

Департамент образования города Москвы
Государственное автономное образовательное учреждение
высшего образования города Москвы
«Московский городской педагогический университет»
Институт цифрового образования
Департамент информатики, управления и технологий

Вариант - 3

Лабораторная работа 2.1

Тема: «Изучение методов хранения данных на
основе NoSQL»

Дисциплина «Инструменты для хранения и обработки больших
данных»

Выполнила:

Арлинская Александра Викторовна

Проверил:

Босенко Тимур Муртазович

Курс обучения: 4

Форма обучения: очная

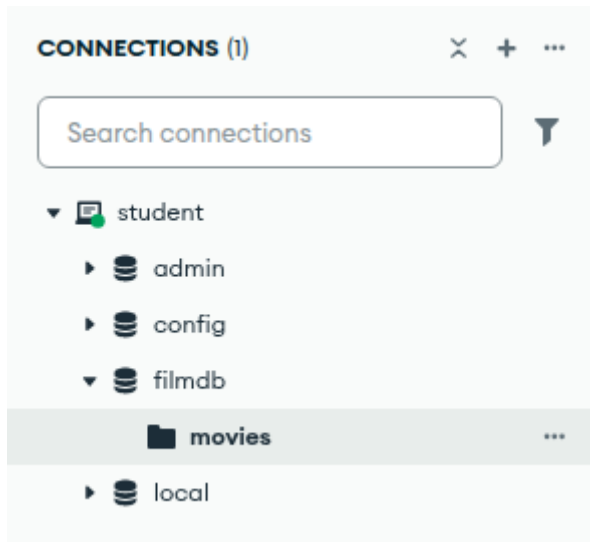
Москва

2025

Цель работы: изучение и практическое применение трех различных типов NoSQL баз данных: документо-ориентированной (MongoDB), графовой (Neo4j) и ключ-значение (Redis). Студенты должны научиться создавать, заполнять и анализировать структуры данных в каждой из систем, а также выполнять запросы для получения необходимой информации, развивая навыки работы с нереляционными моделями данных.

Работа с MongoDB

1. Открыли MongoDB Compass
2. Выполнили подключение к существующей MongoDB и добавили новую базу



3. Создали документ и поле id автоматически добавилось

```
_id: ObjectId('6900ed6b7b85af5e3d443ac0')
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/MV5BNGNhMDIzZTUtNTBlZi00MTRlLWFjM2..."
▶ actors: Array (12)
▶ directors: Array (1)
▶ producers: Array (7)
```

Структура документа:

Документ содержит:

- Простые поля (id, title, year, etc.)
- Массивы (languages, genres)
- Вложенные массивы объектов (actors, directors, producers)

4. Поиск документа

The screenshot shows the MongoDB Compass interface. The query bar contains the JSON query: `{ "title": "Pulp Fiction" }`. Below the query bar, the document is displayed in a collapsible tree view. The document structure is as follows:

```
{
  "_id": ObjectId('6900ed6b7b85af5e3d443ac0'),
  "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/MV5BNhMDIzZTUeNTBlZi00MTRLLWFjM2...",
  "actors": Array (12),
  "directors": Array (1),
  "producers": Array (7)
}
```

Док-во, что действительно работает

The screenshot shows the MongoDB Compass interface. The query bar contains the JSON query: `{ genres: "Detective" }`. Below the query bar, the document is displayed in a collapsible tree view. The document structure is as follows:

```
{
  "_id": ObjectId('6900f43d7b85af5e3d443acd'),
  "id": "0133093",
  "title": "The Matrix",
  "year": 1999,
  "runtime": 136,
  "languages": Array (1),
  "rating": 8.7,
  "votes": 1496538,
  "genres": Array (2),
  "plotOutline": "Thomas A. Anderson is a man living two lives. By day he is an average ...",
  "coverUrl": "https://m.media-amazon.com/images/M/MV5BNzQzOTk3OTAtNDQ0Zi00ZTVkLWI0MT...",
  "actors": Array (12),
  "directors": Array (2),
  "producers": Array (3)
}
```

Below the document, a message states: "No results. Try modifying your query to get results."

