МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Федеральное государственное бюджетное образовательное учреждение высшего образования

Московский авиационный институт

«Национальный исследовательский университет»

Институт № 3 «Системы управления, информатика и электроэнергетика»

Кафедра 304: «Вычислительные машины, системы и сети»

Отчёт по лабораторной работе №27 «Аналитика данных c помощью Grafana»

по учебной дисциплине «Web-технологии»

Выполнил:

студент группы М3О-107БВ-24

Коркина К.В.

Приняла: Масленикова Т.Ю.

Титов Ю.П.

Москва 2025

1. Развернуть локальный сервер с Grafana на своем ПК с помощью Docker.
2. Развернуть локальный сервер с Postgres на своем ПК с помощью Docker. (*Допустимо использовать тот же сервер, что использует Grafana*.)

**Код на docker-compose.yaml:**

*version: '3.8'*

*services:*

*postgres:*

*image: postgres:15*

*container\_name: postgres*

*environment:*

*POSTGRES\_USER: grafana*

*POSTGRES\_PASSWORD: grafana*

*POSTGRES\_DB: analytics*

*volumes:*

*- postgres\_data:/var/lib/postgresql/data*

*- C:\Users\Admin\Documents\Grafana\sql\ddl.sql:/docker-entrypoint-initdb.d/ddl.sql*

*- C:\Users\Admin\Documents\Grafana\sql\dml.sql:/docker-entrypoint-initdb.d/dml.sql*

*ports:*

*- "5432:5432"*

*restart: unless-stopped*

*grafana:*

*image: grafana/grafana:latest*

*container\_name: grafana*

*depends\_on:*

*- postgres*

*ports:*

*- "3000:3000"*

*volumes:*

*- grafana\_data:/var/lib/grafana*

*restart: unless-stopped*

*volumes:*

*postgres\_data:*

*grafana\_data:*

Команды для запуска (командная строка):

**docker-compose up -d** - запуск контейнера:

*PS C:\Users\Admin\Documents\Grafana> docker compose up -d*

*time="2025-05-17T19:00:30+03:00" level=warning msg="C:\\Users\\Admin\\Documents\\Grafana\\docker-compose.yaml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion"*

