Департамент образования города Москвы

Государственное автономное образовательное учреждение

высшего образования города Москвы

«Московский городской педагогический университет»

Институт цифрового образования

Департамент информатики, управления и технологий

Семеняченко Данил Юрьевич

# Лабораторная работа по «Инструменты для хранения и обработки больших данных»

Выполнил:

Семеняченко Данил Юрьевич

Проверил:

Босенко Тимур Муртазович

Курс обучения: 3

Форма обучения: очная

Москва

2025

# **Лабораторная работа 2.1. Изучение методов хранения данных на основе NoSQL**

**Цель работы:** изучение и практическое применение трех различных типов NoSQL баз данных, а именно: документо-ориентированной MongoDB, графовой Neo4j и ключ-значение Redis.

**Оборудование и программное обеспечение:**

* Операционная система Ubuntu
* Язык программирования Python (с библиотеками pymongo, redis, neo4j). CSV файлы с данными.

# Краткая теоретическая справка

NoSQL — это подход к проектированию баз данных, которые не являются реляционными и могут использовать различные модели данных. Они оптимизированы для конкретных сценариев использования и обеспечивают высокую масштабируемость и гибкость.

MongoDB ­­­­­­— документо-ориентированная СУБД. MongoDB хранит данные в гибких, JSON-подобных документах. Это означает, что поля могут варьироваться от документа к документу и структура данных может быть изменена со временем.

Основные концепции:

* База данных (Database): контейнер для коллекций.
* Коллекция (Collection): аналог таблицы в реляционных СУБД. Хранит группу связанных документов, но не требует от них
* Документ (Document): аналог строки (записи) в реляционных СУБД. Представляет собой структуру данных в формате BSON (бинарный JSON).
* Поле (Field): аналог столбца, пара ключ-значение в документе.
* Ключевая особенность: гибкая схема, позволяющая хранить в одной коллекции документы с разным набором полей.

Neo4j — это графовая база данных, разработанная для хранения и обработки связанных данных. Она идеально подходит для анализа сложных взаимосвязей, 2 таких как социальные сети, рекомендательные системы и управление зависимостями.

Основные компоненты графовой модели:

* Узлы (Nodes): представляют сущности (например, "Человек", "Фильм"). Аналогичны записям в реляционной таблице.
* Отношения (Relationships): представляют связи между узлами (например, ACTED\_IN, DIRECTED). Отношения всегда имеют направление и тип.
* Свойства (Properties): пары ключ-значение, которые хранятся внутри узлов и отношений (например, name: 'Tom Hanks').
* Язык запросов: Cypher — декларативный язык для запросов к графу.

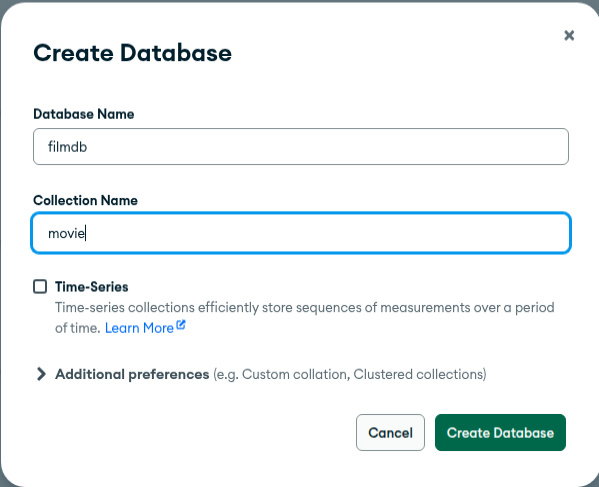
Redis (Remote Dictionary Server) — это высокопроизводительное хранилище данных в памяти, работающее по принципу "ключ-значение". Оно часто используется для кэширования, управления сессиями, очередей сообщений и в качестве брокера сообщений.