5. Добавление еще одного фильма

The screenshot shows the MongoDB Compass interface. The query bar contains the JSON query: `{ genres: "Detective" }`. Below the query bar, the document is displayed in a collapsible tree view. The document structure is as follows:

```
{
  "_id": ObjectId('6900f43d7b85af5e3d443acd'),
  "id": "0133093",
  "title": "The Matrix",
  "year": 1999,
  "runtime": 136,
  "languages": Array (1),
  "rating": 8.7,
  "votes": 1496538,
  "genres": Array (2),
  "plotOutline": "Thomas A. Anderson is a man living two lives. By day he is an average ...",
  "coverUrl": "https://m.media-amazon.com/images/M/MV5BNzQzOTk3OTAtNDQ0Zi00ZTVkLWI0MT...",
  "actors": Array (12),
  "directors": Array (2),
  "producers": Array (3)
}
```

Работа с Redis:

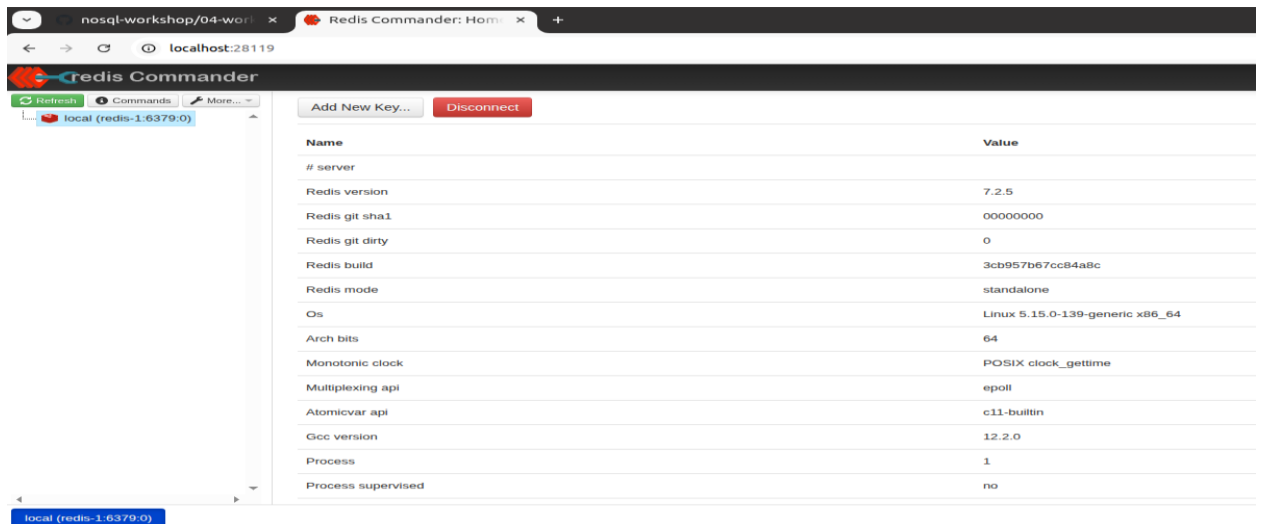
1. Начало работы

```

ngpu@ngpu-vn:~$ docker run -it --rm --network nosql-platform bitnami/redis redis-cli -h redis-1 -p 6379
Unable to find image 'bitnami/redis:latest' locally
latest: Pulling from bitnami/redis
bc946fda1c85: Pull complete
Digest: sha256:f8fc4ba70daf1885971f82c4f655b9f4afbd701851efbea11c77faf299465210
Status: Downloaded newer image for bitnami/redis:latest
redis 17:09:48.07 INFO ==>
redis 17:09:48.08 INFO ==> Welcome to the Bitnami redis container
redis 17:09:48.09 INFO ==> Subscribe to project updates by watching https://github.com/bitnami/containers
redis 17:09:48.10 INFO ==> NOTICE: Starting August 28th, 2025, only a limited subset of images/charts will remain available for free. Backup will
for some time at the 'Bitnami Legacy' repository. More info at https://github.com/bitnami/containers/issues/83267
redis 17:09:48.11 INFO ==>
redis-1:6379> AUTH abc123!
(error) ERR AUTH <password> called without any password configured for the default user. Are you sure your configuration is correct?
redis-1:6379> PING
PONG