*[+] Running 2/2*

*✔Container postgres Running 0.0s*

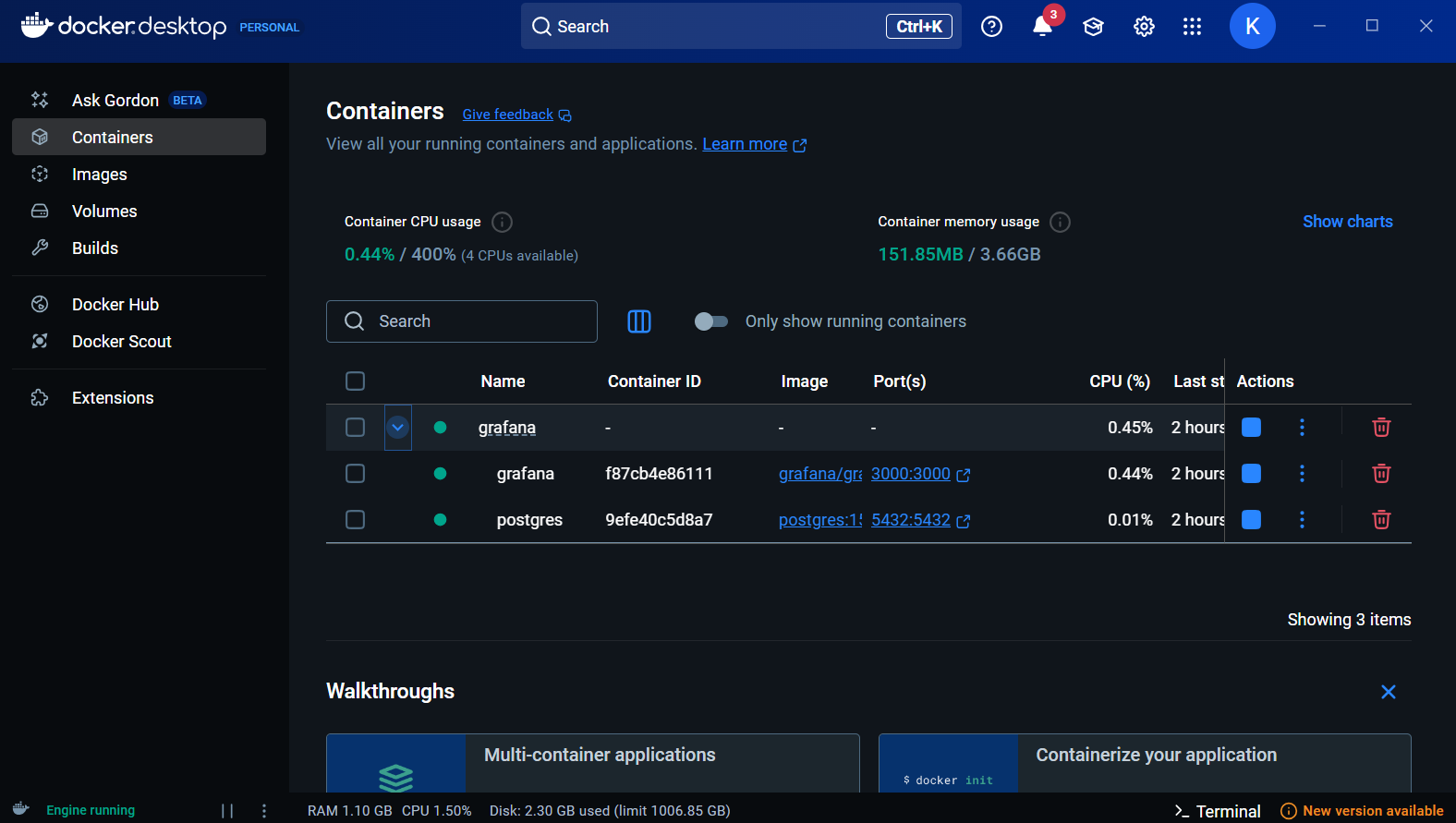
*✔ Container grafana Running*

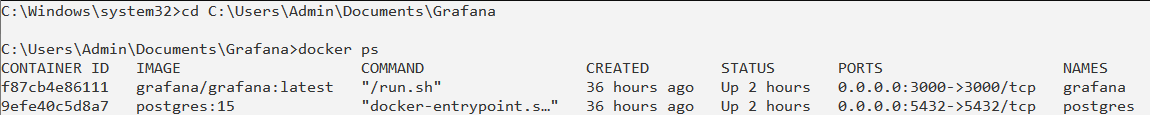
*0.0s*

**docker ps** - проверка состояния

**docker-compose down -v** - остановка контейнера

На скриншоте – Docker в рабочем состоянии и показано, что контейнеры запущены.





Вместо ручного запуска контейнеров PostgreSQL и Grafana отдельными командами *docker run*, используется один файл docker-compose.yml, который описывает все сервисы PostgreSQL и Grafana, настраивает их взаимодействие Grafana зависит от PostgreSQL и управляет томами (volumes) для сохранения данных.

1. Создать новую базу данных в Postgres и добавить в неё 10-15 записей в качестве данных, которые будут анализироваться внутри Grafana.

**ddl.sql: (для создания структуры)**

*CREATE TABLE products(*

*products\_id SERIAL PRIMARY KEY,*

*product\_name VARCHAR(50),*

*category VARCHAR(50),*

*price DECIMAL(10, 2),*

*amount INT,*

*sale\_data DATE*

*)*

**dml.sql: (для наполнения структуры содержимым)**

*INSERT INTO products (product\_name, category, price, amount, sale\_data)*

*VALUES*

*('Молоко', 'Молочная продукция', 100.00, 5, '2025-03-25'),*

*('Сыр', 'Молочная продукция', 120.00, 50, '2025-03-26'),*

*('Творог', 'Молочная продукция', 300.25, 10, '2025-02-27'),*

*('Йогурт', 'Молочная продукция', 150.03, 3, '2025-04-20'),*

*('Курица', 'Мясо', 1000.00, 100, '2025-01-15'),*

*('Свинина', 'Мясо', 2000.00, 200, '2025-01-25'),*

*('Говядина', 'Мясо', 3000.00, 300, '2025-01-30'),*

*('Тушенка', 'Мясо', 500.00, 5, '2025-01-14'),*

*('Ветчина', 'Мясо', 500.00, 10, '2025-01-02'),*

*('Яблоко', 'Фрукты', 50.07, 50, '2025-05-01'),*

*('Бананы', 'Фрукты', 100.00, 40, '2025-05-01'),*

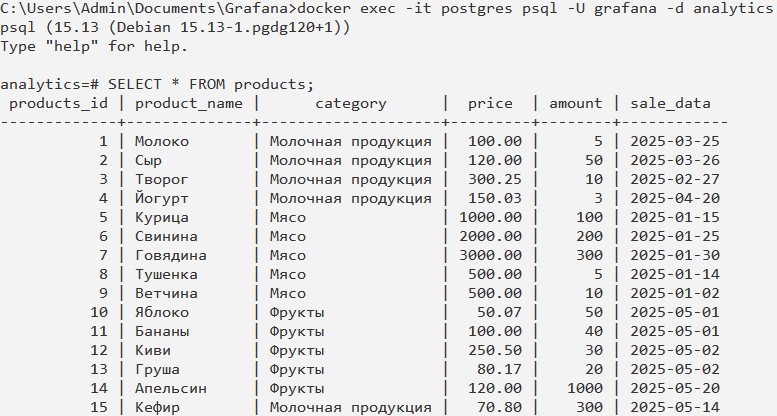
*('Киви', 'Фрукты', 250.50, 30, '2025-05-02'),*

