

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

Create Database

Database Name

filmdb

Collection Name

movie

☐ Time-Series

Time-series collections efficiently store sequences of measurements over a period of time. [Learn More](#)

> Additional preferences (e.g. Custom collation, Clustered collections)

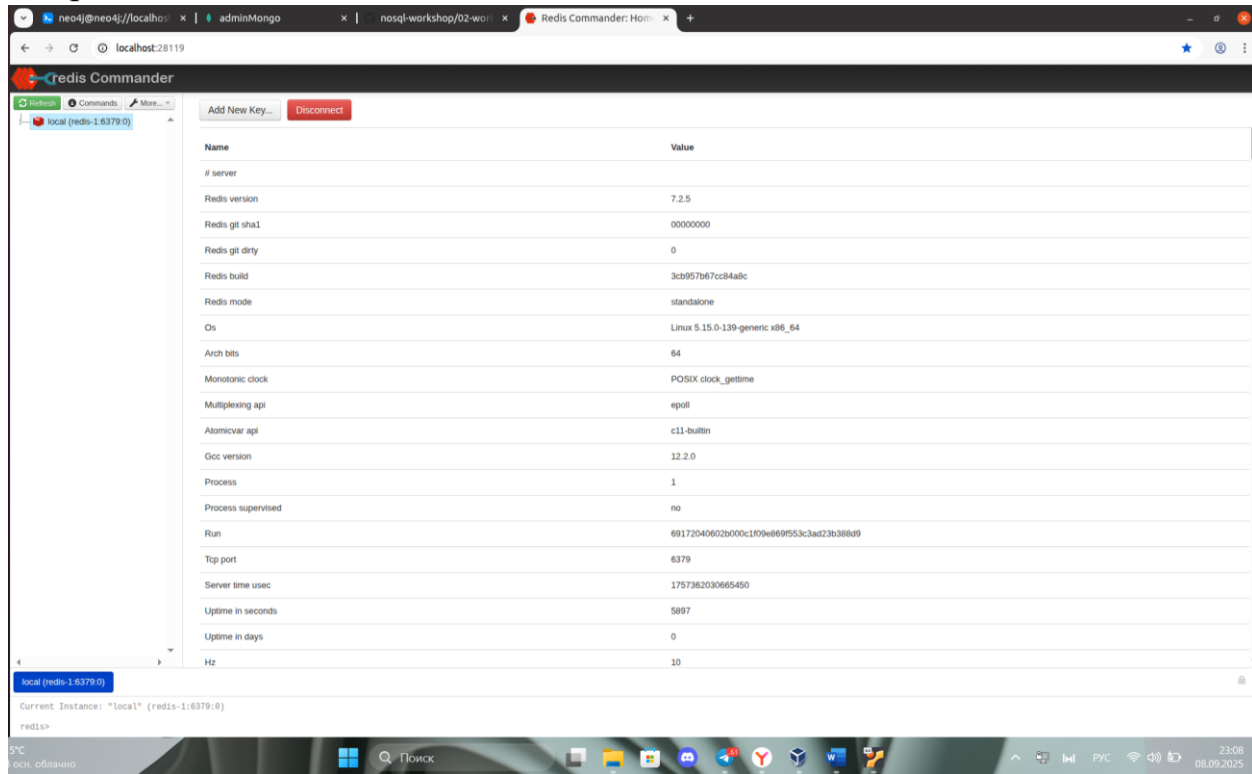
Cancel Create Database

3. Загружаем данные с помощью JSON

```
{
  "_id": ObjectId("68bf28c1b23dd4f0f1f670ad"),
  "id": "0110912",
  "title": "Pulp Fiction",
  "year": 1994,
  "runtime": 154,
  "languages": Array (3),
  "rating": 8.9,
  "votes": 2084331,
  "genres": Array (2),
  "plotOutline": "Jules Winnfield (Samuel L. Jackson) and Vincent Vega (John Travolta) a...",
  "coverUrl": "https://m.media-amazon.com/images/M/MV5BNGNhMDIzZTUtNTBlZi00MTRlLWJmZ2..."
  "actors": Array (12),
  "directors": Array (1),
  "producers": Array (7)
}
```

