**Итоговая аттестация по курсу Java.**

Модуль: Клиент-серверные приложения на Java (JDBC)  
Тема: PostgreSQL, JDBC, Maven, работа с БД из Java

**Цель работы**

Разработать и развернуть приложение для учёта заказов, используя PostgreSQL.  
Приложение должно включать:

1. Создание и миграцию схемы БД (DDL + тестовые данные).

2. Набор SQL-запросов на чтение/изменение/удаление.

3. Java-приложение на Maven, которое подключается к этой БД через JDBC и выполняет базовые операции CRUD, выводя результаты в консоль.

**Результат**

1. Репозиторий на GitHub (ветка + Pull Request).

2. Скриншоты:

o Установка/подключение PostgreSQL.  
o Выполнение миграций.  
o ER-диаграмма или схема БД.  
o Запуск Java-приложения и вывод CRUD-операций.  
o Результаты выполнения test-queries.sql в IDE.

**Отчет о проделанной работе:**

**Задача 1. Установка и настройка PostgreSQL**

Версию PostgreSQL 12 не удалось установить на ПК (постоянно выпадала ошибка).

Было принято решение установить версию PostgreSQL 13.

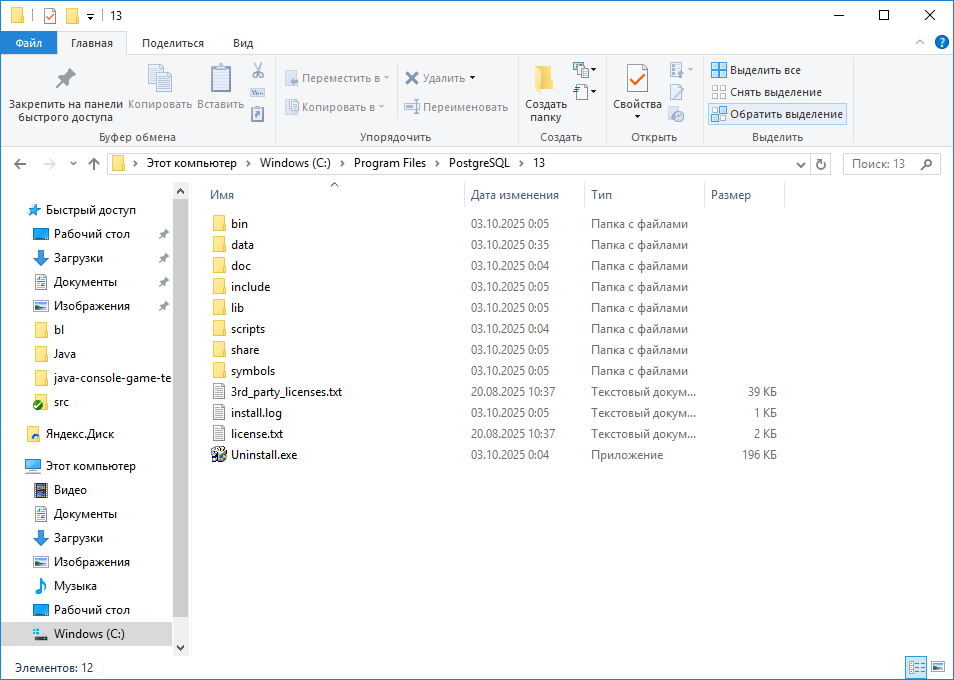


Рис. Установленное ПО PostgreSQL

В процессе настройки подключения к СУБД PostgreSQL возникла необходимость сбросить пароль пользователя postgres через изменение метода аутентификации в файле pg\_hba.conf

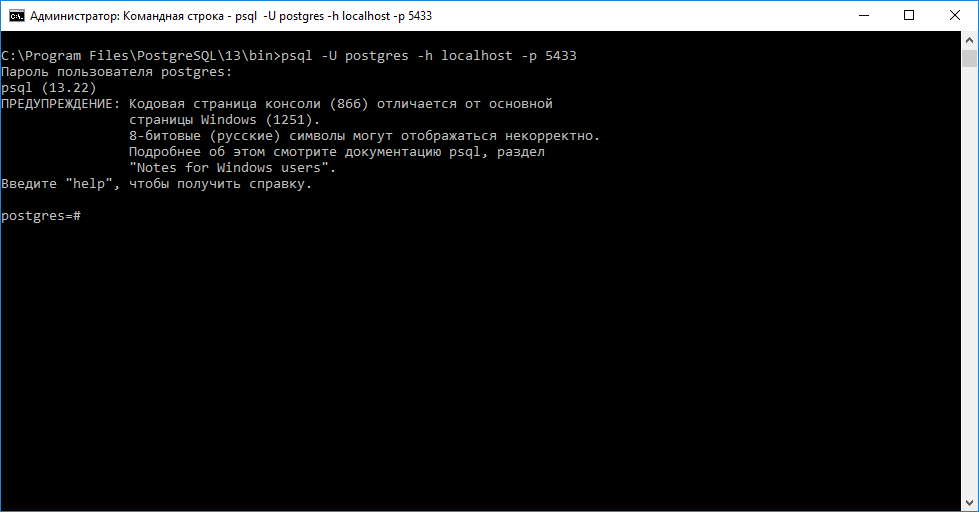


Рис. Установлено успешное соединение к PostgreSQL

Для подключения к серверу PostgreSQL была использована среда разработки Visual Studio Code с установленным расширением "PostgreSQL" by Chris Kolkman.

