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

Кузьмина Дарья Юрьевна БД-241м

Инструменты хранения и анализа больших данных

Лабораторная работа 2.1 Часть 1.
Изучение методов хранения данных на основе NoSQL
Вариант 11

Направление подготовки/специальность
38.04.05 - Бизнес-информатика
Бизнес-аналитика и большие данные
(очная форма обучения)

Руководитель дисциплины:
Босенко Т.М., доцент департамента
информатики, управления и технологий,
доктор экономических наук

Москва
2025

Содержание

Введение

Цель

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

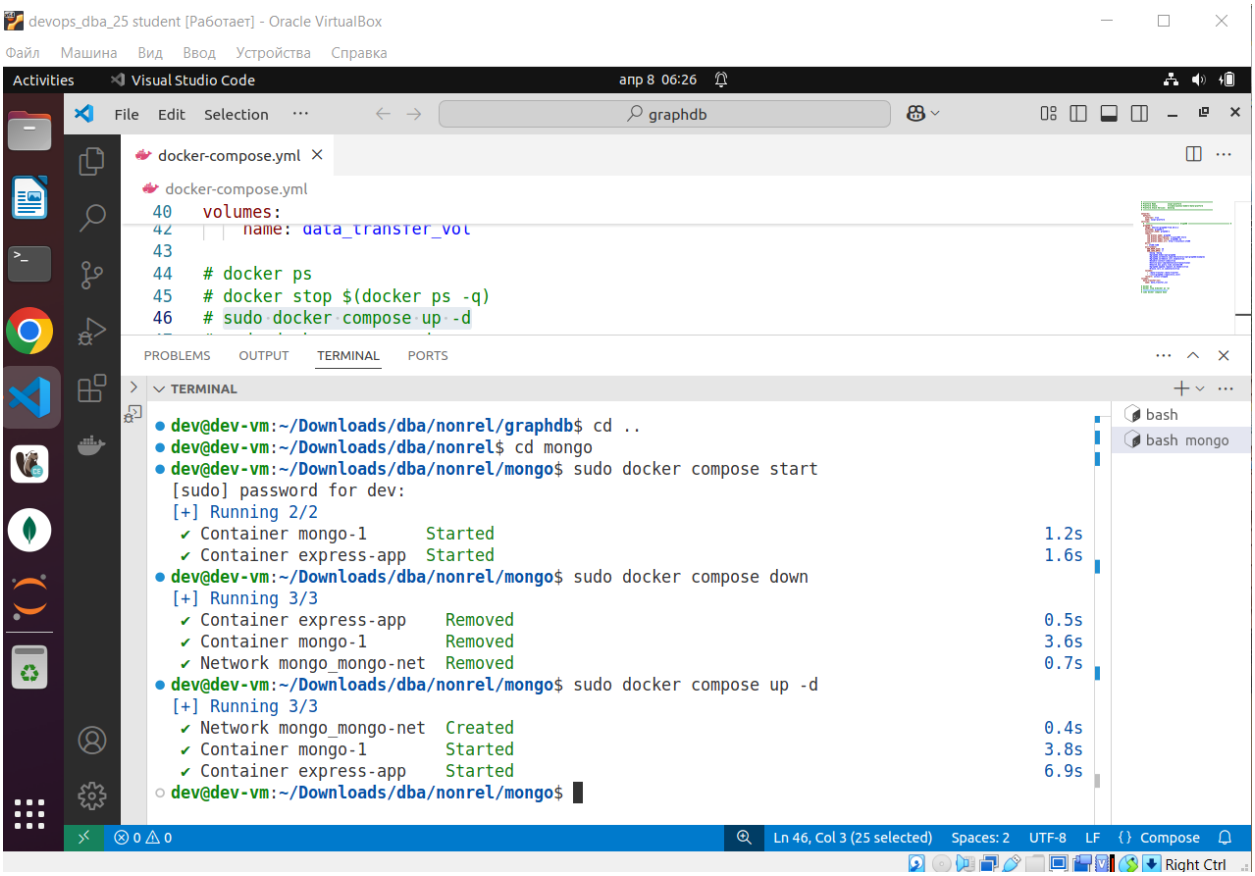
Задачи:
1. Изучить основы работы с MongoDB:
- Подключение к базе данных.
- Создание коллекций.
- Добавление документов.
2. Освоить основные операции с документами:
- Поиск документов.
- Обновление документов.
- Удаление документов.
3. Научиться работать с индексами:
- Создание индексов.
- Использование индексов для оптимизации.
4. Освоить базовые принципы агрегации данных:
- Группировка данных.
- Подсчет статистики.
- Сортировка результатов.
5. Научиться выполнять текстовый поиск:
- Создание текстовых индексов.
- Поиск по текстовым полям.

