

ДЕПАРТАМЕНТ ОБРАЗОВАНИЯ И НАУКИ ГОРОДА МОСКВЫ  
Государственное автономное образовательное учреждение  
Высшего образования города Москвы  
«Московский городской педагогический университет»  
(ГАОУ ВО МГПУ)

Институт цифрового образования

Лабораторная работа №2.1  
«Изучение методов хранения данных на основе NoSQL»  
По дисциплине «Инструменты для хранения и обработки больших  
данных»

Выполнила: Татаринова Екатерина Михайловна

Группа: АДЭУ-221

Преподаватель: доцент Босенко Т.М.

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.

### **Краткое описание процесса выполнения:**

1. Подключение к базе данных MongoDB, создание коллекций, документов, заполнение документов, работа с документами, их изменение и удаление.
2. Подключение к Redis Commander, рассмотрение разных структур данных (string, set, hash и т.д.)
3. Подключение к Neo4j Browser, создание узлов, отношений, атрибутов

## Вариант 15.

### Подготовка окружения.

The screenshot shows a terminal window titled 'docker' with the command `sudo docker compose up -d` being run. The output shows the creation and start of several Docker containers, including 'admin-mongo', 'mongo-1', 'mongo-express', 'cassandra-1', 'redis-commander', 'redis-1', 'neo4j-1', and 'cassandra-web'. The containers are listed with their status as 'Started' and creation times: 33.4s, 37.2s, and 43.8s.

```
Nov 18 10:25
File Edit Selection View Go ...
docker-compose.yml x
File Edit Selection View Go ...
dockerdocker
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
mgpu@mgpu-vm:~/Downloads/ibd/nosql-workshop/01-environment/docker$ sudo docker compose up -d
[+] Running 10/10
  ✓ Network nosql-platform
    ✓ Container admin-mongo      Started
    ✓ Container mongo-1          Started
    ✓ Container mongo-express    Started
    ✓ Container cassandra-1     Started
    ✓ Container redis-commander Started
    ✓ Container redis-1          Started
    ✓ Container neo4j-1          Started
    ✓ Container cassandra-web   Started
33.4s
37.2s
43.8s
You have Docker installed on your system. Do you want to ...
bash + x 37.2s 43.8s
Right Ctrl
```

### Задание для самостоятельной работы MongoDB

№1. Используя агрегацию, посчитать количество фильмов, выпущенных в каждое десятилетие.

The screenshot shows a terminal window titled 'docker' with the command `sudo docker exec -ti mongo-1 mongosh -u "root" -p "abc123!"` being run. The output shows the connection to the MongoDB shell on port 27017, using version 2.5.6. The user 'root' is connected with password 'abc123!'.

```
Dec 2 12:29
File Edit Selection View Go ...
docker-compose.yml x
File Edit Selection View Go ...
dockerdocker
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
mgpu@mgpu-vm:~/Downloads/ibd/nosql-workshop/01-environment/docker$ sudo docker exec -ti mongo-1 mongosh -u "root" -p "abc123!"
Current Mongosh Log ID: 692eb158ab90c465c989b03c
Connecting to: mongodb://<credentials>@127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.5.6
Using MongoDB: 7.0.23
Using Mongosh: 2.5.6
For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/
To help improve our products, anonymous usage data is collected and sent to MongoDB periodically (https://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.
Add context (#), exte
Ask v > v
Ln 203, Col 30, Spaces: 2, UTF-8, LF, { } Compose, & &
```

The screenshot shows the MongoDB Compass interface. On the left is a sidebar with various icons for different tools like MongoDB, MySQL, PostgreSQL, and Redis. The main area has a title bar "Dec 2 13:14" and "MongoDB Compass - localhost:27017/Shell". Below the title bar, there's a navigation bar with tabs: "movies", "mongosh:localhost:27017", and "mongosh:localhost:27017". The main content area displays a MongoDB query in the "MONGOSH" shell:

```
>_MONGOSH
{
  $group: {
    _id: { $floor: { $divide: ["$year", 10] } },
    count: { $sum: 1 }
  },
  {
    $project: {
      decade: { $concat: [{ $toString: { $multiply: ["$_id", 10] } }, "s"] },
      count: 1,
      _id: 0
    }
  },
  {
    $sort: { decade: 1 }
  }
}
<(
  {
    count: 2,
    decade: '1990s'
  }
  {
    count: 10,
    decade: '2020s'
  }
)
filmdb>
```

