

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

Нургалеева Гузель Рустэмовна БД-241м

Инструменты хранения и анализа больших данных

Практическая работа 2.1. Изучение методов хранения данных на основе NoSQL

Вариант задания: 18

Направление подготовки/специальность

38.04.05 - Бизнес-информатика

Бизнес-аналитика и большие данные

(очная форма обучения)

Москва

2025

Содержание:

1 часть. Mongo DB

2 часть. Cassandra

1 часть. Mongo DB

Цель работы: получить практические навыки работы с документо-ориентированной базой данных MongoDB путем выполнения основных операций по управлению данными.

Вариант 18.

1. Создайте коллекцию "quotes" с известными цитатами из фильмов
2. Найдите фильмы с максимальным количеством продюсеров
3. Добавьте поле "prequels" для фильмов, имеющих предыстории
4. Создайте индекс по полю tradeMarks в коллекции persons
5. Определите режиссеров с наибольшим количеством фильмов в базе

Используемое программное обеспечение:

MongoDB

Mongo Compass

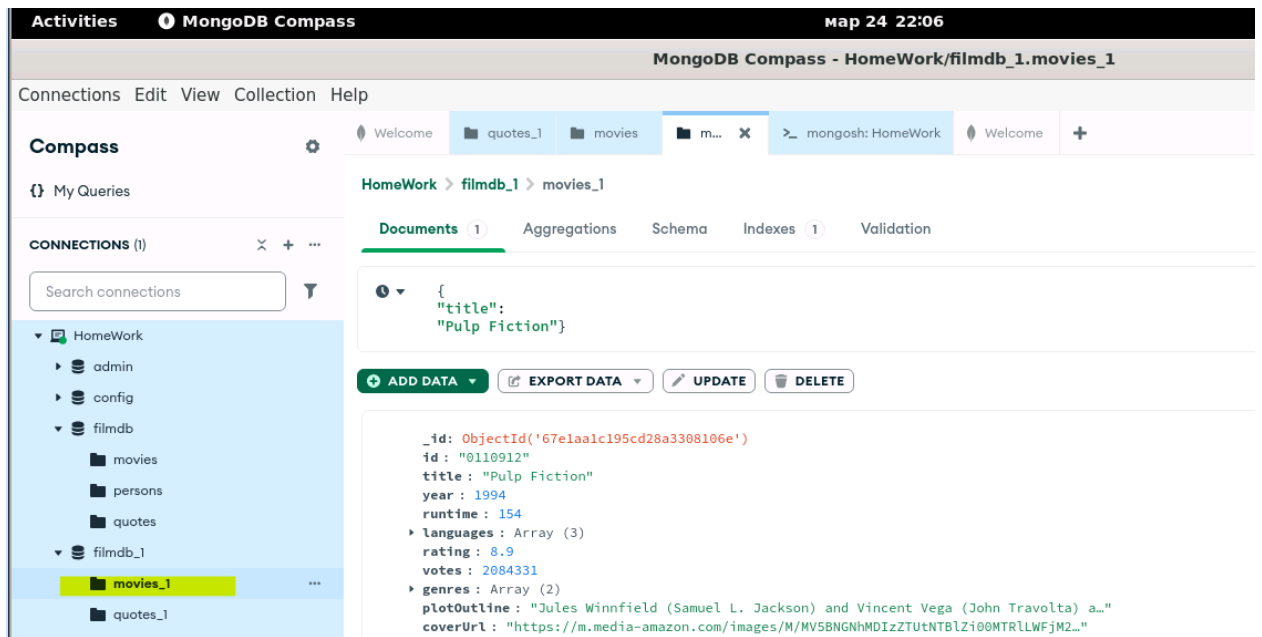
Листинг команд и скриншоты результатов выполнения команд

```
cd nonrel/  
cd mongo/  
ls  
sudo docker compose stop  
sudo docker compose start  
sudo docker compose down  
sudo docker compose up -d  
sudo docker exec -ti mongo-1 mongosh -u "root" -p "abc123!"
```

