Министерство транспорта Российской Федерации

Федеральное государственное автономное образовательное  
учреждение высшего образования  
«РУТ (МИИТ)»

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

КУРСОВОЙ ПРОЕКТ

по дисциплине

**«Основы построения защищенных баз данных»**

**на тему  
«Овощной магазин»**

Выполнил: ст. гр. ТКИ-441

Плуталов Е.А.

Проверил: доц., к.т.н.

Васильева М. А.

Москва 2024

Оглавление

[ЦЕЛЬ КУРСОВОГО ПРОЕКТА \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_5](#_Toc136201526)

[ЗАДАНИЕ НА КУРСОВОВОЙ ПРОЕКТ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 5](#_Toc136201527)

[ПРОЕКТИРОВАНИЕ РЕЛЯЦИОННОЙ БАЗЫ ДАННЫХ\_\_\_\_\_ 6](#_Toc136201528)

[1. Инфологическое проектирование 6](#_Toc136201529)

[1.1 Анализ предметной области 6](#_Toc136201530)

[1.2 Анализ информационных задач и круга пользователей системы 6](#_Toc136201531)

[2. Логическое проектирование реляционной БД 7](#_Toc136201532)

[2.1 Составление реляционных отношений 7](#_Toc136201533)

[3. Физическое проектирование БД 13](#_Toc136201534)

[3.1 Разработка скриптов на создание базы данных и таблиц 13](#_Toc136201535)

[3.2 Разработка скриптов на добавление данных в таблицы 14](#_Toc136201536)

[3.3 Разработка необходимых запросов: 14](#_Toc136201537)

[3.4 Разработка необходимых запросов, представлений (view), процедур, функций и триггеров: 18](#_Toc136201538)

[Список литературы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_21](#_Toc136201539)

# ЦЕЛЬ КУРСОВОГО ПРОЕКТА

Целью курсового проекта является изучение методов и закрепление знаний в проектировании реляционных баз данных (РБД) в системе управления базами данных (СУБД) Система управления базами данных PostgreSQL.

# ЗАДАНИЕ НА КУРСОВОВОЙ ПРОЕКТ

В данном курсовом проекте ставится задача разработать РБД в СУБД PostgreSQL по заданной теме «Овощной магазин». Проектирование РБД проводится с помощью метода «Сущность-связь». Проверка построенной модели РБД осуществляется с помощью метода нормализации отношений.

**Описание предметной области.**

БД создается для информационного обслуживания сотрудников магазина. Магазин получает товар со склада и расфасовывает его для продажи. Некоторые овощи могут быть испорчены.

**Готовые запросы:**

* Показывать ассортимент товара на выбранный день.
* Показывать приход-расход выбранного товара.
* Показывать прибыль магазина (без учета налогов и т. д.) по

указанному товару (за отчетный период времени).

* Показывать в % отношении количество хорошего и испорченного

товара для выбранного товара.

* Показывать количество проданного данного товара за отчетный

период.

# ПРОЕКТИРОВАНИЕ РЕЛЯЦИОННОЙ БАЗЫ ДАННЫХ

## Инфологическое проектирование

### Анализ предметной области

База данных создаётся сотрудников овощного магазина.

Выделим базовые сущности этой предметной области:

* Клиенты
* Поставщики
* Овощи
* Заказы

ER-диаграмма БД, приведена на (Рисунке 1).

