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:

Инструменты для хранения и обработки больших данных

**Лабораторная работа №2.1**

**Тема:**

«Изучение методов хранения данных на основе NoSQL в

MongoDB, Redis, Neo4j»

Выполнил: Егоров А. С., группа: АДЭУ-211

Преподаватель: Босенко Т. М.

Москва

2024

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc179460033)

[ТЕОРЕТИЧЕСКАЯ ЧАСТЬ 4](#_Toc179460034)

[MongoDB: документо-ориентированная NoSQL база данных, где данные хранятся в формате JSON-подобных документов 4](#_Toc179460035)

[Redis: высокопроизводительная база данных типа "ключ-значение", часто используемая для кеширования и временного хранения данных. 5](#_Toc179460036)

[Neo4j: графовая база данных, которая позволяет хранить данные в виде вершин и рёбер графа, что удобно для моделирования сложных взаимосвязей. 6](#_Toc179460037)

[ХОД РАБОТЫ 7](#_Toc179460038)

[Шаг 1. Установка и настройка MongoDB в Jupyter Hub 7](#_Toc179460039)

[Шаг 2. Установка и настройка Redis 10](#_Toc179460040)

[Шаг 3. Запрос, чтобы получить список всех сотрудников, которые участвуют в создании хотя бы одного курса в роли автора, диктора или редактора на Neo4j. 18](#_Toc179460041)

[ЗАКЛЮЧЕНИЕ 20](#_Toc179460042)

# ВВЕДЕНИЕ

**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 файл с данными.

# ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

## MongoDB: документо-ориентированная NoSQL база данных, где данные хранятся в формате JSON-подобных документов

MongoDB — документоориентированная система управления базами данных, не требующая описания схемы таблиц. Считается одним из классических примеров NoSQL-систем, использует JSON-подобные документы и схему базы данных.

Основные принципы работы:

• вместо таблиц MongoDB использует коллекции, где хранятся документы — аналоги строк в реляционной БД;

• MongoDB хранит данные в виде JSON-документов с различным уровнем глубины, кодируя их в бинарном формате BSON;

• разные коллекции могут иметь разные структуры, причём даже в рамках одной коллекции у документов может различаться набор полей и их типы данных;

• будучи NoSQL-хранилищем, MongoDB не поддерживает классические SQL-запросы и по-своему реализует некоторые операции.

MongoDB подходит для следующих применений: регистрация и хранение информации о событиях, системы управления документами и контентом, электронная коммерция, игры, данные мониторинга, датчиков, мобильные приложения, хранилище операционных данных веб-страниц (например, хранение комментариев, рейтингов, профилей пользователей, сеансы пользователей).

Система написана на языке программирования C++, что позволяет портировать её на различные платформы, включая Windows, Linux, MacOS, Solaris.

## Redis: высокопроизводительная база данных типа "ключ-значение", часто используемая для кеширования и временного хранения данных.

Redis — это высокопроизводительная база данных типа «ключ-значение», которая используется в качестве базы данных, кэша, брокера сообщений и очереди.

Некоторые особенности Redis:

* Высокая производительность. Данные хранятся в оперативной памяти сервера, что значительно ускоряет работу с ними.
* Персистентность. Есть возможность сохранить снимки базы данных на диск в зависимости от количества обновлённых значений. Также есть режим AOF — сохранение на диск журнала операций с возможностью восстановить из него данные при следующем запуске.
* Удобство. Есть встроенная поддержка широко используемых структур данных (строки, списки, хеши, множества, сортированные множества).
* Масштабируемость. Поддерживается шардирование, репликация, отказоустойчивость и другие возможности распределённых систем.

Redis часто используется для кеширования результатов запросов к базам данных, длительных сессий, веб-страниц и часто используемых объектов, таких как изображения, файлы и метаданные. Например, можно временно хранить такие пользовательские данные, как товары в корзине покупок интернет-магазина.

Также Redis применяют для хранения промежуточных данных (поток сообщений на стене, голосовалки, таблицы результатов), как брокер сообщений (стратегия «издатель-подписчик» позволяет создавать новостные ленты, групповые чаты) и для хранения «быстрых» данных — когда важны скорость и критичны задержки передачи (аналитика и анализ данных, финансовые и торговые сервисы).

Neo4j: графовая база данных, которая позволяет хранить данные в виде вершин и рёбер графа, что удобно для моделирования сложных взаимосвязей. Neo4j — графовая база данных, которая позволяет хранить данные в виде вершин и рёбер графа, что удобно для моделирования сложных взаимосвязей.