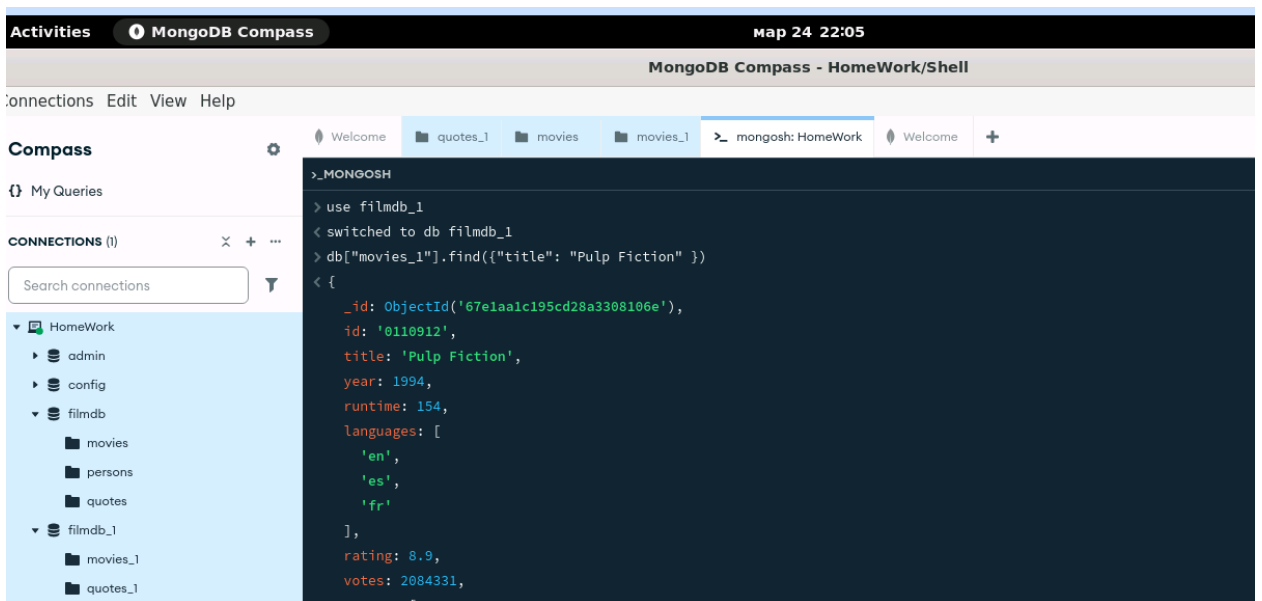
```
dev@dev-vm:~/Downloads/dba/nonrel/mongo$ sudo docker exec -ti mongo-1 mongosh -u "root" -p "abc123!"  
Current Mongosh Log ID: 67e1a44bd35a746d9ce43268  
Connecting to:      mongodb://<credentials>@127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.3.9  
Using MongoDB:      7.0.17-rc1  
Using Mongosh:       2.3.9  
  
For mongosh info see: https://www.mongodb.com/docs/mongosh-shell/  
  
-----  
The server generated these startup warnings when booting  
2025-03-24T18:25:00.659+00:00: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://dochub.mongodb.org/core/pr  
odnotes-filesystem  
2025-03-24T18:25:05.069+00:00: vm.max_map_count is too low  
-----  
test>
```

Создала дополнительную БД (filmbd_1), чтобы создать в ней коллекцию movies_1 и внести фильмы.

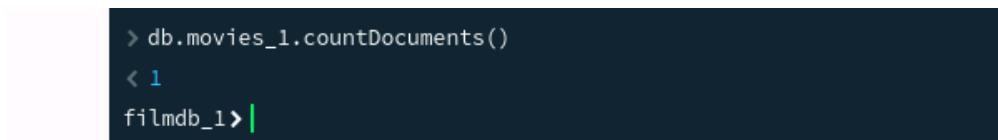
Внесение первого фильма и поиск его:



Проверка вставки



Подсчет документов в коллекции



Добавление документа в коллекцию movies_1:

```
> db.movies_1.insertOne(
  {
    "id": "0133093",
    "title": "The Matrix",
    "year": 1999,
    "runtime": 136,
    "languages": ["en"],
    "rating": 8.7,
    "votes": 1496538,
    "genres": ["Action", "Sci-Fi"],
    "plotOutline": "Thomas A. Anderson is a man living two",
    "coverUrl": "https://m.media-amazon.com/images/M/MV5BN"
  }
)
```

Подсчет количества документов:

```
> db.movies_1.countDocuments()
< 2
filmdb_1>
```

Вывод названий коллекций в БД filmdb_1:

```
> db.getCollectionNames()
< [ 'movies_1', 'quotes_1' ]
```

Индексы в коллекции movies_1:

```
> db.movies_1.getIndexes()
< [ { v: 2, key: { _id: 1 }, name: '_id_' } ]
```

----- Далее все действия в БД filmdb -----

В Коллекцию persons (filmdb) добавила 2 актеров

```
> db.persons.countDocuments()
< 4
```

