SQLS CREATE COMANDOS

CREATE TABLE users (

userid SERIAL PRIMARY KEY,

email VARCHAR(255) UNIQUE NOT NULL,

senha VARCHAR(255) NOT NULL,

telefone VARCHAR(20),

cnpj VARCHAR(14)

);

=============================================================================================

CREATE TABLE userMercado (

user\_mercado\_id SERIAL PRIMARY KEY,

nome\_mercado VARCHAR(255) NOT NULL,

refresh\_token VARCHAR(255),

access\_token VARCHAR(255),

userid INT,

FOREIGN KEY (userid) REFERENCES users(userid)

);

=============================================================================================

CREATE TABLE userMagalu (

user\_magalu\_id SERIAL PRIMARY KEY,

nome\_magalu VARCHAR(255) NOT NULL,

refresh\_token VARCHAR(255),

access\_token VARCHAR(255),

userid INT,

FOREIGN KEY (userid) REFERENCES users(userid)

);

==============================================================================================

CREATE TABLE productsMercado (

product\_sku VARCHAR(200) PRIMARY KEY,

title VARCHAR(255) NOT NULL,

price NUMERIC(10, 2) NOT NULL,

status VARCHAR(50) NOT NULL,

pictureUrls TEXT NOT NULL,

color VARCHAR(50),

diameter VARCHAR(50),

userid INT NOT NULL,

available\_quantity INTEGER

date\_created TIMESTAMP,

last\_updated TIMESTAMP,

UNIQUE (product\_sku, userid)

);

-- Function to update the updated\_at column

CREATE OR REPLACE FUNCTION update\_updated\_at\_column()

RETURNS TRIGGER AS $$

BEGIN

NEW.updated\_at = NOW();

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

-- Trigger to automatically update the updated\_at column on update

CREATE TRIGGER set\_updated\_at

BEFORE UPDATE ON productsMercado

FOR EACH ROW

EXECUTE FUNCTION update\_updated\_at\_column();

==============================================================================================

CREATE TABLE stock (

SKU VARCHAR(55) PRIMARY KEY,

Nome\_do\_Produto VARCHAR(255),

Apelido\_do\_Produto VARCHAR(255),

Categorias VARCHAR(255),

Codigo\_de\_Barras VARCHAR(50),

Data\_de\_Lancamento DATE,

Status\_da\_Venda VARCHAR(10) CHECK (Status\_da\_Venda IN ('Ativo', 'Inativo')),

Vendedor VARCHAR(100),

Preco\_de\_Varejo DECIMAL(10, 2),

Custo\_de\_Compra DECIMAL(10, 2),

Descricao TEXT,

Link\_do\_Fornecedor VARCHAR(255),

Marca VARCHAR(100),

Tamanho VARCHAR(50),

Peso\_do\_Pacote DECIMAL(10, 2),

Tamanho\_de\_Embalagem VARCHAR(50),

Link\_do\_Video VARCHAR(255),

NCM VARCHAR(20),

CEST VARCHAR(20),

Unidade VARCHAR(10),

Origem VARCHAR(50),

quantidade INTEGER,

transito INTEGER,

disponivel INTEGER,

quantidade\_total INTEGER,

skuMercado VARCHAR(50),

userid VARCHAR(100),

);

CREATE TABLE stock (

SKU VARCHAR(55) PRIMARY KEY,

Nome\_do\_Produto VARCHAR(255),

Apelido\_do\_Produto VARCHAR(255),

Categorias VARCHAR(255),

Codigo\_de\_Barras VARCHAR(50),

Data\_de\_Lancamento DATE,

Status\_da\_Venda VARCHAR(10) CHECK (Status\_da\_Venda IN ('Ativo', 'Inativo')),

Vendedor VARCHAR(100),

Preco\_de\_Varejo DECIMAL(10, 2),

Custo\_de\_Compra DECIMAL(10, 2),

Descricao TEXT,

Link\_do\_Fornecedor VARCHAR(255),

Marca VARCHAR(100),

Tamanho VARCHAR(50),

Peso\_do\_Pacote DECIMAL(10, 2),

Tamanho\_de\_Embalagem VARCHAR(50),

Link\_do\_Video VARCHAR(255),

NCM VARCHAR(20),

CEST VARCHAR(20),

Unidade VARCHAR(10),

Origem VARCHAR(50),

quantidade INTEGER,

transito INTEGER,

disponivel INTEGER,

quantidade\_total INTEGER,

skuMercado VARCHAR(50),

userid INT,

FOREIGN KEY (userid) REFERENCES users(userid)

);

==============================================================================================

