Департамент образования и науки города Москвы

Государственное автономное образовательное учреждение высшего образования города Москвы

«Московский городской педагогический университет»

Институт цифрового образования

Департамент информатики, управления и технологий

ДИСЦИПЛИНА:

Распределенные системы

**Лабораторная работа 2.1**

**Изучение методов хранения данных на основе NoSQL в**

**MongoDB, Redis, Neo4j**

Выполнил(а): Шведова С.С., группа: АДЭУ-211

Преподаватель: Босенко Т.М.

Москва

2024

**Цель работы:** изучить и освоить методы хранения и работы с данными в NoSQL базах данных MongoDB, Redis и Neo4j. Научиться загружать данные из CSV файлов в указанные СУБД и выполнять базовые операции по работе с данными.

**Оборудование и ПО:**

- Операционная система Ubuntu.

- Установленные пакеты для работы с NoSQL базами данных: MongoDB, Redis, Neo4j.

- Язык программирования Python (с библиотеками pymongo, redis, neo4j).

- CSV файлы с данными.

**Ход работы**

1. **Mongo**

Сначала надо запустить Mongo Express (это мощный административный инструмент, который предоставляет удобный способ взаимодействия с базами данных MongoDB)

Как можно заметить, Mongo Express смог подключиться к MongoDB (рисунок 1 и 2)