Параметры подключения:

* Хост: localhost
* Порт: 5433
* Пользователь: postgres
* База данных: order\_management
* Тип соединения: стандартное

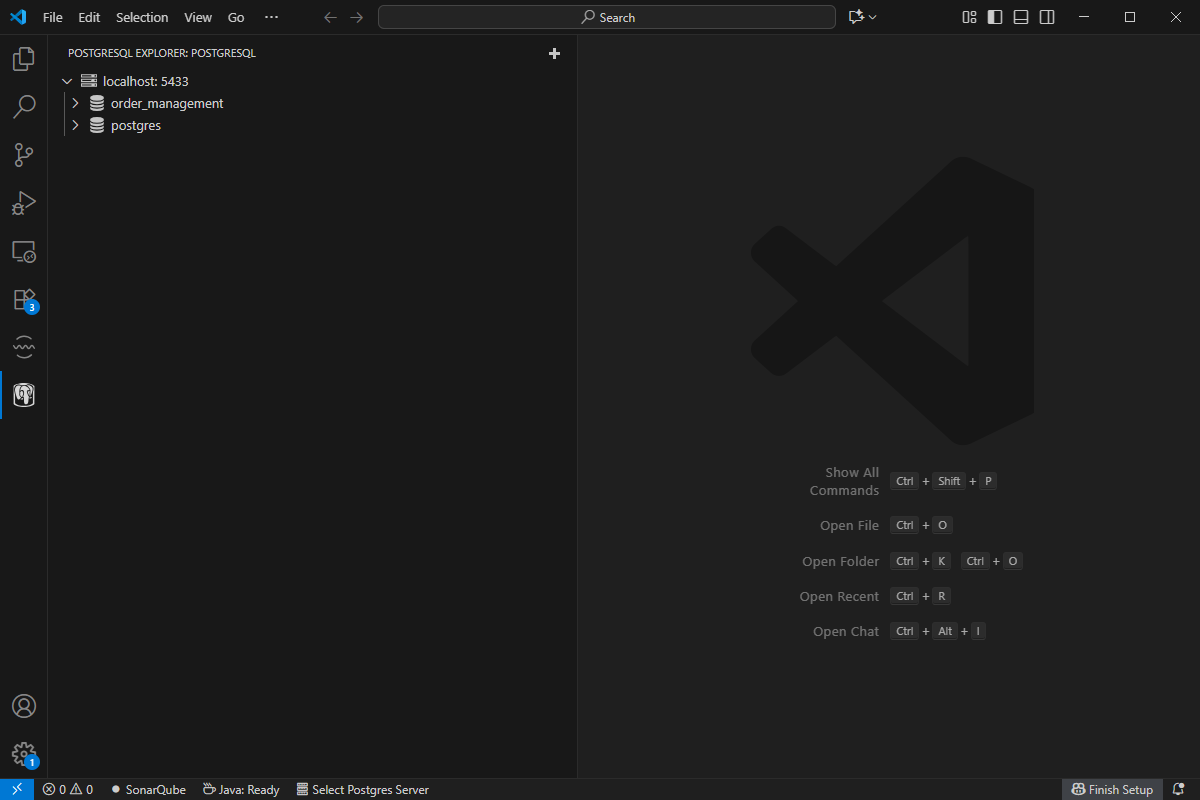
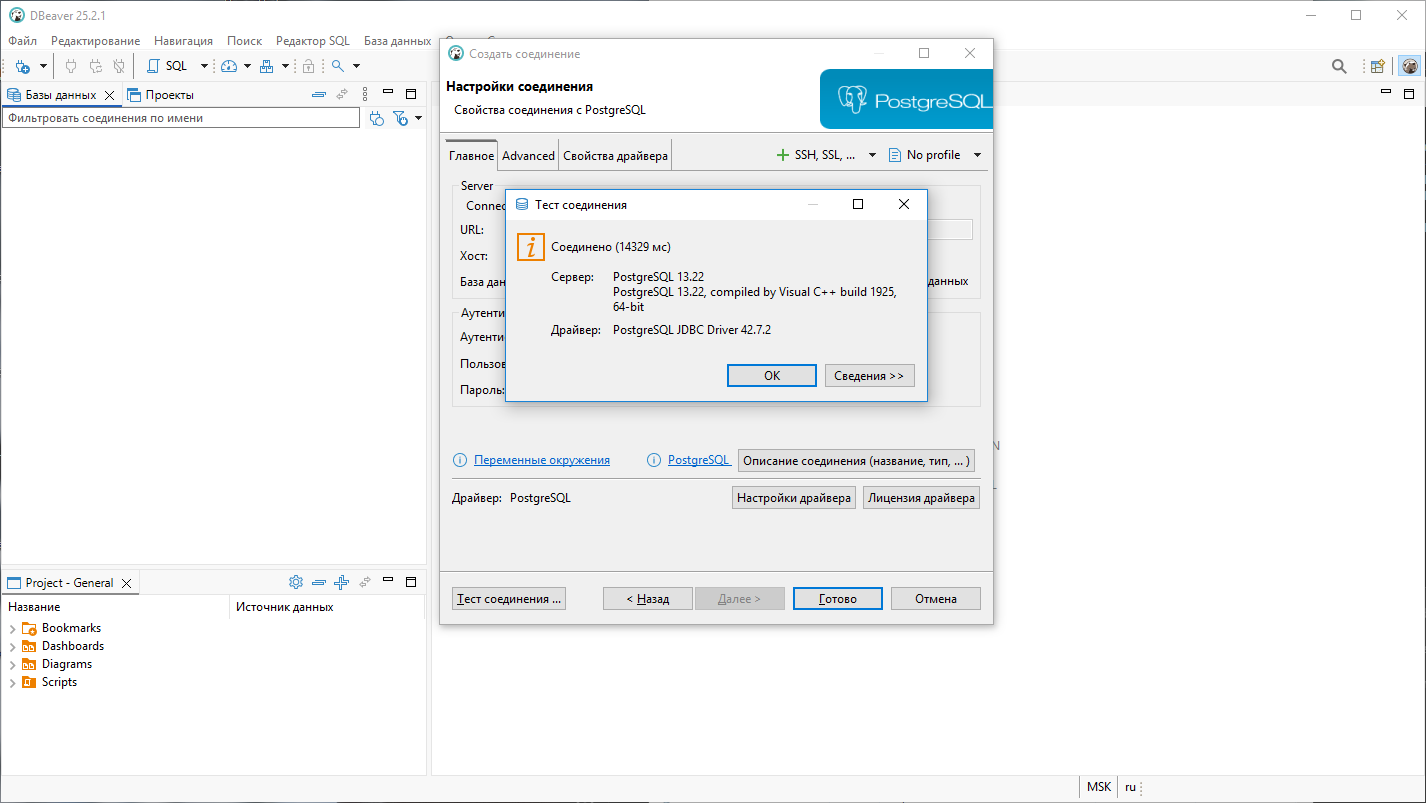