*('Груша', 'Фрукты', 80.17, 20, '2025-05-02');*

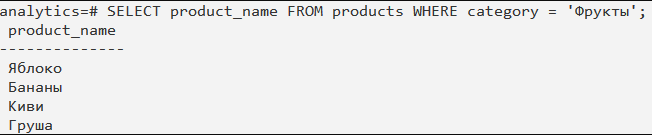
Проверка: вывод данных при разных запросах

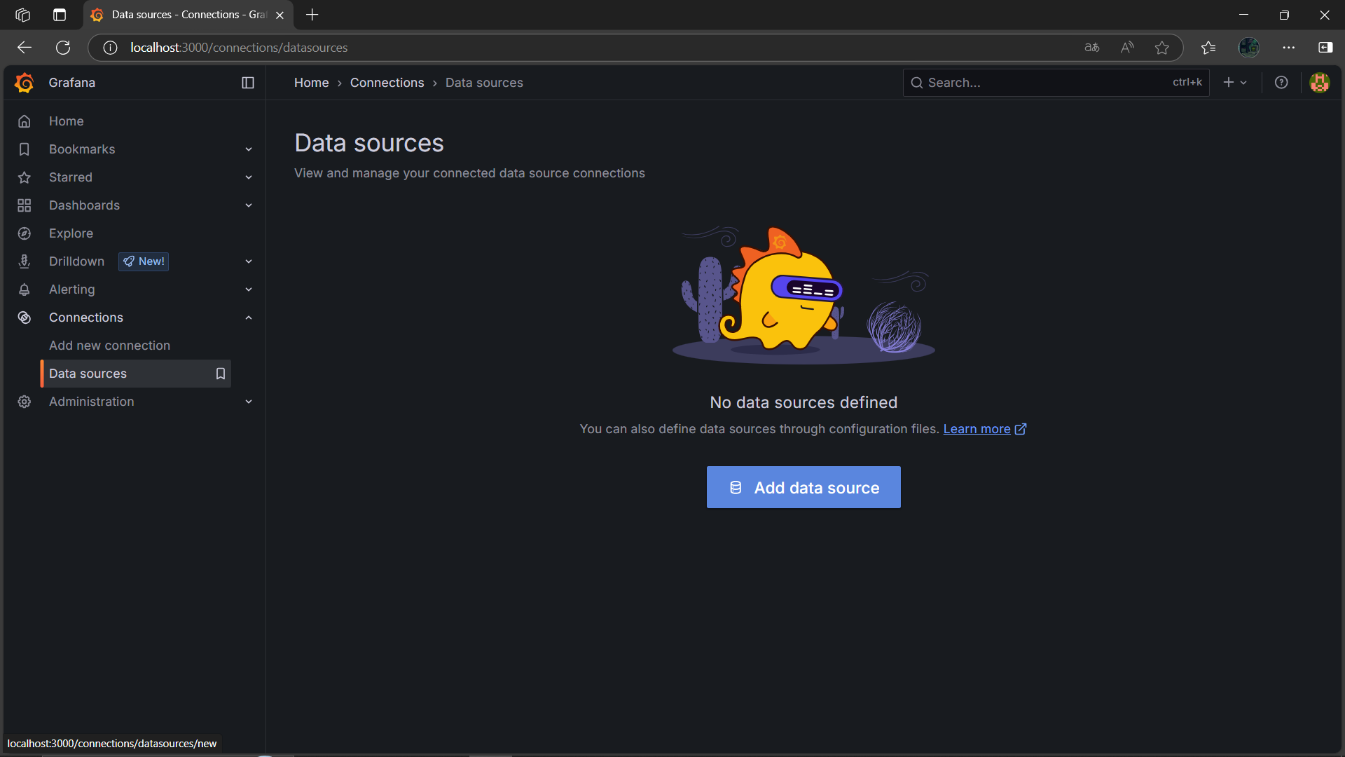
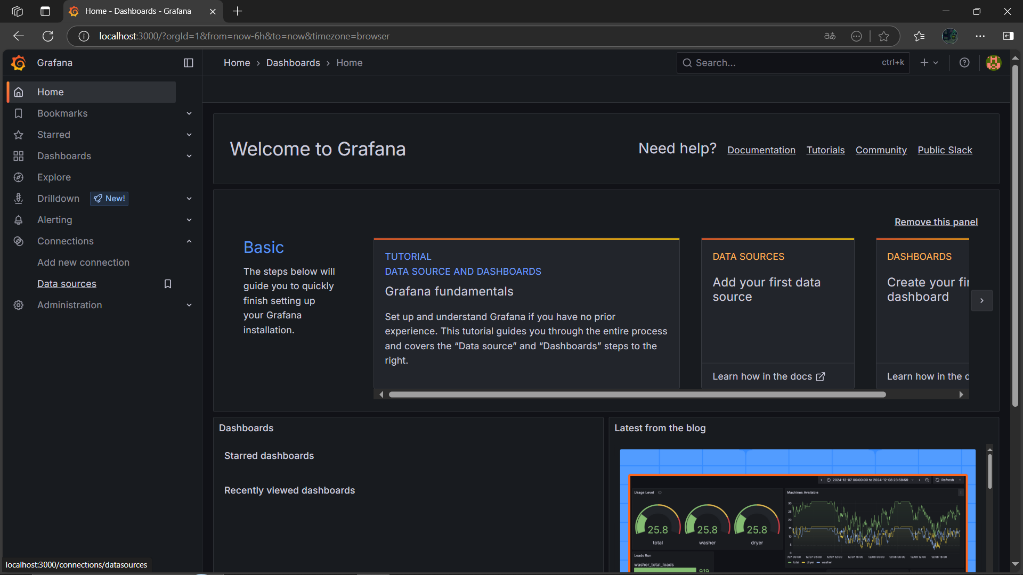
**docker exec -it postgres psql -U grafana -d analytics –** для работы с запросами

