CREATE DATABASE Агентство\_недвижимости;

GO

USE Агентство\_недвижимости;

GO

-- Таблица "Агенты"

CREATE TABLE Агенты (

id\_агента INT PRIMARY KEY IDENTITY(1,1),

Имя VARCHAR(50) NOT NULL,

Фамилия VARCHAR(50) NOT NULL,

Телефон VARCHAR(20) NOT NULL,

Адрес VARCHAR(100) NOT NULL

);

GO

-- Таблица "Клиенты"

CREATE TABLE Клиенты (

id\_клиента INT PRIMARY KEY IDENTITY(1,1),

Имя VARCHAR(50) NOT NULL,

Фамилия VARCHAR(50) NOT NULL,

Телефон VARCHAR(20) NOT NULL,

Адрес VARCHAR(100) NOT NULL

);

GO

-- Таблица "Недвижимость"

CREATE TABLE Недвижимость (

id\_недвижимости INT PRIMARY KEY IDENTITY(1,1),

Тип\_недвижимости VARCHAR(50) NOT NULL,

Адрес VARCHAR(100) NOT NULL,

Цена DECIMAL(10, 2) NOT NULL

);

GO

-- Таблица "Сделки"

CREATE TABLE Сделки (

id\_сделки INT PRIMARY KEY IDENTITY(1,1),

id\_агента INT,

id\_клиента INT,

id\_недвижимости INT,

Дата\_сделки DATE NOT NULL,

FOREIGN KEY (id\_агента) REFERENCES Агенты (id\_агента),

FOREIGN KEY (id\_клиента) REFERENCES Клиенты (id\_клиента),

FOREIGN KEY (id\_недвижимости) REFERENCES Недвижимость (id\_недвижимости)

);

GO

-- Заполнение таблицы "Агенты"

INSERT INTO Агенты (Имя, Фамилия, Телефон, Адрес)

VALUES ('Иван', 'Иванов', '123456789', 'ул. Центральная 1'),

('Петр', 'Петров', '987654321', 'ул. Солнечная 10'),

('Анна', 'Сидорова', '555555555', 'ул. Парковая 5');

GO

-- Заполнение таблицы "Клиенты"

INSERT INTO Клиенты (Имя, Фамилия, Телефон, Адрес)

VALUES ('Елена', 'Смирнова', '111111111', 'ул. Садовая 2'),

('Михаил', 'Козлов', '222222222', 'ул. Лесная 15'),

('Ольга', 'Васильева', '333333333', 'ул. Речная 7');

GO

-- Заполнение таблицы "Недвижимость"

INSERT INTO Недвижимость (Тип\_недвижимости, Адрес, Цена)

VALUES ('Квартира', 'ул. Пушкина 10', 1500000),

('Дом', 'ул. Ленина 5', 3000000),

('Земельный участок', 'ул. Садовая 20', 500000);

GO

-- Заполнение таблицы "Сделки"

INSERT INTO Сделки (id\_агента, id\_клиента, id\_недвижимости, Дата\_сделки)

VALUES (1, 2, 1, '2023-05-15'),

(2, 3, 2, '2023-05-20'),

(3, 1, 3, '2023-05-25');

GO

CREATE VIEW Список\_сделок AS

SELECT Сделки.id\_сделки, Агенты.Фамилия AS Фамилия\_агента, Клиенты.Фамилия AS Фамилия\_клиента, Недвижимость.Адрес AS Адрес\_недвижимости, Сделки.Дата\_сделки

FROM Сделки

JOIN Агенты ON Сделки.id\_агента = Агенты.id\_агента

JOIN Клиенты ON Сделки.id\_клиента = Клиенты.id\_клиента

JOIN Недвижимость ON Сделки.id\_недвижимости = Недвижимость.id\_недвижимости;

GO

CREATE VIEW Список\_свободной\_недвижимости AS

SELECT Недвижимость.id\_недвижимости, Недвижимость.Тип\_недвижимости, Недвижимость.Адрес, Недвижимость.Цена

FROM Недвижимость

LEFT JOIN Сделки ON Недвижимость.id\_недвижимости = Сделки.id\_недвижимости

WHERE Сделки.id\_сделки IS NULL;

GO

CREATE VIEW Список\_клиентов\_суммарных\_затрат AS

SELECT Клиенты.id\_клиента, Клиенты.Фамилия, Клиенты.Телефон, SUM(Недвижимость.Цена) AS Суммарные\_затраты

FROM Клиенты

JOIN Сделки ON Клиенты.id\_клиента = Сделки.id\_клиента

JOIN Недвижимость ON Сделки.id\_недвижимости = Недвижимость.id\_недвижимости

GROUP BY Клиенты.id\_клиента, Клиенты.Фамилия, Клиенты.Телефон;

GO

--Триггер, который автоматически устанавливает текущую дату при вставке новой сделки:

CREATE TRIGGER Установка\_даты\_сделки

ON Сделки

AFTER INSERT

AS

BEGIN

UPDATE Сделки

SET Дата\_сделки = GETDATE()

WHERE id\_сделки IN (SELECT id\_сделки FROM inserted);

END;

GO

--Этот триггер проверяет, что цена недвижимости во вставленной сделке не превышает максимально допустимую цену из таблицы "Недвижимость".