Рис. Проверка подключения в среде разработки

Дополнительно для более простой визуализации БД, был установлен Dbeaver и проведено тестовое соединение к созданной БД:



**Задача 2. Проектирование и создание схемы БД**

1. Файл schema.sql должен содержать:

Таблицы:

▪ product — id, описание, стоимость, количество, категория.

▪ customer — id, имя, фамилия, телефон, email.

▪ order — id, product\_id (FK), customer\_id (FK), дата заказа, количество, статус.

▪ order\_status — справочник статусов заказов (id, имя статуса).

Первичные и внешние ключи, ограничения NOT NULL, CHECK (например, цена ≥ 0, количество ≥ 0).

Индексы по внешним ключам и дате заказа.

Комментарии к таблицам и ключевым полям.

CREATE TABLE IF NOT EXISTS, чтобы скрипт можно было запускать  
повторно.  
  
2. Заполнение тестовыми данными — минимум 10 строк в каждой таблице.

\* Схема данных может быть выбрана любой на усмотрение студента. Можно проявить фантазию и создать Базу Данных на ваш лад.

**Файл schema.sql:**

-- product — id, описание, стоимость, количество, категория

CREATE TABLE IF NOT EXISTS product (

id SERIAL PRIMARY KEY,

description TEXT NOT NULL,

price DECIMAL(10,2) NOT NULL CHECK (price >= 0),

quantity INTEGER NOT NULL CHECK (quantity >= 0),

category VARCHAR(100) NOT NULL

);

-- Комментарии к таблицам и ключевым полям

COMMENT ON TABLE product IS 'Таблица товаров';

COMMENT ON COLUMN product.id IS 'ID товара';

COMMENT ON COLUMN product.description IS 'Описание товара';

COMMENT ON COLUMN product.price IS 'Стоимость товара';

COMMENT ON COLUMN product.quantity IS 'Количество товара в наличии';

COMMENT ON COLUMN product.category IS 'Категория товара';

-- customer — id, имя, фамилия, телефон, email

CREATE TABLE IF NOT EXISTS customer (

id SERIAL PRIMARY KEY,

first\_name VARCHAR(50) NOT NULL,

last\_name VARCHAR(50) NOT NULL,

phone VARCHAR(20),

email VARCHAR(100) UNIQUE

);

-- Комментарии к таблицам и ключевым полям

COMMENT ON TABLE customer IS 'Таблица клиентов';

COMMENT ON COLUMN customer.id IS 'ID клиента';

COMMENT ON COLUMN customer.first\_name IS 'Имя клиента';

COMMENT ON COLUMN customer.last\_name IS 'Фамилия клиента';

COMMENT ON COLUMN customer.phone IS 'Телефон клиента';

COMMENT ON COLUMN customer.email IS 'Email клиента';

-- order — id, product\_id (FK), customer\_id (FK), дата заказа, количество, статус

