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  
«Магазин гель-лаков»**

Выполнил: ст. гр. ТКИ-441

Станчук П. Н.

Проверил: доц., к.т.н.

Васильева М. А.

Москва 2023

Оглавление

[ЦЕЛЬ КУРСОВОГО ПРОЕКТА \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_5](#_Toc136619567)

[ЗАДАНИЕ НА КУРСОВОВОЙ ПРОЕКТ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_5](#_Toc136619568)

[ПРОЕКТИРОВАНИЕ РЕЛЯЦИОННОЙ БАЗЫ ДАННЫХ\_\_\_\_\_\_ 6](#_Toc136619569)

[1. Инфологическое проектирование 6](#_Toc136619570)

[1.1 Анализ предметной области 6](#_Toc136619571)

[1.2 Анализ информационных задач и круга пользователей системы 6](#_Toc136619572)

[2. Логическое проектирование реляционной БД 7](#_Toc136619573)

[2.1 Составление реляционных отношений 7](#_Toc136619574)

[3. Физическое проектирование БД 10](#_Toc136619575)

[3.1 Разработка скриптов на создание базы данных и таблиц 10](#_Toc136619576)

[3.2 Разработка скриптов на добавление данных в таблицы 11](#_Toc136619577)

[3.3 Разработка необходимых запросов: 12](#_Toc136619578)

[3.4 Разработка необходимых запросов, представлений (view), процедур, функций и триггеров: 13](#_Toc136619579)

[Список литературы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_15](#_Toc136619580)

# 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 «Магазин гель-лаков». Проектирование РБД проводится с помощью метода «Сущность-связь». Проверка построенной модели РБД осуществляется с помощью метода нормализации отношений.

**Описание предметной области.**

БД создается для учёта продажи гель-лаков в магазинах от разных поставщиков.

**Готовые запросы:**

• Выдавать список клиентов, купивших товар в конкретном магазине.

• Выдавать список поставщиков, поставляющих товар в данный магазин.

• Выдавать список клиентов из данной страны.

• Выдавать количество поставщиков для данного склада.



# ПРОЕКТИРОВАНИЕ РЕЛЯЦИОННОЙ БАЗЫ ДАННЫХ

## Инфологическое проектирование

### Анализ предметной области

База данных создаётся для информационного обслуживания продаж лаков в магазинах.

Выделим базовые сущности этой предметной области:

* Клиент
* Поставщик
* Склад
* Накладная
* Магазин

ER-диаграмма БД, приведена на (Рисунке 1). На данной ER-диаграмме отсутствует связь многие -ко– многим, поэтому дополнительная промежуточная сущностей не требуется.

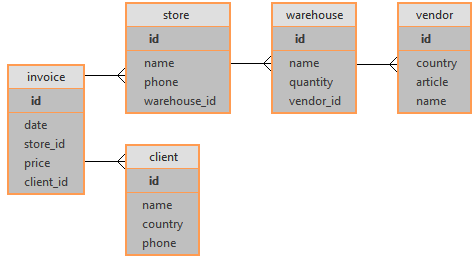


Рисунок 1 – ER-диаграмма

### 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:**

* Выдавать список клиентов, купивших товар в конкретном магазине.
* Выдавать список поставщиков, поставляющих товар в данный магазин.
* Выдавать список клиентов из данной страны.
* Выдавать количество поставщиков для данного склада.

## 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 "store" - содержит информацию о магазине.
2. Таблица "warehouse" – содержит информацию о складе.
3. Таблица "client" - содержит список клиентов.
4. Таблица "vendor" - содержит список поставщиков.
5. Таблица "invoice" - содержит список накладных.