```

2. Открыли Redis Commander:



The screenshot shows the Redis Commander web interface in a browser. The address bar indicates the connection is to localhost:28119. The interface displays the details for the 'local (redis-1:6379:0)' instance. On the right, there is a table with system information:

Name	Value
# server	
Redis version	7.2.5
Redis git sha1	00000000
Redis git dirty	0
Redis build	3cb957b67cc84a8c
Redis mode	standalone
Os	Linux 5.15.0-139-generic x86_64
Arch bits	64
Monotonic clock	POSIX clock_gettime
Multiplexing api	epoll
Atomicvar api	c11-builtin
Gcc version	12.2.0
Process	1
Process supervised	no

3. Выполнение команд



The screenshot shows the Redis Commander web interface with the 'local (redis-1:6379:0)' instance selected. The command history on the left shows the following commands and their outputs:

```

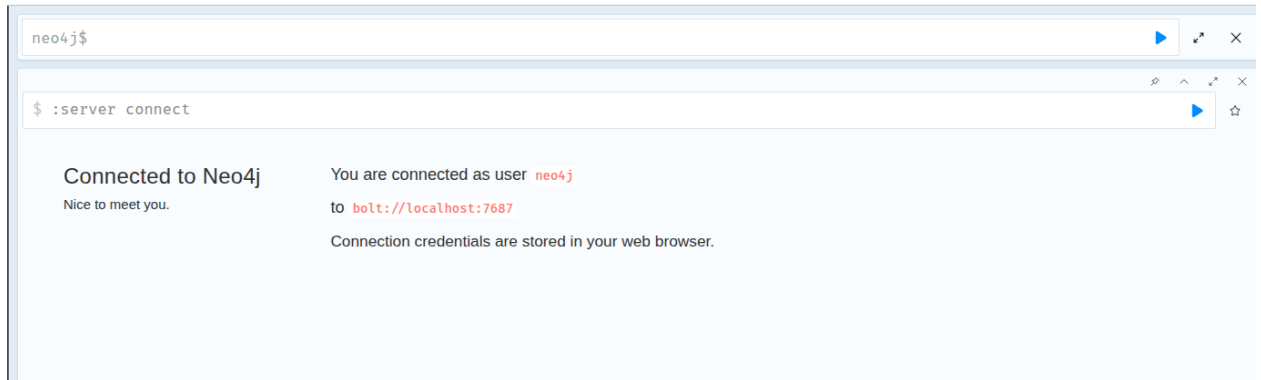
GET server:name
"redis-server"
EXISTS server:name
1
KEYS server*
1) "server:name"
SET connections 10
"OK"
GET connections
"10"
SET connections 20
"OK"
GET connections
"20"
MSET key1 10 key2 20 key3 30
"OK"
MGET key1 key3
1) "10"
2) "30"
SET connections 10
"OK"
INCR connections
11
INCRBY connections 10
21
DECR connections
20
DECRBY connections 10
10
SET resource:lock "Redis Demo"
"OK"
EXPIRE resource:lock 120
1
TTL resource:lock
110

