Министерство науки и высшего образования РФ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт
Программная инженерия
кафедра

ОТЧЕТ О ПРАКТИЧЕСКОЙ РАБОТЕ №10

Проектирование базы данных и ее реализация в среде СУБД PostgreSQL
тема

Преподаватель _____ Вожжов А.Д.
подпись, дата — инициалы, фамилия

Студент КИ23-17/1б 032318988 _____ Александров Е.А.
номер зачетной книжки — подпись, дата — инициалы, фамилия

Красноярск 2025

1 Цель

Изучить основы проектирования базы данных и ее реализации в среде СУБД PostgreSQL. Выполнить указанные в файле задания.

2 Ход работы

Я выбрал предметную область Интернет-магазин. Предметная область "Интернет-магазин" описывает систему для продажи товаров онлайн. Основные сущности в этой системе включают: покупателей (которые совершают заказы), товары (которые продаются), категории товаров (для организации каталога), заказы (представляющие покупки, сделанные покупателями) и позиции заказов (детализация, какие именно товары и в каком количестве входят в каждый заказ). Система должна отслеживать информацию о покупателях, доступные товары (включая их запасы), структуру заказов и их состав.

Логическая и концептуальная модели представлены на рисунках с 1 по 2. Физическая модель представлена на рисунках с 3 по 8.