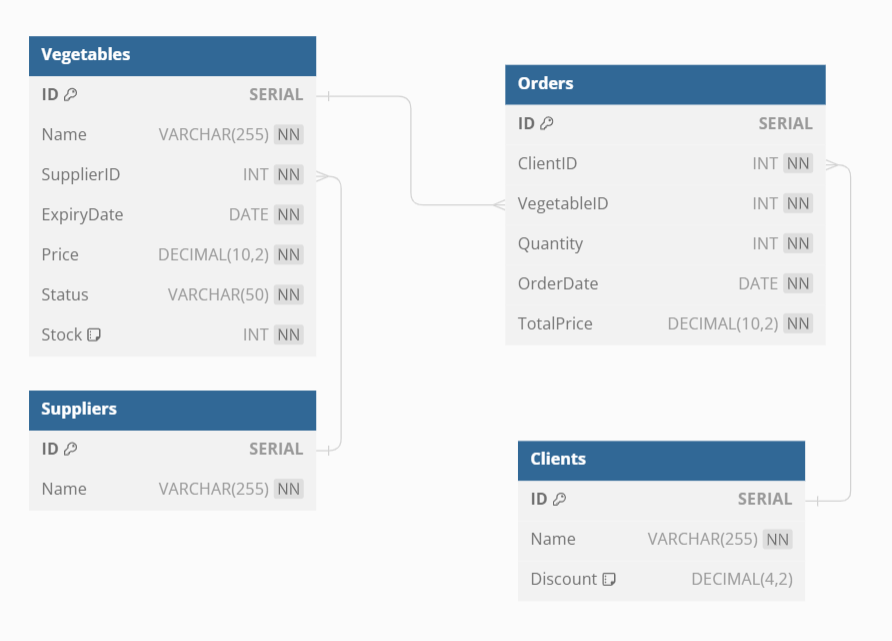


Рисунок 1 – ER-диаграмма

### Анализ информационных задач и круга пользователей системы

Определим границы информационной поддержки пользователей:

**1) Функциональные возможности:**

* ведение БД (запись, чтение, модификация, удаление);
* обеспечение логической непротиворечивости БД;
* реализация наиболее часто встречающихся запросов в готовом виде.

**2) Готовые запросы:**

• Показывать ассортимент товара на выбранный день.

• Показывать приход-расход выбранного товара.

• Показывать прибыль магазина (без учета налогов и т. д.) по

указанному товару (за отчетный период времени).

• Показывать в % отношении количество хорошего и испорченного

товара для выбранного товара.

• Показывать количество проданного данного товара за отчетный

период.

## Логическое проектирование реляционной БД

### Составление реляционных отношений

Каждое реляционное отношение соответствует одной сущности (объекту ПрО), и в него вносятся все атрибуты сущности. Для каждого отношения необходимо определить первичный ключ и внешние ключи (если они есть). В том случае, если базовое отношение не имеет потенциальных ключей, вводится *суррогатный первичный ключ*, который не несёт смысловой нагрузки и служит только для идентификации записей.

Были созданы следующие сущности:

1. Таблица "Clients" содержит информацию о клиентах овощного магазина. Основные поля включают уникальный идентификатор клиента (ID), имя клиента (Name) и скидку (Discount), которая может быть применена к покупкам клиента. Поле ID является первичным ключом, обеспечивающим уникальность каждой записи. Имя клиента является обязательным полем, длиной до 255 символов. Скидка указана в виде числа с плавающей запятой с точностью до двух знаков после запятой и имеет максимальное значение 100, указывая процент скидки, которую клиент может получить.
2. Таблица "Suppliers" содержит информацию о поставщиках, которые снабжают магазин овощами. Основные поля включают уникальный идентификатор поставщика (ID) и название поставщика (Name). Поле ID является первичным ключом и гарантирует уникальность каждой записи. Название поставщика является обязательным полем и может содержать до 255 символов.
3. Таблица "Vegetables" содержит информацию о различных овощах, доступных в магазине. Основные поля включают уникальный идентификатор овоща (ID), название овоща (Name), идентификатор поставщика (SupplierID), срок годности (ExpiryDate), цену (Price) и состояние (Status). Поле ID является первичным ключом. Название овоща и идентификатор поставщика являются обязательными полями, где Name может содержать до 255 символов, а SupplierID ссылается на идентификатор поставщика из таблицы "Suppliers". Срок годности и цена также являются обязательными полями, где ExpiryDate указывает дату истечения срока годности, а Price указана в виде числа с плавающей запятой. Поле состояния (Status) содержит описание состояния овоща, например, 'хороший' или 'испорченный'. Stock: Количество товара на складе, по умолчанию 0.
4. Таблица "Orders" содержит информацию о заказах, сделанных клиентами. Основные поля включают уникальный идентификатор заказа (ID), идентификатор клиента (ClientID), идентификатор овоща (VegetableID), количество заказанных товаров (Quantity), дату заказа (OrderDate) и общую стоимость заказа (TotalPrice). Поле ID является первичным ключом. Идентификатор клиента и идентификатор овоща являются обязательными полями и ссылаются соответственно на таблицы "Clients" и "Vegetables". Поле количества (Quantity) указывает количество заказанных товаров, а дата заказа (OrderDate) указывает дату, когда был сделан заказ. Общая стоимость заказа (TotalPrice) указана в виде числа с плавающей запятой и рассчитывается на основе количества и цены овощей.