--Если цена превышает максимальное значение, то транзакция откатывается и выводится сообщение об ошибке.

CREATE TRIGGER Проверка\_максимальной\_цены

ON Сделки

AFTER INSERT

AS

BEGIN

DECLARE @Максимальная\_цена int = 1000000; -- Максимально допустимая цена

IF EXISTS (SELECT 1 FROM inserted i INNER JOIN Недвижимость n ON i.id\_недвижимости = n.id\_недвижимости WHERE n.Цена > @Максимальная\_цена)

BEGIN

RAISERROR('Цена недвижимости превышает максимально допустимую.', 16, 1);

ROLLBACK TRANSACTION;

END

END;

GO

--Триггер для контроля ограничения на количество записей в таблице "Клиенты". Он позволяет добавлять только определенное количество записей:

CREATE TRIGGER Ограничение\_количества\_клиентов

ON Клиенты

AFTER INSERT

AS

BEGIN

DECLARE @Максимальное\_количество INT = 100; -- Максимальное количество клиентов

IF (SELECT COUNT(\*) FROM Клиенты) > @Максимальное\_количество

BEGIN

RAISERROR('Достигнуто максимальное количество клиентов.', 16, 1);

ROLLBACK TRANSACTION;

END;

END;

GO

CREATE FUNCTION ПолучитьКоличествоСделокПоАгенту(@id\_агента INT)

RETURNS INT

AS

BEGIN

DECLARE @количество INT;

SELECT @количество = COUNT(\*) FROM Сделки WHERE id\_агента = @id\_агента;

RETURN @количество;

END;

go

CREATE FUNCTION РассчитатьОбщуюСуммуСделокКлиента(@id\_клиента INT)

RETURNS DECIMAL(10, 2)

AS

BEGIN

DECLARE @сумма DECIMAL(10, 2);

SELECT @сумма = SUM(Цена) FROM Сделки INNER JOIN Недвижимость ON Сделки.id\_недвижимости = Недвижимость.id\_недвижимости

WHERE id\_клиента = @id\_клиента;

RETURN @сумма;

END;

go

CREATE FUNCTION ПолучитьСписокДоступнойНедвижимости()

RETURNS TABLE

AS

RETURN

(

SELECT id\_недвижимости, Тип\_недвижимости, Адрес, Цена

FROM Недвижимость

WHERE id\_недвижимости NOT IN (SELECT id\_недвижимости FROM Сделки)

);

go

--Процедура "Добавить сделку":

CREATE PROCEDURE Добавить\_сделку

@id\_агента INT,

@id\_клиента INT,

@id\_недвижимости INT,

@дата\_сделки DATE

AS

BEGIN

INSERT INTO Сделки (id\_агента, id\_клиента, id\_недвижимости, Дата\_сделки)

VALUES (@id\_агента, @id\_клиента, @id\_недвижимости, @дата\_сделки);

END;

--Процедура "Обновить информацию о клиенте":

CREATE PROCEDURE Обновить\_информацию\_о\_клиенте

@id\_клиента INT,

@новое\_имя NVARCHAR(50),

@новая\_фамилия NVARCHAR(50),

@новый\_телефон NVARCHAR(50)

AS

BEGIN

UPDATE Клиенты

SET Имя = @новое\_имя, Фамилия = @новая\_фамилия, Телефон = @новый\_телефон

WHERE id\_клиента = @id\_клиента;

END;

--Процедура "Удалить недвижимость":

CREATE PROCEDURE Удалить\_недвижимость

@id\_недвижимости INT

AS

BEGIN

DELETE FROM Недвижимость

WHERE id\_недвижимости = @id\_недвижимости;

END;

1. **Перечислите основные принципы построения концептуальной, логической и физической модели данных.**

Основные принципы построения концептуальной, логической и физической моделей данных включают:

Концептуальная модель данных:

• Абстракция: Она должна описывать сущности, их атрибуты и связи на высоком уровне, отражая предметную область без учета деталей хранения данных.

• Унификация: Концептуальная модель должна объединять данные из различных источников и представлять общую семантику и структуру данных.

• Простота и понятность: Модель должна быть простой и легко понятной для заинтересованных сторон, включая аналитиков, разработчиков и пользователей.

Логическая модель данных:

• Структурированность: Она должна быть организована таким образом, чтобы обеспечить эффективное хранение и доступ к данным.

• Нормализация: Нормализация помогает устранить аномалии и повысить эффективность работы с данными.

• Согласованность: Логическая модель должна быть согласованной с концептуальной моделью и отражать ее семантику и структуру.

Физическая модель данных:

• Оптимизация: Она должна быть оптимизирована для обеспечения высокой производительности и эффективности работы с данными.

• Индексирование: Индексы помогают сократить время выполнения запросов и улучшить общую производительность системы.

• Физические ограничения: Физическая модель определяет ограничения целостности данных, такие как уникальность, ссылочная целостность и другие правила, которые должны соблюдаться при работе с данными на физическом уровне.