CREATE DATABASE Real\_Estate\_Management\_System;

USE Real\_Estate\_Management\_System;

-- Client Table

CREATE TABLE Client (

ClientID INT IDENTITY(1,1) PRIMARY KEY,

FirstName VARCHAR(50) NOT NULL,

LastName VARCHAR(50) NOT NULL,

Email VARCHAR(100) UNIQUE NOT NULL,

Phone VARCHAR(20) -- Increased length for phone numbers

);

-- Agent Table

CREATE TABLE Agent (

AgentID INT IDENTITY(1,1) PRIMARY KEY,

FirstName VARCHAR(50) NOT NULL,

LastName VARCHAR(50) NOT NULL,

Email VARCHAR(100) UNIQUE NOT NULL,

Phone VARCHAR(20) -- Increased length for phone numbers

);

-- PropertyType Table

CREATE TABLE PropertyType (

PropertyTypeID INT IDENTITY(1,1) PRIMARY KEY,

TypeName VARCHAR(50) NOT NULL,

Description VARCHAR(255)

);

-- Property Table

CREATE TABLE Property (

PropertyID INT IDENTITY(1,1) PRIMARY KEY,

AgentID INT NOT NULL,

PropertyTypeID INT NOT NULL,

Address VARCHAR(255) NOT NULL,

City VARCHAR(100) NOT NULL,

State VARCHAR(50) NOT NULL,

ZipCode VARCHAR(10) NOT NULL,

FOREIGN KEY (AgentID) REFERENCES Agent(AgentID),

FOREIGN KEY (PropertyTypeID) REFERENCES PropertyType(PropertyTypeID)

);

-- Appointment Table

CREATE TABLE Appointment (

AppointmentID INT IDENTITY(1,1) PRIMARY KEY,

ClientID INT NOT NULL,

AgentID INT NOT NULL,

AppointmentDate DATE NOT NULL,

AppointmentTime TIME NOT NULL,

Purpose VARCHAR(255),

FOREIGN KEY (ClientID) REFERENCES Client(ClientID),

FOREIGN KEY (AgentID) REFERENCES Agent(AgentID)

);

-- LeaseAgreement Table

CREATE TABLE LeaseAgreement (

LeaseID INT IDENTITY(1,1) PRIMARY KEY,

PropertyID INT NOT NULL,

ClientID INT NOT NULL,

AgentID INT NOT NULL,

LeaseStartDate DATE NOT NULL,

LeaseEndDate DATE NOT NULL,

MonthlyRent DECIMAL(15, 2) NOT NULL,

FOREIGN KEY (PropertyID) REFERENCES Property(PropertyID),

FOREIGN KEY (ClientID) REFERENCES Client(ClientID),

FOREIGN KEY (AgentID) REFERENCES Agent(AgentID)

);

-- Transaction Table

CREATE TABLE [Transaction] (

TransactionID INT IDENTITY(1,1) PRIMARY KEY,

ClientID INT NOT NULL,

PropertyID INT NOT NULL,

AgentID INT NOT NULL,

TransactionDate DATE NOT NULL,

Amount DECIMAL(15, 2) NOT NULL,

PaymentStatus VARCHAR(50) NOT NULL,

FOREIGN KEY (ClientID) REFERENCES Client(ClientID),

FOREIGN KEY (PropertyID) REFERENCES Property(PropertyID),

FOREIGN KEY (AgentID) REFERENCES Agent(AgentID)

);

-- Payment Table

CREATE TABLE Payment (

PaymentID INT IDENTITY(1,1) PRIMARY KEY,

TransactionID INT NOT NULL,

PaymentDate DATE NOT NULL,

PaymentMethod VARCHAR(50),

Amount DECIMAL(15, 2) NOT NULL,

FOREIGN KEY (TransactionID) REFERENCES [Transaction](TransactionID)

);

-- Inspection Table

CREATE TABLE Inspection (

InspectionID INT IDENTITY(1,1) PRIMARY KEY,

PropertyID INT NOT NULL,

AgentID INT NOT NULL,

InspectionDate DATE NOT NULL,

InspectionTime TIME NOT NULL,

Notes VARCHAR(255),

FOREIGN KEY (PropertyID) REFERENCES Property(PropertyID),

FOREIGN KEY (AgentID) REFERENCES Agent(AgentID)

);

-- Documents Table

CREATE TABLE Documents (

DocumentID INT IDENTITY(1,1) PRIMARY KEY,

PropertyID INT NOT NULL,

TransactionID INT NOT NULL,

DocumentType VARCHAR(50) NOT NULL,

FilePath VARCHAR(255) NOT NULL,

UploadDate DATETIME NOT NULL,

FOREIGN KEY (PropertyID) REFERENCES Property(PropertyID),

FOREIGN KEY (TransactionID) REFERENCES [Transaction](TransactionID)

);

-- MaintenanceRequest Table

CREATE TABLE MaintenanceRequest (

RequestID INT IDENTITY(1,1) PRIMARY KEY,

PropertyID INT NOT NULL,

ClientID INT NOT NULL,

RequestDate DATE NOT NULL,

Status VARCHAR(50) CHECK (Status IN ('Pending', 'In Progress', 'Completed')),

Description VARCHAR(255),

FOREIGN KEY (PropertyID) REFERENCES Property(PropertyID),

FOREIGN KEY (ClientID) REFERENCES Client(ClientID)