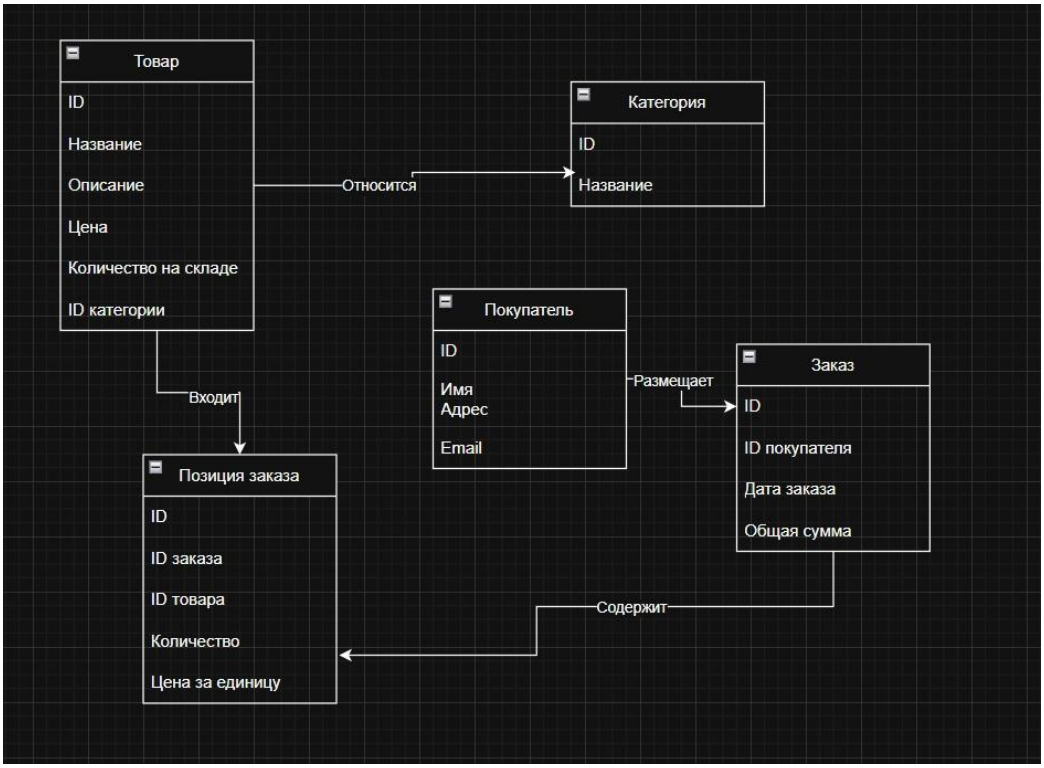

Рисунок 1 – Концептуальная модель

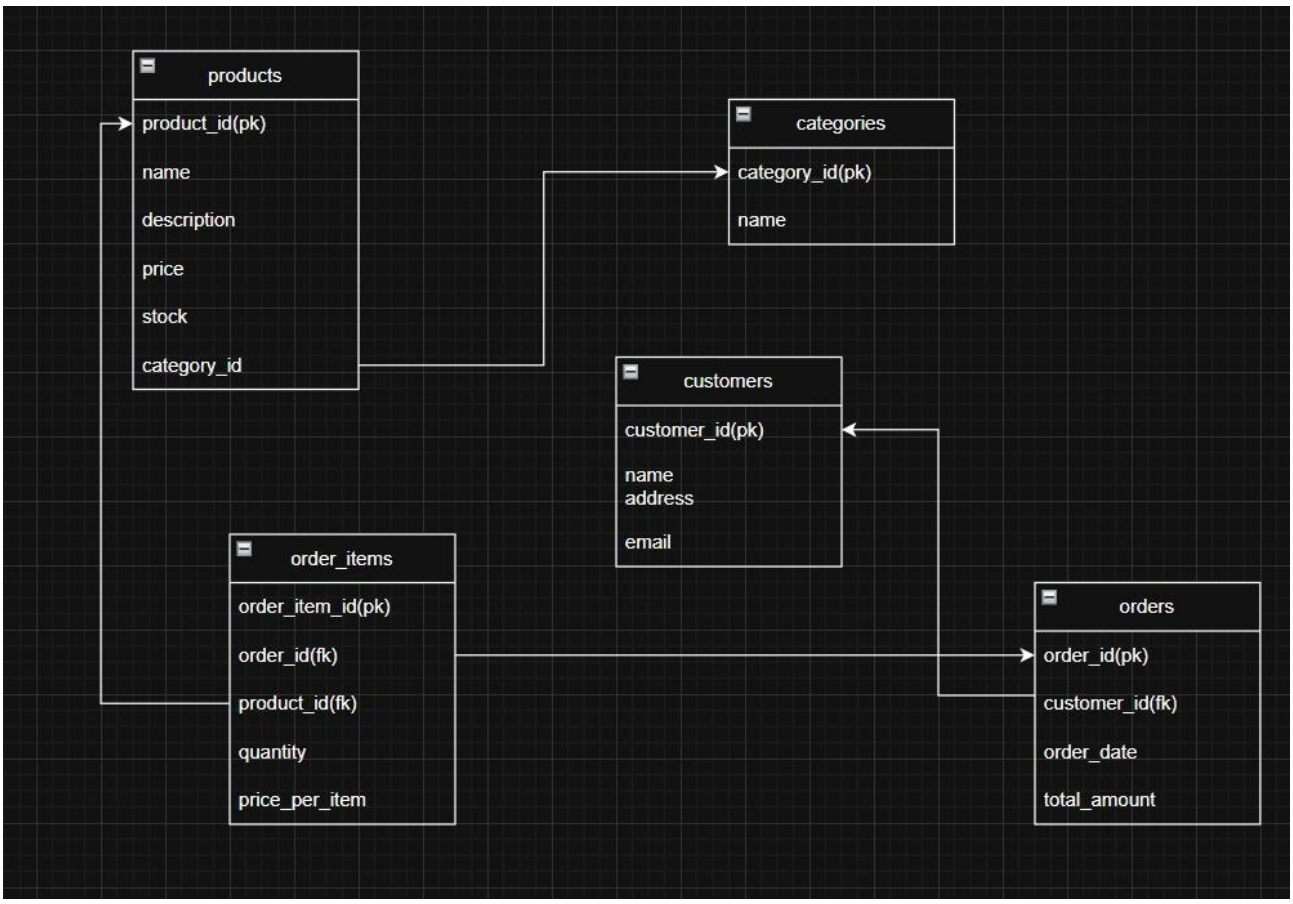

Рисунок 2 – Логическая модель

```
demo=# DROP TABLE IF EXISTS order_items CASCADE;
ЗАМЕЧАНИЕ:  таблица "order_items" не существует, пропускается
DROP TABLE
demo=# DROP TABLE IF EXISTS orders CASCADE;
ЗАМЕЧАНИЕ:  таблица "orders" не существует, пропускается
DROP TABLE
demo=# DROP TABLE IF EXISTS products CASCADE;
ЗАМЕЧАНИЕ:  таблица "products" не существует, пропускается
DROP TABLE
demo=# DROP TABLE IF EXISTS categories CASCADE;
ЗАМЕЧАНИЕ:  таблица "categories" не существует, пропускается
DROP TABLE
demo=# DROP TABLE IF EXISTS customers CASCADE;
ЗАМЕЧАНИЕ:  таблица "customers" не существует, пропускается
DROP TABLE
demo=# CREATE TABLE customers (
demo(#     customer_id SERIAL PRIMARY KEY,
demo(#     name VARCHAR(100) NOT NULL,
demo(#     email VARCHAR(100) UNIQUE NOT NULL,
demo(#     address TEXT
demo(# );
CREATE TABLE
demo=# CREATE TABLE categories (
demo(#     category_id SERIAL PRIMARY KEY,
demo(#     name VARCHAR(100) UNIQUE NOT NULL
demo(# );
CREATE TABLE
```