```

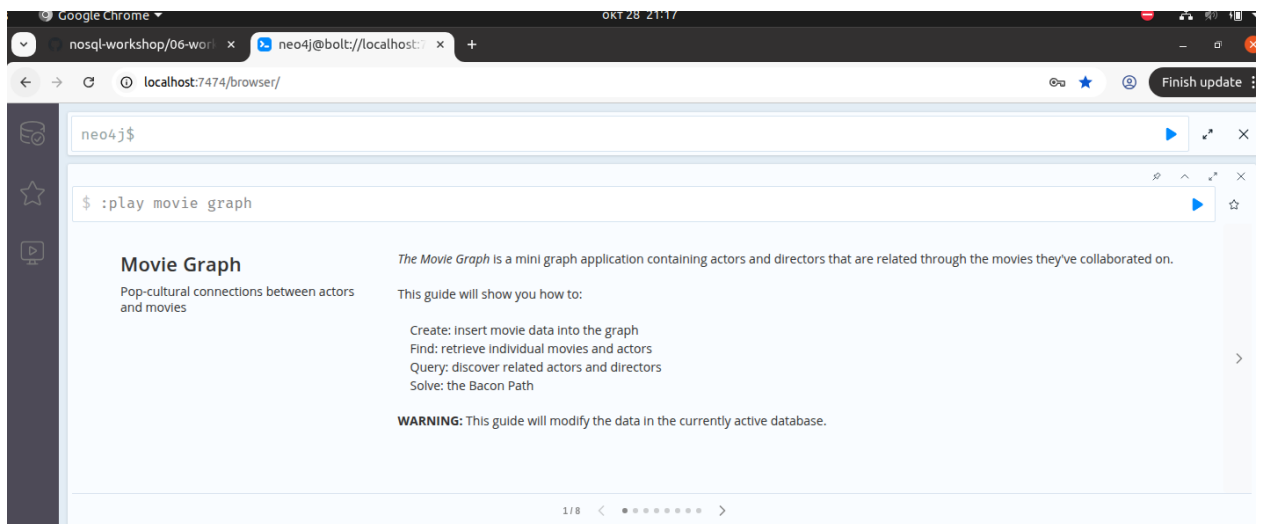
At the bottom, it indicates the 'Current Instance: "local" (redis-1:6379:0)'.