Необходимое ПО:

- **Операционная система**: Ubuntu 22.
- **СУБД**: GraphDB.
- **Docker**: для запуска контейнера с MongoDB.
- **Среда разработки**: MongoDB Compass - редактор для выполнения запросов.

Основная часть

Задача 1.

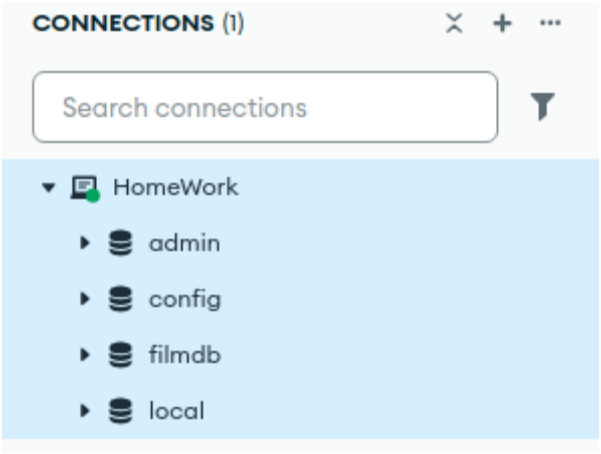


Запускаем докер композер

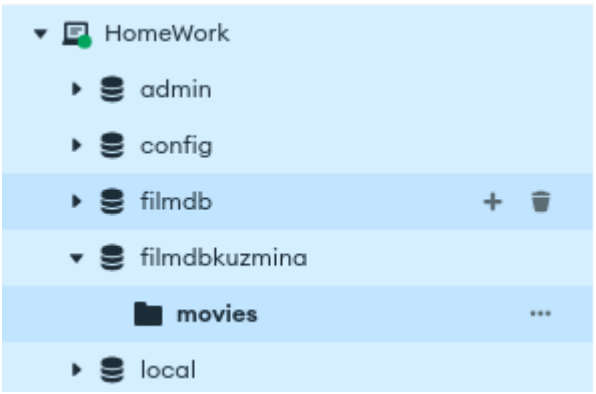
```
 ✔ Container express-app    Started                                                    6.9s
dev@dev-vm:~/Downloads/dba/nonrel/mongo$ sudo docker exec -ti mongo-1 mongosh -u "root" -p "ab
c123!"
Current Mongosh Log ID: 67f497f8a7715f1636e43268
Connecting to:          mongodb://<credentials>@127.0.0.1:27017/?directConnection=true&serverSelection
TimeoutMS=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/mongodb-shell/

------
   The server generated these startup warnings when booting
   2025-04-08T03:26:29.785+00:00: Using the XFS filesystem is strongly recommended with the WiredTiger
 storage engine. See http://dochub.mongodb.org/core/prodnotes-filesystem
   2025-04-08T03:26:37.369+00:00: vm.max_map_count is too low
------

test>
```

Проверяем, запущена ли служба в контейнере.