**Таблица 1 – Схема отношения** **Orders**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Длина** | **Примечания** |
| Первичный ключ | ID | Целое число | - | первичный ключ |
| Идентификатор клиента | ClientID | Целое число | - | обязательное поле |
| Идентификатор овоща | VegetableID | Целое число | - | обязательное поле |
| Общая стоимость заказа | TotalPrice | Число с плавающей запятой | 10, 2 | обязательное поле |
| Количество | Quantity | Целое число | - | обязательное поле |
| Дата заказа | OrderDate | Дата | - | обязательное поле |

Таблица 2 – Схема отношения Clients

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Длина** | **Примечания** |
| Первичный ключ | ID | Целое число | - | первичный ключ |
| Имя клиента | Name | Строка | 255 | обязательное поле |
| Скидка | Discount | Число с плавающей запятой | 4, 2 | обязательное поле |

Таблица 3 – Схема отношения Suppliers

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Длина** | **Примечания** |
| Первичный ключ | ID | Целое число | - | первичный ключ |
| Название поставщика | Name | Строка | 255 | обязательное поле |

Таблица 4 – Схема отношения Vegetables

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Длина** | **Примечания** |
| Первичный ключ | ID | Целое число | - | первичный ключ |
| Наименование овоща | Name | Строка | 255 | обязательное поле |
| Идентификатор поставщика | SupplierID | Целое число | - | Обязательное поле |
| Срок годности | ExpiryDate | Дата | - | Обязательное поле |
| Цена | Price | Число с плавающей запятой | 10, 2 | Обязательное поле |
| Состояние | Status | Строка | 50 | Обязательное поле |
| Количество товара на складе | Stock | Целое число | - | По умолчанию 0 |

Нормализация полученных отношений

**1НФ.** Для приведения таблиц к 1НФ требуется составить таблицы (один атрибут – один столбец) и разбить сложные атрибуты на простые, а многозначные атрибуты вынести в отдельные отношения.  
В наших таблицах все атрибуты простые.

**2НФ.** Все отношения находятся во 2НФ, так как для каждого из них введен уникальный ключ.

**3НФ**. Все неключевые атрибуты зависят только от первичного ключа и не зависят друг от друга. Поэтому спроектированная **база данных находится в третьей нормальной форме**.

Созданную схему Вы можете показана на рисунке 2.

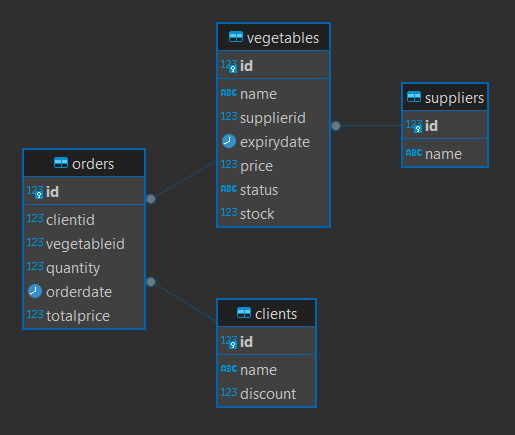


Рисунок 2 – ER-диаграмма в DBeaver

## Физическое проектирование БД

### Разработка скриптов на создание базы данных и таблиц

БД «Овощной магазин» проектируется в среде Система управления базами данных PostgresSQL.

