CREATE DATABASE База\_ГСМ;

GO

USE База\_ГСМ;

GO

-- Создание таблицы "Товары"

CREATE TABLE Товары (

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

Наименование VARCHAR(100) NOT NULL,

Тип VARCHAR(50) NOT NULL,

Цена DECIMAL(10, 2) NOT NULL,

Количество INT NOT NULL

);

GO

-- Создание таблицы "Поставщики"

CREATE TABLE Поставщики (

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

Наименование VARCHAR(100) NOT NULL,

Адрес VARCHAR(200) NOT NULL,

Телефон VARCHAR(20) NOT NULL,

Email VARCHAR(100)

);

GO

-- Создание таблицы "Поставки"

CREATE TABLE Поставки (

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

id\_товара INT,

id\_поставщика INT,

Дата\_поставки DATE NOT NULL,

Количество INT NOT NULL,

FOREIGN KEY (id\_товара) REFERENCES Товары(id\_товара),

FOREIGN KEY (id\_поставщика) REFERENCES Поставщики(id\_поставщика)

);

GO

-- Создание таблицы "Клиенты"

CREATE TABLE Клиенты (

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

Имя VARCHAR(100) NOT NULL,

Фамилия VARCHAR(100) NOT NULL,

Телефон VARCHAR(20) NOT NULL,

Email VARCHAR(100)

);

GO

-- Создание таблицы "Продажи"

CREATE TABLE Продажи (

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

id\_товара INT,

id\_клиента INT,

Дата\_продажи DATE NOT NULL,

Количество INT NOT NULL,

FOREIGN KEY (id\_товара) REFERENCES Товары(id\_товара),

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

);

GO

use База\_ГСМ

go

-- Заполнение таблицы "Товары"

INSERT INTO Товары (Наименование, Тип, Цена, Количество)

VALUES

('Моторное масло', 'Масло', 100.00, 50),

('Трансмиссионное масло', 'Масло', 80.00, 30),

('Тормозная жидкость', 'Жидкость', 50.00, 20),

('Антифриз', 'Жидкость', 60.00, 15),

('Смазка универсальная', 'Смазка', 40.00, 40);

-- Заполнение таблицы "Поставщики"

INSERT INTO Поставщики (Наименование, Адрес, Телефон, Email)

VALUES

('ООО "Нефтепром"', 'г. Москва, ул. Ленина, 1', '+7 (495) 123-4567', 'info@nefteprom.ru'),

('ОАО "Роснефть"', 'г. Санкт-Петербург, ул. Пушкина, 10', '+7 (812) 987-6543', 'info@rosneft.ru'),

('ЗАО "Газпромнефть"', 'г. Новосибирск, ул. Гагарина, 5', '+7 (383) 111-2222', 'info@gazpromneft.ru');

-- Заполнение таблицы "Поставки"

INSERT INTO Поставки (id\_товара, id\_поставщика, Дата\_поставки, Количество)

VALUES

(1, 1, '2023-05-01', 20),

(2, 2, '2023-05-02', 10),

(3, 3, '2023-05-03', 15),

(4, 1, '2023-05-04', 8),

(5, 2, '2023-05-05', 25);

-- Заполнение таблицы "Клиенты"

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

VALUES

('Иван', 'Петров', '+7 (495) 111-1111', 'ivan@mail.com'),

('Алексей', 'Сидоров', '+7 (495) 222-2222', 'alexey@mail.com'),

('Елена', 'Иванова', '+7 (495) 333-3333', 'elena@mail.com');

-- Заполнение таблицы "Продажи"

INSERT INTO Продажи (id\_товара, id\_клиента, Дата\_продажи, Количество)

VALUES

(1, 1, '2023-06-01', 5),

(2, 2, '2023-06-01', 3),

(3, 3, '2023-06-01', 2),

(4, 1, '2023-06-01', 1),

(5, 2, '2023-06-01', 4);

USE База\_ГСМ;

GO

CREATE FUNCTION Общая\_Стоимость\_Товара (@id\_товара INT)

RETURNS DECIMAL(10, 2)

AS

BEGIN

DECLARE @общая\_стоимость DECIMAL(10, 2);

SELECT @общая\_стоимость = Цена \* Количество

FROM Товары

WHERE id\_товара = @id\_товара;

RETURN @общая\_стоимость;

END;

go

CREATE FUNCTION Количество\_Поставок\_Товара (@id\_товара INT)

RETURNS INT

AS

BEGIN

DECLARE @количество\_поставок INT;

SELECT @количество\_поставок = COUNT(\*)

FROM Поставки

WHERE id\_товара = @id\_товара;

RETURN @количество\_поставок;

END;

go

CREATE FUNCTION Общая\_Сумма\_Продаж (@id\_клиента INT)

RETURNS DECIMAL(10, 2)

AS

BEGIN

DECLARE @общая\_сумма DECIMAL(10, 2);

SELECT @общая\_сумма = SUM(Товары.Цена \* Продажи.Количество)