CREATE TABLE IF NOT EXISTS orders (

id SERIAL PRIMARY KEY,

product\_id INTEGER NOT NULL,

customer\_id INTEGER NOT NULL,

order\_date DATE NOT NULL DEFAULT CURRENT\_DATE,

quantity INTEGER NOT NULL CHECK (quantity > 0),

status\_id INTEGER NOT NULL,

FOREIGN KEY (product\_id) REFERENCES product(id),

FOREIGN KEY (customer\_id) REFERENCES customer(id),

FOREIGN KEY (status\_id) REFERENCES order\_status(id)

);

-- Комментарии к таблицам и ключевым полям

COMMENT ON TABLE orders IS 'Таблица заказов';

COMMENT ON COLUMN orders.id IS 'ID заказа';

COMMENT ON COLUMN orders.product\_id IS 'Ссылка на товар (внешний ключ)';

COMMENT ON COLUMN orders.customer\_id IS 'Ссылка на клиента (внешний ключ)';

COMMENT ON COLUMN orders.order\_date IS 'Дата заказа';

COMMENT ON COLUMN orders.quantity IS 'Количество товара в заказе';

COMMENT ON COLUMN orders.status\_id IS 'Статус заказа';

-- order\_status — справочник статусов заказов (id, имя статуса)

CREATE TABLE IF NOT EXISTS order\_status (

id SERIAL PRIMARY KEY,

status\_name VARCHAR(50) NOT NULL UNIQUE

);

-- Комментарии к таблицам и ключевым полям

COMMENT ON TABLE order\_status IS 'Справочник статусов заказов';

COMMENT ON COLUMN order\_status.id IS 'ID статуса';

COMMENT ON COLUMN order\_status.status\_name IS 'Наименование статуса';

-- Индексы по внешним ключам и дате заказа

CREATE INDEX IF NOT EXISTS idx\_orders\_product\_id ON orders(product\_id);

CREATE INDEX IF NOT EXISTS idx\_orders\_customer\_id ON orders(customer\_id);

CREATE INDEX IF NOT EXISTS idx\_orders\_order\_date ON orders(order\_date);

-- Заполнение тестовыми данными — минимум 10 строк в каждой таблице.

-- Товары для детского магазина

INSERT INTO product (description, price, quantity, category) VALUES

('Конструктор LEGO Classic', 2500.00, 25, 'Конструкторы'),

('Детский велосипед 12"', 8500.00, 8, 'Транспорт'),

('Кукла Barbie', 1800.00, 30, 'Куклы'),

('Набор для рисования 100 предметов', 1200.00, 40, 'Творчество'),

('Детский рюкзак с героями мультфильмов', 1500.00, 35, 'Аксессуары'),

('Интерактивная игрушка "Умный щенок"', 3200.00, 12, 'Интерактивные игрушки'),

('Набор детской посуды "Hello Kitty"', 800.00, 50, 'Посуда'),

('Детский столик и стульчик', 4500.00, 6, 'Мебель'),

('Настольная игра "Дженга"', 900.00, 28, 'Настольные игры'),

('Толокар-машинка', 5200.00, 5, 'Транспорт'),

('Пазл 60 элементов "Тачки"', 450.00, 60, 'Настольные игры'),

('Детский костюм супергероя', 2100.00, 15, 'Одежда'),

('Музыкальный коврик для танцев', 3800.00, 7, 'Интерактивные игрушки'),

('Набор для песочницы', 650.00, 45, 'Творчество'),

('Детские наручные часы с GPS', 4200.00, 10, 'Аксессуары')

ON CONFLICT DO NOTHING;

-- Клиенты (родители)

INSERT INTO customer (first\_name, last\_name, phone, email) VALUES

('Анна', 'Иванова', '+79161234567', 'anna.ivanova@mail.ru'),

('Сергей', 'Петров', '+79167654321', 'sergey.petrov@mail.ru'),

('Ольга', 'Сидорова', '+79162345678', 'olga.sidorova@mail.ru'),

('Дмитрий', 'Козлов', '+79163456789', 'dmitry.kozlov@mail.ru'),

('Екатерина', 'Николаева', '+79164567890', 'ekaterina.nikolaeva@mail.ru'),

('Алексей', 'Васильев', '+79165678901', 'alexey.vasiliev@mail.ru'),

('Мария', 'Смирнова', '+79166789012', 'maria.smirnova@mail.ru'),

('Иван', 'Попов', '+79167890123', 'ivan.popov@mail.ru'),

('Юлия', 'Морозова', '+79168901234', 'yulia.morozova@mail.ru'),

('Андрей', 'Волков', '+79169012345', 'andrey.volkov@mail.ru'),

('Татьяна', 'Зайцева', '+79160123456', 'tatyana.zaytseva@mail.ru'),

('Павел', 'Семенов', '+79161234567', 'pavel.semenov@mail.ru')

ON CONFLICT DO NOTHING;

-- Заказы с датами с 1 сентября по 6 октября 2025 года

INSERT INTO orders (product\_id, customer\_id, order\_date, quantity, status\_id) VALUES

(1, 1, '2025-09-01', 1, 1), -- Конструктор LEGO (1 сентября)

(3, 2, '2025-09-03', 2, 2), -- Куклы Barbie (3 сентября)