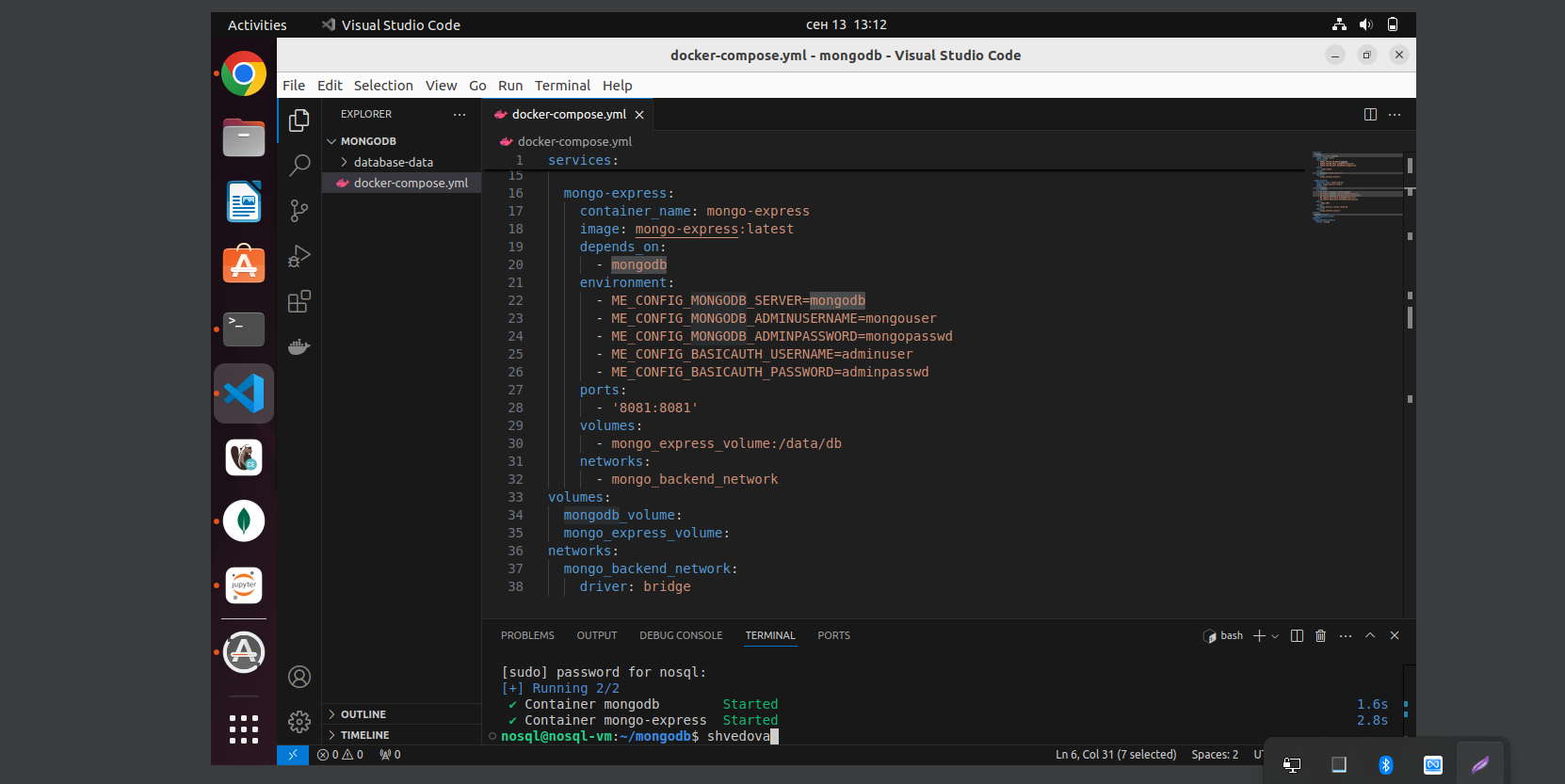


Рисунок 1. Подключение к Mongo Express

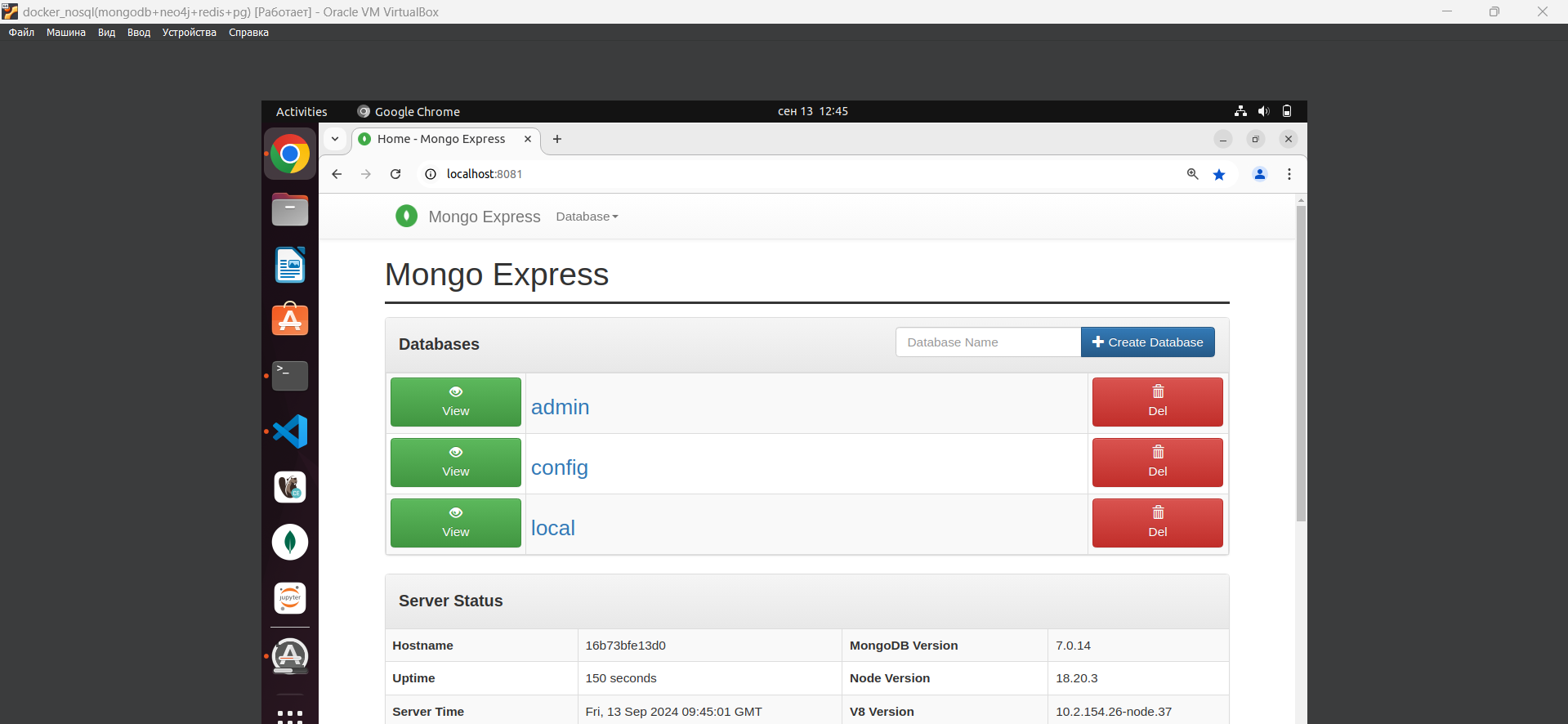


Рисунок 2. Запуск Mongo Express на localhost:8081

Далее надо зайти в MongoDB Compass (инструмент с графическим интерфейсом пользователя (GUI), разработанный для упрощения работы с базами данных MongoDB), а для этого нужно ввести логин и пароль, который был представлен ранее на рисунке 1. Как видно на рисунке 3, все успешно получилось.