Работа Neo4j

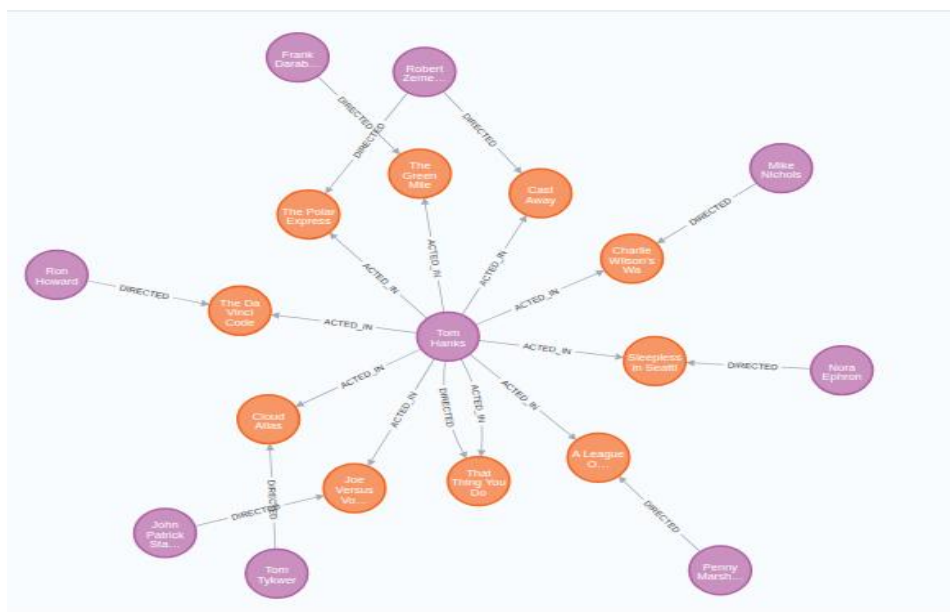
1. Выполнили подключение



2. Вводим команду для работы с графами

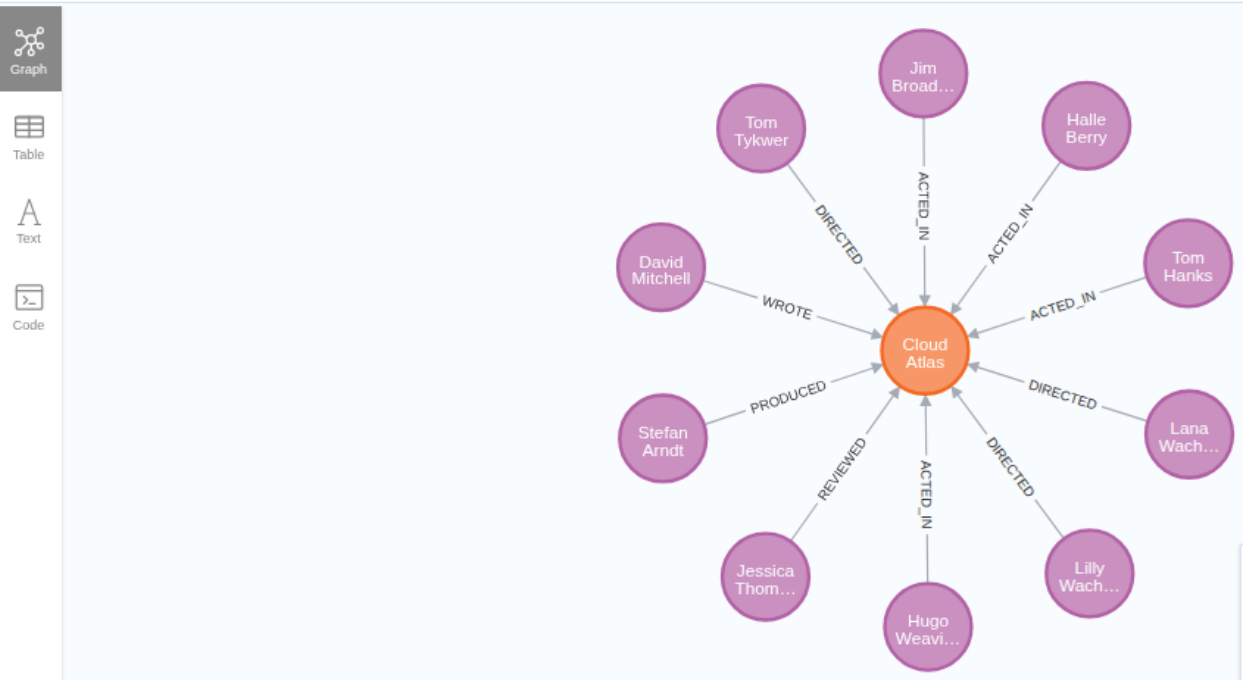


3. Выполнили команду Create



4. Команды для поиска

```
neo4j$ MATCH (cloudAtlas {title: "Cloud Atlas"}) RETURN cloudAtlas
```