(2, 3, '2025-09-05', 1, 3), -- Детский велосипед (5 сентября)

(4, 4, '2025-09-08', 1, 4), -- Набор для рисования (8 сентября)

(5, 5, '2025-09-10', 1, 5), -- Детский рюкзак (10 сентября)

(6, 6, '2025-09-12', 1, 6), -- Интерактивная игрушка (12 сентября)

(7, 7, '2025-09-15', 3, 7), -- Набор детской посуды (15 сентября)

(8, 8, '2025-09-18', 1, 8), -- Детский столик (18 сентября)

(9, 9, '2025-09-20', 2, 1), -- Настольная игра (20 сентября)

(10, 10, '2025-09-22', 1, 2), -- Толокар-машинка (22 сентября)

(11, 11, '2025-09-25', 1, 3), -- Пазл (25 сентября)

(12, 12, '2025-09-28', 1, 4), -- Костюм супергероя (28 сентября)

(13, 1, '2025-10-01', 1, 5), -- Музыкальный коврик (1 октября)

(14, 2, '2025-10-03', 2, 6), -- Набор для песочницы (3 октября)

(15, 3, '2025-10-06', 1, 7) -- Детские часы с GPS (6 октября)

ON CONFLICT DO NOTHING;

-- Статусы заказов

INSERT INTO order\_status (status\_name) VALUES

('Новый'), ('Подтвержден'), ('В обработке'), ('Отправлен'), ('В пути'), ('Доставлен'), ('Задерживается'), ('Отменен')

ON CONFLICT (status\_name) DO NOTHING;

**Задача 3. SQL-запросы**

Создать файл test-queries.sql, включающий не менее 10 запросов:

* 5 запросов на чтение (в том числе JOIN с несколькими таблицами, агрегаты, сортировка, фильтрация).
* 3 запроса на изменение (UPDATE).
* 2 запроса на удаление (DELETE).

Примеры:

* Список всех заказов за последние 7 дней с именем покупателя и описанием товара.
* Топ-3 самых популярных товара.
* Обновление количества на складе при покупке.
* Удаление клиентов без заказов

-- SQL-запросы для магазина детских товаров

-- \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

-- ЗАПРОСЫ НА ЧТЕНИЕ

-- \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

-- 1. Список всех заказов за последние 7 дней с именем покупателя и описанием товара

SELECT

o.id as order\_id,

o.order\_date,

c.first\_name || ' ' || c.last\_name as customer\_name,

p.description as product\_name,

o.quantity,

os.status\_name,

(p.price \* o.quantity) as total\_price

FROM orders o

JOIN customer c ON o.customer\_id = c.id

JOIN product p ON o.product\_id = p.id

JOIN order\_status os ON o.status\_id = os.id

WHERE o.order\_date >= CURRENT\_DATE - INTERVAL '7 days'

ORDER BY o.order\_date DESC;

-- 2. Топ-3 самых популярных товаров (по количеству заказов)

SELECT

p.description as product\_name,

p.category,

COUNT(o.id) as order\_count,

SUM(o.quantity) as total\_ordered

FROM product p

JOIN orders o ON p.id = o.product\_id

GROUP BY p.id, p.description, p.category

ORDER BY order\_count DESC, total\_ordered DESC

LIMIT 3;

-- 3. Товары с низким остатком (менее 10 шт)

SELECT

p.description as product\_name,

p.category as category,

p.quantity as current\_stock,

p.price as price,

CASE

WHEN p.quantity < 5 THEN 'CRITICALLY LOW'

WHEN p.quantity < 10 THEN 'LOW'

ELSE 'NORMAL'

END as stock\_level

FROM product p

WHERE p.quantity < 10

ORDER BY p.quantity ASC;

-- 4. Статистика заказов за текущий год

SELECT

EXTRACT(MONTH FROM o.order\_date) as month\_number,

TO\_CHAR(o.order\_date, 'Month') as month\_name,

COUNT(o.id) as order\_count,

SUM(o.quantity) as total\_items,

SUM(p.price \* o.quantity) as total\_revenue

FROM orders o

JOIN product p ON o.product\_id = p.id

WHERE EXTRACT(YEAR FROM o.order\_date) = EXTRACT(YEAR FROM CURRENT\_DATE)

GROUP BY month\_number, month\_name

ORDER BY month\_number;

-- 5. Заказы по статусам с общей суммой

SELECT

os.status\_name as status,

COUNT(o.id) as order\_count,

SUM(o.quantity) as total\_items,

SUM(p.price \* o.quantity) as total\_amount

FROM orders o

JOIN order\_status os ON o.status\_id = os.id

JOIN product p ON o.product\_id = p.id

GROUP BY os.status\_name

ORDER BY order\_count DESC;

-- \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

-- ЗАПРОСЫ НА ИЗМЕНЕНИЕ (UPDATE)

-- \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

-- 6. Обновление количества товара на складе для недавно отправленных заказов

UPDATE product

SET quantity = quantity - (

SELECT COALESCE(SUM(o.quantity), 0)