Граф Neo4j состоит из четырёх компонентов:

* Узлы (Nodes) — вершины графа, которые представляют объект или сущности. Их можно промаркировать.
* Отношения (Relationships) — связи между любыми двумя узлами. Отношения имеют тип и направление.
* Метки или ярлыки (Labels), которые маркируют категорию узлов.
* Свойства (Properties) — конкретная информация каждого узла и отношения.

База данных Neo4j позволяет исследовать различные пути и связи между данными и максимально эффективно запрашивать их. Кроме того, можно легко извлекать сложные данные из базы данных, даже если они сильно связаны.

Для базы данных Neo4j создан декларативный язык запросов Cypher, обеспечивающий эффективное чтение и запись данных.

# ХОД РАБОТЫ

## Шаг 1. Установка и настройка MongoDB в Jupyter Hub

Запускаем Jupyter Hub. В интерфейсе проектов выбираем MongoDB\_Redis.ipynb. Необходимо установить библиотеку pymongo (!pip install pymongo). Далее нужно подключиться к MongoDB с аутентификацией. Создадим подключение и проверим его успешность.

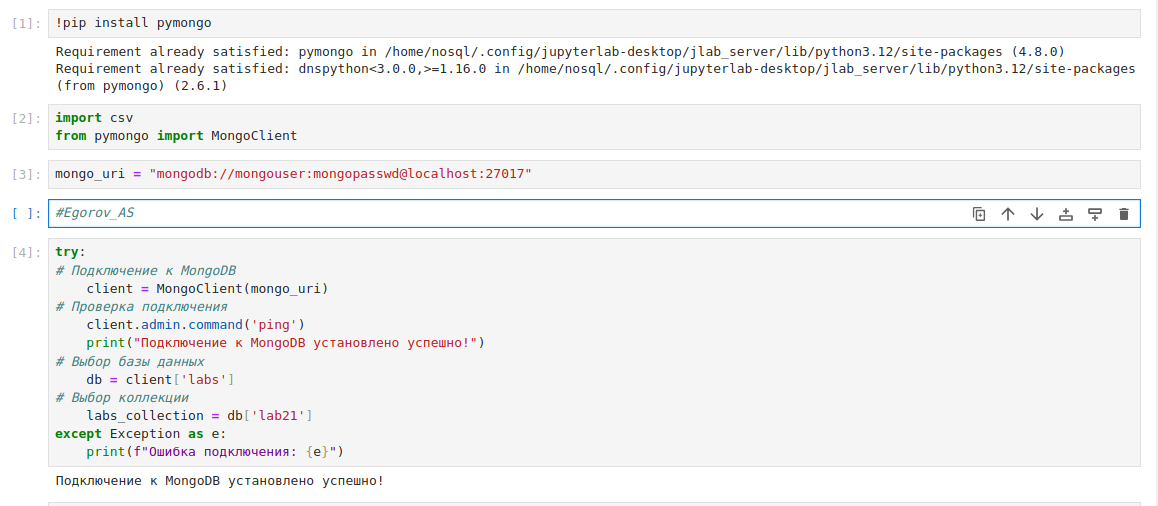
****

Рисунок 1 – Подключение к MongoDB

Для выполнения индивидуального задания были сгенерированы 120 записей (Курсы и преподаватели в университете) по 4 варианту, хранящие информацию об:

- ID курса

- Название курса

- ID преподавателя

- Имя преподавателя

- Фамилия преподавателя

- Факультет

- Семестр

- Год

- Аудитория

Присвоим сгенерированным 120 записям значение переменной test\_data.



Рисунок 2 – Присвоим значения переменной

Загрузим данные из переменной test\_data в коллекцию labs.

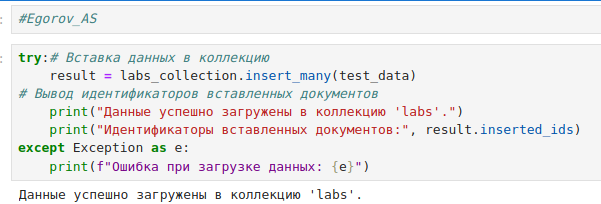


Рисунок 3 – Загрузка данных в labs

Проверим в MongoDB Compass корректность загрузки данных.

