Департамент образования города Москвы

Государственное автономное образовательное учреждение

высшего образования города Москвы

«Московский городской педагогический университет»

Институт цифрового образования

Департамент информатики, управления и технологий

**Лабораторная работа 4-1**

**по дисциплине «Инструменты для хранения и обработки больших данных»**

**Тема: «Сравнение подходов хранения больших данных»**

Направление подготовки 38.03.05 – бизнес-информатика

Профиль подготовки «Аналитика данных и эффективное управление»

(очная форма обучения)

Выполнила:

Студентка группы АДЭУ-211

Белик Мария Константиновна

Преподаватель:

Босенко Т.М.

Москва

2024

**ВВЕДЕНИЕ**

Цель работы: сравнить производительность и эффективность различных подходов к хранению и обработке больших данных на примере реляционной базы данных PostgreSQL и документоориентированной базы данных MongoDB.

Оборудование и ПО:

* операционная система Ubuntu;
* MongoDB;
* PostgreSQL
* язык программирования Python;
* Библиотеки: psycopg2, pymongo, pandas, matplotlib.

**ТЕОРЕТИЧЕСКАЯ ЧАСТЬ**

В современном мире объемы данных растут экспоненциально, что приводит к необходимости использования эффективных методов их хранения и обработки. Существует два основных подхода к хранению больших данных:

1. Реляционные базы данных (например, PostgreSQL)

2. NoSQL базы данных (например, MongoDB)

Каждый из этих подходов имеет свои преимущества и недостатки, которые будут рассмотрены в ходе выполнения лабораторной работы.

**ПРАКТИЧЕСКАЯ ЧАСТЬ**

**Шаг 1. Подготовка к работе**

В первую очередь требуется запустить виртуальную машину, на которой установлена операционная система Ubuntu:

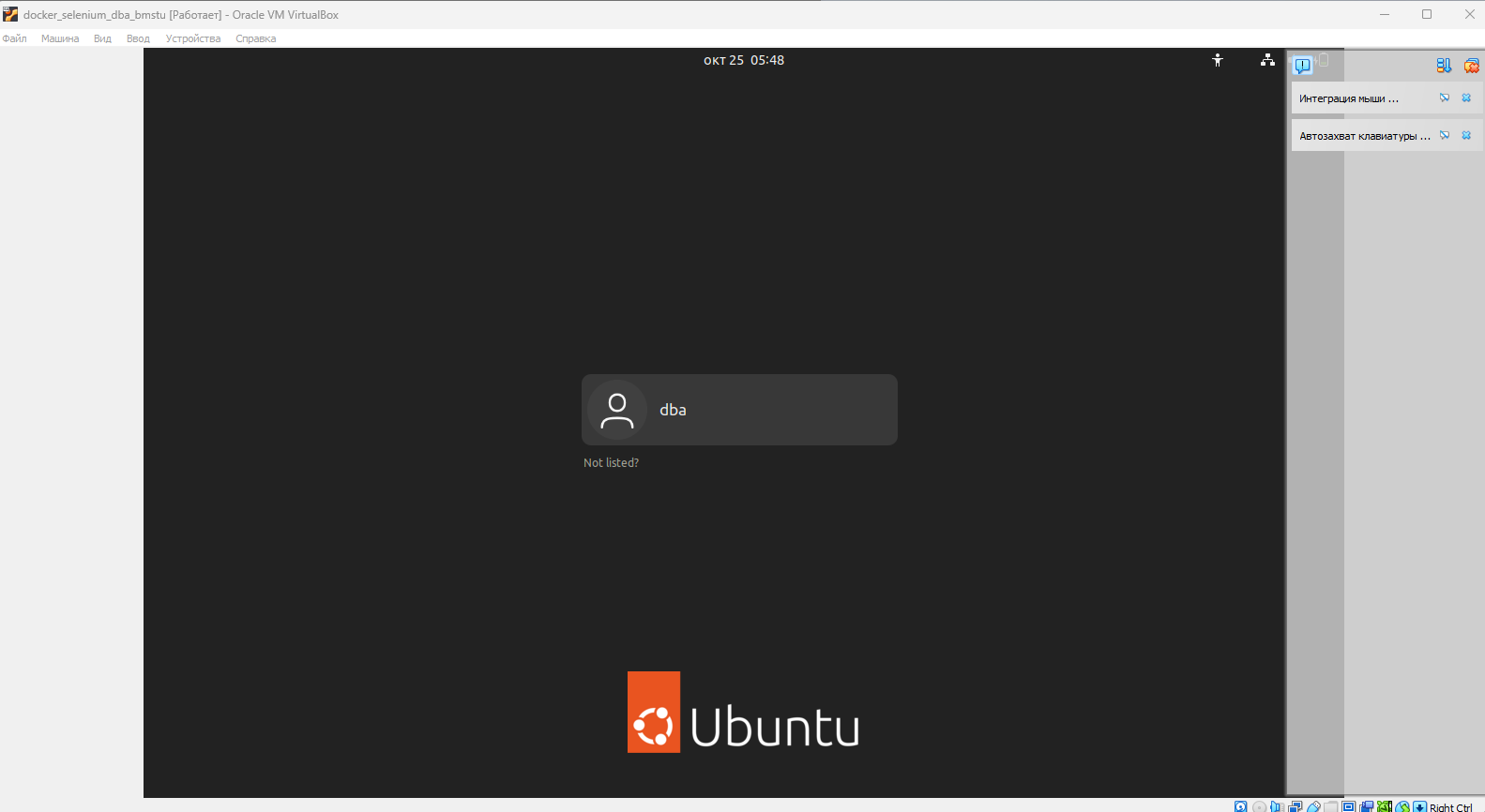


Рис. 1 – Запуск виртуальной машины

Для удобства дальнейшей работы подключимся к запущенной виртуальной машине через подключение к удаленному рабочему столу:

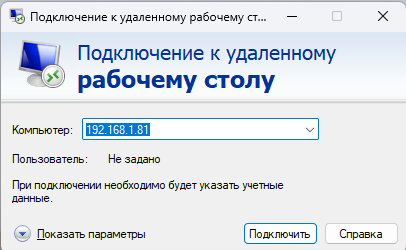


Рис. 2 – Указание настроек при подключении к удаленному рабочему столу

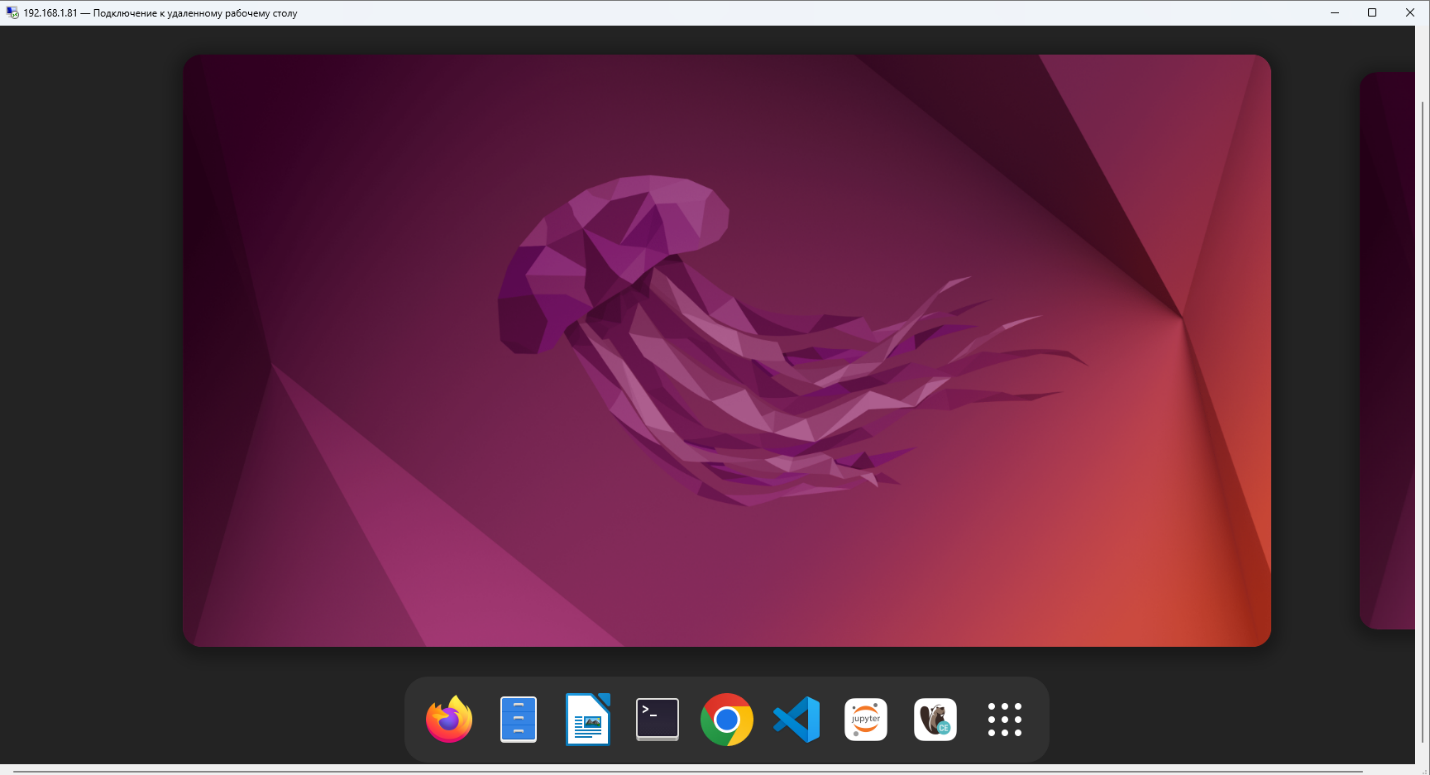


Рис. 3 – Успешное подключение к удаленному рабочему столу

Открываем файл *docker-compose.yml* в приложении “Visual Studio Code”, после чего запускаем докер-контейнеры командой sudo docker compose up -d*:*