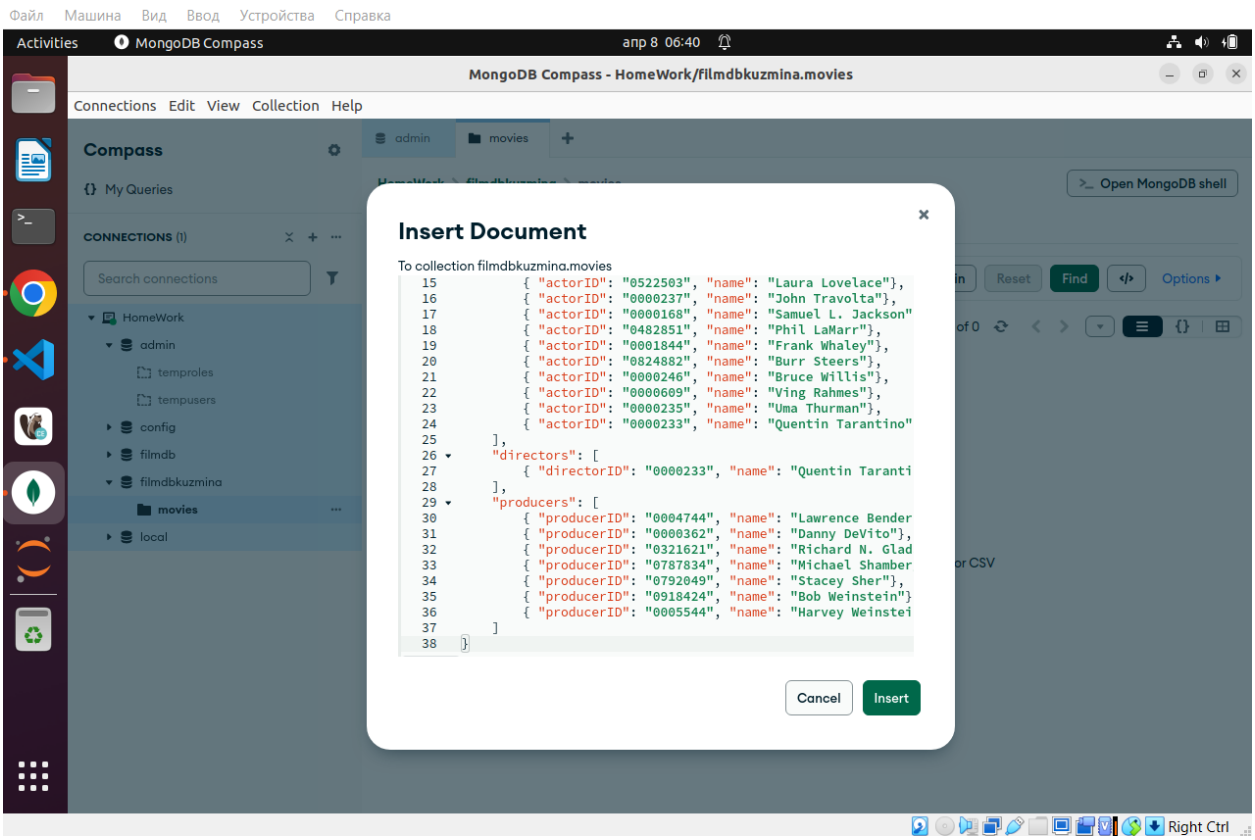
Сервер доступен, версия верная, стабильная, команды будут работать, мы находимся в тестовой базе данных.