FROM orders o

WHERE o.product\_id = product.id

AND o.status\_id IN (SELECT id FROM order\_status WHERE status\_name IN ('Отправлен', 'В пути'))

AND o.order\_date >= CURRENT\_DATE - INTERVAL '3 days'

)

WHERE id IN (

SELECT DISTINCT product\_id

FROM orders

WHERE status\_id IN (SELECT id FROM order\_status WHERE status\_name IN ('Отправлен', 'В пути'))

AND order\_date >= CURRENT\_DATE - INTERVAL '3 days'

);

-- 7. Обновление статуса заказов, которые "В пути" дольше 5 дней

UPDATE orders

SET status\_id = (SELECT id FROM order\_status WHERE status\_name = 'Задерживается')

WHERE status\_id = (SELECT id FROM order\_status WHERE status\_name = 'В пути')

AND order\_date <= CURRENT\_DATE - INTERVAL '5 days';

-- 8. Скидка 10% на товары, которые на складе больше 3 месяцев

UPDATE product

SET price = price \* 0.9

WHERE quantity > 20

AND id NOT IN (

SELECT DISTINCT product\_id

FROM orders

WHERE order\_date >= CURRENT\_DATE - INTERVAL '3 months'

);

-- \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

-- ЗАПРОСЫ НА УДАЛЕНИЕ (DELETE)

-- \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

-- 9. Удаление клиентов без заказов за последние 6 месяцев

DELETE FROM customer

WHERE id NOT IN (

SELECT DISTINCT customer\_id

FROM orders

WHERE order\_date >= CURRENT\_DATE - INTERVAL '6 months'

);

-- 10. Удаление отмененных заказов старше 1 года

DELETE FROM orders

WHERE status\_id = (SELECT id FROM order\_status WHERE status\_name = 'Отменен')

AND order\_date < CURRENT\_DATE - INTERVAL '1 year';

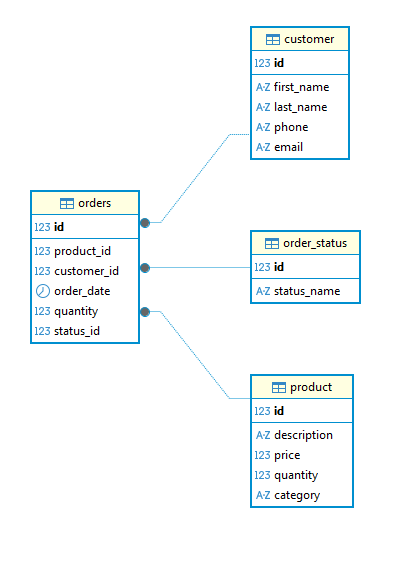
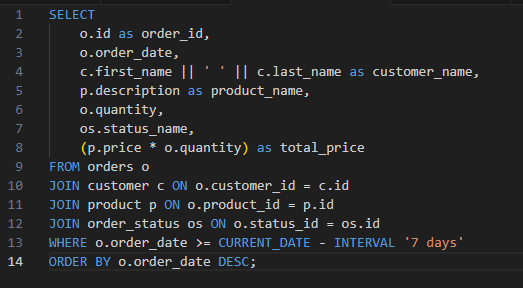
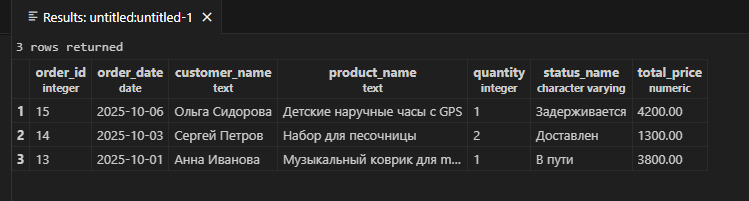


Рис. ERD-диаграмма (взято из Dbeaver)

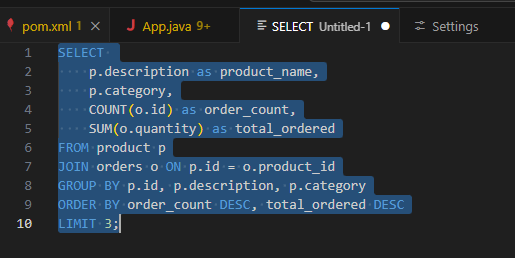
Запуск запросов из файла test-queries.sql в VS Code:

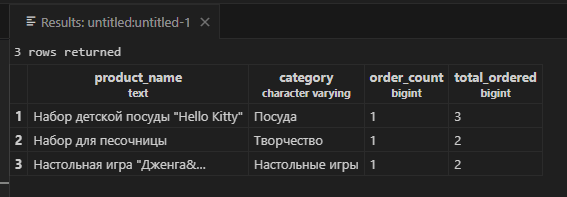
1. Список всех заказов за последние 7 дней с именем покупателя и описанием товара