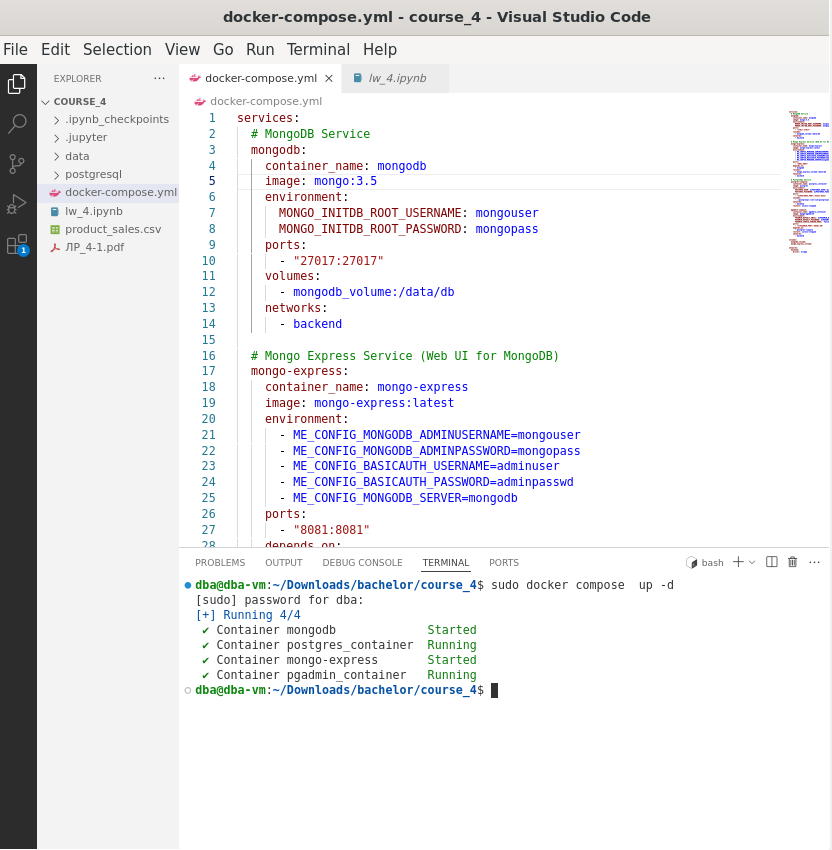
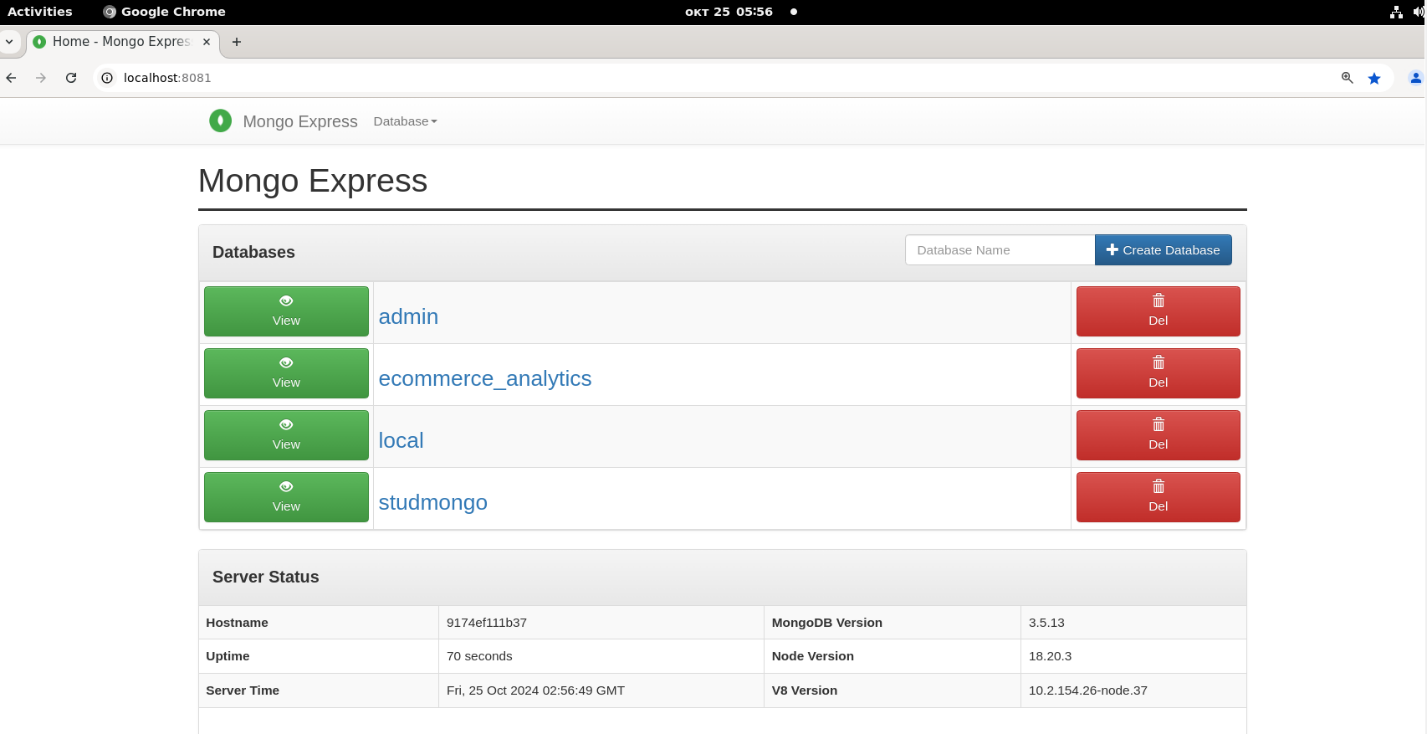


Рис. 4 – Запуск докер-контейнеров

После успешного запуска докер-контейнеров проверяем доступ к MongoDB (http://localhost:8081) и pgAdmin (http://localhost:5050) с помощью Google Chrome:

Рис. 5 – Открытие Mongo Express

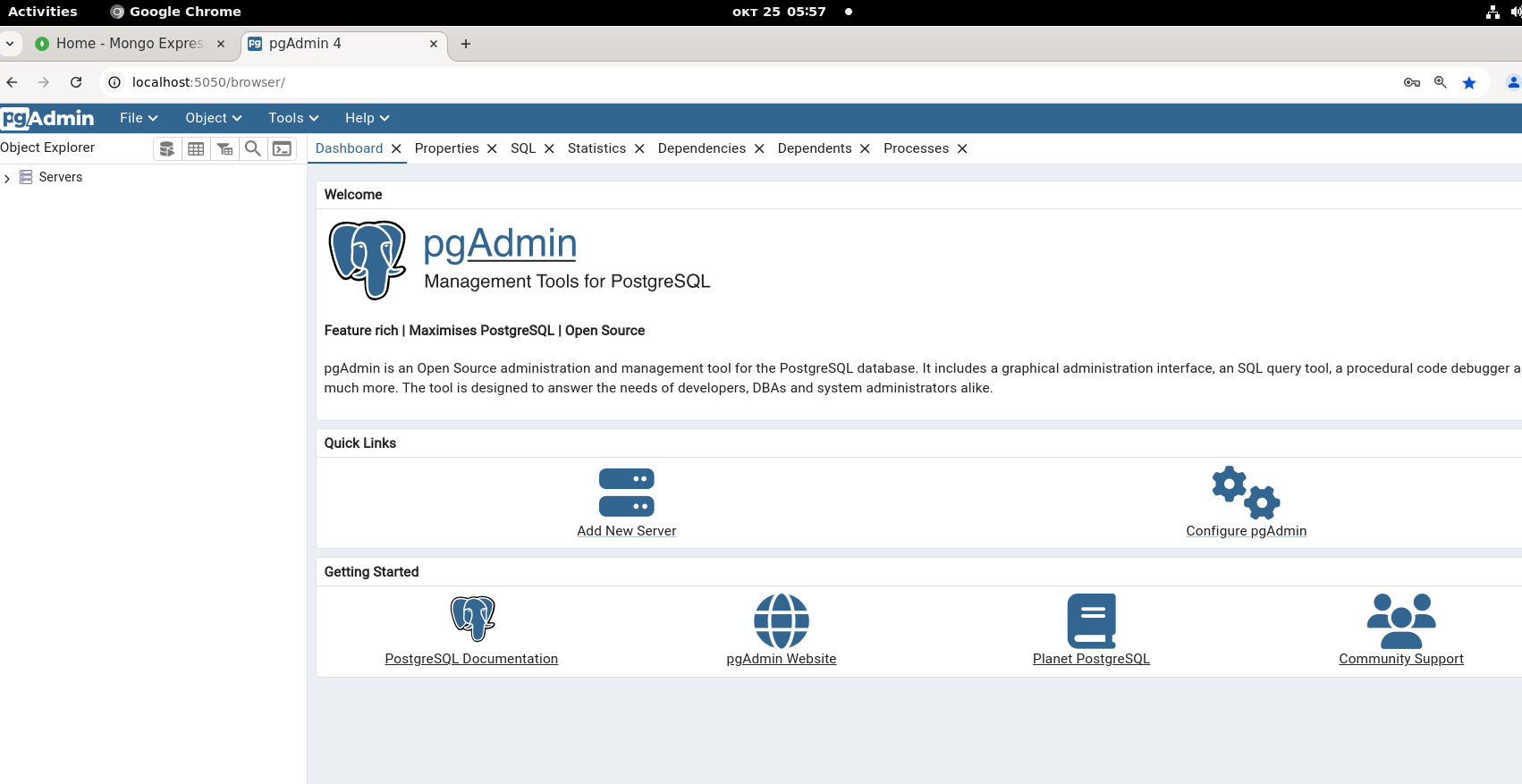


Рис. 6 – Открытие pgAdmin

**Шаг 2. Работа с MongoDB**

Для того, чтобы начать работу с MongoDB необходимо создать новый блокнот в приложении “Jupiter Notebook”.

Далее требуется установить и скачать все необходимые для работы библиотеки и прописать функции для подключения к базам данных:

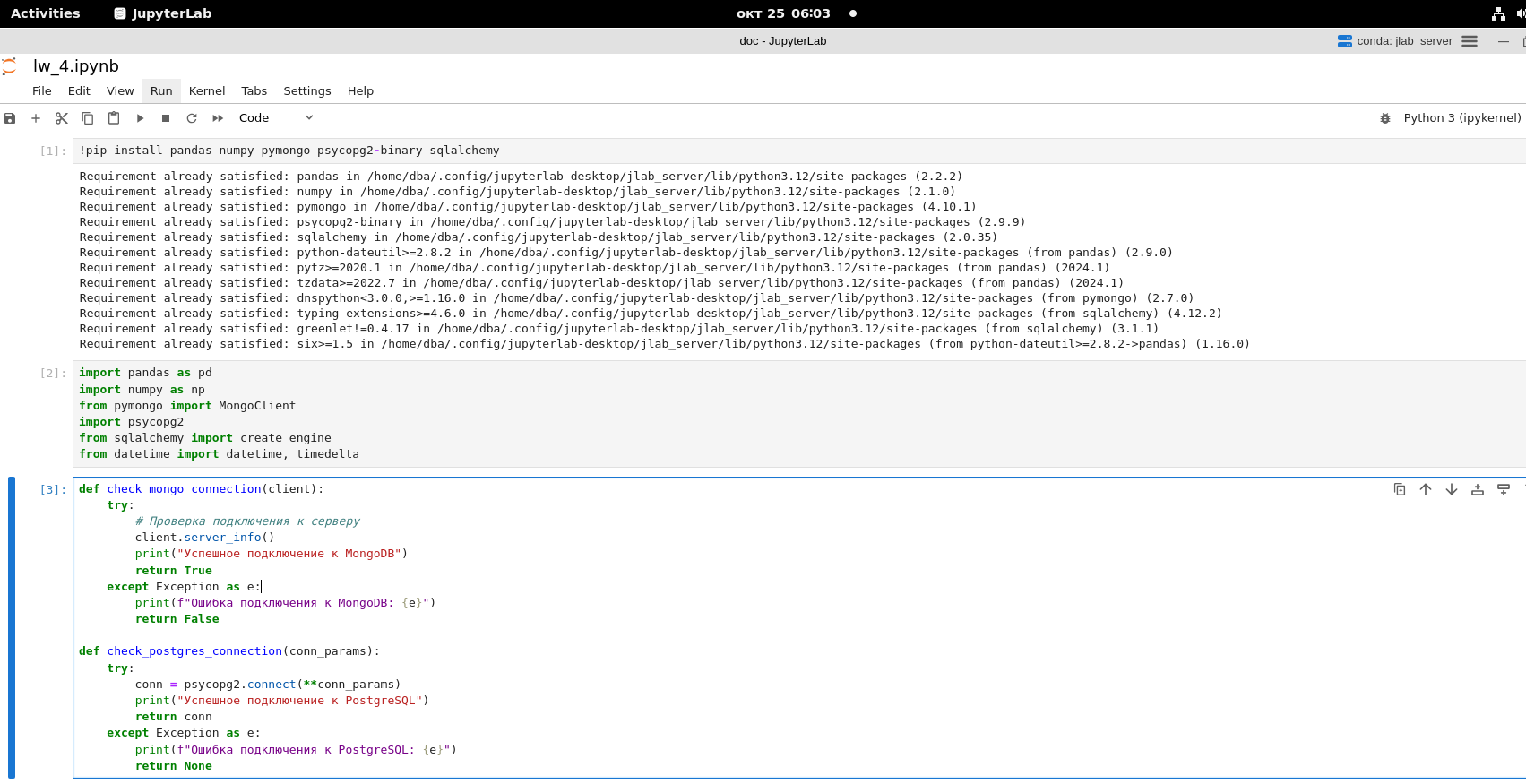


Рис. 7 – Скачивание необходимых для работы библиотек

Перейдем к генерированию данных. По моему варианту (1) необходимо сгенерировать данные о продажах в интернет-магазине, согласно следующему коду примеру:

import random

from datetime import datetime, timedelta

import json

import psycopg2

from pymongo import MongoClient

from faker import Faker

fake = Faker()

def generate\_sales\_data(n\_records):

sales = []

for \_ in range(n\_records):

sale = {

"order\_id": fake.uuid4(),

"customer\_id": fake.uuid4(),

"product\_id": fake.uuid4(),

"quantity": random.randint(1, 10),

"price": round(random.uniform(10, 1000), 2),

"date": fake.date\_time\_this\_year()

}

sales.append(sale)

return sales

Т.к. у меня возникли проблемы с Faker, то была сделана генерация требуемых данных по принципу, представленному в примере (который мы делали на лабораторном занятии):