**Таблица 1 – Схема отношения** **Магазин** (store)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Длина** | **Примечания** |
| Идентификатор | id | Целое число | - | Первичный ключ, автоинкремент |
| Имя | name | Текст | - | Обязательное поле |
| Телефон | phone | Текст | - | Обязательное поле |
| Идентификатор склада | warehouse\_id | Целое число | - | Внешний ключ |

Таблица 2 – Схема отношения Поставщик (Vendor)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Длина** | **Примечания** |
| Идентификатор | id | Целое число | - | Первичный ключ, автоинкремент |
| Страна | country | Текст | - | Обязательное поле |
| Артикул | article | Целое число | - | Обязательное поле |
| Название | name | Текст | - | Обязательное поле |

Таблица 3 – Схема отношения Склад (Warehouse)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Длина** | **Примечания** |
| Идентификатор | id | Целое число | - | Первичный ключ, автоинкремент |
| Название | name | Текст | - | Обязательное поле |
| Количество товаров | quantity | Целое число | - | Обязательное поле |
| Идентификатор поставщика | id | Целое число | - | Внешний ключ |

Таблица 4 – Схема отношения Клиент (Client)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Длина** | **Примечания** |
| Идентификатор | id | Целое число | - | Первичный ключ, автоинкремент |
| ФИО | name | Текст | - | Обязательное поле |
| Страна | country | Текст | - | Обязательное поле |
| Телефон | phone | Текст | - | Обязательное поле |

Таблица 5 – Схема отношения Накладная (Invoice)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Длина** | **Примечания** |
| Идентификатор | id | Целое число | - | Первичный ключ, автоинкремент |
| Дата продажи | date | Дата | - | Обязательно поле |
| Цена | price | Денежный | - | Обязательное поле |
| Идентификатор магазина | store\_id | Целое число | - | Внешний ключ |
| Идентификатор клиента | client\_id | Целое число | - | Внешний ключ |

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 наблюдать на рисунке 2.

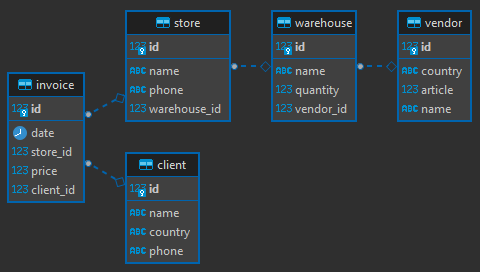


Рисунок 2 – ER-диаграмма в DBeaver

## Физическое проектирование БД

### Разработка скриптов на создание базы данных и таблиц

БД «Магазин гель-лаков» проектируется в среде «Система управления базами данных PostgresSQL».

**-- Создание Vendor**

**CREATE TABLE Vendor (**

**id SERIAL PRIMARY KEY,**

**country TEXT,**

**article INT,**

**name TEXT**

**);**

**-- Создание Warehouse**

**CREATE TABLE Warehouse (**

**id SERIAL PRIMARY KEY,**

**name TEXT,**

**quantity INT,**

**vendor\_id INT,**

**FOREIGN KEY (vendor\_id) REFERENCES Vendor(id)**

**);**

**-- Создание Store**

**CREATE TABLE Store (**

**id SERIAL PRIMARY KEY,**

**name TEXT,**

**phone TEXT,**

**warehouse\_id INT,**

**FOREIGN KEY (warehouse\_id) REFERENCES Warehouse(id)**

**);**

**-- Создание Client**

**CREATE TABLE Client (**

**id SERIAL PRIMARY KEY,**

**name TEXT,**

**country TEXT,**

**phone TEXT**

**);**

**-- Создание Invoice**

**CREATE TABLE Invoice (**

**id SERIAL PRIMARY KEY,**

**date DATE,**

**store\_id INT,**

**price MONEY,**

**client\_id INT,**

**FOREIGN KEY (store\_id) REFERENCES Store(id),**

**FOREIGN KEY (client\_id) REFERENCES Client(id)**

**);**

### 3.2 Разработка скриптов на добавление данных в таблицы

-- Заполнение Vendor

INSERT INTO Vendor (country, article, name) VALUES

('Russia', 12345, 'Поставщик 1'),

('Germany', 54321, 'Поставщик 2');

-- Заполнение Warehouse

INSERT INTO Warehouse (name, quantity, vendor\_id) VALUES

('Склад 1', 100, 1),

('Склад 2', 200, 2);