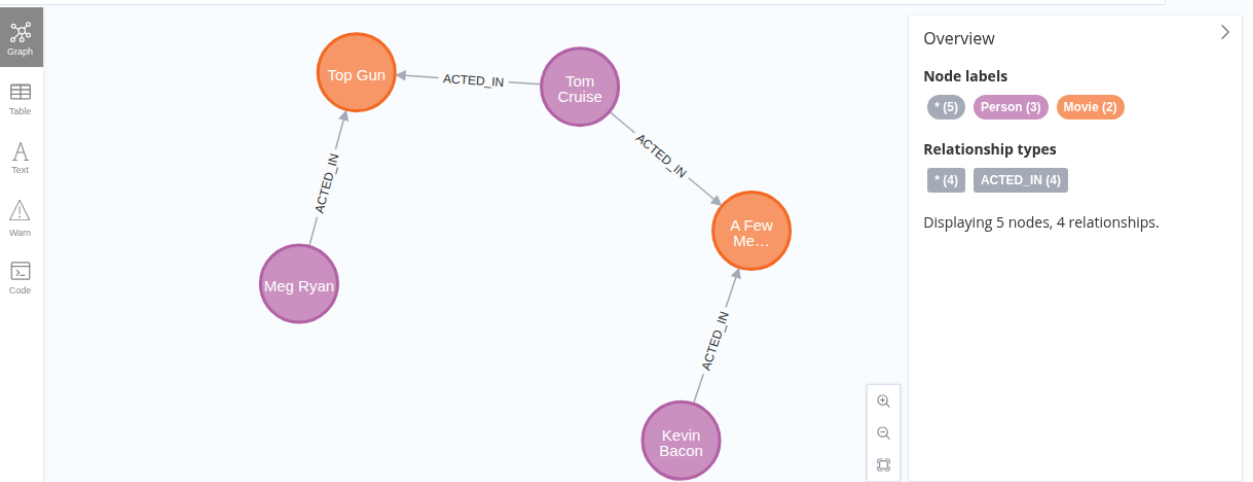


```
neo4j$ MATCH (tom:Person {name:"Tom Hanks"})-[:ACTED_IN]→(m)←[:ACTED_IN]-(coActors) RETURN coActors.name
```

	coActors.name
1	"Ed Harris"
2	"Gary Sinise"
3	"Kevin Bacon"
4	"Bill Paxton"
5	"Parker Posey"
6	"Greg Kinnear"
7	

Started streaming 39 records after 14 ms and completed after 16 ms.

```
neo4j$ MATCH p=shortestPath( (bacon:Person {name:"Kevin Bacon"})-[*]-(meg:Person {name:"Meg Ryan"}) ) RET...
```



Индивидуальные задания:

MongoDB

1. Переключаемся на нужную БД

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

2. Добавление фильма с соответствующим годом

```
> db.movies.insertOne({  
  title: " Passemgers",  
  year: 2008,  
  director: "Rodrigo Garsia",  
  genres: ["Triler", "Action"],  
  ratings: {  
    imdb: 8.8,  
    metacritic: 74  
  }  
})  
< {  
  acknowledged: true,  
  insertedId: ObjectId('690101c54acdce84b7634dd')  
}
```

3. Добавление поля rank для фильмов 1994 и 2008 года

```
> db.movies.updateMany(  
  { year: { $in: [2008]} },  
  { $set: {rank: 1 } }  
);  
< {  
  acknowledged: true,  
  insertedId: null,  
  matchedCount: 1,  
  modifiedCount: 1,  
  upsertedCount: 0  
}  
filmdb> |
```


4. Проверка

```
▶ _id: ObjectId('690101c54acdcce84b7634dd')
  title: " Passemgers"
  year: 2008
  director: "Rodrigo Garsia"
  ▶ genres: Array (2)
  ▶ ratings: Object
    rank: 1
```

5. Найти все фильмы, выпущенные в 1994 или 2008 году (\$in), и удалить у них поле rank (\$unset).

```
> db.movies.updateMany(
  { year: { $in: [1994, 2008] } },
  { $unset: { rank: "" } }
);
< {
  acknowledged: true,
  insertedId: null,
  matchedCount: 2,
  modifiedCount: 2,
  upsertedCount: 0
}
filmdb>
```

6. Проверка

```
▶ _id: ObjectId('690101c54acdcce84b7634dd')
  title: " Passemgers"
  year: 2008
  director: "Rodrigo Garsia"
  ▶ genres: Array (2)
  ▶ ratings: Object
```

```
▶ _id: ObjectId('6900ed6b7b85af5e3d443ac0')
  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/MV5BNGNhMDIzZTUtNTBlZi00MTRlLWFjM2..."
  ▶ actors: Array (12)
  ▶ directors: Array (1)
  ▶ producers: Array (7)
```