);

-- UserRoles Table

CREATE TABLE UserRoles (

RoleID INT IDENTITY(1,1) PRIMARY KEY,

RoleName VARCHAR(50) NOT NULL

);

-- UserAccounts Table

CREATE TABLE UserAccounts (

UserID INT IDENTITY(1,1) PRIMARY KEY,

Username VARCHAR(50) UNIQUE NOT NULL,

Password VARCHAR(100) NOT NULL,

RoleID INT NOT NULL,

FOREIGN KEY (RoleID) REFERENCES UserRoles(RoleID)

);

-- Admin Table

CREATE TABLE Admin (

AdminID INT IDENTITY(1,1) PRIMARY KEY,

Username VARCHAR(50) UNIQUE NOT NULL,

Password VARCHAR(100) NOT NULL,

FirstName VARCHAR(50),

LastName VARCHAR(50),

Email VARCHAR(100),

Phone VARCHAR(20) -- Increased length for phone numbers

);

-- Comments Table

CREATE TABLE Comments (

CommentID INT IDENTITY(1,1) PRIMARY KEY,

ClientID INT NOT NULL,

AgentID INT NOT NULL,

PropertyID INT NOT NULL,

CommentText TEXT NOT NULL,

CommentDate DATETIME NOT NULL,

FOREIGN KEY (ClientID) REFERENCES Client(ClientID),

FOREIGN KEY (AgentID) REFERENCES Agent(AgentID),

FOREIGN KEY (PropertyID) REFERENCES Property(PropertyID)

);

-- AdminActivityLog Table

CREATE TABLE AdminActivityLog (

LogID INT IDENTITY(1,1) PRIMARY KEY,

AdminID INT NOT NULL,

Action VARCHAR(50) NOT NULL,

Timestamp DATETIME NOT NULL,

FOREIGN KEY (AdminID) REFERENCES Admin(AdminID)

);

-- Insert Initial Data with Conflict Handling for SQL Server

-- Using "IF NOT EXISTS" for SQL Server

IF NOT EXISTS (SELECT 1 FROM Client WHERE Email = 'jane.smith@example.com')

BEGIN

INSERT INTO Client (FirstName, LastName, Email, Phone)

VALUES

('Jane', 'Smith', 'jane.smith@example.com', '987-654-3210'),

('Michael', 'Johnson', 'michael.johnson@example.com', '456-789-0123'),

('Emily', 'Williams', 'emily.williams@example.com', '789-012-3456'),

('David', 'Brown', 'david.brown@example.com', '321-654-9870');

END;

IF NOT EXISTS (SELECT 1 FROM Agent WHERE Email = 'alice.johnson@example.com')

BEGIN

INSERT INTO Agent (FirstName, LastName, Email, Phone)

VALUES

('Alice', 'Johnson', 'alice.johnson@example.com', '111-222-3333'),

('Bob', 'Smith', 'bob.smith@example.com', '444-555-6666'),

('Carol', 'Williams', 'carol.williams@example.com', '777-888-9999'),

('Daniel', 'Jones', 'daniel.jones@example.com', '000-999-8888'),

('Eva', 'Brown', 'eva.brown@example.com', '333-444-5555');

END;

IF NOT EXISTS (SELECT 1 FROM PropertyType WHERE TypeName = 'Apartment')

BEGIN

INSERT INTO PropertyType (TypeName, Description)

VALUES

('Apartment', 'A residential unit in a building'),

('House', 'A standalone residential structure'),

('Condo', 'A unit in a multi-unit building with shared common areas'),

('Commercial', 'A property used for business purposes'),

('Land', 'A plot of land available for development');

END;

IF NOT EXISTS (SELECT 1 FROM Property WHERE Address = '123 Main St')

BEGIN

INSERT INTO Property (AgentID, PropertyTypeID, Address, City, State, ZipCode)

VALUES

(1, 1, '123 Main St', 'Springfield', 'IL', '62701'),

(2, 2, '456 Oak Ave', 'Chicago', 'IL', '60601'),

(3, 3, '789 Pine Rd', 'Naperville', 'IL', '60563'),

(4, 4, '101 State St', 'Peoria', 'IL', '61602'),

(5, 5, '202 Green Ln', 'Champaign', 'IL', '61820');

END;

-- AddTransaction procedure remains unchanged

-- View for Appointment Details

DROP VIEW IF EXISTS vw\_AppointmentDetails;

CREATE VIEW vw\_AppointmentDetails AS

SELECT

a.AppointmentID,

a.AppointmentDate,

a.AppointmentTime,

a.Purpose,

CONCAT(c.FirstName, ' ', c.LastName) AS ClientName,

ag.FirstName AS AgentFirstName,

ag.LastName AS AgentLastName

FROM

Appointment a

JOIN

Client c ON a.ClientID = c.ClientID

JOIN

Agent ag ON a.AgentID = ag.AgentID;