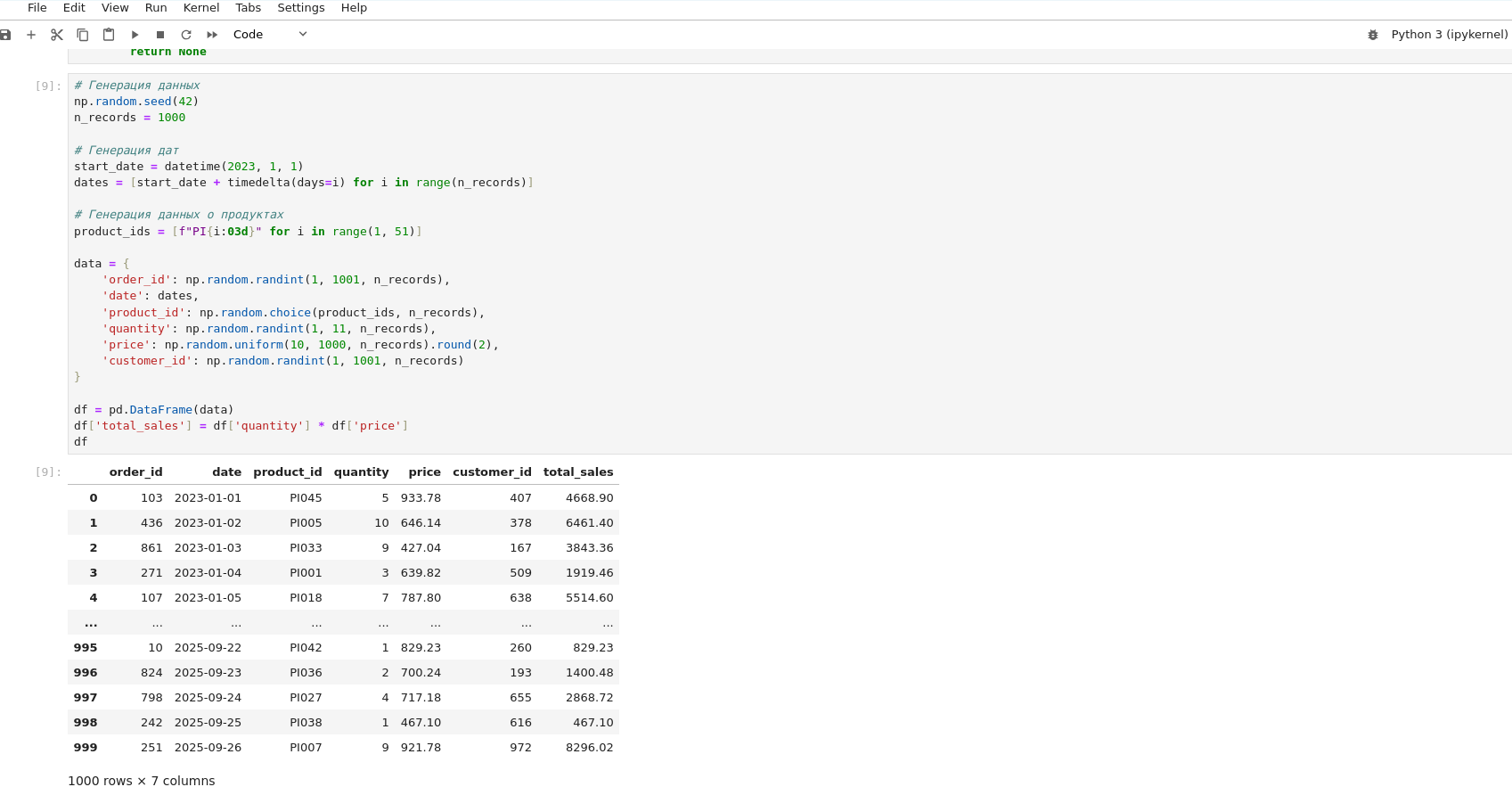


Рис. 8 – Генерация данных о продажах интернет-магазина

Полученные данные необходимо выгрузить в файл формата csv:

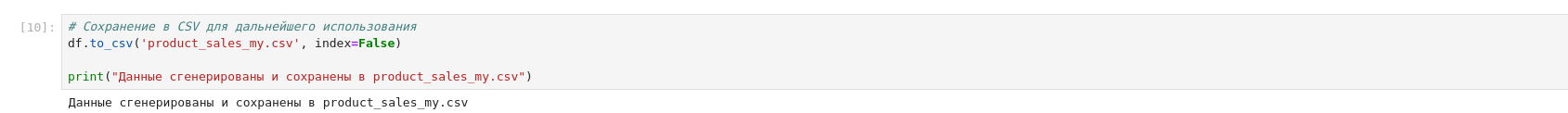


Рис. 9 – Выгрузка данных в файл «product\_sales\_my.csv»

Полученные данные о продажах интернет-магазина нужно загрузить в базу данных MongoDB, для этого подключаемся к MongoDB, загружаем данные, проверяем успешность выполнения действий, делая запрос к базе данных, результатом которого является вывод id товара, общей суммы, на которую было продано данного товара, и средней цены данного товара с фильтрацией по общей сумме (от наибольшей к наименьшей):