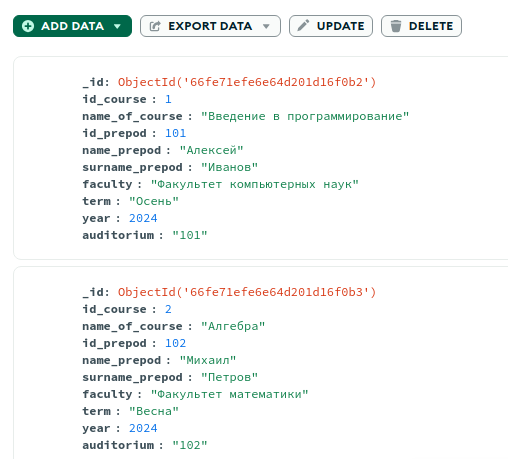


Рисунок 4 – Проверка загрузки данных в MongoDB

Обновим данные. Предположим, что преподаватель с id ‘103’ Ольга Сидорова сменила после свадьбы фамилию на ‘Столбова’.

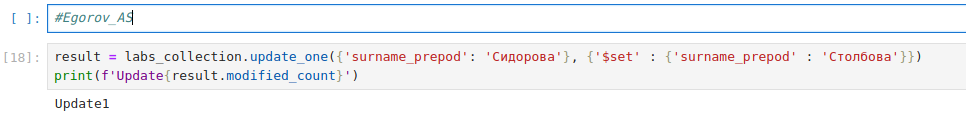


Рисунок 5 – Обновление фамилии преподавателя

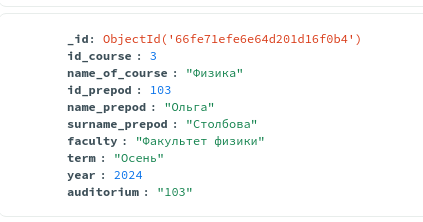
****

Рисунок 6 – Проверка изменения фамилии

Предположим, что преподаватель ‘Столбова’, которая недавно сменила фамилию, решила больше не вести свой курс, тогда удалим ее запись из БД.

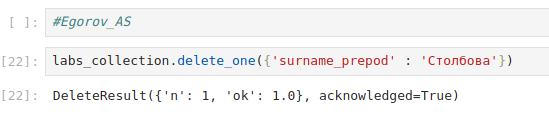


Рисунок 7 – Удаление преподавателя

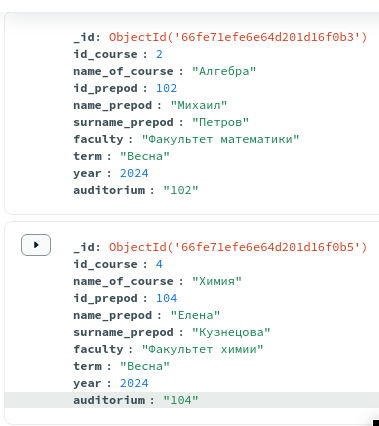


Рисунок 8 – Проверка удаления

Как видно на рисунке выше, отсутствует запись о преподавателе с фамилией Столбова, у которой был id\_prepod = 103.

Закроем соединение с помощью команды: client.close()

**Вывод к 1 шагу**: была установлена библиотека pymongo, пройдена аутентификация и создано соединение. Сгенерированные записи по варианту 4 были загружены в MongoDB. Далее была продемонстрирована возможность менять и удалять записи в MongoDB и закрыто соединение.

## Шаг 2. Установка и настройка Redis

Необходимо установить библиотеку redis (!pip install redis). Далее нужно импортировать другие необходимые библиотеки и подключиться к Redis с аутентификацией. Создадим подключение и проверим его успешность.



Рисунок 9 – Подключение к Redis

Сгенерируем десять записей и загрузим их в базу данных Redis.

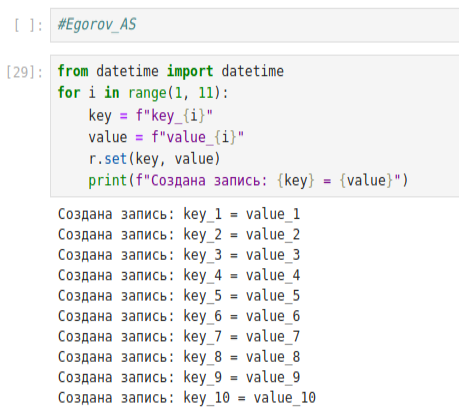
****

Рисунок 10 – Генерация 10 записей

Проверка созданных записей



Рисунок 11 – Выполнение проверки генерации

Создание по 5 записей различных типов данных.