Основные структуры данных:

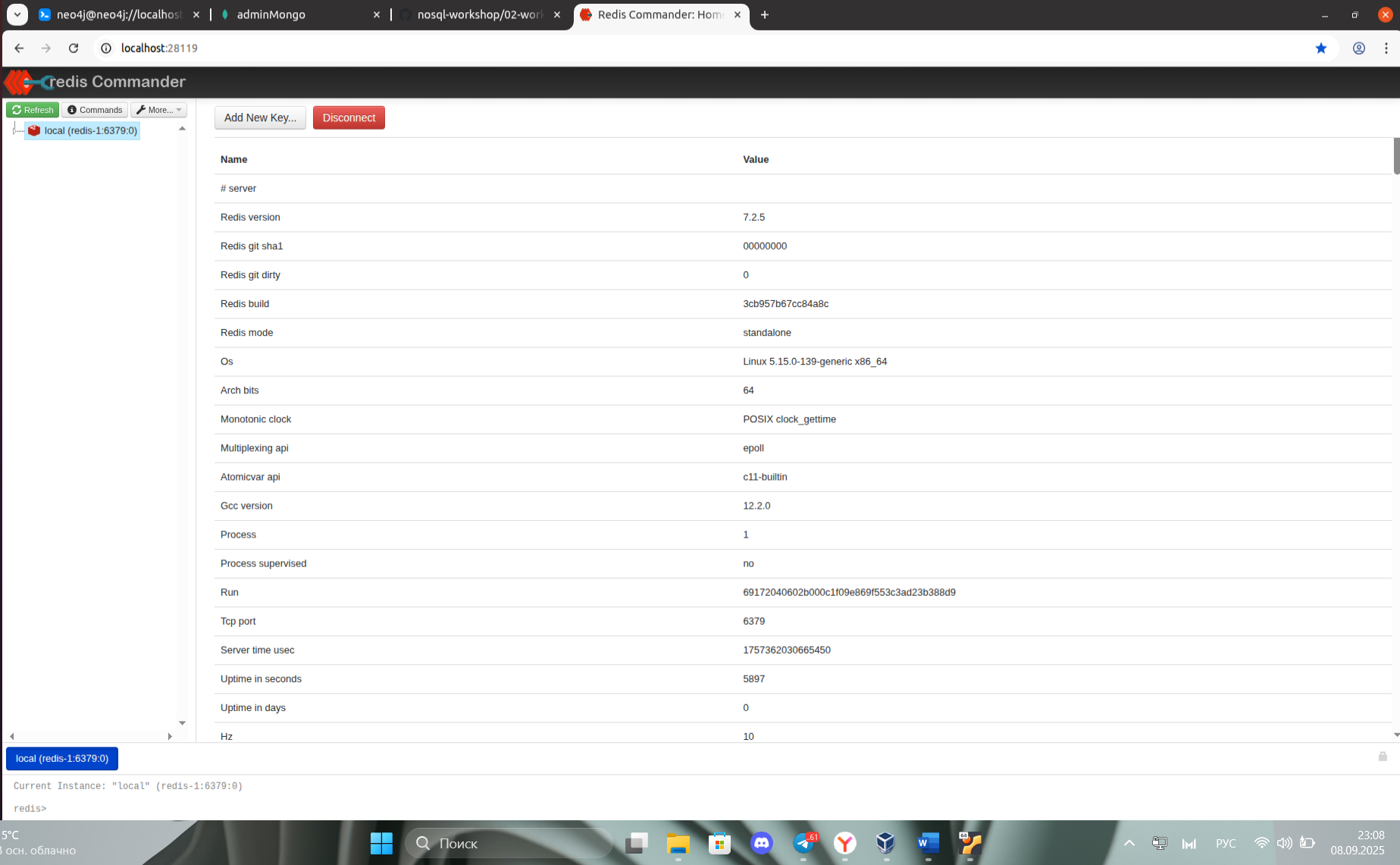
* Строки (Strings): простейший тип, где одному ключу соответствует одно строковое значение.
* Списки (Lists): последовательности строк, упорядоченные по порядку вставки.
* Множества (Sets): неупорядоченные коллекции уникальных строк
* Хэши (Hashes): структуры для хранения объектов, состоящие из полей и их значений.
* Упорядоченные множества (Sorted Sets): множества, где каждый элемент связан с числовым значением (оценкой), которое используется для сортировки.

