

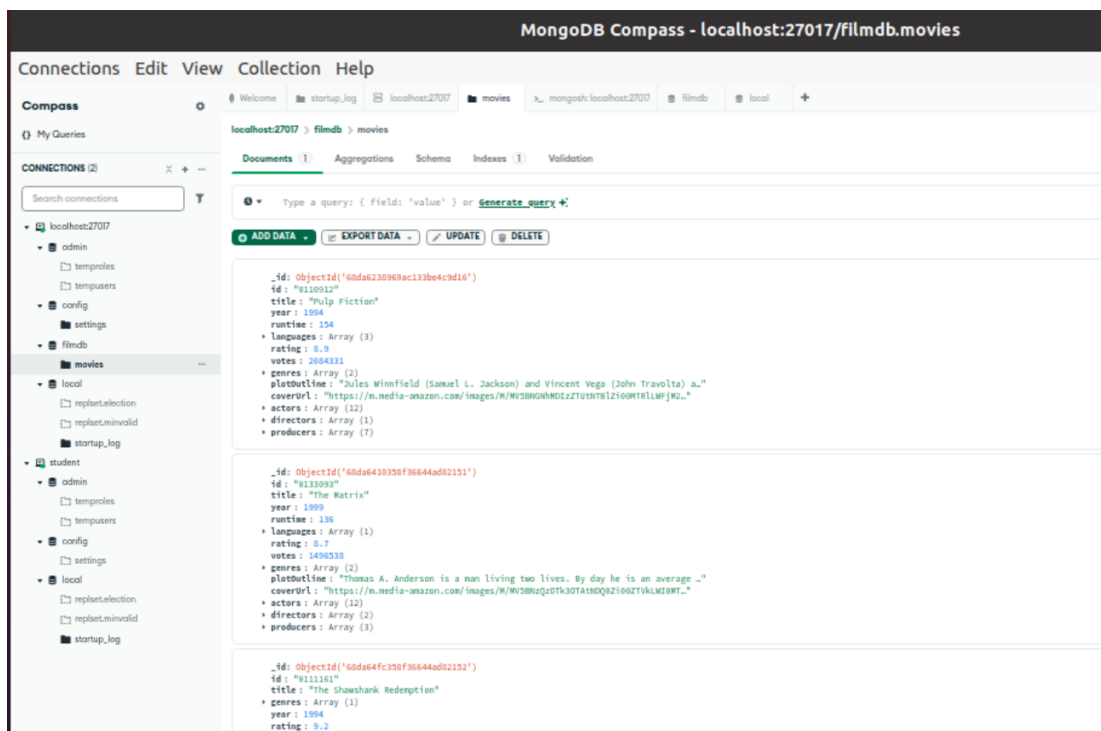
## Отчет по лабораторной работе 2.1

Выполнил Бойко Константин

Студент группы АДЭУ-221

### Практическое задание **Вариант 20**

**Задание 1. Найти все фильмы с рейтингом 9.0 или выше. Вывести только их названия и год выпуска, исключив поле `_id`. MongoDB**



Предварительно мы создали 50 фильмов в MongoCompass

```
> db.movies.find().count()
< 50
```

Функция `gte` - greater than or equal поможет нам найти фильмы с рейтингом `>= 9,0`

```

> db.movies.find(
  { "rating": { $gte: 9 } },
  { title: 1, year: 1, _id: 0 }
)
< {
  title: 'The Shawshank Redemption',
  year: 1994
}
{
  title: 'The Godfather',
  year: 1972
}
{
  title: 'The Godfather: Part II',
  year: 1974
}
{
  title: 'The Dark Knight',
  year: 2008
}

```

Получаем искомое

## Задание 2. Найти актеров, которые снимались только в одном фильме из представленных в базе. NEO4J

Всего в базе 133 актера. Найдём всех актеров с одним фильмом в портфолио:

```

MATCH (p:Person)-[:ACTED_IN]->(m:Movie) WITH p, COUNT(m) AS moviesCount
WHERE moviesCount = 1 RETURN p.name AS Актёр

```

Мы получили список из 67 актеров с одним фильмом в портфолио, на графе это проверить визуально не представляется возможным

|    |                 |
|----|-----------------|
| 62 | "Ed Harris"     |
| 63 | "Diane Keaton"  |
| 64 | "Julia Roberts" |
| 65 | "Madonna"       |
| 66 | "Geena Davis"   |
| 67 | "Lori Petty"    |

**Задание 3. В список queue:priority добавить 3 задачи. Переместить последнюю добавленную задачу в начало списка. Redis**

Redis — это in-memory база данных, часто используемая для очередей (queues). Мы будем работать со структурой list, ключом queue:priority. Задачи — это простые строки (например, "task1", "task2", "task3"). Добавление в очередь: используем RPUSH (добавляем в конец списка). Перемещение: вынимаем последнюю (из конца) и добавляем в начало.

Подключаемся к докеру Redis-cli, на всякий случай удаляем ключ queue:priority

```
mgpu@mgpu-vm:~$ docker exec -it redis-1 redis-cli
127.0.0.1:6379> DEL queue:priority
(integer) 1
127.0.0.1:6379> DEL queue:priority
(integer) 0
```

4

Добавляем новые задачи в ключе, он создается автоматически. Проверяем их наличие

```

127.0.0.1:6379> RPush queue:priority "task1"
(integer) 1
127.0.0.1:6379> RPush queue:priority "task2"
(integer) 2
127.0.0.1:6379> RPush queue:priority "task3"
(integer) 3
127.0.0.1:6379> LRange queue:priority 0 -1
1) "task1"
2) "task2"
3) "task3"

```

Вытаскиваем из очереди 3 задачу вставляем ее первой. Порядок изменился

```

127.0.0.1:6379> RPop queue:priority
"task3"
127.0.0.1:6379> LPush queue:priority "task3"
(integer) 3
127.0.0.1:6379> LRange queue:priority 0 -1
1) "task3"
2) "task1"
3) "task2"

```

В RedisComander отражаются изменения

The screenshot shows the Redis Commander web interface. On the left, a tree view shows the selected key 'queue:priority' under the 'local (redis-1:6379:0)' instance. The main panel displays the details for 'Key: queue:priority', including 'TTL: -1' and 'Type: List (3 items)'. Below this, a table lists the elements of the list in order from index 0 to 2.

| # | Value |
|---|-------|
| 0 | task3 |
| 1 | task1 |
| 2 | task2 |