## Задание для самостоятельной работы Redis

The screenshot shows a terminal window and a Redis Commander interface. The terminal window shows the command `docker run -it --rm --network nosql-platform bitnami/redis redis -cli -h redis-1 -p 6379` being run, followed by Redis startup logs:

```
mgpu@mgpu-vm:~$ docker run -it --rm --network nosql-platform bitnami/redis redis
redis 10:23:07.43 INFO ==>
redis 10:23:07.43 INFO ==> Welcome to the Bitnami redis container
redis 10:23:07.44 INFO ==> Subscribe to project updates by watching https://git
hub.com/bitnami/containers
redis 10:23:07.44 INFO ==> NOTICE: Starting August 28th, 2025, only a limited s
ubset of images/charts will remain available for free. Backup will be available
for some time at the 'Bitnami Legacy' repository. More info at https://github.co
m/bitnami/containers/issues/83267
redis 10:23:07.45 INFO ==>
```

The Redis Commander interface shows a sidebar with file explorer entries like "11-pinecone", "99-misc", "archive", "examp", and "vv-working-with-influxdb". The main pane shows Redis commands and their results:

- GET server:name
- И Redis ОТВЕТИТ "redis-server".
- EXISTS server:name

## №2 Задание 3 (Redis)

Создать список `log_errors` и добавить в него 3 сообщения об ошибках. Получить последнее добавленное сообщение, не удаляя его из списка (`LINDEX`)

The screenshot shows two terminal windows and a Jupyter Notebook interface.

**Terminal Window 1:** Shows the Redis CLI running in a Docker container. It displays the Bitnami Redis welcome message, some INFO logs, and a PING/PONG interaction. The command history shows:

```
redis> PING
(error) ERR unknown command 'redis-1:6379>', with args beginning with: 'PING'
"ERR redis-1:6379> PING
PONG
redis-1:6379> "
```

**Terminal Window 2:** Shows the Redis CLI running in a Docker container. It displays help information and a series of error messages from the `RPUSH log_errors` command. The command history shows:

```
"help @<group>" to get a list of commands in <group>
"help <command>" for help on <command>
"help <tab>" to get a list of possible help topics
"quit" to exit

To set redis-cli preferences:
  ":set hints" enable online hints
  ":set nohints" disable online hints
Set your preferences in ~/.redisclirc
redis-1:6379> RPUSH log_errors "Error: Database connection timeout"
(integer) 1
redis-1:6379> RPUSH log_errors "Error: File 'config.json' not found"
(integer) 2
redis-1:6379> RPUSH log_errors "Error: Unauthorized API access attempt"
(integer) 3
redis-1:6379> RPUSH log_errors "Error: Database connection timeout"
(integer) 4
redis-1:6379> RPUSH log_errors "Error: File 'config.json' not found"
(integer) 5
redis-1:6379> RPUSH log_errors "Error: Database connection timeout"
(integer) 6
redis-1:6379> LINDEX log_errors -1
"Error: Database connection timeout"
redis-1:6379>
```