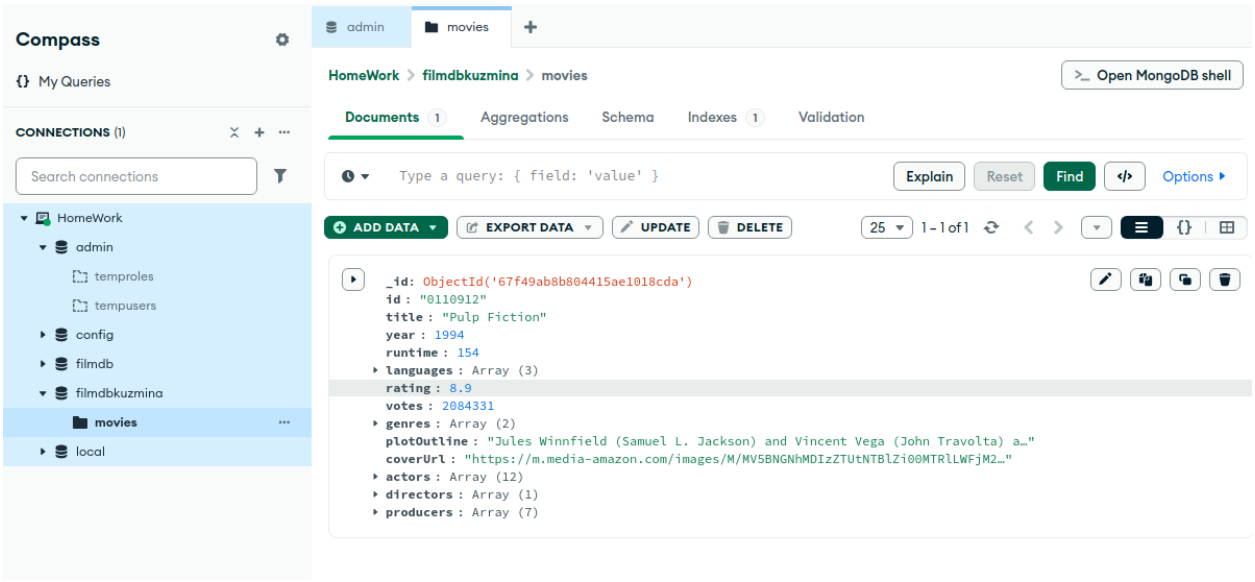


Подключились монго дБ компас

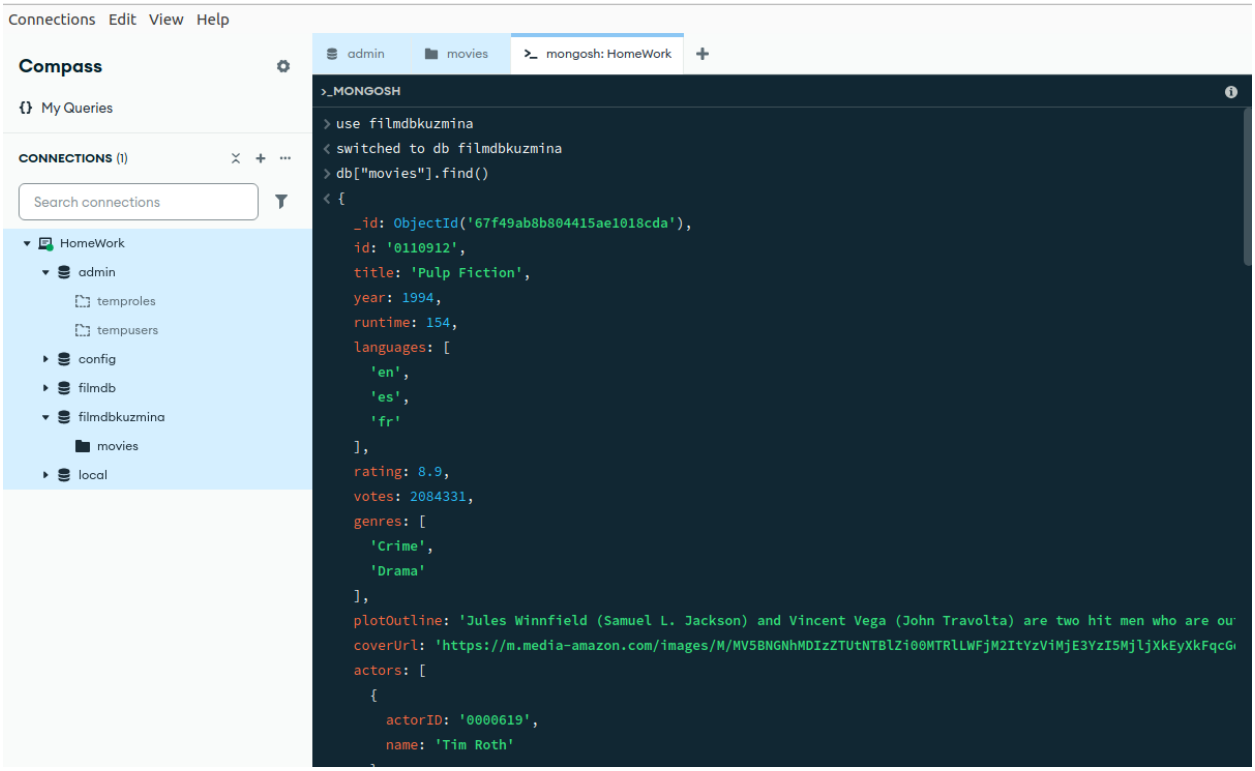


Удаляем ссылку, вставляем свой документ для импорта данных

