(

Telefone LIKE ‘[0-9][0-9][0-9]-[0-9][0-9][0-9][0-9]’

OR

Telefone LIKE ‘([0-9][0-9][0-9]) [0-9][0-9][0-9]-[0-9][0-9][0-9][0-

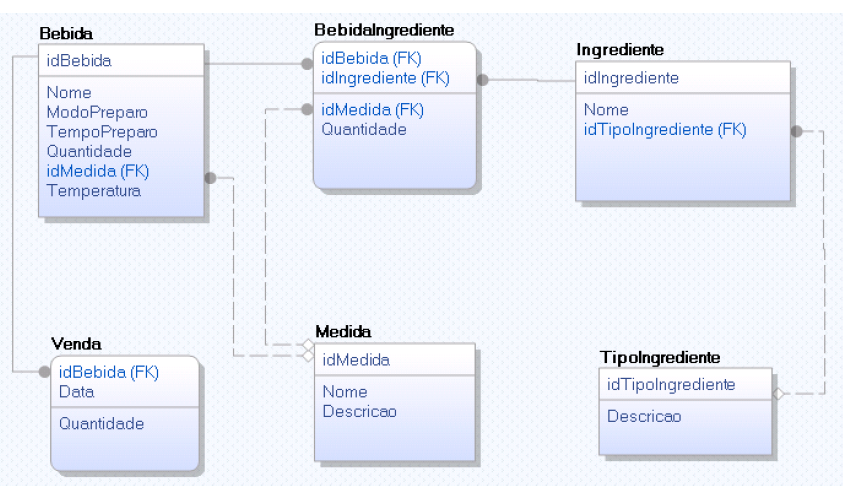
9])’

)

);

Exercícios página 19 da apostila do curso

Crie as tabelas do modelo abaixo escolhendo com cuidado os tipos de dados e respeitando todas as restrições (chave primária, chave estrangeira, unicidade, verificação, default e permissão de valores NULL):



Obs.: Para consulta de seu modelo use SP\_HELP.

30

**Resposta:**

CREATE TABLE Venda

(

idVenda INT NOT NULL IDENTITY(1,1) PRIMARY KEY,

Data DATATIME2 INT NOT NULL

Quantidade INT INT NOT NULL

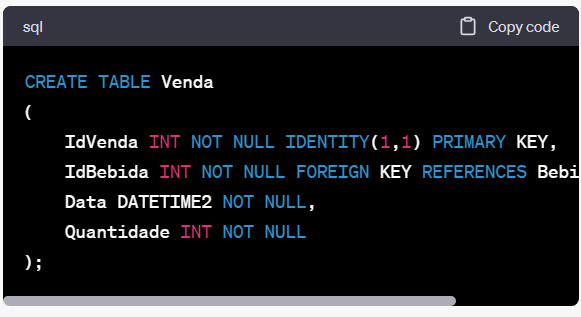
)

, CONSTRAINT pk idVenda PRIMARY KEY (Venda)

, CONSTRAINT fkidBebida FOREIGN KEY (Bebida)

);

**Resposta ChatGPT:**

****

CREATE TABLE Bebida

(

IdBebida INT NOT NULL IDENTITY(1,1) PRIMARY KEY,

NomeDaBebida VARCHAR(30) NOT NULL UNIQUE,

ModoDePreparo VARCHAR(Max),

TempoDePreparo Time,

Quantidade INT,

)

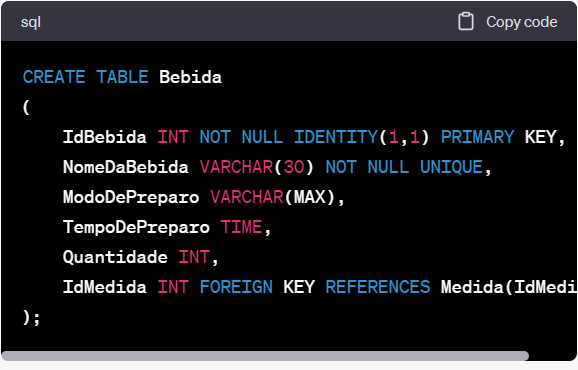
, CONSTRAINT pk IdBebida PRIMARY KEY (idBebida)

, CONSTRAINT fkidMedida FOREIGN KEY (idMedida)

REFERENCIAS Medida(Idmedida)

);

**Resposta ChatGPT:**

****

**Artigo**

[**https://www.devmedia.com.br/mer-e-der-modelagem-de-bancos-de-dados/14332**](https://www.devmedia.com.br/mer-e-der-modelagem-de-bancos-de-dados/14332)

**\*\*Obs: Este artigo é um resumo da aula de fundamentos de Banco de Dados.**