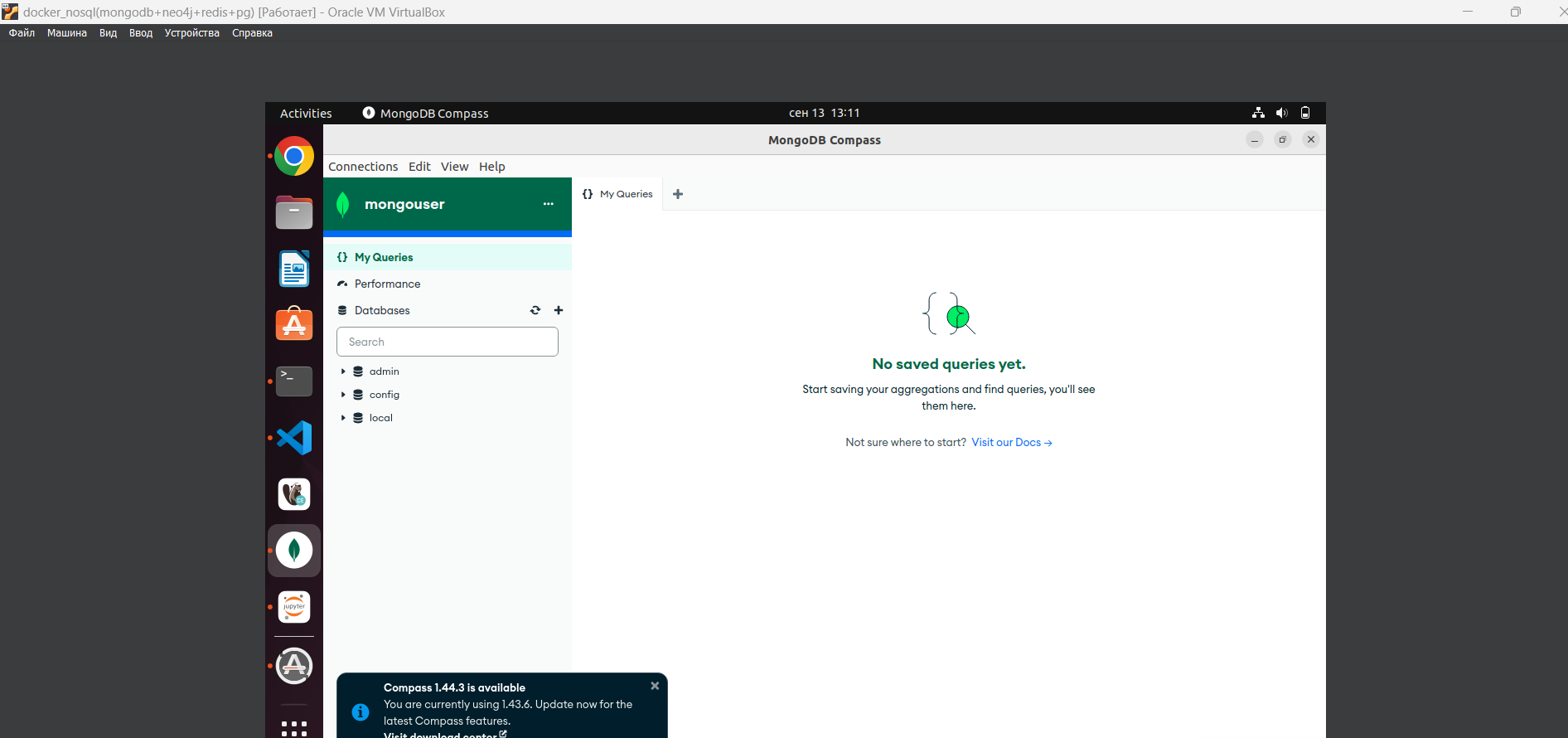


Рисунок 3. Запуск MongoDB Compass

Сначала надо сделать установку и настройку MongoDB, а для этого необходимо установить библиотеки. Сначала надо установить библиотеку pymongo в Jupiter Notebook (рисунок 4)

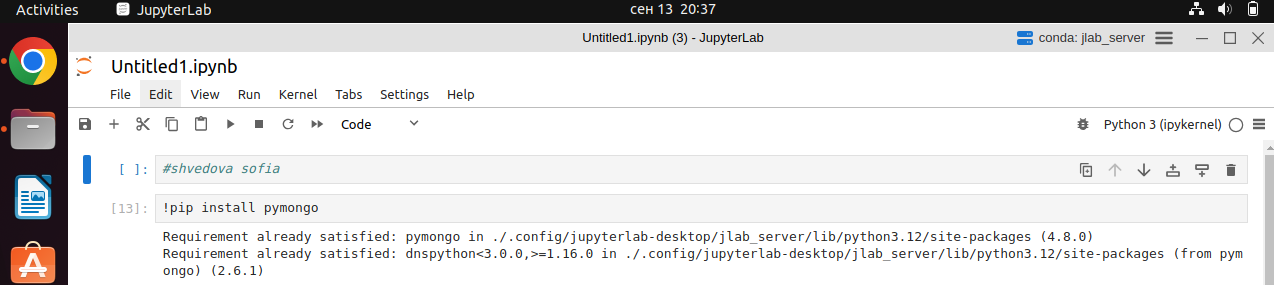


Рисунок 4. Библиотека pymongo

Далее надо подключиться к MongoDB с аутентификацией. Есть следующие параметры для подключения:

- URI MongoDB: mongodb://localhost:27017/

- Имя базы данных: labs

- Имя пользователя: mongouser

- Пароль: mongopasswd

- Имя коллекции (таблицы): lab21

Теперь нужно создать подключение и проверить его успешность. Как можно заметить, подключение прошло успешно, так как никаких ошибок нет (рисунок 5)



Рисунок 5. Подключение к MongoDB прошло успешно

Теперь надо создать текстовые данные, это представлено на рисунке 6.