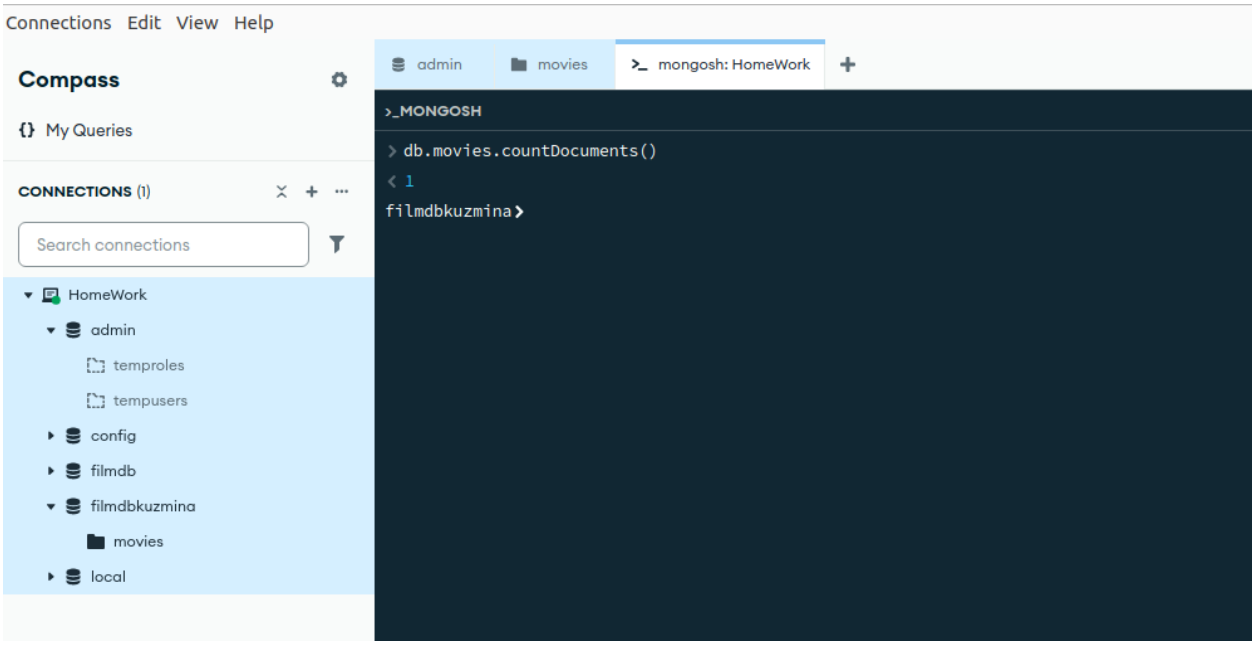




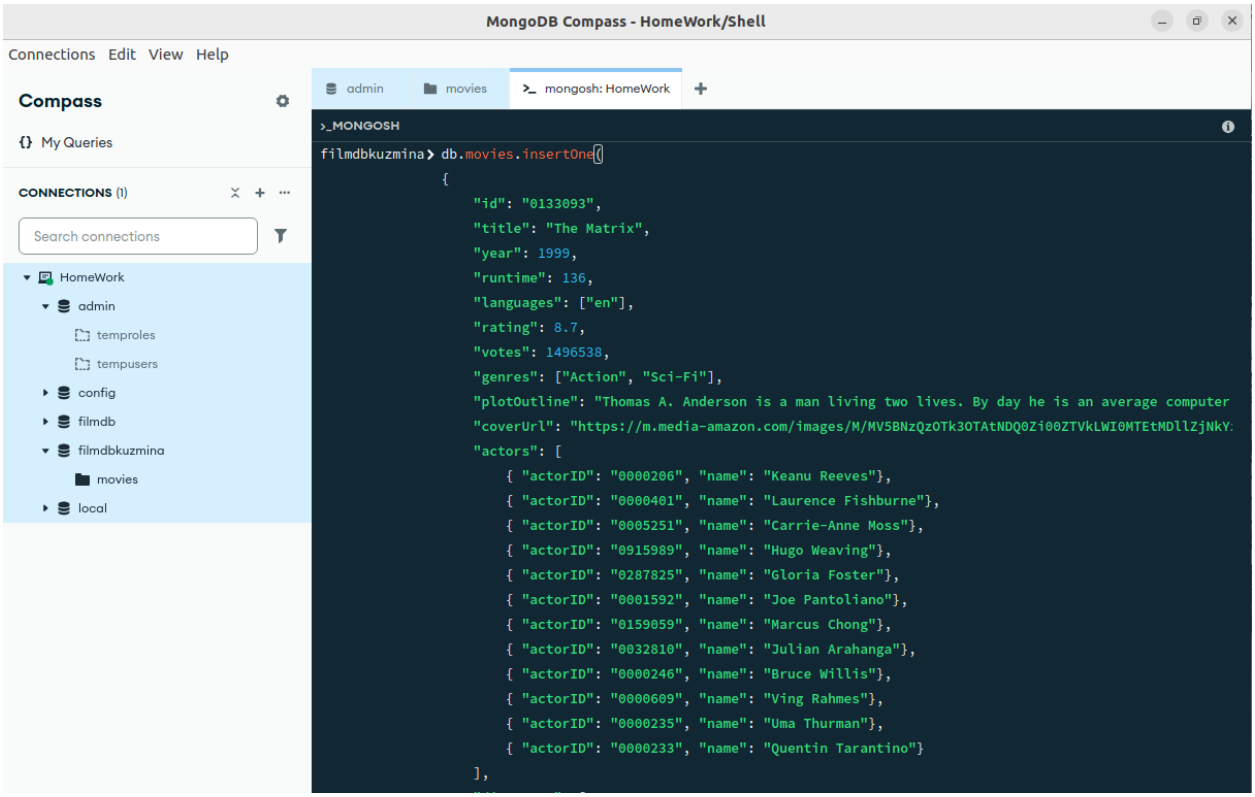
Проверяем видимость коллекции через терминал