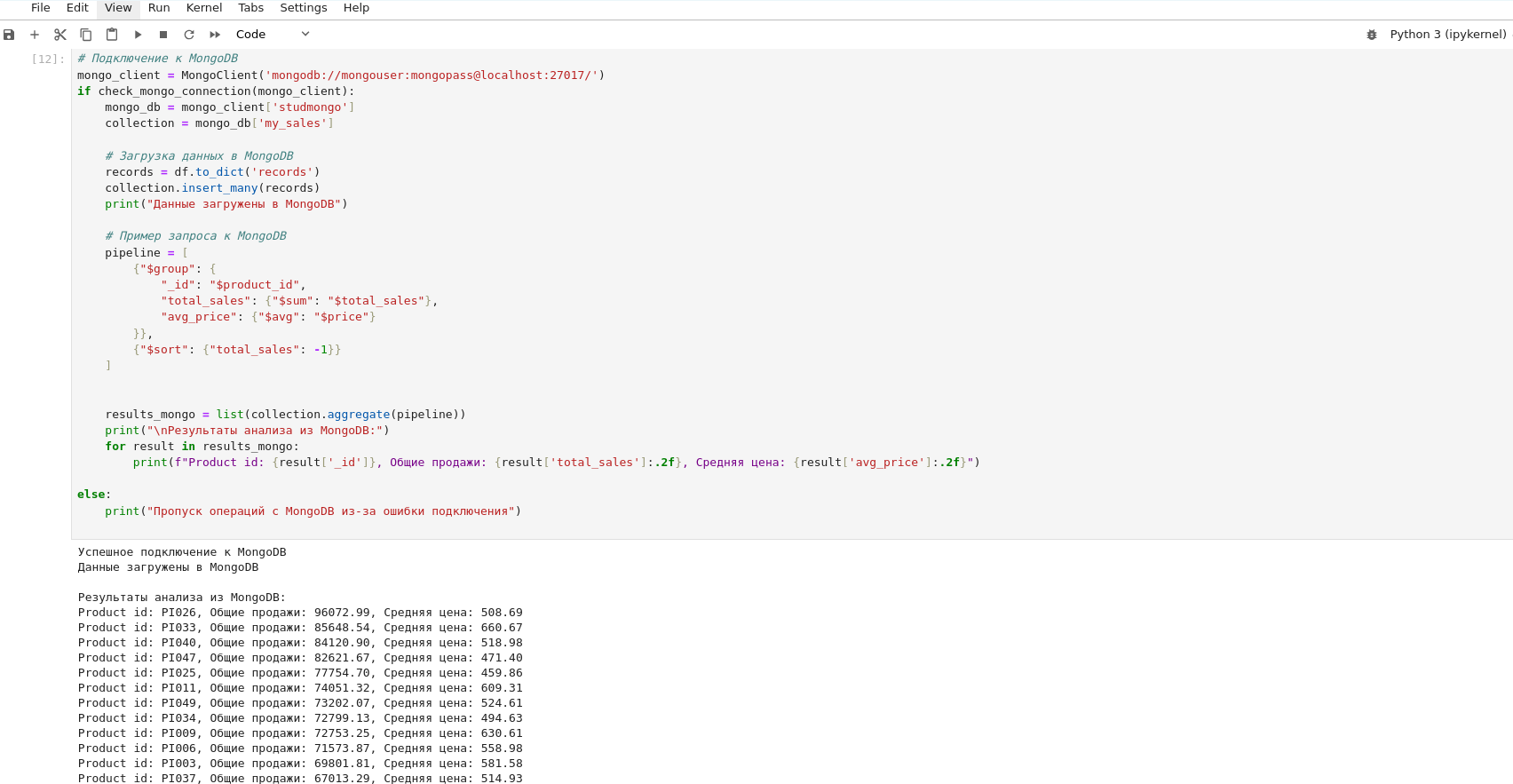


Рис. 10 – Подключение к MongoDB, загрузка данных и выполнение запроса

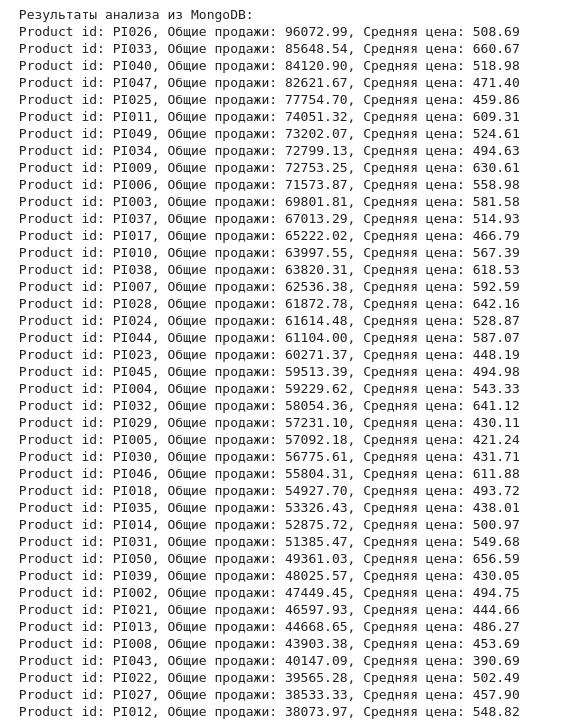


Рис. 11 – Результат анализа из MongoDB

**Шаг 3. Работа с PostgreSQL**

Первым этапом работы аналогично является подключение к PostgreSQL:

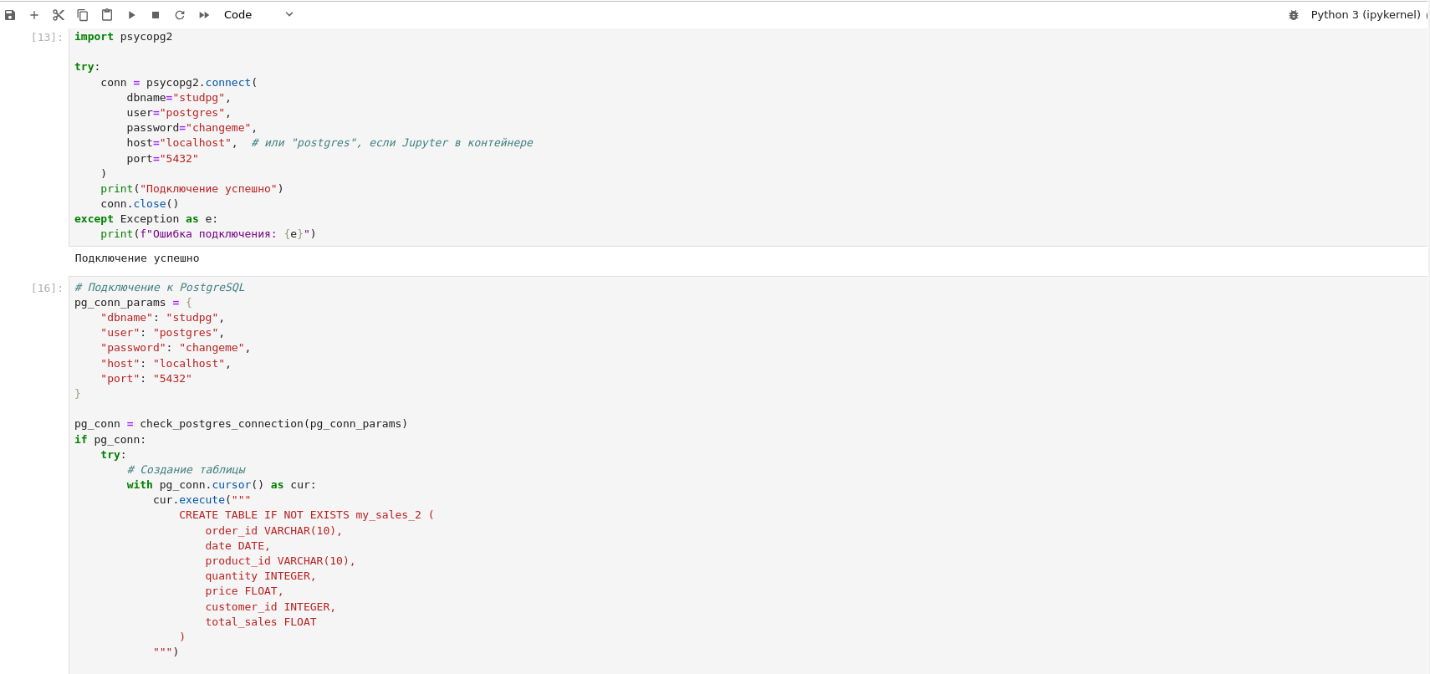


Рис. 12 – Подключение к PostgreSQL

Далее загрузим сгенерированные данные о продажах в интернет-магазине и выполним запрос, результатом которого является вывод id товара, общей суммы, на которую было продано данного товара, и средней цены данного товара с фильтрацией по id товара:

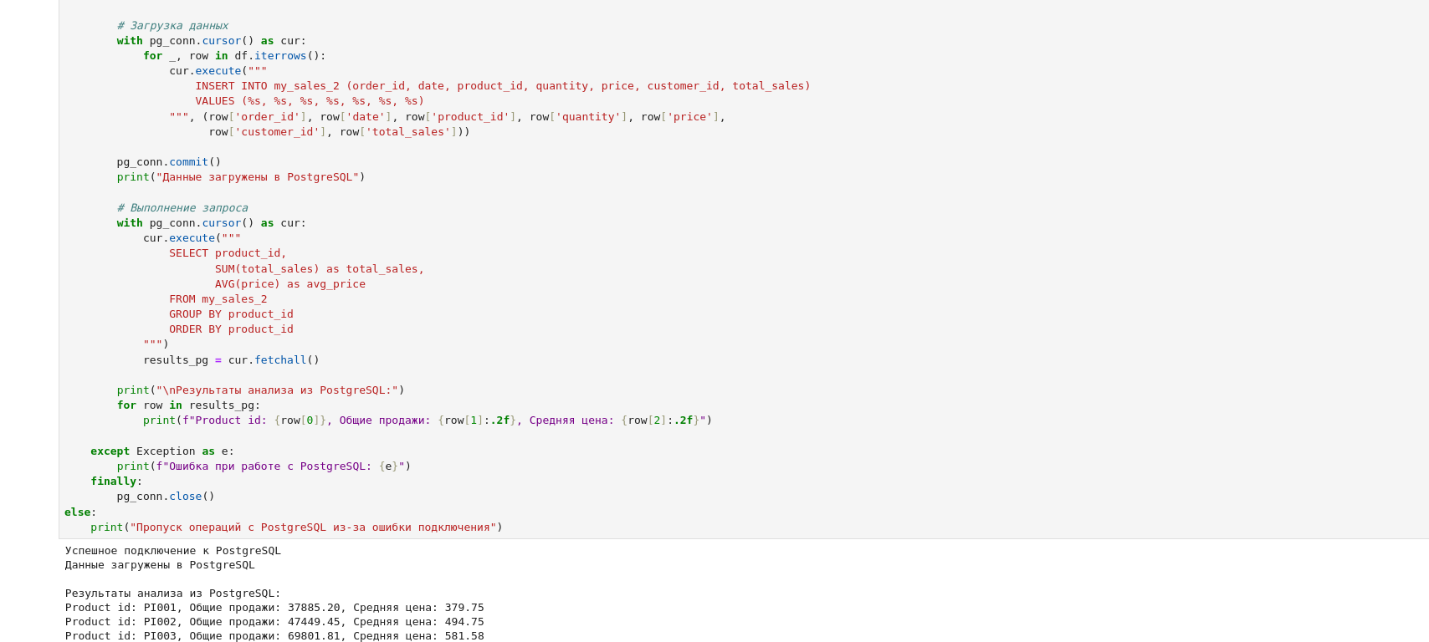


Рис. 13 – Загрузка данных в PostgreSQL, выполнение запроса

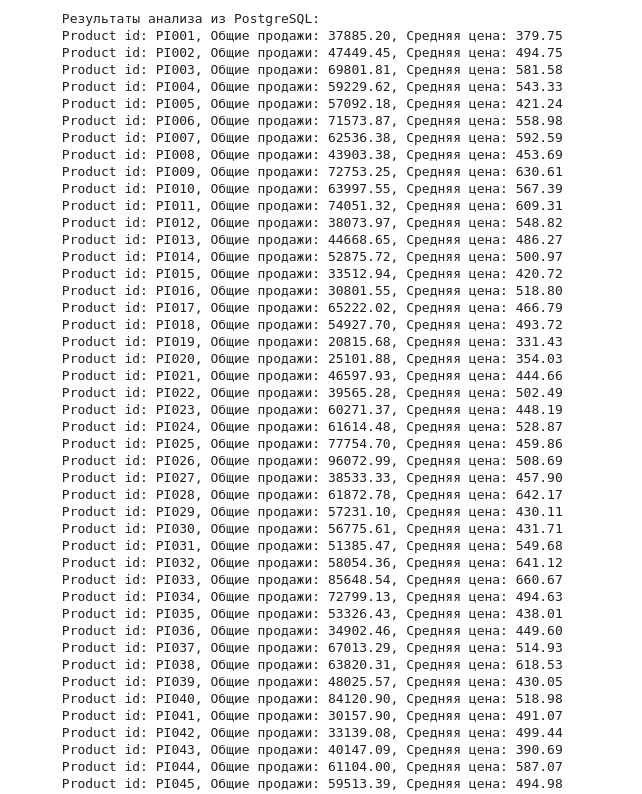


Рис. 14 – Результат анализа из PostgreSQL

**ВЫВОДЫ**

Таким образом, в ходе выполнения данной лабораторной работы:

* был отработан DevOps подход (развертывание необходимого ПО с помощью докер-контейнеров);
* была продемонстрирована работа с реляционной базой данных PostgreSQL;
* была продемонстрирована работа с базой данных NoSQL – MongoDB.

Также хочется отметить преимущества NoSQL базы данных над реляционной, т.к. все-таки она позволяет добавлять объекты нового типа без изменения схемы базы данных, позволяет работать с различными типами данных (не встраиваемыми в одну структуру и собираемыми из разных источников), а так же позволяет обрабатывать сложные запросы быстрее.