Предполагаемое кол-во на основе метаданных

```
> db.persons.estimatedDocumentCount()
< 4
```

В movies добавлены фильмы

```
> db.movies.countDocuments()
< 50
```

Поиск фильмов по жанру

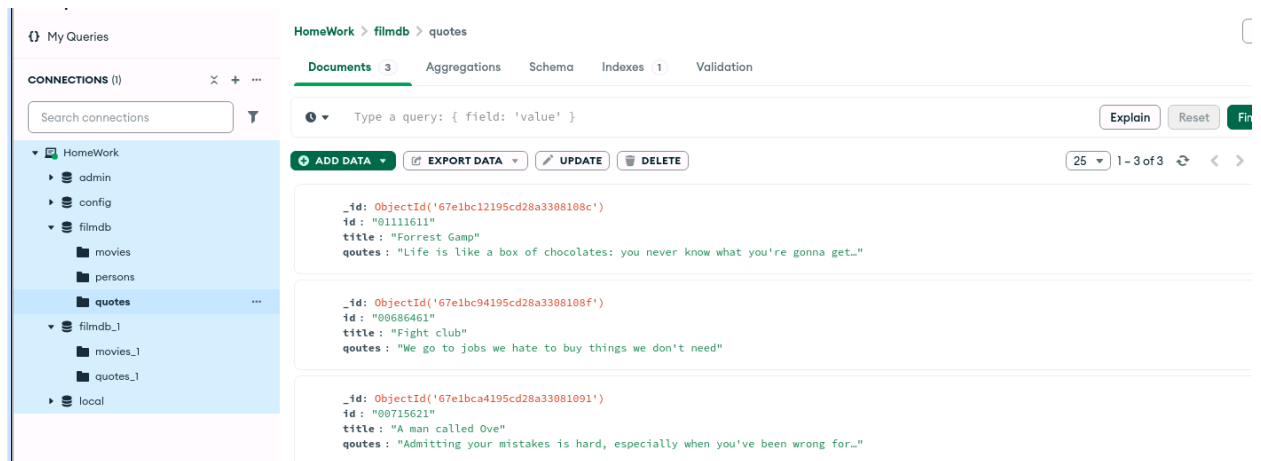
```
> db.movies.find({"genres": "Family"})
< {
  _id: ObjectId('67e1b433dfd46e1a4f1b47a1'),
  id: '0038650',
  title: "It's a Wonderful Life",
  genres: [
    'Drama',
    'Family',
    'Fantasy'
  ],
  year: 1946,
  rating: 8.6,
  rank: 25
}
```

-----Вариант 18-----

Вариант 18.

1. Создайте коллекцию "quotes" с известными цитатами из фильмов
2. Найдите фильмы с максимальным количеством продюсеров
3. Добавьте поле "prequels" для фильмов, имеющих предыстории
4. Создайте индекс по полю tradeMarks в коллекции persons
5. Определите режиссеров с наибольшим количеством фильмов в базе

1)



2)

```
db.movies.aggregate([
  {$unwind: "$producers" },
  {$group: { _id: '$title',
    number :{ $sum:1 }
  }},
  {$sort:{number:-1}} ])
```

```

> db.movies.aggregate([
    {$unwind: "$producers" },
    {$group: {_id:'$title',
              number :{ $sum:1 }
            }},
    {$sort:{number:-1}} ])

< {
  _id: 'Pulp Fiction',
  number: 7
}
{
  _id: 'The Matrix',
  number: 3
}
filmdb>

```

3)

```

> db.movies.updateOne ( {title: 'The Lord of the Rings: The Fellowship of the Ring'} , { $set: {'prequel': 'yes'} } )
< {
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
filmdb>

```

Documents 2
Aggregations
Schema
Indexes 1
Validation

🕒
{"title": "The Lord of the Rings: The Fellowship of the Ring"}

+ ADD DATA
EXPORT DATA
UPDATE
DELETE

```

_id: ObjectId('67e1b433dfd46e1a4f1b4795')
id: "0120737"
title: "The Lord of the Rings: The Fellowship of the Ring"
▶ genres: Array (3)
year: 2001
rating: 8.8
rank: 12
prequel: "yes"

```

4)

Name & Definition	Type	Size	Usage	Properties	Status
> _id	REGULAR	36.9 KB	1 (since Mon Mar 24 2025)	UNIQUE	READY
tradeMarks_1	REGULAR	20.5 KB	0 (since Mon Mar 24 2025)		READY