*Создание базы данных* «Овощной магазин»:

Создание таблицы Clients:

-- Создание таблицы "Clients"

CREATE TABLE Clients (

ID SERIAL PRIMARY KEY,

Name VARCHAR(255) NOT NULL,

Discount DECIMAL(4,2) DEFAULT 0

);

Создание таблицы Suppliers:

-- Создание таблицы "Suppliers"

CREATE TABLE Suppliers (

ID SERIAL PRIMARY KEY,

Name VARCHAR(255) NOT NULL

);

Создание таблицы Vegetables:

CREATE TABLE Vegetables (

ID SERIAL PRIMARY KEY,

Name VARCHAR(255) NOT NULL,

SupplierID INT NOT NULL,

ExpiryDate DATE NOT NULL,

Price DECIMAL(10,2) NOT NULL,

Status VARCHAR(50) NOT NULL,

Stock INT NOT NULL DEFAULT 0, -- Количество товара на складе

FOREIGN KEY (SupplierID) REFERENCES Suppliers(ID)

);

Создание таблицы Orders:

-- Создание таблицы "Orders"

CREATE TABLE Orders (

ID SERIAL PRIMARY KEY,

ClientID INT NOT NULL,

VegetableID INT NOT NULL,

Quantity INT NOT NULL,

OrderDate DATE NOT NULL,

TotalPrice DECIMAL(10,2) NOT NULL,

FOREIGN KEY (ClientID) REFERENCES Clients(ID),

FOREIGN KEY (VegetableID) REFERENCES Vegetables(ID)

);

### 3.2 Разработка ограничений

-- Таблица "Orders"

ALTER TABLE Orders

ALTER COLUMN ClientID SET NOT NULL,

ALTER COLUMN VegetableID SET NOT NULL,

ALTER COLUMN Quantity SET NOT NULL,

ALTER COLUMN OrderDate SET NOT NULL,

ALTER COLUMN TotalPrice SET NOT NULL

ADD CONSTRAINT chk\_quantity CHECK (Quantity > 0),

ADD CONSTRAINT chk\_total\_price CHECK (TotalPrice >= 0),

ADD CONSTRAINT fk\_client FOREIGN KEY (ClientID) REFERENCES Clients(ID) ON DELETE CASCADE,

ADD CONSTRAINT fk\_vegetable FOREIGN KEY (VegetableID) REFERENCES Vegetables(ID) ON DELETE CASCADE;

-- Таблица "Vegetables"

ALTER TABLE Vegetables

ALTER COLUMN Name SET NOT NULL,

ALTER COLUMN SupplierID SET NOT NULL,

ALTER COLUMN ExpiryDate SET NOT NULL,

ALTER COLUMN Price SET NOT NULL,

ALTER COLUMN Status SET NOT NULL,

ALTER COLUMN Stock SET NOT NULL

ADD CONSTRAINT chk\_expiry\_date CHECK (ExpiryDate > CURRENT\_DATE),

ADD CONSTRAINT chk\_price CHECK (Price >= 0),

ADD CONSTRAINT chk\_status CHECK (Status IN ('хороший', 'испорченный')),

ADD CONSTRAINT chk\_stock CHECK (Stock >= 0),

ADD CONSTRAINT fk\_supplier FOREIGN KEY (SupplierID) REFERENCES Suppliers(ID) ON DELETE CASCADE;

-- Таблица "Suppliers"