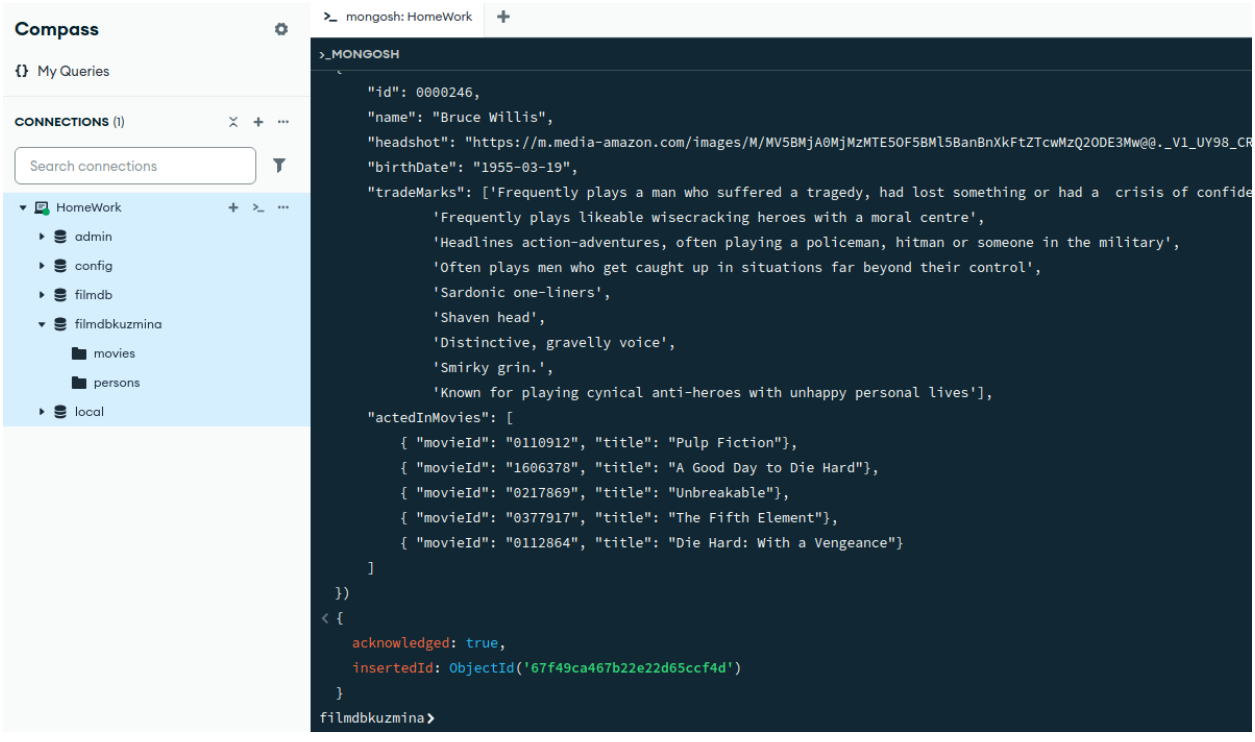
Выполнили подсчет документов в базе

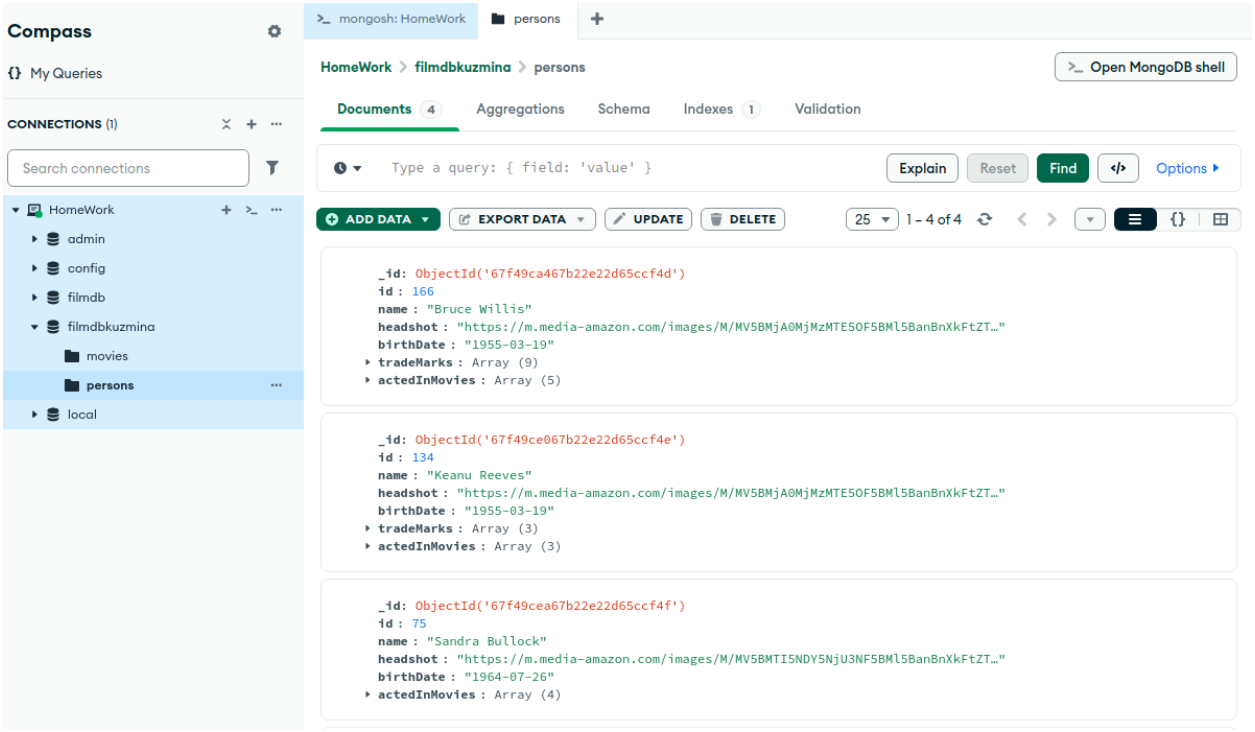


Добавим еще один фильм



Добавим персон





Задаем новый рейтинг.

```
> db.movies.updateOne ( {title: 'Fight Club'} , { $set: {rating: 9} } )
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
filmdbkuzmina>
```

Оптимизация производительности создаем тайтл.

```
> db.movies.createIndex( {title: 1} );
< title_1
filmdbkuzmina>
```

Индивидуальное задание **Вариант 11**

Пояснения к заданиям варианта 11:

1. Создайте коллекцию "locations" с информацией о местах съёмок для 5 фильмов

2. Найдите все фильмы с exactly 2 режиссерами

3. Добавьте поле "remakes" в виде массива для фильмов до 1980 года

4. Создайте индекс по полю runtime

5. Посчитайте количество фильмов каждого жанра для каждого десятилетия

1. Создание коллекции "locations" с информацией о местах съёмок для 5 фильмов

Цель задания — создать новую коллекцию для хранения данных о локациях съемок фильмов. Для этого используется команда «insertMany()», которая добавляет несколько документов за один запрос. Каждый документ содержит идентификатор фильма («filmId»), его название («title») и массив мест съемок («locations»). Примеры данных включают города и страны, где снимались фильмы, такие как "Los Angeles, USA" для "Pulp Fiction".

```
db.locations.insertMany([
  {
    filmId: "tt0110912",
    title: "Pulp Fiction",
    locations: ["Los Angeles, USA", "Paris, France"]
  },
  {
    filmId: "tt0133093",
    title: "The Matrix",
    locations: ["Sydney, Australia", "San Francisco, USA"]
  },
  {
    filmId: "tt1375666",
    title: "Inception",
    locations: ["Tokyo, Japan", "London, UK", "Morocco"]
  },
```