*SELECT \* FROM products;* - получить всю таблицу



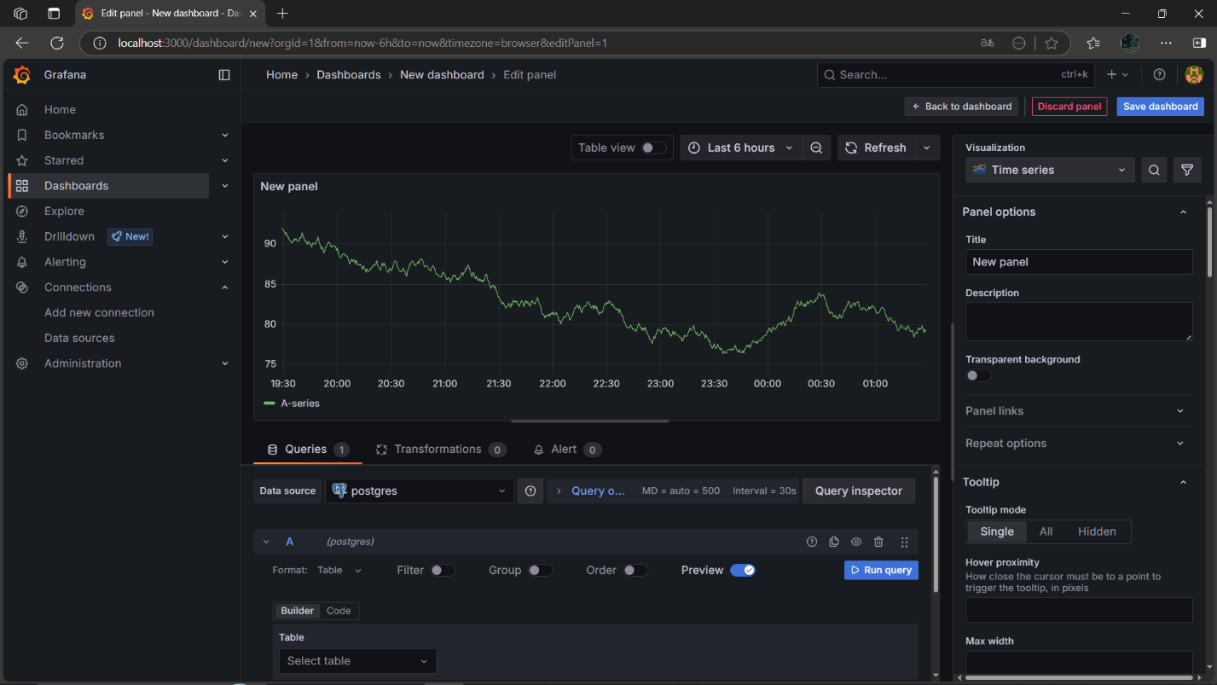
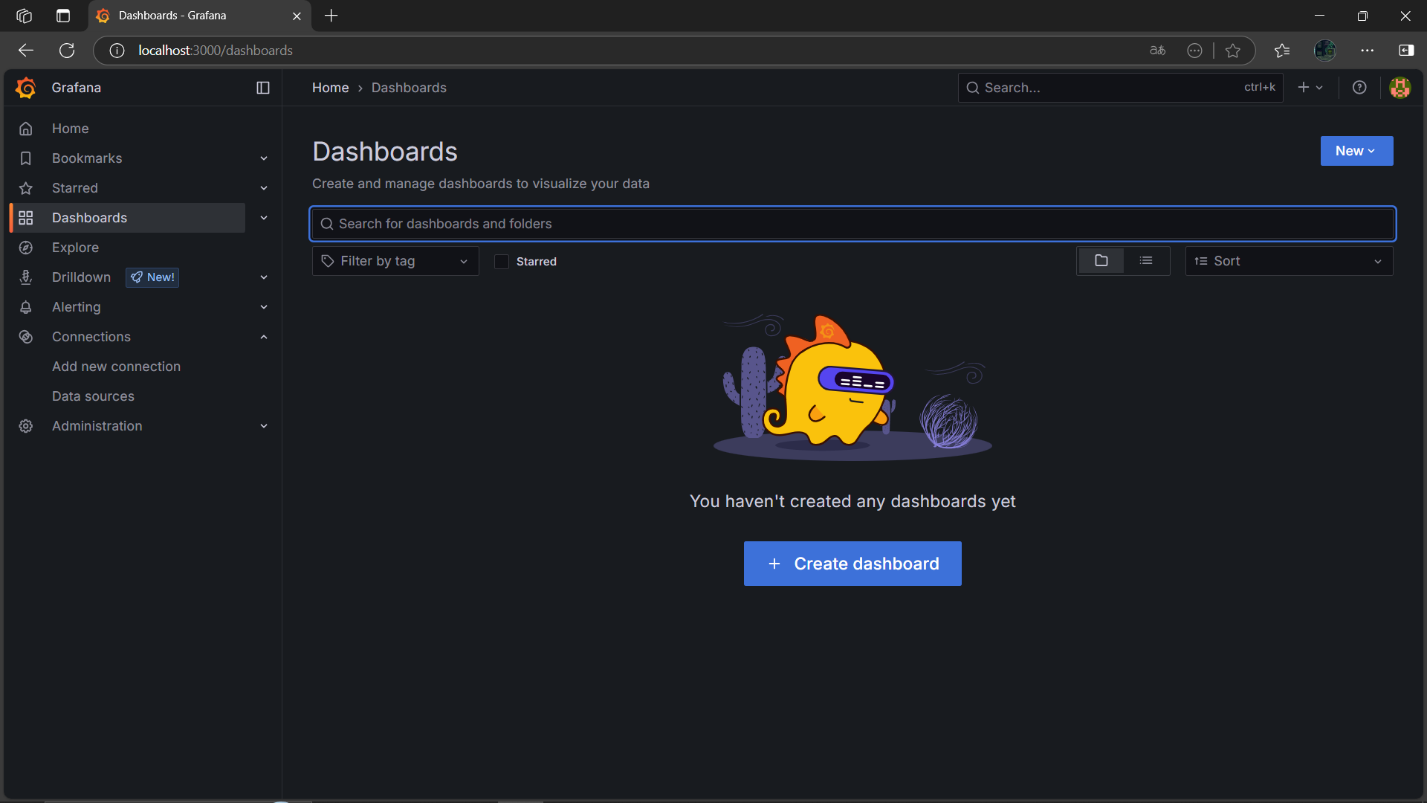