-- Заполнение Store

INSERT INTO Store (name, phone, warehouse\_id) VALUES

('Магазин 1', '123-456-7890', 1),

('Магазин 2', '987-654-3210', 2),

('Магазин 3', '555-555-5555', 1);

-- Заполнение Client

INSERT INTO Client (name, country, phone) VALUES

('Petr Ivanov', 'Russia', '111-111-1111'),

('Thomas Becker', 'Germany', '222-222-2222'),

('Josh Smith', 'USA', '333-333-3333');

-- Заполнение Invoice

INSERT INTO Invoice (date, store\_id, price, client\_id) VALUES

('2023-05-20', 1, 50.00, 1),

('2023-05-21', 2, 75.00, 2),

('2023-05-22', 1, 30.00, 3);

### Разработка необходимых запросов:

* Выдавать список клиентов, купивших товар в конкретном магазине.

**SELECT client\_id**

**FROM invoice**

**WHERE store\_id = 1**

* Выдавать список поставщиков, поставляющих товар на данный склад.

**SELECT vendor\_id**

**FROM warehouse**

**WHERE id = 1**

* Выдавать список клиентов из данной страны.

**SELECT id**

**FROM client**

**WHERE country = 'Russia'**

* Выдавать количество поставщиков для данного склада.

**SELECT COUNT(vendor\_id) AS vendor\_count**

**FROM warehouse**

**WHERE id = 1;**

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

Создание представления view:

Вывести всех поставщиков из России:

**CREATE VIEW Vendor\_Russia AS**

**SELECT \***

**FROM vendor**

**WHERE country = 'Russia'**

**SELECT \***

**FROM Vendor\_Russia**



Рисунок 3 – Результат представления

Пример процедуры:

Процедура вывода клиентов из заданной страны:

**CREATE OR REPLACE PROCEDURE GetClientsByCountry(IN country\_param TEXT)**

**AS**

**$$**

**BEGIN**

**PERFORM \***

**FROM Client**

**WHERE country = country\_param;**

**END;**

**$$**

LANGUAGE plpgsql;

**CALL GetClientsByCountry('Russia');**

Пример функции для подсчёта количества поставщиков из данной страны:

**CREATE OR REPLACE FUNCTION CountVendorsByCountry(country\_param TEXT)**

**RETURNS INTEGER AS**

**$$**

**DECLARE**

**supplier\_count INTEGER;**

**BEGIN**

**SELECT COUNT(\*) INTO supplier\_count**

**FROM Vendor**

**WHERE country = country\_param;**

**RETURN supplier\_count;**

**END;**

**$$**

**LANGUAGE plpgsql;**

**SELECT CountVendorsByCountry('Russia');**



Рисунок 4 – Результат функции

Пример триггера на удаление накладной:

**CREATE OR REPLACE FUNCTION DeleteInvoiceRow()**

**RETURNS TRIGGER AS**

**$$**

**BEGIN**

**RAISE NOTICE 'Строка с id % была удалена из таблицы Invoice.', OLD.id;**

**RETURN OLD;**

**END;**

**$$**

**LANGUAGE plpgsql;**

**-- Создание триггера**

**CREATE TRIGGER DeleteInvoiceTrigger**

**AFTER DELETE ON Invoice**

**FOR EACH ROW**

**EXECUTE FUNCTION DeleteInvoiceRow();**

**DELETE FROM invoice WHERE id=1**



Рисунок 5 – Результат триггера

# Список литературы

* + - 1. Васильева М.А., Хобта Д.О., Фильтрация набора данных. Рекомендации по выполнению работы и перечень типовых заданий: Учебно-методическое пособие. Издание второе, исправленное и дополненное–М.:РУТ(МИИТ). 2023.–105с.
      2. Васильева М.А., Меркулов Д.А. Группировка и обобщение данных. Рекомендации по выполнению работы и перечень типовых заданий. Учебно-методическое пособие. М.:РУТ(МИИТ), 2023. 46–с.
      3. Васильева М.А., Ракинцев Н.А. Соединение данных из множества таблиц. Рекомендации по выполнению работы и перечень типовых заданий. Учебно-методическое пособие. М.:РУТ(МИИТ), 2023. 63–с.
      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.