Redis:

1. Открыл Redis Commander



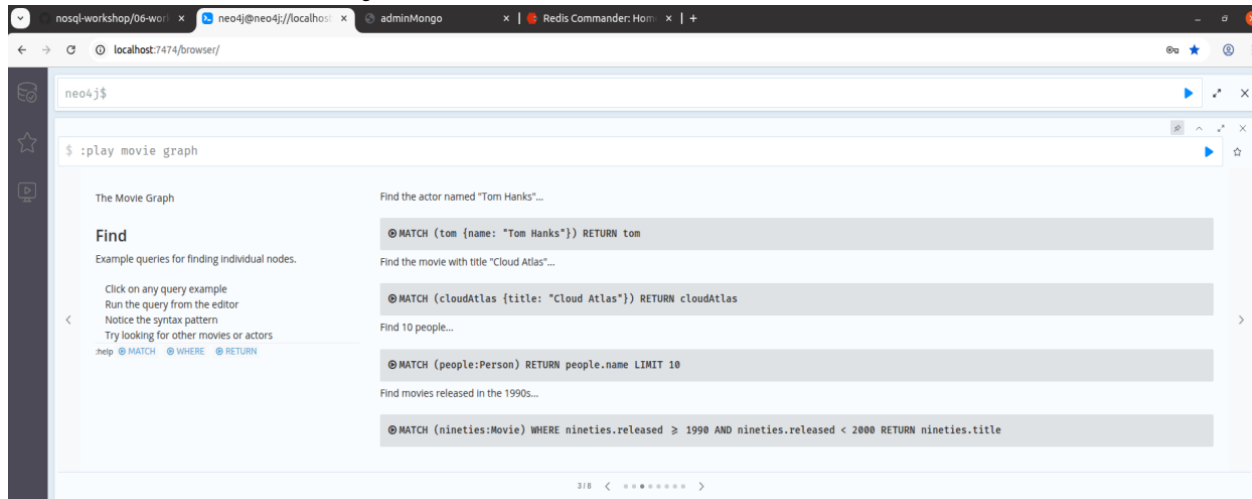
2. Выполнил все шаги из гит хаба:

```
help @string
"ERR unknown command 'help', with args beginning with: '@string' "
SET server:name "redis-server"
"OK"
GET server:name
"redis-server"
EXISTS server:name (integer) 1
1
KEYS server* 1) "server:name"
"ERR wrong number of arguments for 'keys' command"
KEYS * 1) "server:name"
"ERR wrong number of arguments for 'keys' command"
KEYS * 1) "server:name"
"ERR wrong number of arguments for 'keys' command"
SET connections 10 OK
"ERR syntax error"
redis:6379> SET connections 10 OK
"ERR unknown command 'redis:6379>', with args beginning with: 'SET' 'connections' '10' 'OK' "
SET connections 10
"OK"
GET connections
"10"

Current Instance: "local" (redis-1:6379:0)
redis>
```

Neo4j:

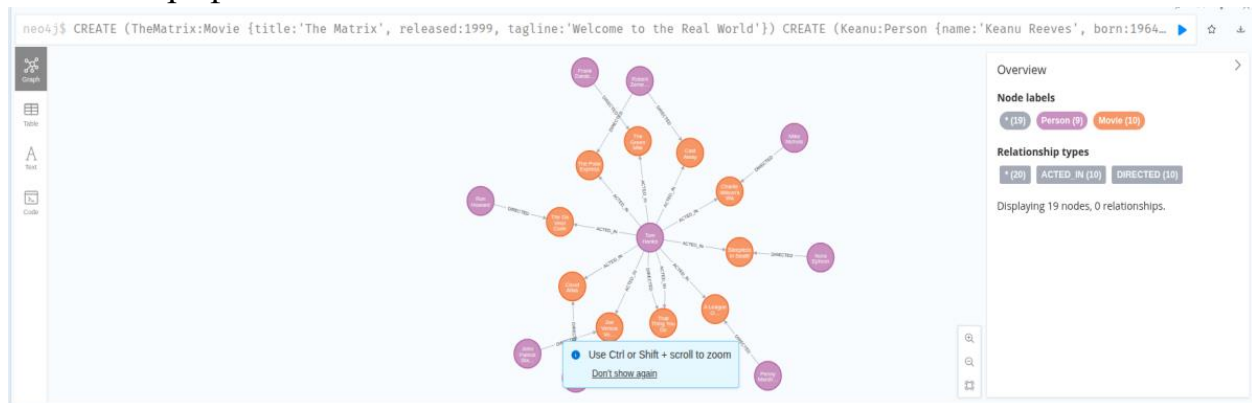
1. Подключаемся в Neo4j Browser



2. Выполняем команду `:play movie graph`

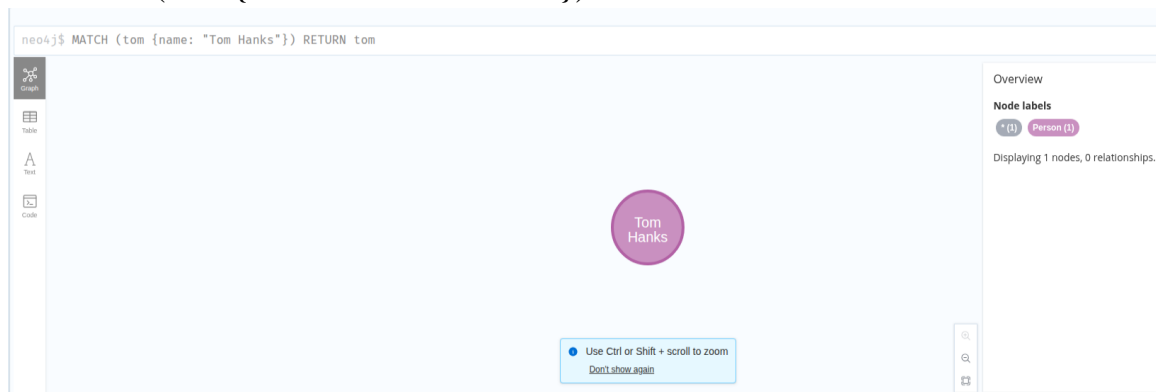


3. Создаём граф

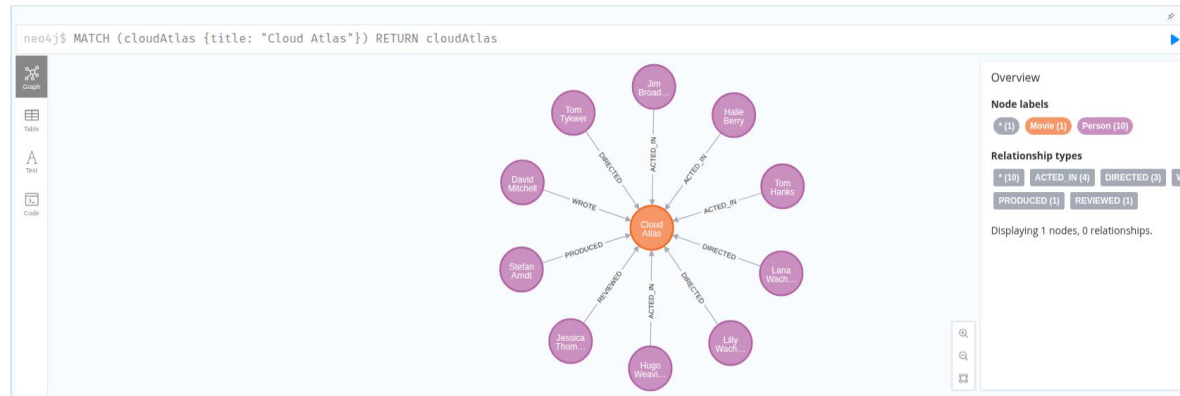


4. Выполняем несколько запросов:

а. `MATCH (tom {name: "Tom Hanks"}) RETURN tom`



b. **MATCH** (cloudAtlas {title: "Cloud Atlas"}) **RETURN** cloudAtlas



c. 1

The image shows a Neo4j Cypher query result for the query: `neo4j$ MATCH (people:Person) RETURN people.name LIMIT 10`. The result is a table with the following data:

people.name
"Keanu Reeves"
"Carrie-Anne Moss"
"Laurence Fishburne"
"Hugo Weaving"
"Lilly Wachowski"
"Lana Wachowski"

Started streaming 10 records after 139 ms and completed after 140 ms.

d.

The image shows a Neo4j Cypher query result for the query: `neo4j$ MATCH (nineties:Movie) WHERE nineties.released >= 1990 AND nineties.released < 2000 RETURN nineties.title`. The result is a table with the following data:

nineties.title
"The Matrix"
"The Devil's Advocate"
"A Few Good Men"
"As Good as It Gets"
"What Dreams May Come"
"Snow Falling on Cedars"

Индивидуальные задания

MongoDB

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

1. Переключаю в консоли бд с test на filmdb

```
> use filmdb
< switched to db filmdb
filmdb> |
```

2. Вписываю код:

```
db.Movie.updateOne(
  { title: "Pulp Fiction" },
  { $pull: { genres: "Crime" } }
)
```

3. Проверяю результат:

```
{
  "_id": ObjectId("68bf28c1b23dd4f0f1f670ad"),
  "id": "0110912",
  "title": "Pulp Fiction",
  "year": 1994,
  "runtime": 154,
  "languages": Array (3),
  "rating": 8.9,
  "votes": 2084331,
  "genres": Array (1)
    0: "Drama"
  "plotOutline": "Jules Winnfield (Samuel L. Jackson) and Vincent Vega (John Travolta) a..."
  "coverUrl": "https://m.media-amazon.com/images/M/MV5BNGNhMDIzZTUEB1Zi00MTRlLWFjM2..."
  "actors": Array (12),
  "directors": Array (1),
  "producers": Array (7)
}
```

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


```

SET page_view:contact 0
"OK"
GET page_view:contact
"0"
INCRBY page_view:contact 15
15
GET page_view:contact
"15"

Current Instance: "local" (redis-1:6379:0)

redis> |

```

Neo4j

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

1.Открываю Neo4j Browser

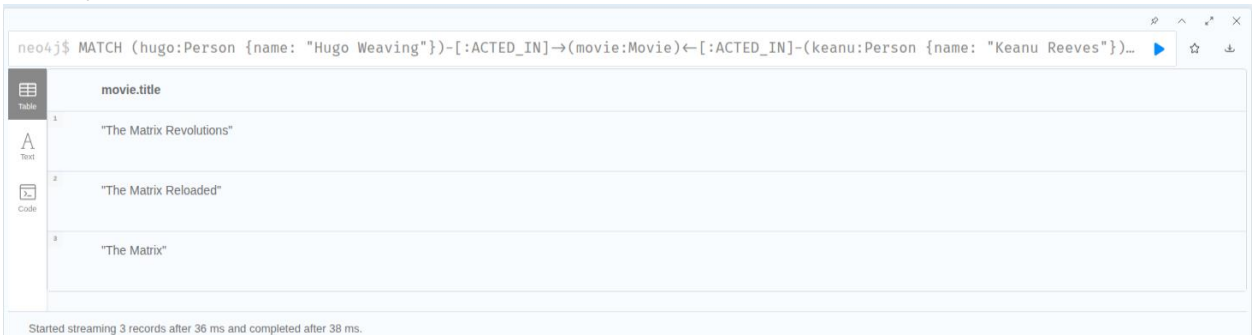
2.Вписываю код:

```

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:



	movie.title
1	"The Matrix Revolutions"
2	"The Matrix Reloaded"
3	"The Matrix"

Started streaming 3 records after 36 ms and completed after 38 ms.

Вывод

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