Рисунок 3 – Физическая модель

```
demo=# CREATE TABLE products (
demo(#     product_id SERIAL PRIMARY KEY,
demo(#     name VARCHAR(255) NOT NULL,
demo(#     description TEXT,
demo(#     price DECIMAL(10, 2) NOT NULL CHECK (price >= 0),
demo(#     stock INTEGER NOT NULL CHECK (stock >= 0),
demo(#     category_id INTEGER REFERENCES categories(category_id)
demo(# );
CREATE TABLE
demo=# CREATE TABLE orders (
demo(#     order_id SERIAL PRIMARY KEY,
demo(#     customer_id INTEGER REFERENCES customers(customer_id) NOT NULL,
demo(#     order_date TIMESTAMP WITH TIME ZONE NOT NULL DEFAULT CURRENT_TIMESTAMP,
demo(#     total_amount DECIMAL(10, 2) NOT NULL DEFAULT 0.00 CHECK (total_amount >= 0)
demo(# );
CREATE TABLE
demo=# CREATE TABLE order_items (
demo(#     order_item_id SERIAL PRIMARY KEY,
demo(#     order_id INTEGER REFERENCES orders(order_id) ON DELETE CASCADE NOT NULL,
demo(#     product_id INTEGER REFERENCES products(product_id) NOT NULL,
demo(#     quantity INTEGER NOT NULL CHECK (quantity > 0),
demo(#     price_per_item DECIMAL(10, 2) NOT NULL CHECK (price_per_item >= 0),
demo(#     UNIQUE (order_id, product_id) -- Опционально, но полезно
demo(# );
CREATE TABLE
```

Рисунок 4 – Физическая модель

```
demo=# CREATE OR REPLACE FUNCTION update_order_total_and_stock()
demo-# RETURNS TRIGGER AS $$
demo$# BEGIN
demo$#     UPDATE orders
demo$#     SET total_amount = (
demo$#         SELECT COALESCE(SUM(quantity * price_per_item), 0)
demo$#         FROM order_items
demo$#         WHERE order_id = NEW.order_id OR order_id = OLD.order_id
demo$#     )
demo$#     WHERE order_id = NEW.order_id OR order_id = OLD.order_id;
demo$#     IF TG_OP = 'INSERT' THEN
demo$#         UPDATE products
demo$#         SET stock = stock - NEW.quantity
demo$#         WHERE product_id = NEW.product_id;
demo$#     ELSIF TG_OP = 'UPDATE' THEN
demo$#         UPDATE products
demo$#         SET stock = stock + OLD.quantity - NEW.quantity
demo$#         WHERE product_id = NEW.product_id;
demo$#     ELSIF TG_OP = 'DELETE' THEN
demo$#         UPDATE products
demo$#         SET stock = stock + OLD.quantity
demo$#         WHERE product_id = OLD.product_id;
demo$#     END IF;
demo$#     IF TG_OP = 'DELETE' THEN
demo$#         RETURN OLD;
demo$#     ELSE
demo$#         RETURN NEW;
demo$#     END IF;
demo$# END;
demo$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
```