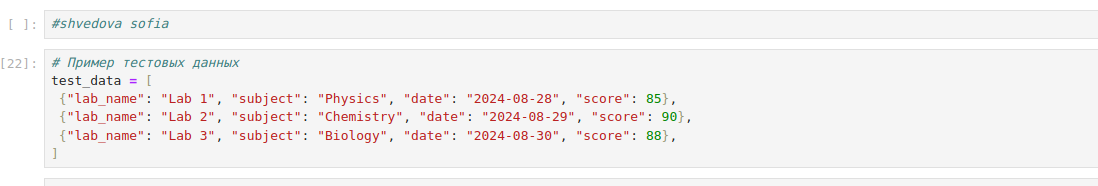


Рисунок 6. Пример текстовых данных

Далее надо загрузить созданные тестовые данные в коллекцию labs. Как можно заметить, данные успешно загружены в коллекцию, вывел выводит идентификаторы вставленных документов с помощью result.inserted\_ids и извлекает все документы из коллекции labs\_collection с помощью метода find, проходясь по каждому документу и выводя его содержимое на экран. (рисунок 7).



Рисунок 7. Загрузка созданных тестовых данные в коллекцию labs

На рисунке 8 демонстрируется закрытие соединения.

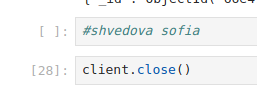


Рисунок 8. Закрытие соединения

Вывод: на рисунках выше видно, что получилось подключиться к MongoDB с использованием аутентификации, выбрать базу данных и коллекцию, выполнить операции с данными и закрыть соединение.

1. **Redis**

Redis широко применяется для кэширования, управления сеансами, разработки игр, создания таблиц лидеров, аналитики в режиме реального времени, работы с геопространственными данными, поддержки служб такси, чатов и сервисов обмена сообщениями, потоковой передачи мультимедиа и приложений с отправкой сообщений по модели «издатель – подписчик» (Pub/Sub).

Redis Commander - веб-приложение node.js, которое используется для просмотра, редактирования и управления базой данных Redis.

Для начала надо установить соединение с Redis (рисунок 9 и 10)