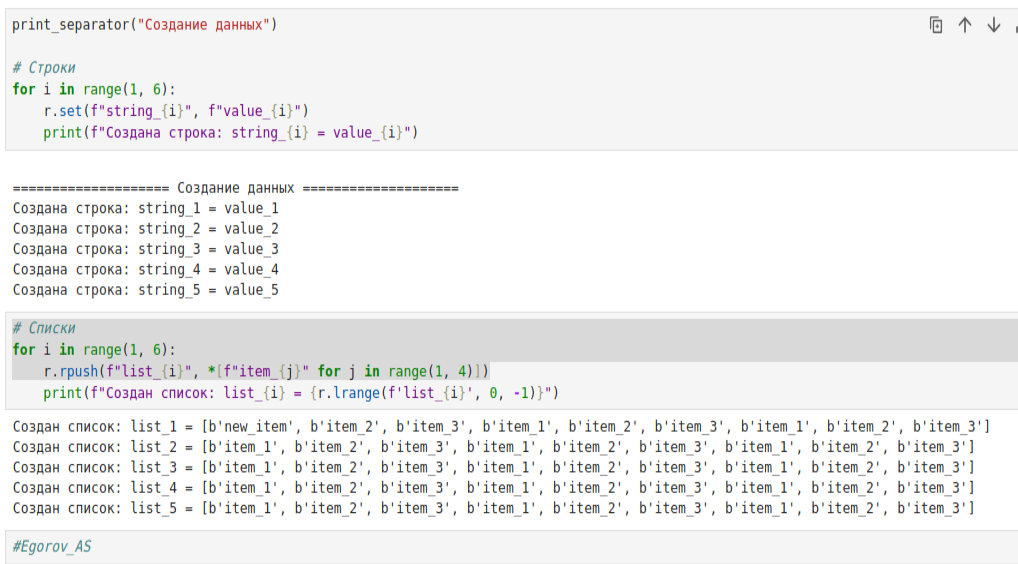


Рисунок 12 – Создание по 5 записей различных типов данных

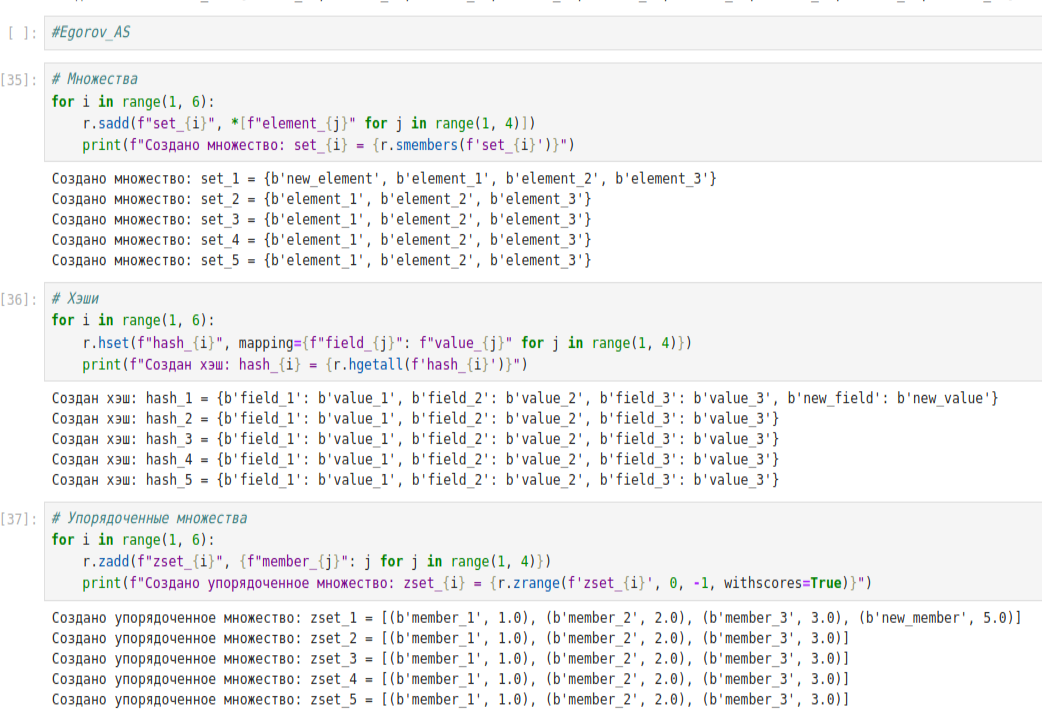


Рисунок 13 – Создание по 5 записей различных типов данных продолжение

Получим и обновим данные по ключу.