# **Процесс выполнения общего из гит хаба**

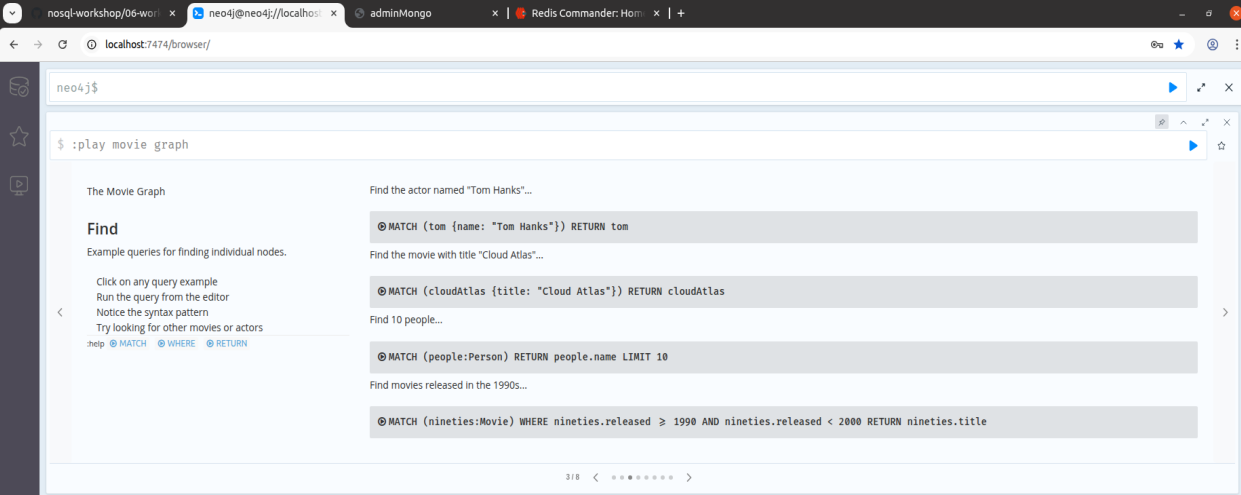