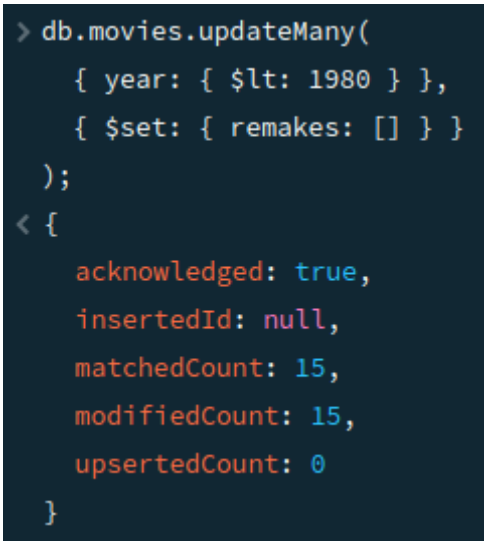
```
  {
    filmId: "tt0068646",
    title: "The Godfather",
    locations: ["New York, USA", "Sicily, Italy"]
  },
  {
    filmId: "tt0088763",
    title: "Back to the Future",
    locations: ["Los Angeles, USA", "Hill Valley (фиктивное место)"]
  }
]);
```

2. Поиск фильмов с точно 2 режиссерами

Задача — найти фильмы, у которых в списке режиссеров («directors») ровно два человека. Для этого применяется оператор «$size», который проверяет длину массива. Запрос «db.movies.find({ "directors": { $size: 2 } })» вернет все подходящие документы. Важно учитывать, что если поле «directors» отсутствует или содержит не массив, такие документы не будут найдены.

3. Добавление поля "remakes" для фильмов до 1980 года

Цель — добавить новое поле «remakes» (ремейки) в виде пустого массива к фильмам, выпущенным до 1980 года. Используется команда «updateMany()» с условием «{ year: { $lt: 1980 } }» для выбора старых фильмов и оператором «$set» для добавления поля. Если таких фильмов нет в коллекции, запрос не изменит документы.



4. Создание индекса по полю runtime

Индекс по полю «runtime» (продолжительность фильма) ускоряет поиск и сортировку по этому параметру. Команда «db.movies.createIndex({ runtime: 1

})» создает индекс с сортировкой по возрастанию. Индексы улучшают производительность чтения, но могут замедлить операции записи.

```
> db.movies.createIndex({ runtime: 1 });
< runtime_1
filmdbkuzmina>
```

5. Подсчет количества фильмов каждого жанра для каждого десятилетия

Задача — проанализировать распределение жанров по десятилетиям. Для этого используется агрегация:

- «$unwind» разбивает массив жанров на отдельные документы.

- «$project» вычисляет десятилетие как «(год // 10) 10» (например, 1994 → 1990).

- «$group» группирует данные по жанру и десятилетию, подсчитывая количество фильмов.

- «$sort» упорядочивает результат по десятилетию и жанру.

```
> db.movies.aggregate([
    { $unwind: "$genres" },
    {
      $project: {
        decade: {
          $floor: { $divide: ["$year", 10] }
        },
        genre: "$genres"
      }
    },
    {
      $group: {
        _id: {
          genre: "$genre",
          decade: { $multiply: ["$decade", 10] }
        },
        count: { $sum: 1 }
      }
    },
    { $sort: { "_id.decade": 1, "_id.genre": 1 } }
  ]);
< {
```

```
{
  _id: {
    genre: 'Drama',
    decade: 1930
  },
  count: 2
}
{
  _id: {
    genre: 'Family',
    decade: 1930
  },
  count: 1
}
{
  _id: {
    genre: 'Romance',
    decade: 1930
  },
  count: 2
}
{
  _id: {
    genre: 'Drama',
    decade: 1940
```


Выгружаем csv.

Заключение

Вывод:

В ходе лабораторной работы были освоены основные операции работы с MongoDB, включая создание и управление базами данных, коллекциями и документами. Изучены методы поиска, обновления и удаления данных, а также работа с индексами и агрегацией. Практические задания позволили закрепить навыки использования командной строки и графических интерфейсов для взаимодействия с MongoDB. Работа продемонстрировала гибкость и эффективность документо-ориентированной СУБД для обработки структурированных и неструктурированных данных.