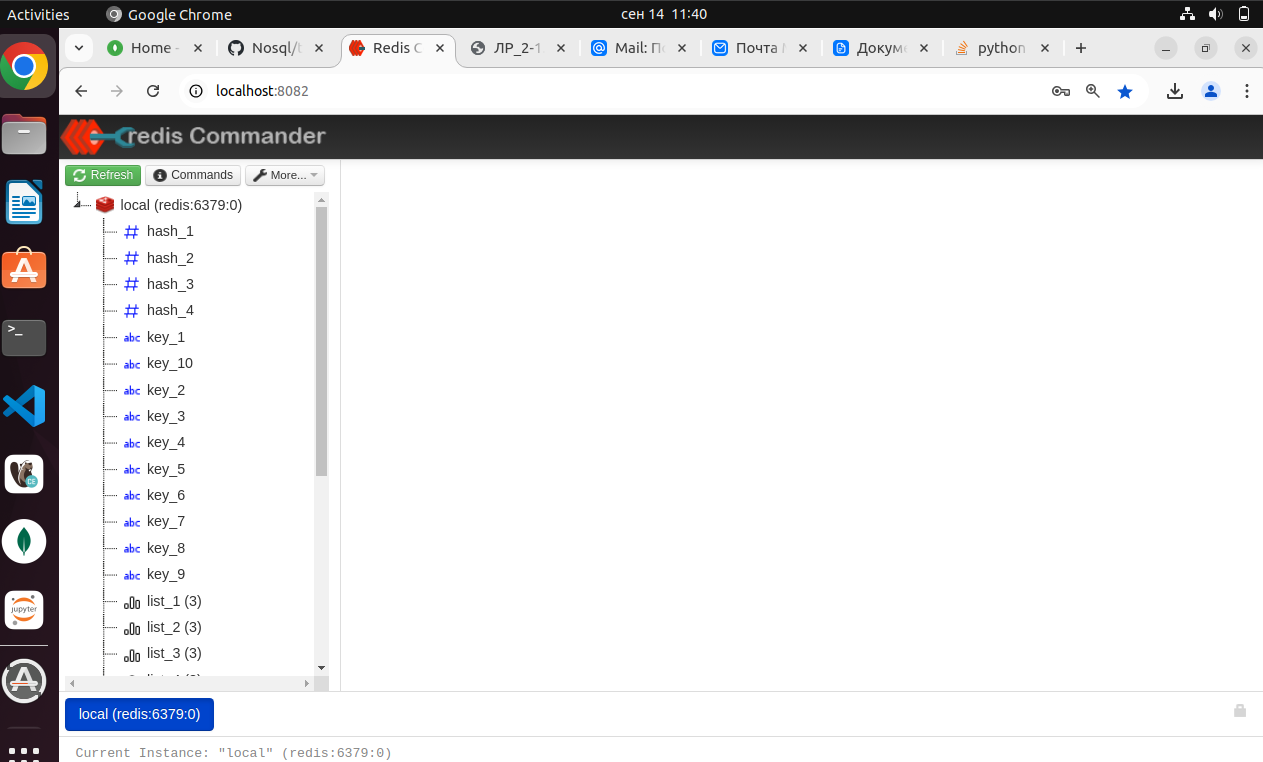


Рисунок 9. Включение Redis Commander



Рисунок 10. Подключение с Redis

Генерация и загрузка данных в Redis. Можно сгенерировать десять записей и загрузить их в базу данных Redis. (рисунок 11)

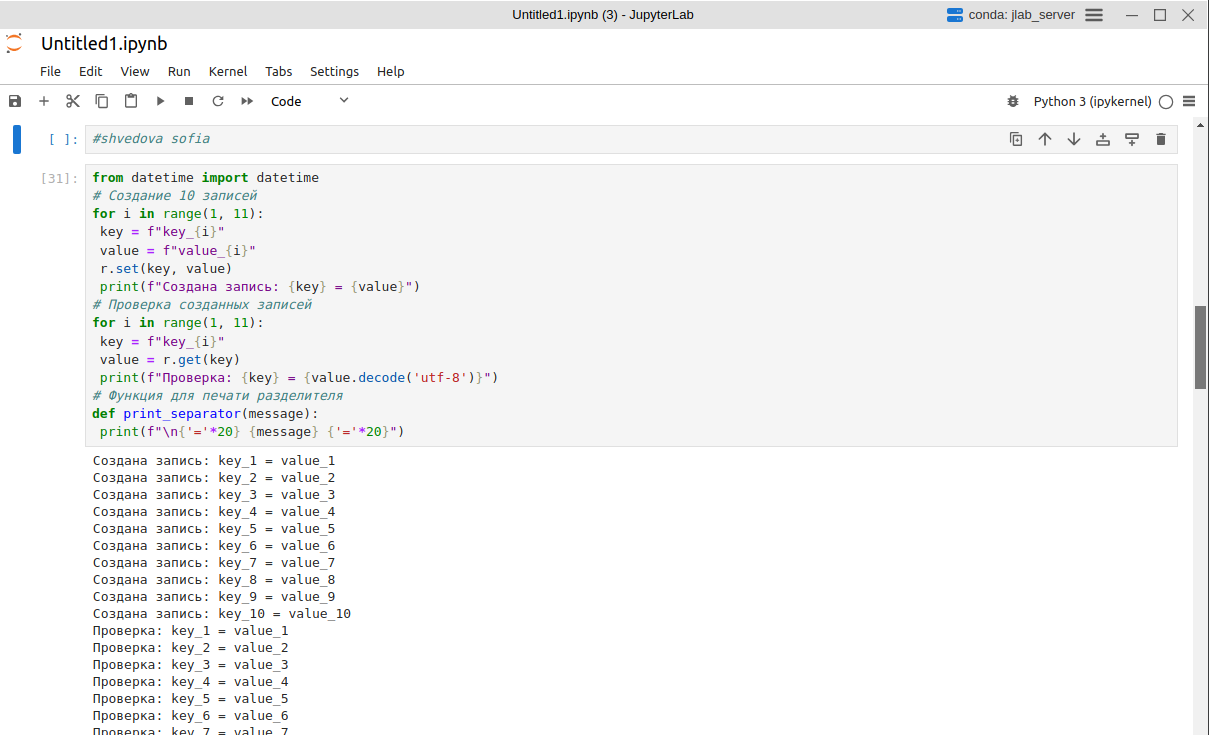


Рисунок 11. Генерация 10 записей

Создание по 5 записей различных типов данных представлено на рисунке 12.