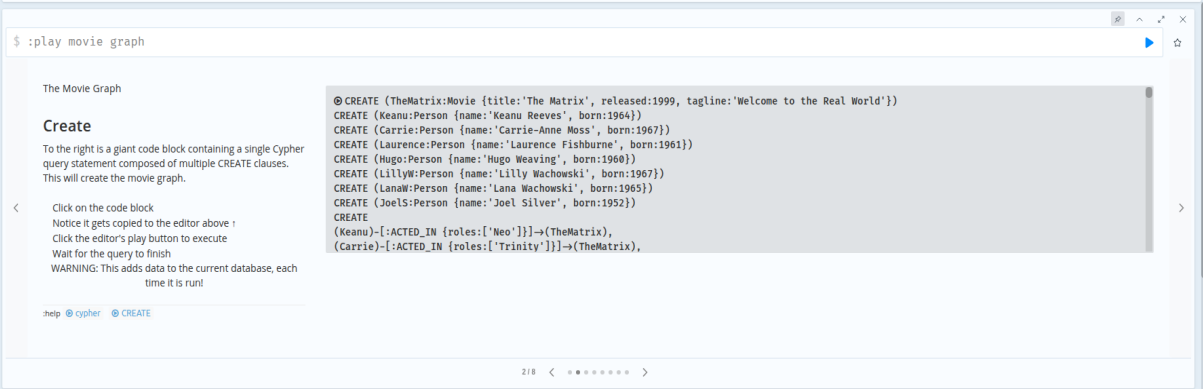
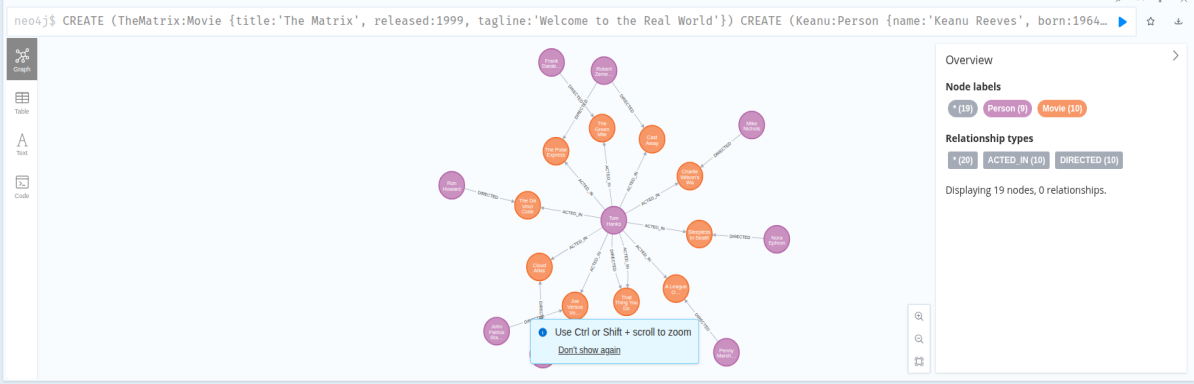