5) для большинства фильмов режиссер не указан

```

> db.movies.aggregate( [{ $group: { _id: '$directors', total: { $sum: 1 } } } ])
< {
  _id: [
    {
      directorID: '0905154',
      name: 'Lana Wachowski'
    },
    {
      directorID: '0905152',
      name: 'Lilly Wachowski'
    }
  ],
  total: 1
},
{
  _id: null,
  total: 48
},
{
  _id: [
    {
      directorID: '0000233',
      name: 'Quentin Tarantino'
    }
  ],
  total: 1
}

```

2 часть. Cassandra

Цель работы: получить практические навыки работы с базой данных Cassandra, изучив основные операции по управлению данными, включая создание и использование ключей, таблиц, выполнение запросов CQL, а также работу с различными инструментами подключения и администрирования.

Вариант 18.

1. Создайте ключейс ecommerce с репликацией SimpleStrategy и коэффициентом репликации 1.
2. Создайте таблицу products в ключейсе ecommerce с полями product_id (int), name (text), price (decimal), category (text), stock (int) и первичным ключом product_id.
3. Вставьте три товара в таблицу products.
4. Выберите все товары, цена которых меньше 100.
5. Обновите поле stock для товара с product_id = 1, увеличив его на 50.

Используемое ПО:

- 1) cassandra:3.11
- 2) apache/zeppelin:0.10.0
- 3) ZEPPELIN_CASSANDRA_VERSION=3.11
- 4) trivadis/cassandra-web

Листинг команд из задания 18:

1) **CREATE KEYSPACE ecommerce WITH replication = {'class': 'SimpleStrategy', 'replication_factor': '1'};**

```
CREATE KEYSPACE ecommerce WITH replication = {'class': 'SimpleStrategy', 'replication_factor': '1'};
```

No Result [Last query execution info](#)

2)

```
DROP TABLE IF EXISTS ecommerce.products;
CREATE TABLE ecommerce.products (product_id int,
    name text,
    price decimal,
    category text,
    stock int,
    PRIMARY KEY (product_id)
);
```

```
DROP TABLE IF EXISTS ecommerce.products;
CREATE TABLE ecommerce.products (product_id int,
    name text,
    price decimal,
    category text,
    stock int,
    PRIMARY KEY (product_id)
);
```

No Result [Last query execution info](#)

3)

```
INSERT INTO ecommerce.products (product_id, name, price, category, stock)
VALUES (1,
```

```
    'apple',
    10.0,
    'fruits',
    20);
```

```
INSERT INTO ecommerce.products (product_id, name, price, category, stock)
VALUES (2,
```

```
    'oranges',
    30.0,
    'fruits',
    100);
```

```
INSERT INTO ecommerce.products (product_id, name, price, category, stock)
VALUES (3,
```

```
    'potatoes',
    10.0,
```



```
INSERT INTO ecommerce.products (product_id, name, price, category, stock)
VALUES (1,
        'apple',
        10.0,
        'fruits',
        20);

INSERT INTO ecommerce.products (product_id, name, price, category, stock)
VALUES (2,
        'oranges',
        30.0,
        'fruits',
        100);

INSERT INTO ecommerce.products (product_id, name, price, category, stock)
VALUES (3,
        'potatoes',
        10.0,
        'vegetables',
        200);

INSERT INTO ecommerce.products (product_id, name, price, category, stock)
VALUES (4,
        'pineapple',
        101.0,
        'fruits',
        10);
```

Last query execution info

FROM ecommerce.products;

FINISHED

product_id	category	name	price	stock
1	fruits	apple	10.0	20
2	fruits	oranges	30.0	100
4	fruits	pineapple	101.0	10
3	vegetables	potatoes	10.0	200

ALLOW FILTERING;

SELECT *
FROM ecommerce.products
WHERE price < 100
ALLOW FILTERING;

FINISHED

settings

product_id	category	name	price	stock
1	fruits	apple	10.0	20
2	fruits	oranges	30.0	100
3	vegetables	potatoes	10.0	200

5)

UPDATE ecommerce.products
SET stock = 70
WHERE product_id = 1;

```
UPDATE ecommerce.products  
SET stock = 70  
WHERE product_id = 1;
```

No Result [Last query execution info](#)

Проверка:

SELECT *
FROM ecommerce.products;

SELECT *
FROM ecommerce.products;

FINISHED

settings

product_id	category	name	price	stock
1	fruits	apple	10.0	70
2	fruits	oranges	30.0	100
4	fruits	pineapple	101.0	10
3	vegetables	potatoes	10.0	200