Рисунок 14 – Получение данных по ключу

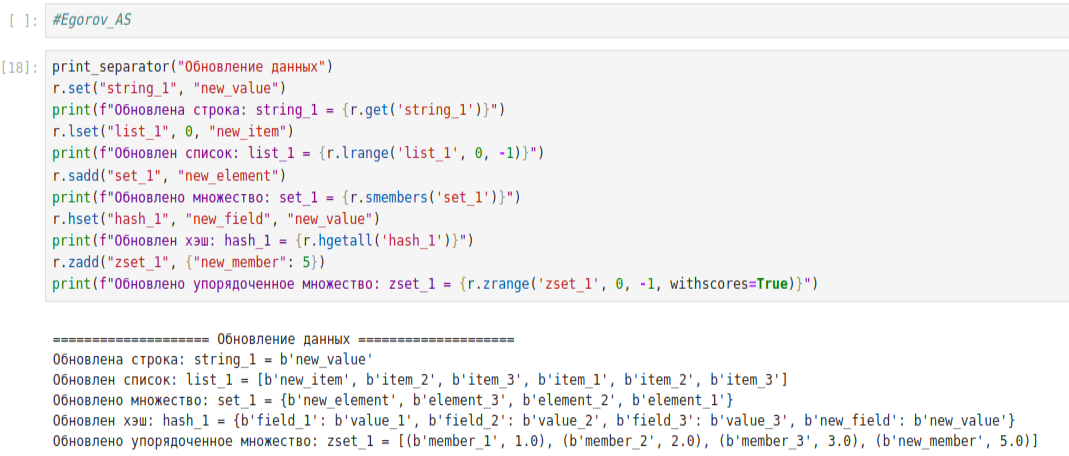


Рисунок 15 – Обновление данных по ключу

Удалим данные по ключу и проверим операцию.



Рисунок 16 – Удаление данных по ключу и запуск проверки

Выгрузить все данные из Redis в csv.



Рисунок 17 – Выгрузка всех данных и сохранение в redis\_dump.csv



Рисунок 18 – Выгрузка всех данных и сохранение в redis\_dump.csv продолжение

**Перейдем к индивидуальному заданию**. Преобразуем созданные 120 записей в csv файл с названием ‘courses.csv’. Прочитаем файл с курсами с помощью pandas.

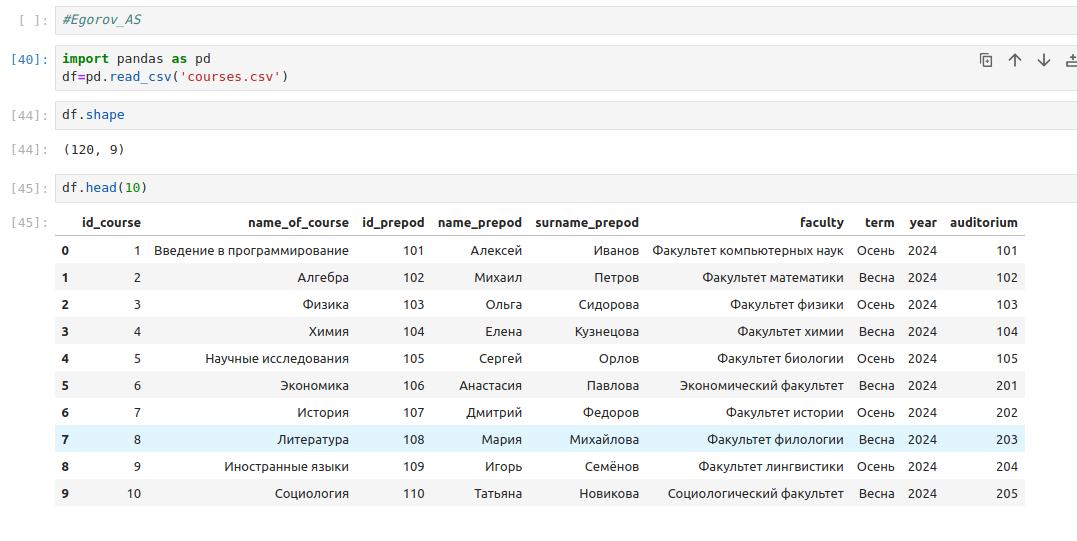


Рисунок 19 – Загрузка данных формата CSV

Загрузим ключи и их значения в Redis.