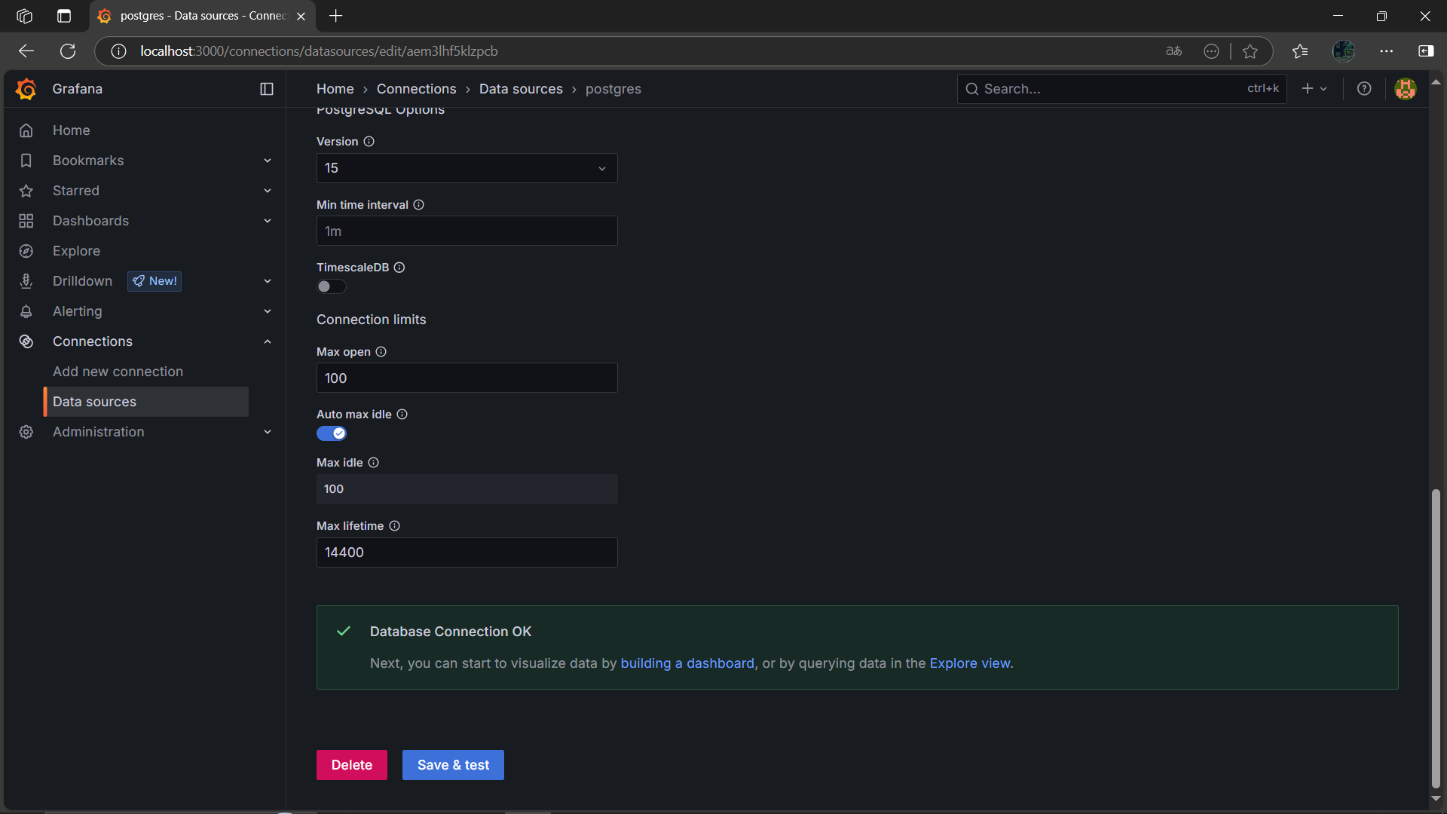
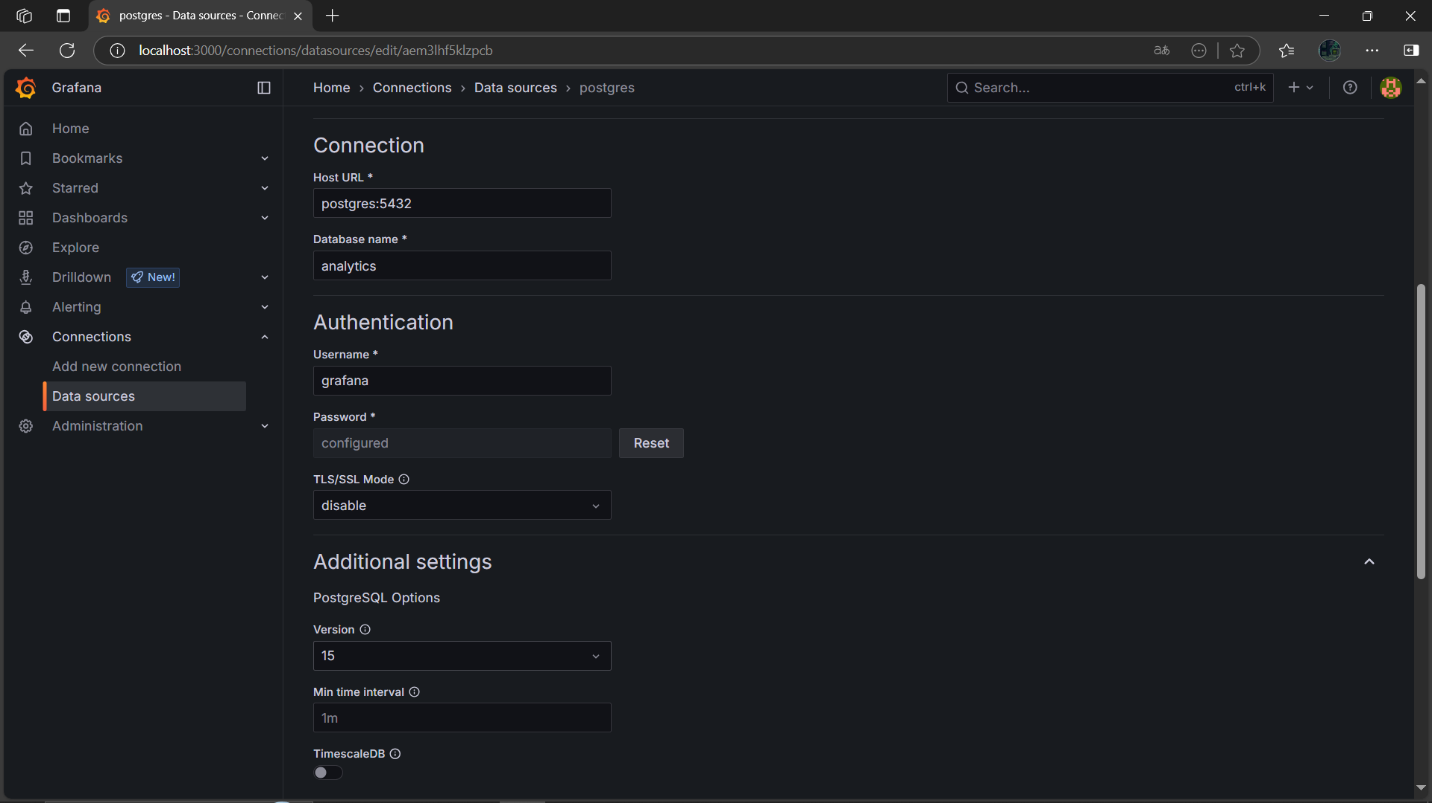
*SELECT product\_name FROM products WHERE category = 'Фрукты';* - получить столбец, содержащий названия продуктов из таблицы products по категории ‘Фрукты’