ALTER TABLE Suppliers

ADD CONSTRAINT uq\_name UNIQUE (Name)

ALTER COLUMN Name SET NOT NULL;

-- Таблица "Clients"

ALTER TABLE Clients

ALTER COLUMN Name SET NOT NULL,

ALTER COLUMN Discount SET NOT NULL;

ADD CONSTRAINT chk\_discount CHECK (Discount >= 0 AND Discount <= 100);

### 3.3 Разработка скриптов на добавление данных в таблицы

-- Вставка данных в таблицу "Clients"

INSERT INTO Clients (Name, Discount) VALUES ('Клиент 1', 5.0);

INSERT INTO Clients (Name, Discount) VALUES ('Клиент 2', 10.0);

-- Вставка данных в таблицу "Suppliers"

INSERT INTO Suppliers (Name) VALUES ('Поставщик 1');

INSERT INTO Suppliers (Name) VALUES ('Поставщик 2');

-- Вставка данных в таблицу "Vegetables"

INSERT INTO Vegetables (Name, SupplierID, ExpiryDate, Price, Status, Stock) VALUES ('Помидор', 1, '2024-06-15', 50.00, 'хороший', 0);

INSERT INTO Vegetables (Name, SupplierID, ExpiryDate, Price, Status, Stock) VALUES ('Огурец', 2, '2024-06-20', 30.00, 'хороший', 0);

-- Вставка данных в таблицу "Orders"

INSERT INTO Orders (ClientID, VegetableID, Quantity, OrderDate, TotalPrice) VALUES (1, 1, 10, '2024-05-01', 500.00);

INSERT INTO Orders (ClientID, VegetableID, Quantity, OrderDate, TotalPrice) VALUES (2, 2, 20, '2024-05-02', 600.00);

### 3.4 Разработка необходимых запросов:

1. Показать ассортимент товара на выбранный день:

SELECT V.Name, V.ExpiryDate, V.Price, V.Status

FROM Vegetables V

WHERE V.ExpiryDate >= '2024-06-03';

Результат:

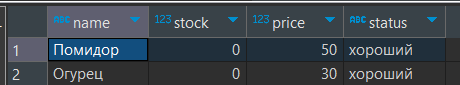


Рисунок 3 – Результат 1 запроса

1. Показывать приход-расход выбранного товара:

-- Приход

SELECT Name, Stock

FROM Vegetables

WHERE ID = [ID выбранного товара];

-- Расход

SELECT Name, SUM(Quantity) AS TotalSold

FROM Orders

JOIN Vegetables ON Orders.VegetableID = Vegetables.ID

WHERE Vegetables.ID = [ID выбранного товара]

GROUP BY Name;

1. Результат:

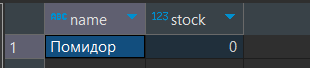


Рисунок 4 – Результат 2 запроса

1. Показывать прибыль магазина по указанному товару за отчетный период времени:

--Показывать прибыль магазина по указанному товару за отчетный период времени

SELECT Name, SUM(TotalPrice) AS TotalProfit

FROM Orders

JOIN Vegetables ON Orders.VegetableID = Vegetables.ID

WHERE Vegetables.ID = 2

AND OrderDate BETWEEN "2024-05-01" AND "2024-05-02"

GROUP BY Name;

1. Показывать список клиентов, имеющих скидку:

--Показывать в % отношении количество хорошего и испорченного товара для выбранного товара

SELECT Name,

(SUM(CASE WHEN Status = 'хороший' THEN Stock ELSE 0 END) \* 100 / SUM(Stock)) AS GoodPercentage,

(SUM(CASE WHEN Status = 'испорченный' THEN Stock ELSE 0 END) \* 100 / SUM(Stock)) AS BadPercentage

FROM Vegetables

WHERE ID = 1

GROUP BY Name;

### Разработка представлений (view), процедур, функций и триггеров:

Создание представления view:

--Просмотр доступных овощей с их состоянием и сроком годности

CREATE VIEW AvailableVegetables AS

SELECT

v.ID,

v.Name,

s.Name AS SupplierName,

v.ExpiryDate,

v.Price,

v.Status,

v.Stock

FROM

Vegetables v

JOIN

Suppliers s ON v.SupplierID = s.ID

WHERE

v.Stock > 0 AND v.ExpiryDate > CURRENT\_DATE;

Пример процедуры:

-- --Добавление нового заказа и обновление склада

CREATE PROCEDURE AddNewOrder(

IN p\_ClientID INT,

IN p\_VegetableID INT,

IN p\_Quantity INT,

IN p\_OrderDate DATE

)

LANGUAGE plpgsql

AS $$

DECLARE

TotalPrice DECIMAL(10,2);

BEGIN

-- Вычисление общей стоимости заказа

SELECT CalculateTotalPriceWithDiscount(p\_ClientID, p\_VegetableID, p\_Quantity) INTO TotalPrice;

-- Вставка нового заказа

INSERT INTO Orders (ClientID, VegetableID, Quantity, OrderDate, TotalPrice)

VALUES (p\_ClientID, p\_VegetableID, p\_Quantity, p\_OrderDate, TotalPrice);

-- Обновление количества товара на складе

UPDATE Vegetables

SET Stock = Stock - p\_Quantity

WHERE ID = p\_VegetableID;

COMMIT;

END;

$$;

Пример функции:

--Вычисление общей стоимости заказа с учетом скидки клиента

CREATE FUNCTION CalculateTotalPriceWithDiscount(ClientID INT, VegetableID INT, Quantity INT)

RETURNS DECIMAL(10,2) AS $$

DECLARE

Price DECIMAL(10,2);

Discount DECIMAL(4,2);

TotalPrice DECIMAL(10,2);

BEGIN

SELECT Price INTO Price FROM Vegetables WHERE ID = VegetableID;

SELECT Discount INTO Discount FROM Clients WHERE ID = ClientID;

TotalPrice := Quantity \* Price \* (1 - Discount / 100);

RETURN TotalPrice;

END;

$$ LANGUAGE plpgsql;

Создание триггера:

CREATE OR REPLACE FUNCTION UpdateVegetableStatus()

RETURNS TRIGGER AS $$

BEGIN

-- Обновление состояния овоща в зависимости от количества на складе

IF NEW.Quantity <= 0 THEN

UPDATE Vegetables

SET Status = 'испорченный'

WHERE ID = NEW.VegetableID;

ELSE

UPDATE Vegetables

SET Status = 'хороший'

WHERE ID = NEW.VegetableID;

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER UpdateVegetableStatusTrigger

AFTER INSERT ON Orders

FOR EACH ROW

EXECUTE FUNCTION UpdateVegetableStatus();

# Список литературы

* + - 1. Васильева М. А, Филипченко К. М. Фильтрация набора данных. Рекомендации по выполнению работы и перечень типовых заданий: Учебно-методическое пособие. Издание третье, исправленное и дополненное. Москва–2024, 133 с.
      2. Васильева М. А., Клепцов М. Я. Группировка и обобщение данных. Рекомендации по выполнению работы и перечень типовых заданий. Учебно-методическое пособие. 2-е издание. 2024.– 59с.
      3. Васильева М. А., Балакина Е. П., Васильева М. В. Соединение данных из множества таблиц. Рекомендации по выполнению работы и перечень типовых заданий. Учебно–методическое пособие. Издание второе, исправленное. – М.: РУТ (МИИТ).2024. – 72с.
      4. Балакина Е.П., Васильева М.А., Филипченко К.М. Информационное обеспечение систем управления. Методические указания к курсовому проектированию. Учебно-методическое пособие. Издание второе, исправленное и дополненное, 2023.102–с.