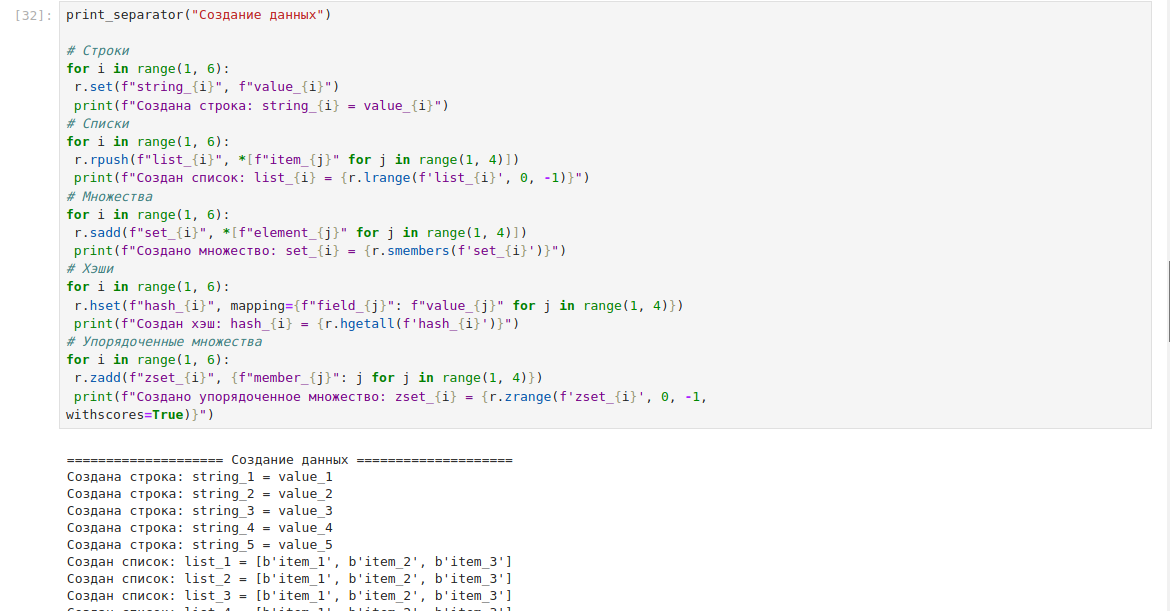


Рисунок 12. Создание по 5 записей различных типов данных

Получение данных по ключу и обновление данных по ключу на рисунке 12.



Рисунок 12. Получение данных по ключу и обновление данных

Удаление данных по ключу представлено на рисунке 13.



Рисунок 13. Удаление данных по ключу

Выгрузка всех данные из Redis в csv на рисунках 14 и 15.



Рисунок 14. Выгрузка всех данных



Рисунок 15. Выгрузка всех данных и сохранение в redis\_dump.csv

**Вывод:**

CSV формат ограничен в представлении сложных структур данных. Для списков, множеств, хешей и упорядоченных множеств мы преобразуем их в JSON-строки. Это позволяет сохранить всю информацию, но может затруднить дальнейшую обработку данных в CSV. Каждая строка в CSV будет содержать три столбца:

- Key: ключ из Redis.

- Type: тип данных (string, list, set, hash, zset).

- Value: значение, преобразованное в строку или JSON.

**Индивидуальное задание**

Был выбран csv файл с режиссерами и фильмами, где 105 строк и 2 колонки. Сначала надо загрузить его (рисунок 16)

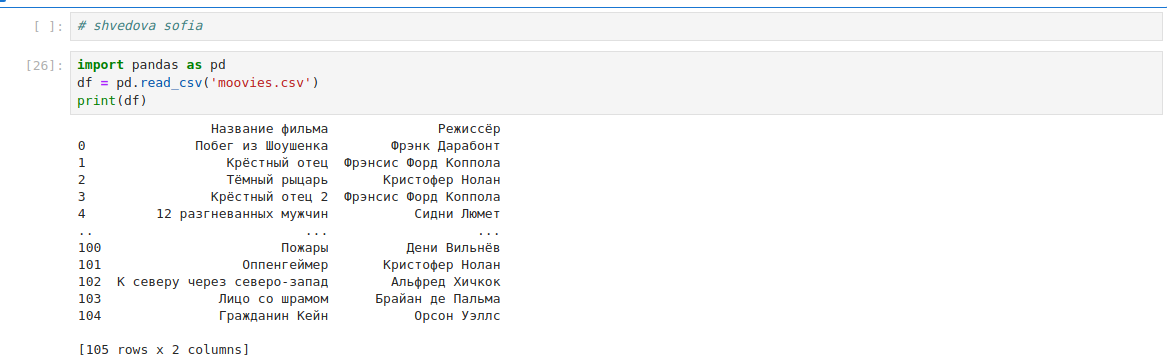


Рисунок 16. Загрузка файла moovies.csv

Далее идет вставка и выборка данных (рисунок 17).



Рисунок 17. Вставка и выборка данных

Как можно увидеть, данные в коллекции присутствуют (рисунок 18)

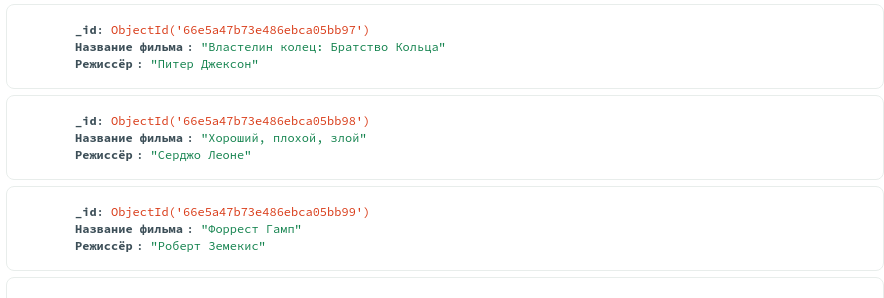


Рисунок 18. MongoDB

Затем уже происходит обновление данных, которое успешно завершено (рисунок 19-20)