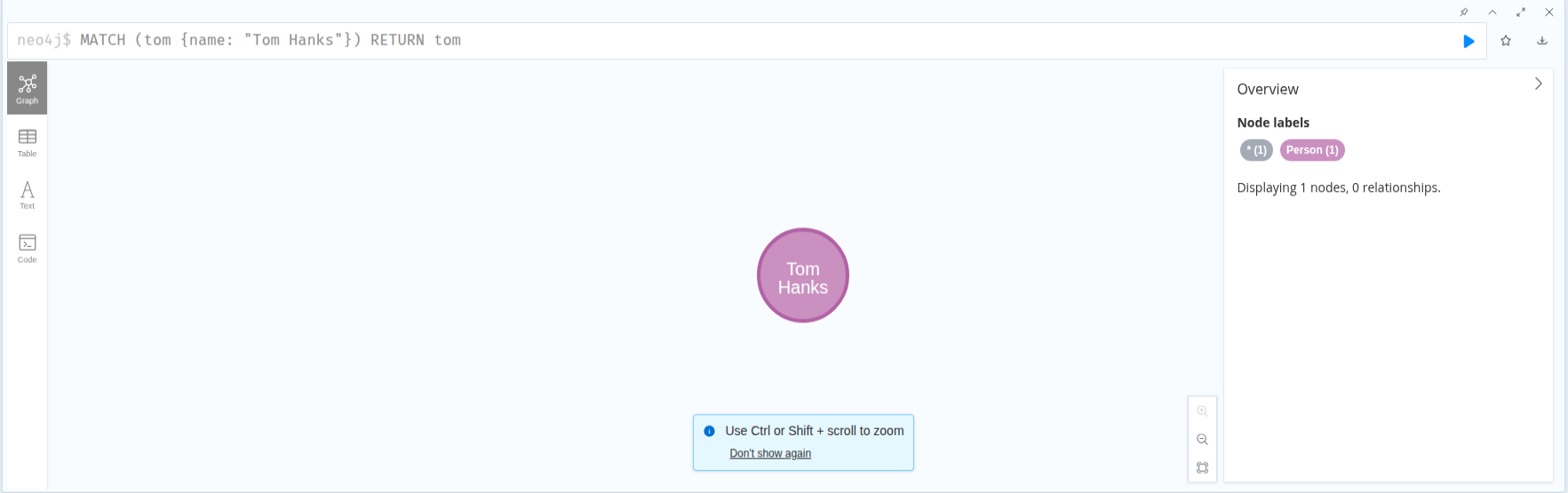
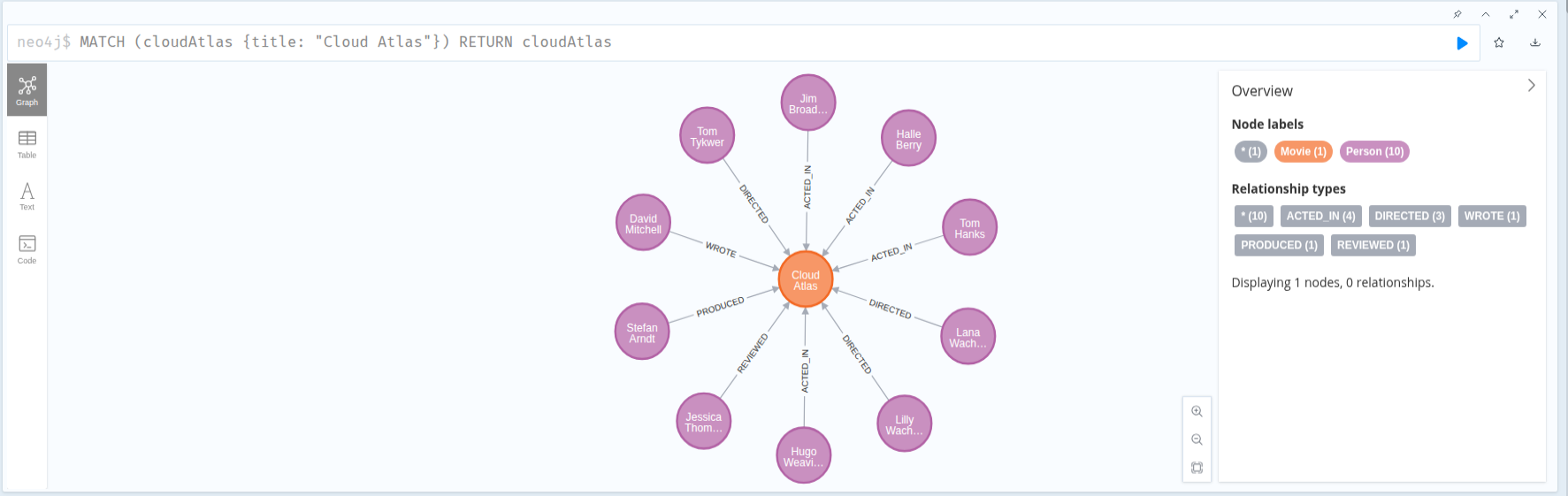