CREATE TABLE stockVariant (

SPU INT PRIMARY KEY,

SKU VARCHAR(255),

Nome\_do\_Produto VARCHAR(255),

Apelido\_do\_Produto VARCHAR(100),

Categorias VARCHAR(255),

Codigo\_de\_Barras VARCHAR(50),

Data\_de\_Lancamento DATE,

Status\_da\_Venda VARCHAR(10) CHECK (Status\_da\_Venda IN ('Ativo', 'Inativo')),

Vendedor VARCHAR(100),

Preco\_de\_Varejo DECIMAL(10, 2),

Custo\_de\_Compra DECIMAL(10, 2),

Descricao TEXT,

Link\_do\_Fornecedor VARCHAR(255),

Marca VARCHAR(100),

Tamanho VARCHAR(50),

Cor VARCHAR(50),

Adicionar VARCHAR(255),

Peso\_do\_Pacote DECIMAL(10, 2),

Tamanho\_de\_Embalagem VARCHAR(50),

Link\_do\_Video VARCHAR(255),

NCM VARCHAR(20),

CEST VARCHAR(20),

Unidade VARCHAR(10),

Origem VARCHAR(50),

quantidade INTEGER,

transito INTEGER,

disponivel INTEGER,

quantidade\_total INTEGER,

userid VARCHAR(100)

);

==============================================================================================

CREATE TABLE stockKit (

SKUKIT VARCHAR(55) PRIMARY KEY,

SKU VARCHAR(255) NOT NULL,

Nome\_do\_Produto VARCHAR(255),

Apelido\_do\_Produto VARCHAR(255),

Categorias VARCHAR(255),

Custo\_de\_Compra DECIMAL(10, 2),

Status\_da\_Venda VARCHAR(10) CHECK (Status\_da\_Venda IN ('Ativo', 'Inativo')),

SkuMercado VARCHAR(50),

quantidade INTEGER,

userid INTEGER(100) NOT NULL,

FOREIGN KEY (userid) REFERENCES usuarios(userid)

);

==============================================================================================

CREATE TABLE ordersmercado (

order\_id VARCHAR(255),

reason VARCHAR(255),

total\_paid\_amount NUMERIC,

buyer\_nickname VARCHAR(255),

date\_last\_modified VARCHAR(90),

total\_amount NUMERIC,

date\_created TIMESTAMP,

seller\_nickname VARCHAR(255),

status VARCHAR(255),

substatus VARCHAR(255),

status\_simc VARCHAR(255),

pack\_id VARCHAR(255),

quantity INTEGER,

shipping\_id VARCHAR(255),

tracking\_number VARCHAR(255),

tracking\_method VARCHAR(255),

street\_name VARCHAR(255),

receiver\_name VARCHAR(255),

address\_line VARCHAR(255),

neighborhood VARCHAR(255),

city VARCHAR(255),

state VARCHAR(255),

zip\_code VARCHAR(50),

country VARCHAR(255)

userid VARCHAR(100),

product\_sku VARCHAR(255),

pictureUrls VARCHAR(255),

unit\_price NUMERIC(10, 2),

color\_name VARCHAR,

sale\_fee NUMERIC,

list\_cost NUMERIC,

invoice\_id VARCHAR(255),

COLUMN invoice\_key VARCHAR(255)

);

==============================================================================================

CREATE TABLE itemVisitsMercado (

user\_id VARCHAR(100) NOT NULL,

date\_from TIMESTAMP NOT NULL,

date\_to TIMESTAMP NOT NULL,

total\_visits INTEGER NOT NULL,

conversion\_rate NUMERIC(5, 2) NOT NULL,

);

CREATE TABLE itemVisitsMercado (

userid INT,

date\_from TIMESTAMP NOT NULL,

date\_to TIMESTAMP NOT NULL,

total\_visits INTEGER NOT NULL,

conversion\_rate NUMERIC(5, 2) NOT NULL,

FOREIGN KEY (userid) REFERENCES users(userid)

);

==============

CREATE TABLE companyInformation (

cnpj VARCHAR(20) PRIMARY KEY, -- CNPJ of the company, set as primary key

serial\_number INTEGER, -- Serial number

company\_name VARCHAR(255) NOT NULL, -- Name of the company

tax\_type VARCHAR(50), -- Type of taxation

company\_type VARCHAR(50), -- Type of the company (e.g., MEI, Ltda, etc.)

state\_registration VARCHAR(50), -- State Registration (Inscrição Estadual)

email VARCHAR(255) NOT NULL, -- Email of the company

postal\_code VARCHAR(10), -- Postal code (CEP)

address VARCHAR(255), -- Address of the company

address\_number VARCHAR(10), -- Address number

neighborhood VARCHAR(100), -- Neighborhood

city VARCHAR(100), -- City

state VARCHAR(2), -- State (e.g., SP, RJ)

userid VARCHAR(100) -- User ID associated with the company

);

—----------------------------------------------------------------------------------------------------------------------------

SHOPE

CREATE TABLE userShopee (

user\_shop\_id BIGINT PRIMARY KEY,

nome\_shopee VARCHAR(255) NOT NULL,

refresh\_token VARCHAR(255),

access\_token VARCHAR(255),

userid INT,

FOREIGN KEY (userid) REFERENCES users(userid)

);