FROM Продажи

INNER JOIN Товары ON Продажи.id\_товара = Товары.id\_товара

WHERE Продажи.id\_клиента = @id\_клиента;

RETURN @общая\_сумма;

END;

go

USE База\_ГСМ;

GO

CREATE TRIGGER Проверка\_Количества\_Товара

ON Продажи

AFTER INSERT

AS

BEGIN

IF EXISTS (

SELECT t.Количество

FROM inserted i

INNER JOIN Товары t ON i.id\_товара = t.id\_товара

WHERE t.Количество < i.Количество

)

BEGIN

RAISERROR ('Недостаточное количество товара.', 16, 1);

ROLLBACK TRANSACTION;

END;

END;

go

CREATE TRIGGER Обновление\_Количества\_Товара

ON Продажи

INSTEAD OF INSERT

AS

BEGIN

DECLARE @productId INT;

DECLARE @quantity INT;

-- Получ1ение id\_товара и количества из вставленной строки

SELECT @productId = id\_товара, @quantity = Количество FROM inserted;

-- Обновление количества товара в таблице "Товары"

UPDATE Товары

SET Количество = Количество - @quantity

WHERE id\_товара = @productId;

-- Вставка данных в таблицу "Продажи"

INSERT INTO Продажи (id\_товара, id\_клиента, Дата\_продажи, Количество)

SELECT id\_товара, id\_клиента, Дата\_продажи, Количество FROM inserted;

END;

go

CREATE TRIGGER Установка\_Дата\_Поставки

ON Поставки

AFTER INSERT

AS

BEGIN

UPDATE Поставки

SET Дата\_поставки = GETDATE()

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

END;

go

USE База\_ГСМ;

GO

CREATE VIEW Продукты\_в\_наличии AS

SELECT Наименование, Тип, Цена, Количество

FROM Товары

WHERE Количество > 0;

go

CREATE VIEW Поставки\_по\_поставщику AS

SELECT Поставщики.Наименование, Поставки.id\_поставки, Товары.Наименование AS Товар, Поставки.Дата\_поставки, Поставки.Количество

FROM Поставки

JOIN Поставщики ON Поставки.id\_поставщика = Поставщики.id\_поставщика

JOIN Товары ON Поставки.id\_товара = Товары.id\_товара;

go

CREATE VIEW Продажи\_по\_клиенту AS

SELECT Клиенты.Имя, Клиенты.Фамилия, Продажи.id\_продажи, Товары.Наименование AS Товар, Продажи.Дата\_продажи, Продажи.Количество

FROM Продажи

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

JOIN Товары ON Продажи.id\_товара = Товары.id\_товара;

go

USE База\_ГСМ;

GO

CREATE PROCEDURE Добавить\_Товар

@Наименование VARCHAR(100),

@Тип VARCHAR(50),

@Цена DECIMAL(10, 2),

@Количество INT

AS

BEGIN

INSERT INTO Товары (Наименование, Тип, Цена, Количество)

VALUES (@Наименование, @Тип, @Цена, @Количество);

END;

go

CREATE PROCEDURE Поставить\_Товар

@id\_товара INT,

@id\_поставщика INT,

@Дата\_поставки DATE,

@Количество INT

AS

BEGIN

INSERT INTO Поставки (id\_товара, id\_поставщика, Дата\_поставки, Количество)

VALUES (@id\_товара, @id\_поставщика, @Дата\_поставки, @Количество);

END;

go

CREATE PROCEDURE Продать\_Товар

@id\_товара INT,

@id\_клиента INT,

@Дата\_продажи DATE,

@Количество INT

AS

BEGIN

INSERT INTO Продажи (id\_товара, id\_клиента, Дата\_продажи, Количество)

VALUES (@id\_товара, @id\_клиента, @Дата\_продажи, @Количество);

END;

go

1. **Перечислите нормальные формы БД**

Вот перечень нормальных форм баз данных (НФБД):

1. Первая нормальная форма (1НФ): Все атрибуты в таблице должны быть атомарными, то есть не должны содержать повторяющихся или составных значений.

2. Вторая нормальная форма (2НФ): Таблица должна быть в 1НФ, и каждый неключевой атрибут должен полностью зависеть от всего составного первичного ключа.

3. Третья нормальная форма (3НФ): Таблица должна быть в 2НФ, и каждый неключевой атрибут должен зависеть только от первичного ключа, а не от других неключевых атрибутов.

4. Четвёртая нормальная форма (4НФ): Таблица должна быть в 3НФ, и неключевые атрибуты не должны иметь зависимых многозначных фактов.

5. Пятая нормальная форма (5НФ) или нормальная форма проекции-соединения (ПСНФ): Таблица должна быть в 4НФ, и любая зависимость должна быть выражена только через ключевые атрибуты.

6. Доменно-ключевая нормальная форма (ДКНФ): Все атрибуты, не входящие в ключевой атрибут, должны полностью зависеть от каждого ключевого атрибута, а не от их комбинации.

7. Шестая нормальная форма (6НФ): Цель этой нормальной формы состоит в устранении избыточности данных, путём разделения многозначных фактов и сохранения их отношений.