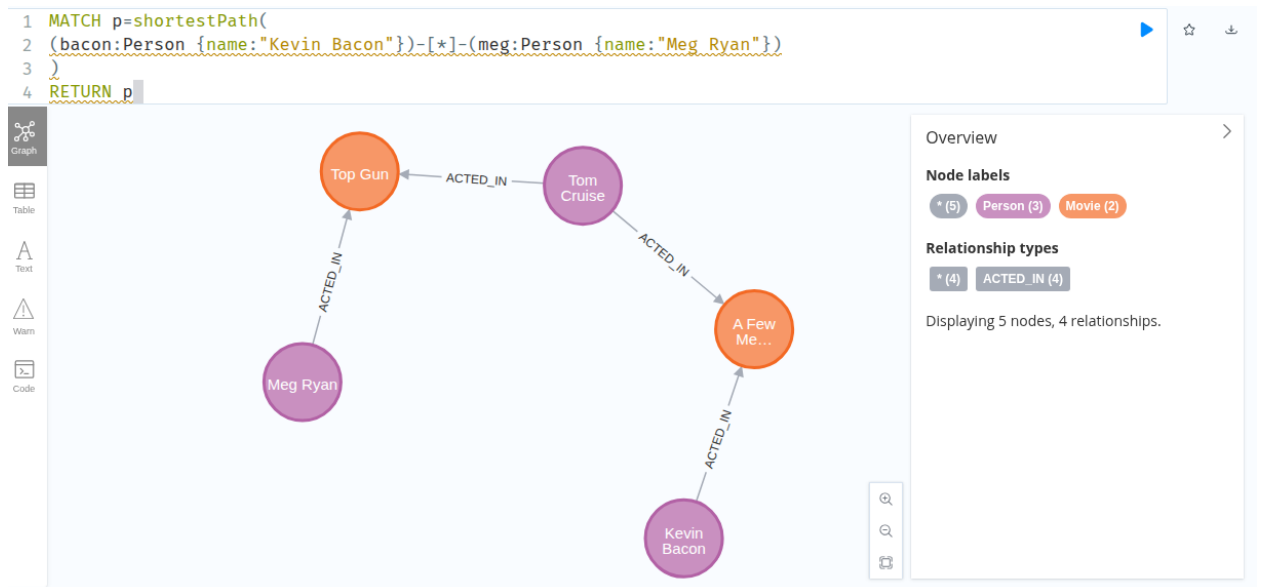
Redis

1. Смоделировать очередь задач, с помощью списка (LPUSH) добавить 5 ID задач в очередь task_queue;
2. Проверили список;
3. Извлекли (RPOP) 2 задачи для обработки;
4. Проверили список.

```
ping
"PONG"
LPUSH task_queue 5 4 3 2 1
5
LRANGE task_queue 0 4
1) "1"
2) "2"
3) "3"
4) "4"
5) "5"
RPOP task_queue
"5"
RPOP task_queue
"4"
LRANGE task_queue
"ERR wrong number of arguments for 'lrange' command"
LRANGE task_queue 0 2
1) "1"
2) "2"
3) "3"
```

Neo4j

1. Нашли кратчайший путь (shortestPath) в графе между актерами "Kevin Bacon" и "Meg Ryan"



Вывод

В ходе выполнения лабораторной работы рассмотрены ключевые направления и особенности различных типов NoSQL баз данных, отличающихся от традиционных реляционных по структуре и подходам к хранению информации. Документо-ориентированная СУБД MongoDB демонстрирует высокую гибкость за счёт хранения данных в JSON-подобных документах с возможностью изменяющейся схемы. Графовая СУБД Neo4j эффективна для анализа сложных взаимосвязей благодаря модели узлов и направленных отношений, что особенно актуально для социальных сетей и рекомендательных систем. Redis, как хранилище "ключ-значение", обеспечивает высокую производительность в задачах кэширования и управления сессиями, предлагая разнообразные структуры данных для эффективной работы с информацией. Таким образом, выбор конкретного типа NoSQL базы зависит от специфики задачи, необходимой масштабируемости и структуры обрабатываемых данных.