1. Подключиться к базе данных с помощью Grafana. 

Добавление источника данных

Стартовая страница Grafana

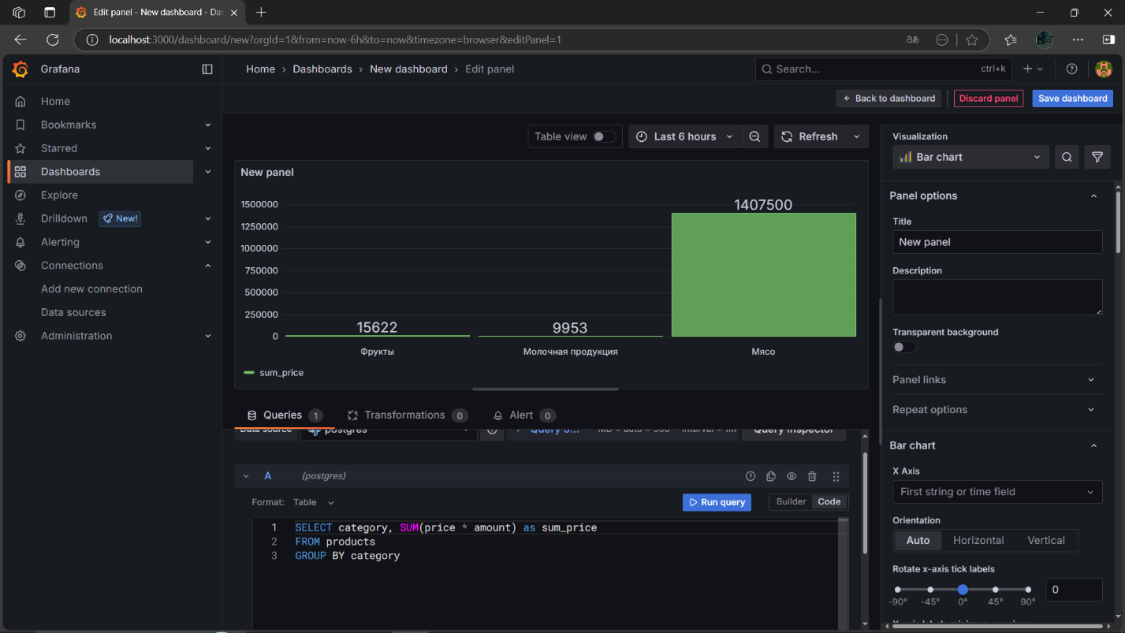


Начальный дашборд в Grafana (надо будет подключить базу данных products для составления запросов)

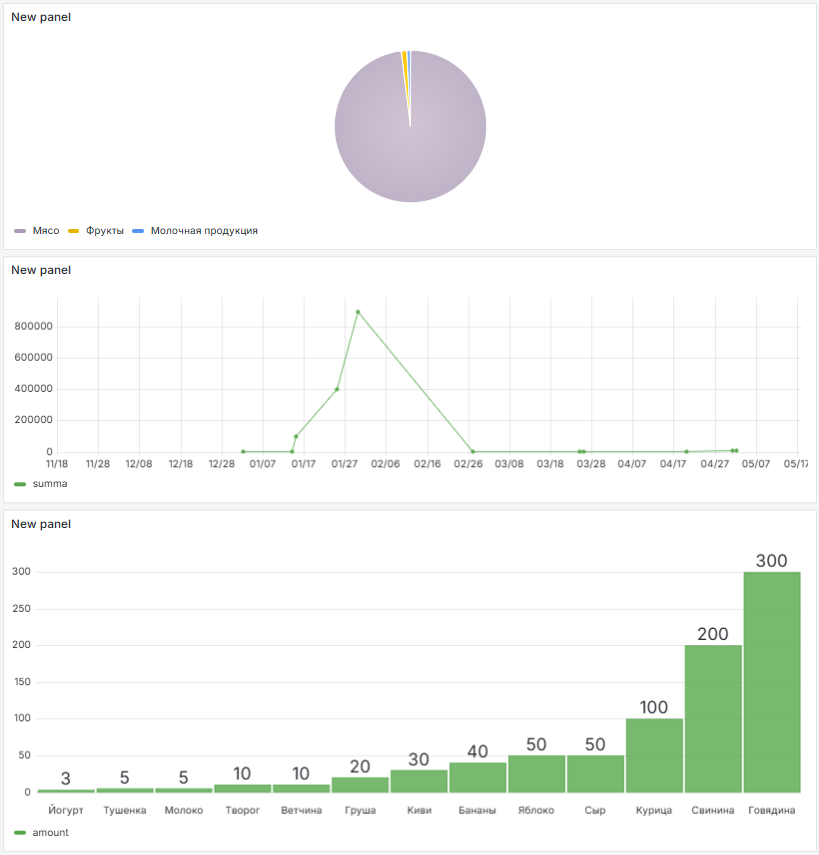
Настройка подключения (данные из docker-compose.yaml)