Рисунок 5 – Физическая модель

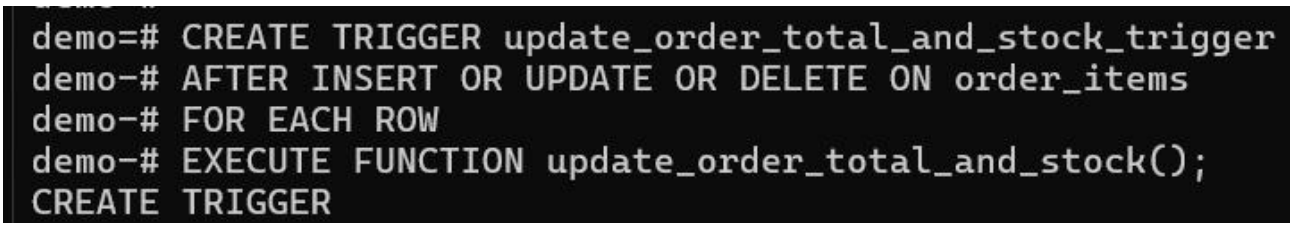

Рисунок 6 – Физическая модель

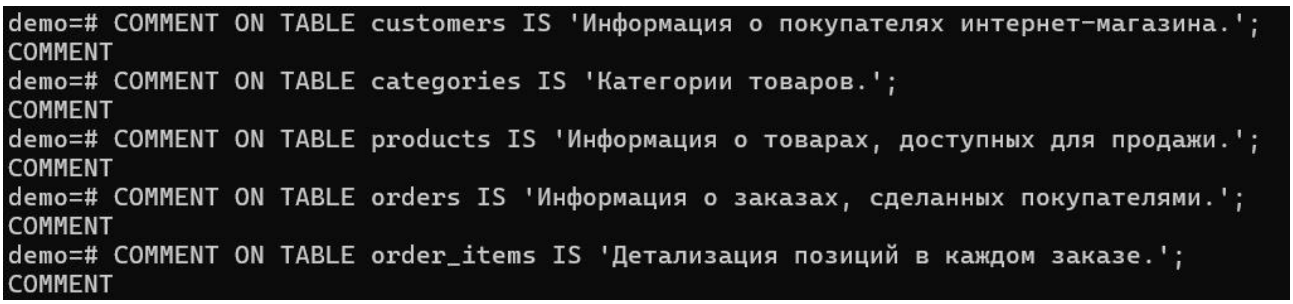

Рисунок 7 – Физическая модель

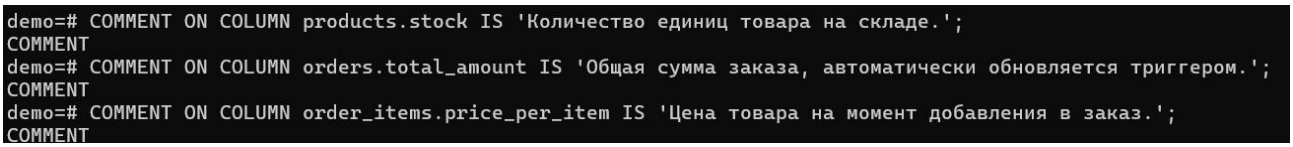

Рисунок 8 – Физическая модель

Номера нормальных форм для каждой таблицы:

customers: Таблица находится в 3НФ (Третья нормальная форма).

1НФ: Нет повторяющихся групп или многозначных атрибутов. Все значения атомарны.

2НФ: Первичный ключ (customer_id) является простым. Все неключевые атрибуты (name, email, address) полностью зависят от первичного ключа.

3НФ: Нет транзитивных зависимостей. Ни один неключевой атрибут не зависит от другого неключевого атрибута.

categories: Таблица находится в 3НФ.

1НФ: Выполнено.

2НФ: Первичный ключ (category_id) простой. name полностью зависит от category_id.

3НФ: Нет транзитивных зависимостей.

products: Таблица находится в 3НФ.

1НФ: Выполнено.

2НФ: Первичный ключ (product_id) простой. Все неключевые атрибуты (name, description, price, stock, category_id) полностью зависят от product_id. category_id здесь является атрибутом, описывающим товар.

3НФ: Нет транзитивных зависимостей. name, description, price, stock, category_id зависят только от product_id.

orders: Таблица находится в 3НФ.

1НФ: Выполнено.

2НФ: Первичный ключ (order_id) простой. Все неключевые атрибуты (customer_id, order_date, total_amount) полностью зависят от order_id.

3НФ: Нет транзитивных зависимостей. customer_id - внешний ключ, но он напрямую связан с заказом. order_date и total_amount также напрямую описывают заказ.

order_items: Таблица находится в 3НФ.

1НФ: Выполнено.

2НФ: Первичный ключ (order_item_id) простой. Все неключевые атрибуты (order_id, product_id, quantity, price_per_item) полностью зависят от order_item_id. (Если бы первичным ключом был композитный (order_id, product_id), то quantity и price_per_item также полностью зависели бы от обоих частей ключа).

3НФ: Нет транзитивных зависимостей. quantity и price_per_item описывают конкретную позицию в заказе (order_item_id), а не зависят от других неключевых атрибутов.

3 Заключение

В ходе практической работы были изучены основы проектирования базы данных и ее реализации в среде СУБД PostgreSQL, были выполнены указанные в файле задания.