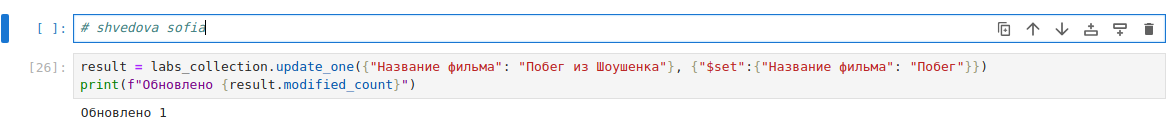


Рисунок 19. Код для обновления данных

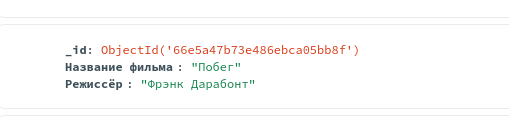


Рисунок 20. Данные обновлены в MongoDB

Теперь нужно удалить запись, это показано на рисунке 21.

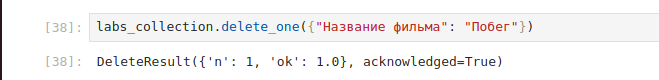


Рисунок 21. Удаление записи

В Redis надо тоже подключиться, после этого надо сделать вставку данных, выборку, обновление и удаление данных (рисунок 22).

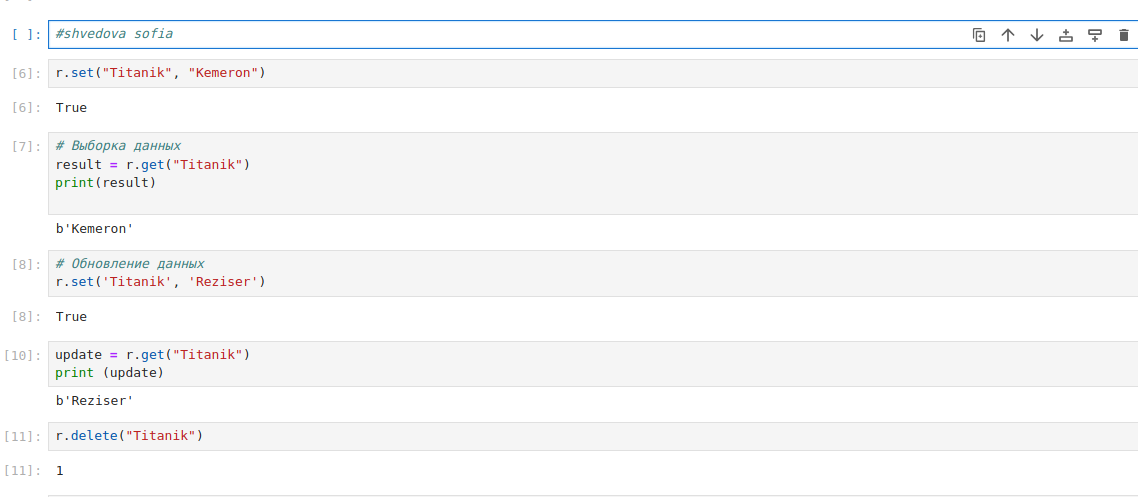


Рисунок 22. Вставка данных, выборка, обновление и удаление данных

**Neo4j**

Neo4j — графовая система управления базами данных с открытым исходным кодом, реализованная на Java.

Для начала надо загрузить исходные данные в эту СУБД (рисунок 23)

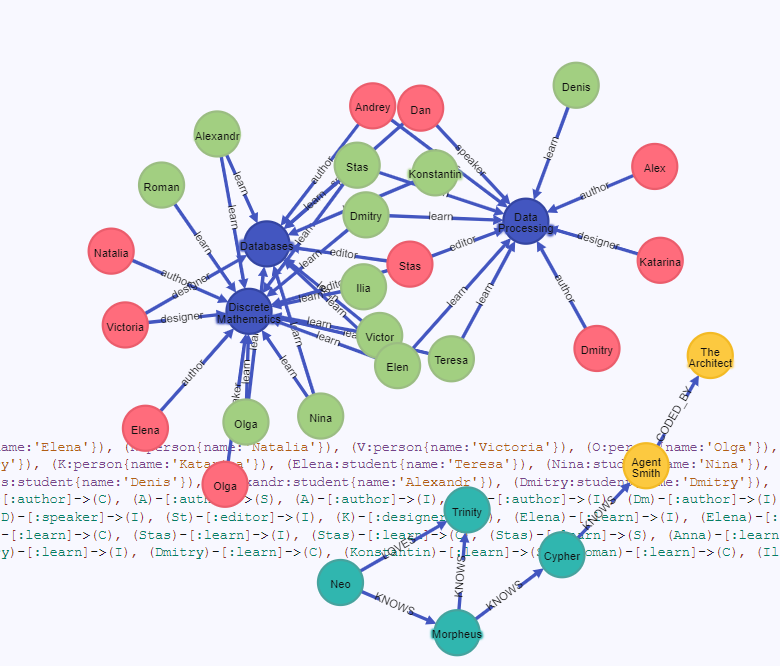


Рисунок 23. Загрузка данных в СУБД Neo4j

Далее нужно Найдите всех студентов, которые не записаны ни на один курс с помощью кода