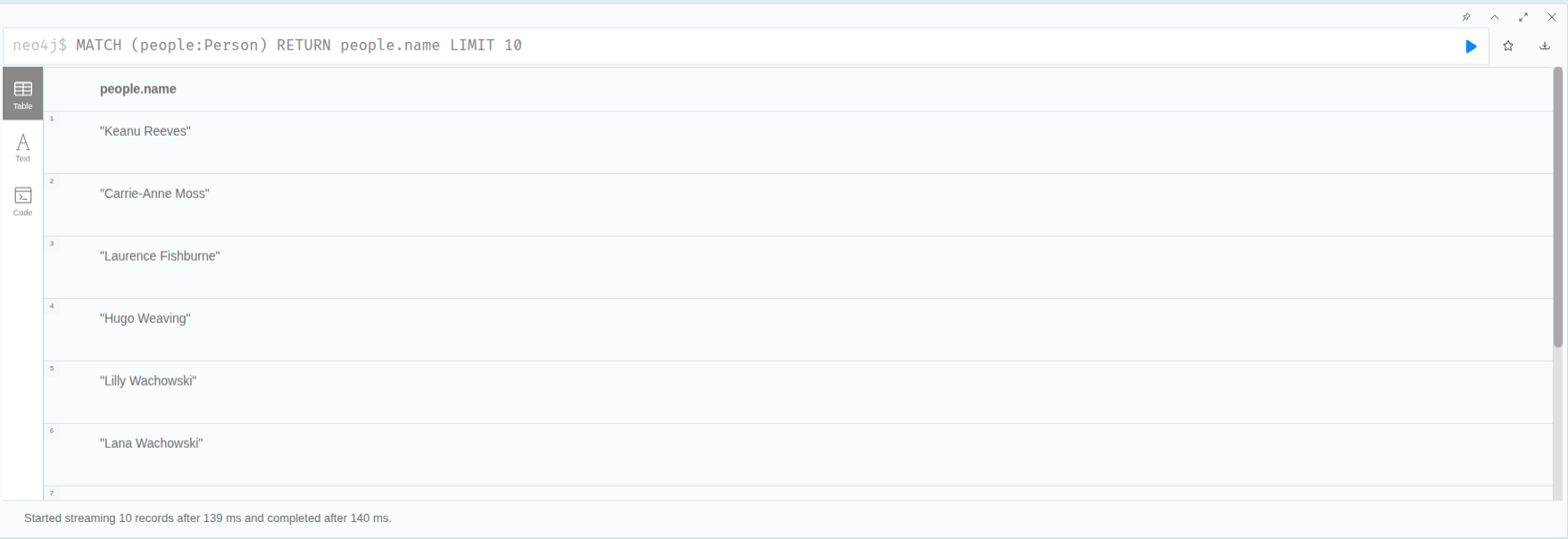
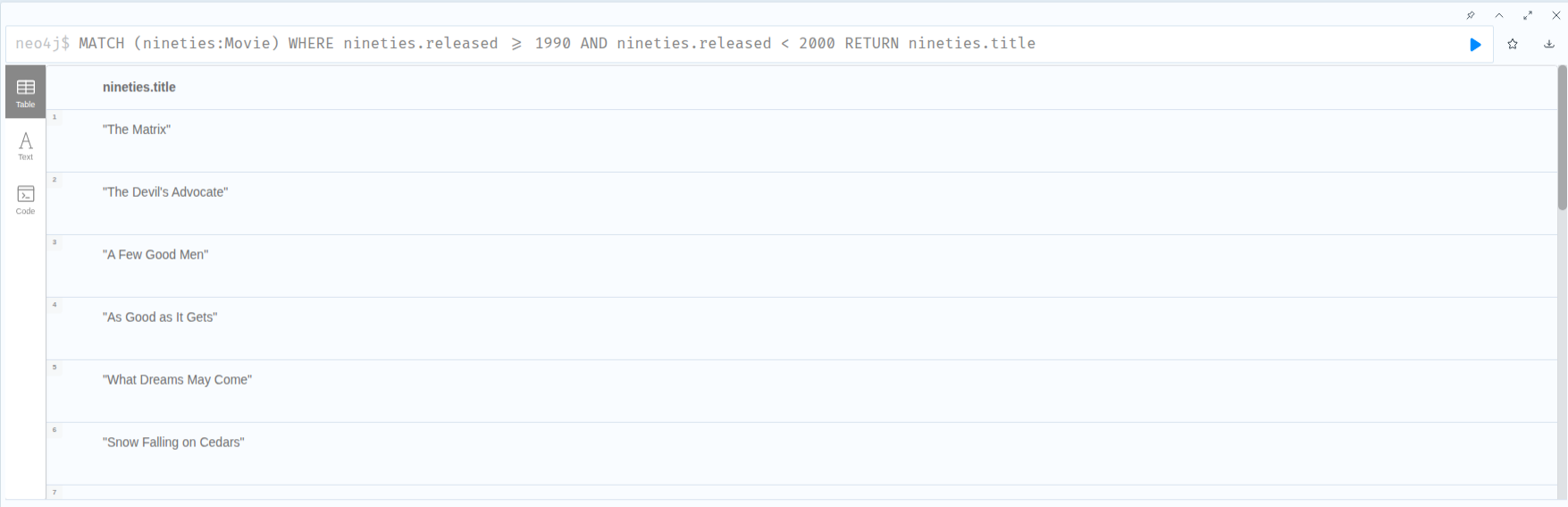
MongoDB:

1. Заходим в Mongo Compass
2. Создаем базу данных filmdb  
   
3. Загружаем данные с помощью JSON  
   

Redis:

1. Открыл Reddis Commander  
   
2. Выполнил все шаги из гит хаба:  
   

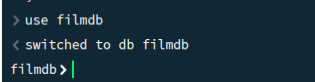
Neo4j:

1. Подключаемся в Neo4j Browser   
   
2. Выполняем команду :play movie graph  
   
3. Создаём граф  
   
4. Выполняем несколько запросов:
   1. MATCH (tom {name: "Tom Hanks"}) RETURN tom   
      
   2. MATCH (cloudAtlas {title: "Cloud Atlas"}) RETURN cloudAtlas  
      
   3. 1  
      
   4. 

# **Индивидуальные задания**

## MongoDB

Цель: Для фильма "Pulp Fiction" удалить из массива genres значение "Crime" ($pull).

1. Переключаю в консоли бд с test на filmdb  
   
2. Вписываю код:

db.Movie.updateOne(

{ title: "Pulp Fiction" },

{ $pull: { genres: "Crime" } }

)

1. Проверяю результат:  
     
   Удалился жанр Криминалистики

## Redis

Цель: Инкрементировать счетчик (INCR) для ключа page\_views:contact 15 раз. Получить итоговое значение.

1.Устанавливаю значение ключа page\_views:contact на 0

2. Прописываю INCRBY для инкрементации 15 раз счетчика  
INCRBY page\_views:contact 15

3.Проверяю значение  
GET page\_views:contact

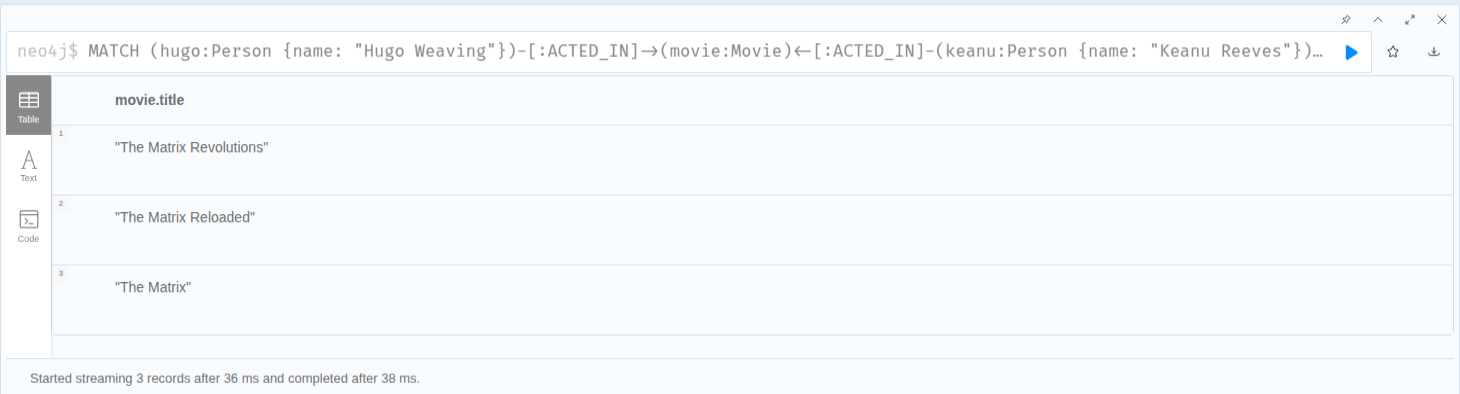


## Neo4j

Цель: Найти фильм, в котором вместе снимались "Hugo Weaving" и "Keanu Reeves".

1.Открываю Neo4j Browser

2.Вписываю код:  
  
MATCH (hugo:Person {name: "Hugo Weaving"})-[:ACTED\_IN]->(movie:Movie)<-[:ACTED\_IN]-(keanu:Person {name: "Keanu Reeves"}) RETURN movie.title AS title, movie.released AS year

3.Результат:  


# **Вывод**

Изучил и применил на практике различные типы NoSQL баз данных: документо-ориентированную (MongoDB), графовую (Neo4j) и ключ-значение (Redis).