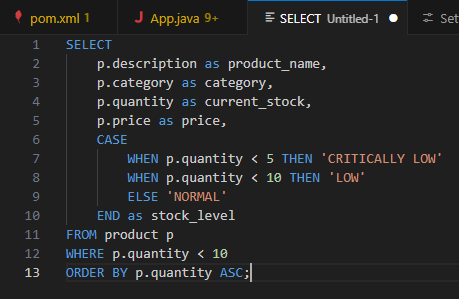


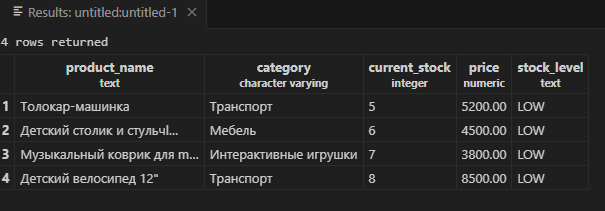
2. Топ-3 самых популярных товаров (по количеству заказов)



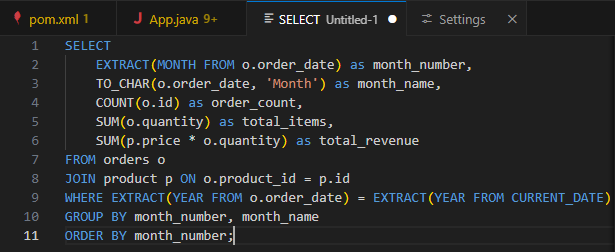


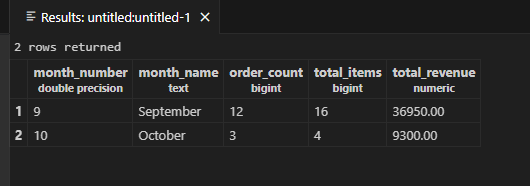
3. Товары, требующие пополнения запаса (менее 20% от максимального количества)