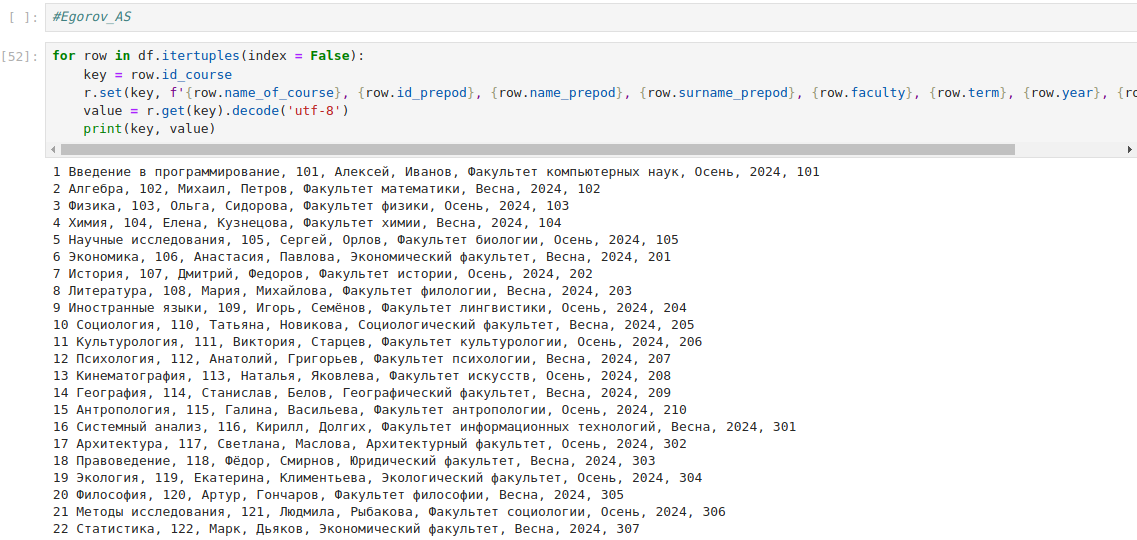


Рисунок 20 – Загрузка данных в Redis

Просмотрим результат загрузки в Redis.

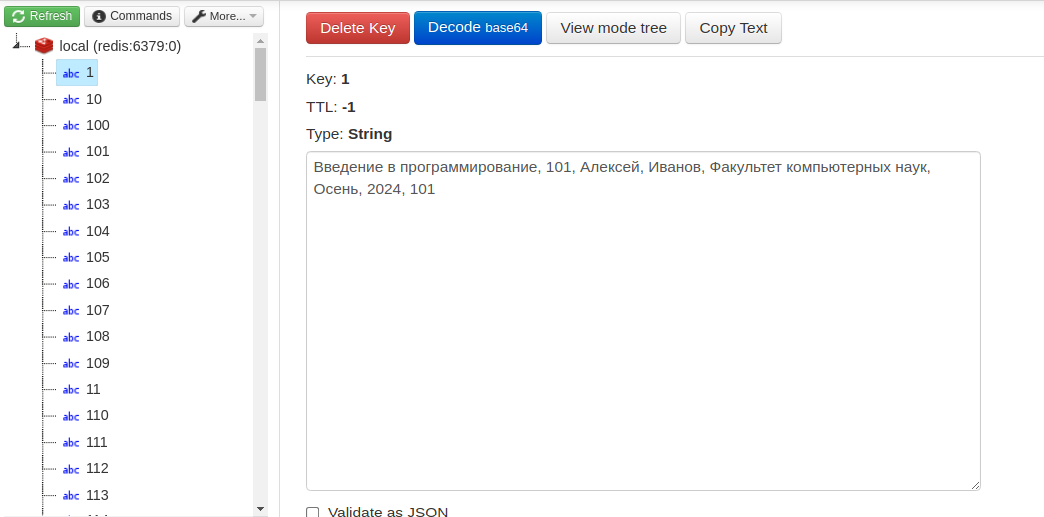


Рисунок 21 – Загруженные данные в Redis

Для id\_course = 1 изменим аудиторию со 101 на 222.