**Jupyter Notebook Cell:** Shows a code cell with the following content:

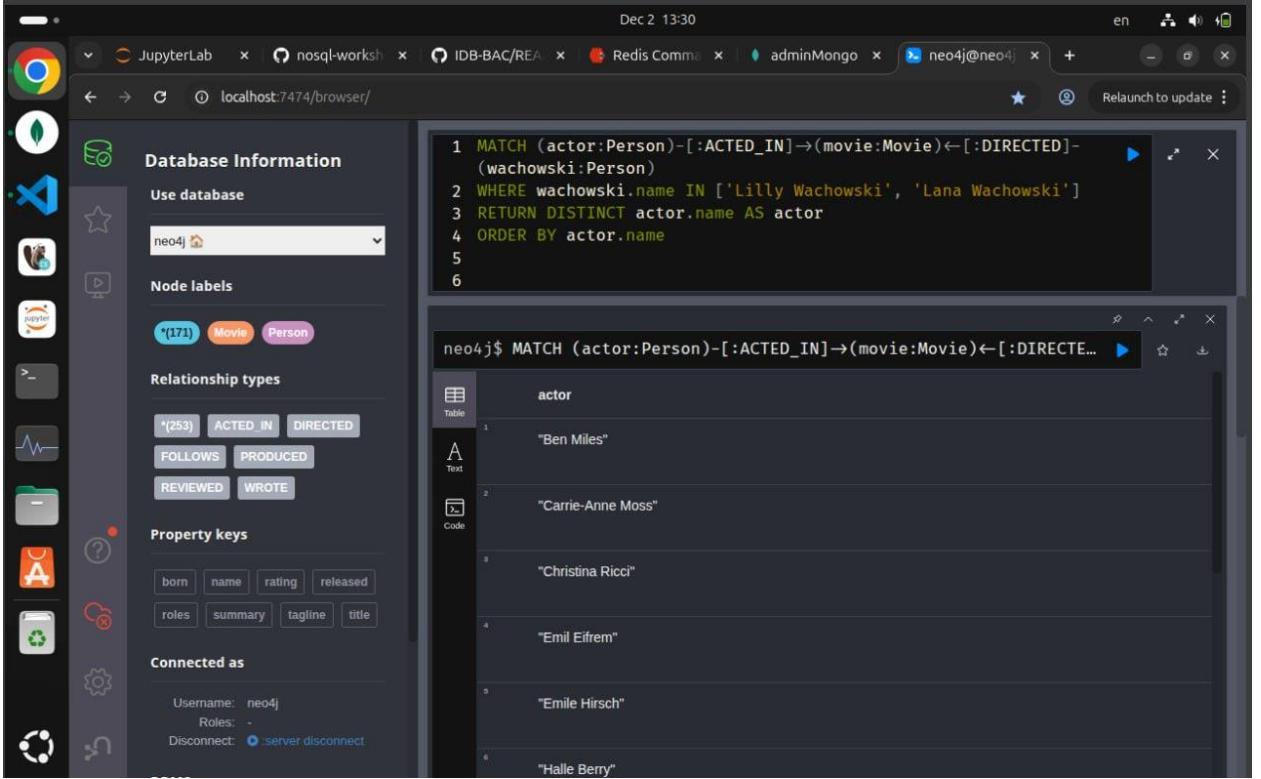
```
In [1]: RPUSH log_errors "Error: Database connection timeout"
RPUSH log_errors "Error: File 'config.json' not found"
RPUSH log_errors "Error: Unauthorized API access attempt"

In [1]: LINDEX log_errors -1
```

**Задание для самостоятельной работы Neo4j**

## №3 Задание 2 (Neo4j)

Найти всех актеров, которые работали с режиссерами Вачовски (Lille или Lana)



The screenshot shows the Neo4j browser interface. On the left, there's a sidebar with various icons and a "Database Information" panel. The main area has a query editor and a results table.

```
1 MATCH (actor:Person)-[:ACTED_IN]-(movie:Movie) <-[:DIRECTED]-> (wachowski:Person)
2 WHERE wachowski.name IN ['Lilly Wachowski', 'Lana Wachowski']
3 RETURN DISTINCT actor.name AS actor
4 ORDER BY actor.name
```

actor
"Ben Miles"
"Carrie-Anne Moss"
"Christina Ricci"
"Emil Eifrem"
"Emile Hirsch"
"Halle Berry"

## Вывод

В ходе выполнения задания были успешно применены три ключевые технологии из экосистемы NoSQL, каждая из которых решает свои специфические задачи:

### 1. MongoDB (документная база данных)

- Показала свою гибкость в работе со структурированными, но схемой-независимыми данными (фильмы с годами выпуска).
- С помощью агрегационного конвейера удалось эффективно группировать данные по десятилетиям, что демонстрирует мощь MongoDB в аналитических задачах без необходимости писать сложный код на стороне приложения.

### 2. Redis (ключ-значение хранилище, in-memory)

- Продемонстрировала скорость и простоту для операций с временными или лог-подобными данными.

- Использование списка `log_errors` и команды `LINDEX` показало, как Redis может служить буфером для логирования, мониторинга или очередей в реальном времени.
3. Neo4j –
- возможность находить связи между сущностями (актёры ↔ режиссёры) иллюстрирует силу графовых БД в задачах, где важны отношения, а не только атрибуты объектов.