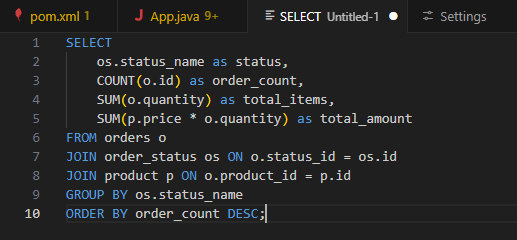


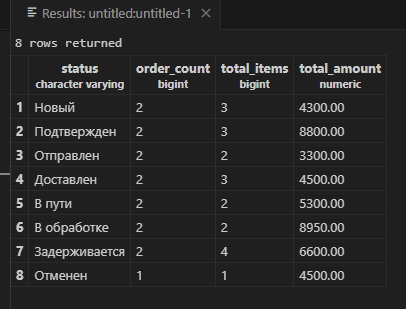
4. Статистика заказов за текущий год



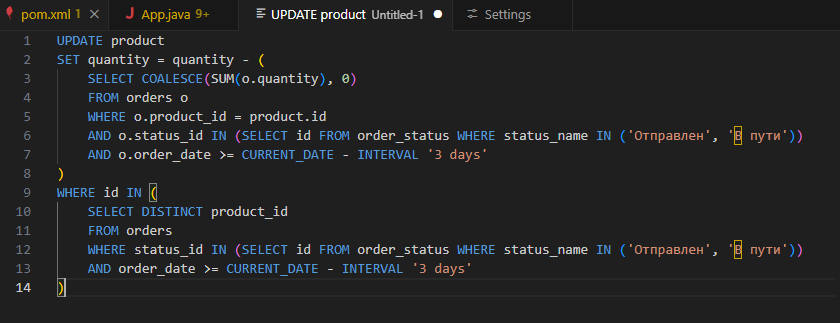


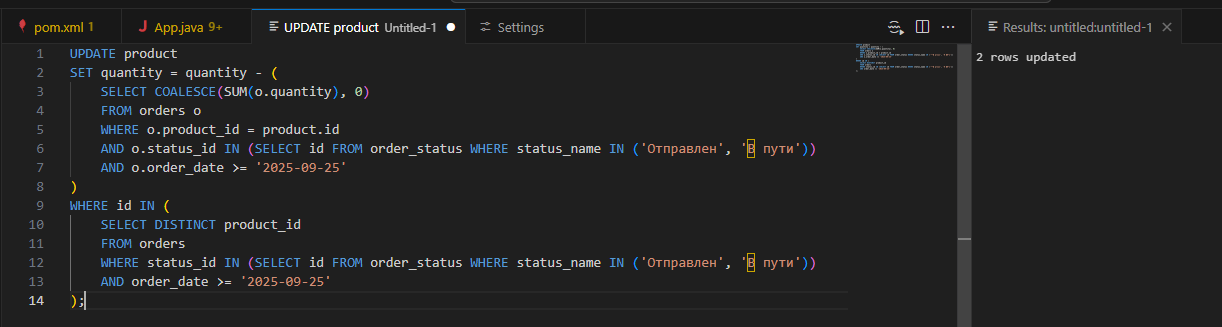
5. Заказы по статусам с общей суммой



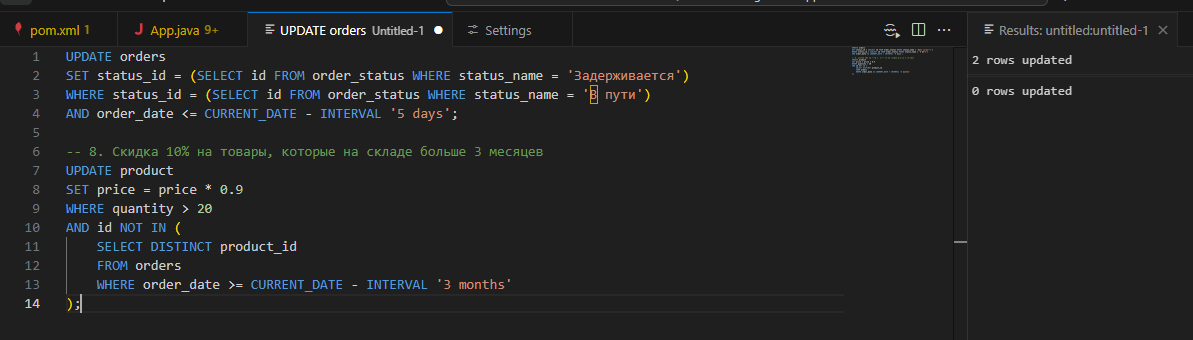


6. Обновление количества товара на складе для недавно отправленных заказов

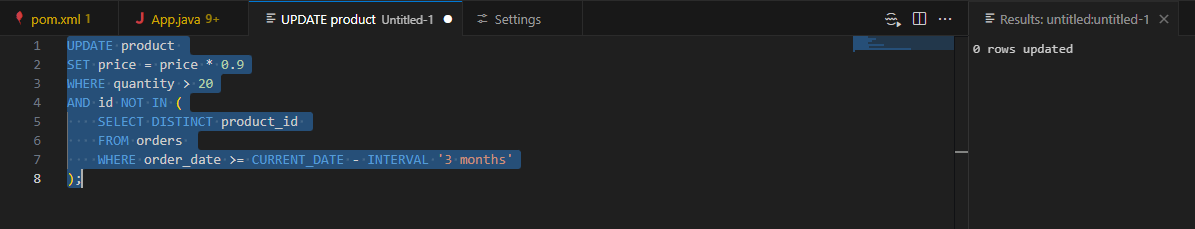




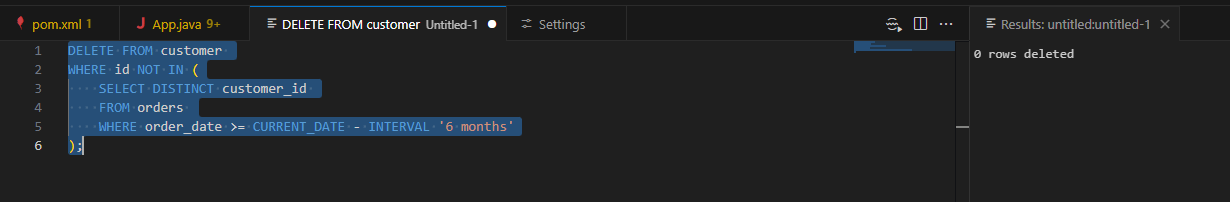
7. Обновление статуса заказов, которые "В пути" дольше 5 дней



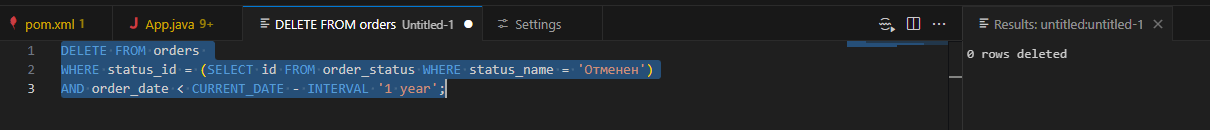
8. Скидка 10% на товары, которые на складе больше 3 месяцев



9. Удаление клиентов без заказов за последние 6 месяцев



10. Удаление отмененных заказов старше 1 года



**Задача 4. Java-приложение (Maven, без Spring)**

1. Создать Maven-проект (Java 17) с зависимостями:
   1. PostgreSQL JDBC Driver
   2. (опционально) Flyway для миграций
2. Реализовать класс App.java, который:
   1. Подключается к БД через JDBC (DriverManager).
   2. Выполняет автоматический запуск миграций (если используется Flyway).
   3. Демонстрирует CRUD-операции:
      1. Вставка нового товара и покупателя (PreparedStatement)
      2. Создание заказа для покупателя.
      3. Чтение и вывод последних 5 заказов с JOIN на товары и покупателей.
      4. Обновление цены товара и количества на складе.
      5. Удаление тестовых записей.
   4. Работает в транзакции с commit() и rollback() при ошибках.
   5. Выводит результаты операций в консоль в удобочитаемом формате.
3. Параметры подключения (URL, user, password) вынести в application.properties

Maven-проект создан на версии Java 21.

App класс:

Подключается к БД через JDBC (DriverManager).

connection = DriverManager.getConnection(url, username, password);

Выполняет автоматический запуск миграций (если используется Flyway).

 // Настройка Flyway

            Flyway flyway = Flyway.configure()

                    .dataSource(url, username, password)

                    .locations("db/migration")

                    .load();

            // Запуск миграций

            flyway.migrate();

Демонстрация CRUD