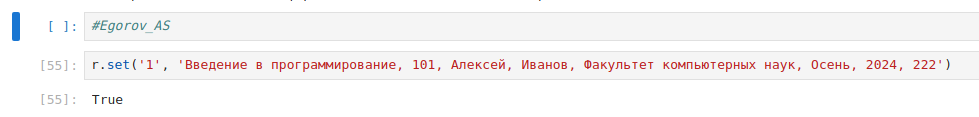


Рисунок 22 – Изменение значения в данных

Проверим результат исполнения команды. На рисунке ниже видно изменение аудитории. На рисунке 16 видны первоначальное значение аудитории.

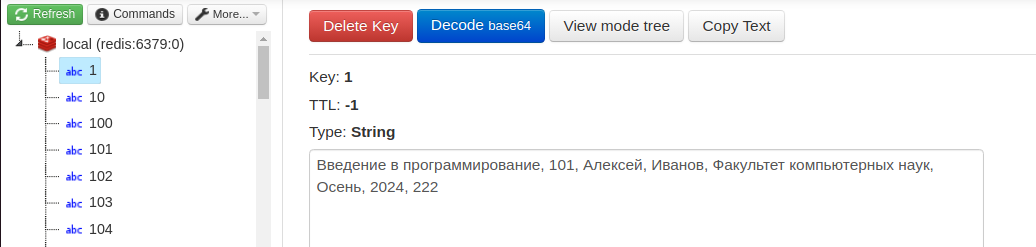


Рисунок 23 – Проверка исполнения команды в Redis

Удалим строку с id\_course = 1 и проверим в Pyton



Рисунок 24 – Удаление данных

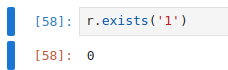


Рисунок 25 – Проверка удаления данных в Pyton

Проверим удаление данных в Redis.

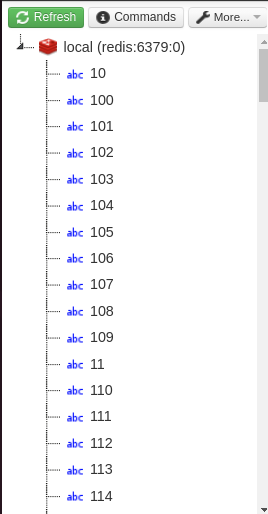


Рисунок 26 – Проверка удаления данных в Redis

**Вывод по шагу 2:** была установлена библиотека redis, пройдена аутентификация и создано соединение. Продемонстрировано решение примеров с генерацией данных, создание записей различных типов, обновление и удаление данных и выгрузка их в csv. Также было выполнено индивидуальное задание с данными из варианта 4, в котором были продемонстрированы загрузка, обновление и удаление данных.

## Шаг 3. Запрос, чтобы получить список всех сотрудников, которые участвуют в создании хотя бы одного курса в роли автора, диктора или редактора на Neo4j.

Данный шаг выполнен с помощью “песочницы”: <http://console.neo4j.org> в контексте использования базы «Учебные курсы», которая на языке Cypher. Загрузим данные.