MATCH (s:student)

WHERE NOT (s)-[:enrolled\_in]->()

RETURN s

Этот запрос найдет всех студентов, у которых отсутствуют отношения "enrolled\_in" с каким-либо курсом

Как можно заметить на рисунке 24, студенты которые не записаны ни на один курс:

(19:student {name:"Teresa"})

(20:student {name:"Nina"})

(21:student {name:"Victor"})

(22:student {name:"Olga"})

(23:student {name:"Stas"})

(24:student {name:"Elen"})

(25:student {name:"Denis"})

(26:student {name:"Alexandr"})

(27:student {name:"Dmitry"})

(28:student {name:"Konstantin"})

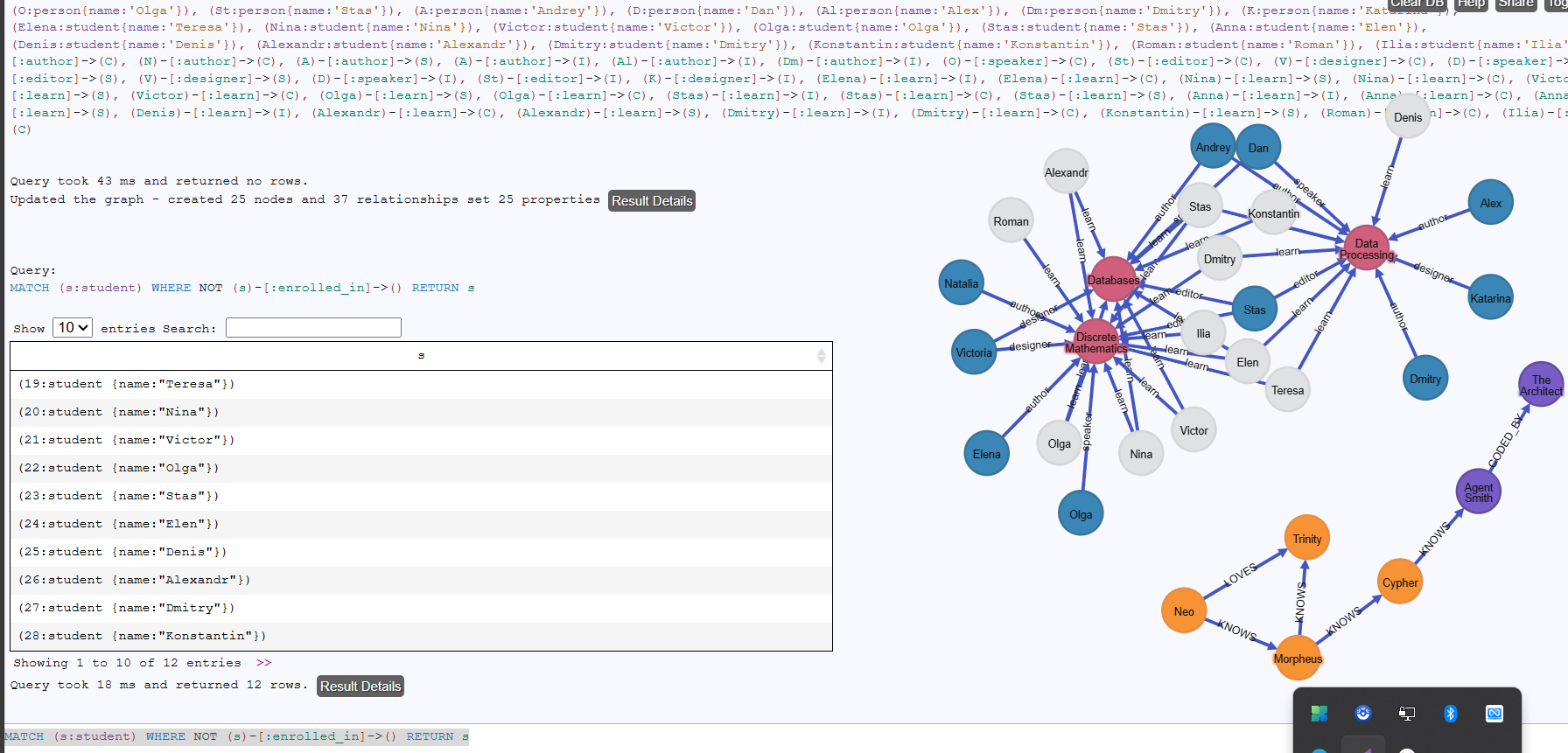


Рисунок 24. Вывод запроса

**Вывод:**

Neo4j позволяет решать следующие задачи на графах:

1. поиск оптимального пути в логистике, информационных или электротехнических сетях;
2. выявление сообществ;
3. анализ социальных связей.

Система поддерживает алгоритмы работы со графами, включая определение меры центральности и степени связанности, алгоритм Дейкстры для нахождения оптимального пути и другие инструменты решения графовых задач.