1. Создать новую панель внутри Grafana для начала работы с данными из Postgres.

Новая панель для работы с данными из Postgres для примера – виджет Bar Chart. Запрос отображает общую сумму продаж всех продуктов (умножает цену каждого продукта на его количества и складывает в соответствующих категориях) по различным категориям (и так же отсортированную по этим категориям).



1. На созданной панели добавить любые три виджета для отображения ранее записанных данных. Например, pie chart или bar chart.



(Сверху сниз): 1) **Pie Chart** (*SELECT category, SUM (price \* amount) as all\_price FROM products GROUP BY category*) отображает сумму продаж продуктов по их категориям;

2) **Time Series** (*SELECT sale\_data, SUM(price \* amount) as summa FROM products GROUP BY sale\_data ORDER BY 1*) отображает общую сумму продаж в определенные даты;

3) **Bar Chart** *(SELECT product\_name, amount FROM products ORDER BY amount)* отображает сумму продаж по категориям

1. Для каждого из них написать запрос в базу данных. (пт. 6)
2. Проверить, что при добавлении данных виджеты обновляются.
3. Результаты проделанной работы отобразить в отчёте поэтапно.

**Добавление данных:**

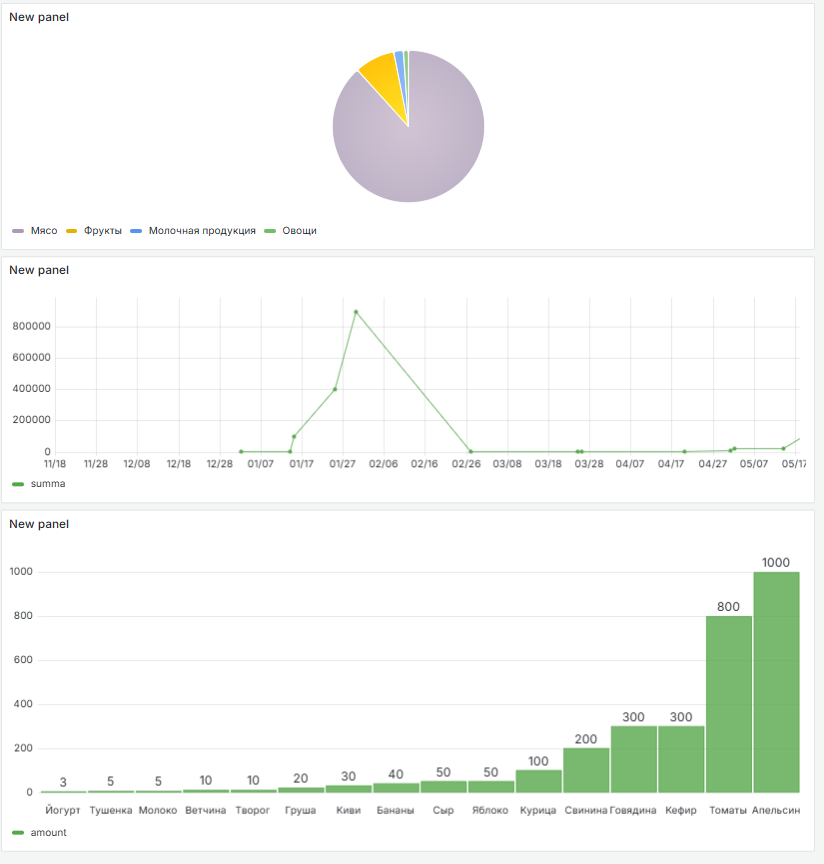
**Запросы sql (на добавление элементов в существующие категории):**

*INSERT INTO products (product\_name, category, price, amount, sale\_data) VALUES ('Апельсин', 'Фрукты', 120.00, 1000, '2025-05-20');*

*INSERT INTO products (product\_name, category, price, amount, sale\_data) VALUES ('Кефир', 'Молочная продукция', 70.80, 300, '2025-05-14');*

**Добавление элемента новой категории:**

*INSERT INTO products(product\_name, category, price, amount, sale\_data) VALUES ('Томаты', 'Овощи', 20.20, 800, '2025-05-02');*

**