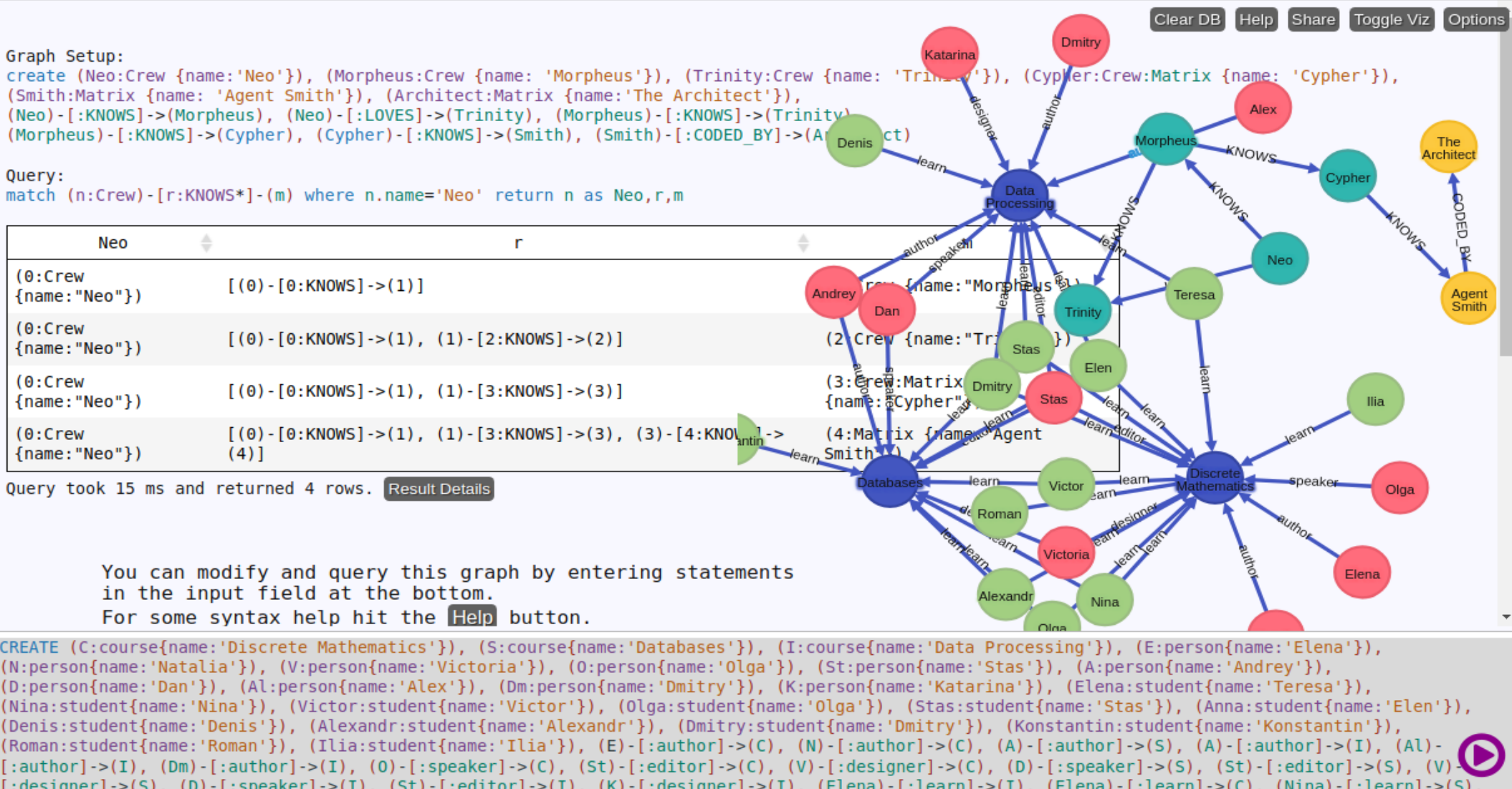


Рисунок 27 – Загрузка данных по курсам, студентам и сотрудникам

Необходимо написать запрос, который вернет список курсов, созданных сотрудником с именем "Andrey".

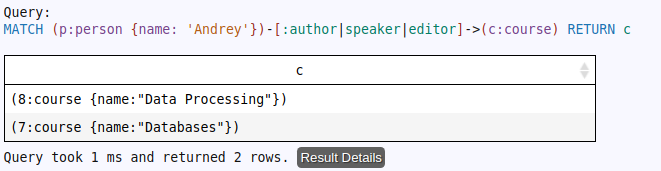


Рисунок 28 – Сам запрос и отображение его результата

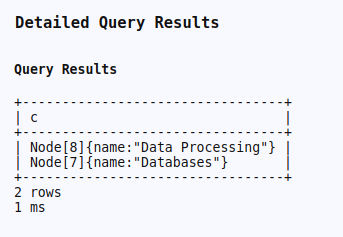


Рисунок 29 – Детальный отчет по результату

**Вывод по шагу 3:** была загружена представленная в задании база и решен запрос 4 из списка индивидуальных заданий.

# ЗАКЛЮЧЕНИЕ

Использование NoSQL-систем предоставляет разработчикам аналог храпения данных. Выбор конкретной системы зависит от специфики проекта и требований к данным, что позволяет создать оптимальное решение для каждого случая.

MongoDB ориентированна на документо-ориентированный подход и обеспечивает гибкость при работе с полуструктурированными данными, позволяя легко масштабировать приложения и менять схему данных. Это делает ее идеальной для проектов, в которых часто меняются требования.

Redis отлично подходит для сценариев, требующих быстрой обработки данных, таких как кэширование, обмен сообщениями и сеансовое хранение. Его простота и скорость позволяют создавать высоконагруженные приложения с минимальными задержками.

Neo4j является графовой базой данных, которая полезна в работе со связанными данными и сложными запросами. Это делает ее особенно полезной для социальных сетей, рекомендационных систем и анализа данных, где важна структура связи между объектами.