

Национальный исследовательский Университет ИТМО
Мегафакультет информационных и трансляционных технологий
Факультет инфокоммуникационных технологий

Проектирование и реализация баз данных

"Работа с БД в СУБД MongoDB"

Лабораторная работа 5.2

Работу
выполнил:
Нестеров В.А.
Группа: К3243
Преподаватель:
Говорова М.М.

Санкт-Петербург
2022

Содержание

1. Цель работы	4
2. Практическое задание	4
3. Выполнение	4
3.1. Создадим базу данных learn	4
3.2. Заполним коллекцию единорогов unicorns	4
3.3. Используя второй способ, вставим в коллекцию единорогов документ . .	4
3.4. Проверим содержимое коллекции с помощью метода find	5
3.5. Сформируем запросы для вывода списков самцов и самок единорогов. Ограничим список самок первыми тремя особями. Отсортируем списки по имени	5
3.6. Найдём всех самок, которые любят carrot. Ограничим этот список первой особью с помощью функций findOne и limit	6
3.7. Модифицируйте запрос для вывода списков самцов единорогов, исключив из результата информацию о предпочтениях и поле	6
3.8. Вывести список единорогов в обратном порядке добавления	6
3.9. Вывести список единорогов с названием первого любимого предпочтения, исключив идентификатор	7
3.10. Вывести список самок единорогов весом от полутонны до 700 кг, исключив вывод идентификатора.	7
3.11. Вывести список самцов единорогов весом от полутонны и предпочитающих ггаре и lemon, исключив вывод идентификатора	7
3.12. Найти всех единорогов, не имеющих ключ vampires	7
3.13. Вывести список упорядоченный список имен самцов единорогов с информацией об их первом предпочтении	8
3.14. Создайте коллекцию towns, включающую следующие документы	8
3.15. Сформировать запрос, который возвращает список городов с независимыми мэрами (party="I"). Вывести только название города и информацию о мэре	8
3.16. Сформировать запрос, который возвращает список беспартийных мэров (party отсутствует). Вывести только название города и информацию о мэре	9
3.17. Сформировать функцию для вывода списка самцов единорогов	9
3.18. Создать курсор для этого списка из первых двух особей с сортировкой в лексикографическом порядке	9
3.19. Вывести результат, используя forEach	9
3.20. Вывести количество самок единорогов весом от полутонны до 600 кг . . .	9
3.21. Вывести список предпочтений	10
3.22. Посчитать количество особей единорогов обоих полов. с учётом Group из следующих заданий	10
3.23. Выполнить команду	10
3.24. Проверить содержимое коллекции unicorns	10
3.25. Для самки единорога Аупа внести изменения в БД: теперь ее вес 800, она убила 51 вампира	11
3.26. Проверить содержимое коллекции unicorns	11

3.27. Для самца единорога Raleigh внести изменения в БД: теперь он любит рэдбул	11
3.28. Проверить содержимое коллекции unicorns	11
3.29. Всем самцам единорогов увеличить количество убитых вампиров на 5 . .	11
3.30. Проверить содержимое коллекции unicorns	12
3.31. Изменить информацию о городе Портланд: мэр этого города теперь беспартийный	12
3.32. Проверить содержимое коллекции towns	12
3.33. Изменить информацию о самце единорога Pilot: теперь он любит и шоколад	12
3.34. Проверить содержимое коллекции unicorns	12
3.35. Изменить информацию о самке единорога Aurora: теперь она любит еще и сахар, и лимоны	12
3.36. Проверить содержимое коллекции unicorns	13
3.37. Создайте коллекцию towns, включающую следующие документы	13
3.38. Удалите документы с беспартийными мэрами	13
3.39. Проверьте содержание коллекции	13
3.40. Очистите коллекцию	14
3.41. Просмотрите список доступных коллекций	14
3.42. Создайте коллекцию зон обитания единорогов, указав в качестве идентификатора кратко название зоны, далее включив полное название и описание	14
3.43. Включите для нескольких единорогов в документы ссылку на зону обитания, используя второй способ автоматического связывания	14
3.44. Проверьте содержание коллекции единорогов	14
3.45. Обновить содержание коллекции единорогов unicorns	15
3.46. Проверьте, можно ли задать для коллекции unicorns индекс для ключа name с флагом unique	15
3.47. Обновить содержание коллекции единорогов unicorns	15
3.48. Получите информацию о всех индексах коллекции unicorns	15
3.49. Удалите все индексы, кроме индекса для идентификатора	16
3.50. Попытайтесь удалить индекс для идентификатора	16
3.51. Создайте объемную коллекцию numbers, задействовав курсор	16
3.52. Выберите последних четыре документа	16
3.53. Проанализируйте план выполнения запроса 2. Сколько потребовалось времени на выполнение запроса?	17
3.54. Создайте индекс для ключа value	17
3.55. Получите информацию о всех индексах коллекции numbers	17
3.56. Выполните запрос 2	17
3.57. Проанализируйте план выполнения запроса с установленным индексом. Сколько потребовалось времени на выполнение запроса?	18
3.58. Сравните время выполнения запросов с индексом и без. Дайте ответ на вопрос: какой запрос более эффективен?	18

4. Выводы 19

1. Цель работы

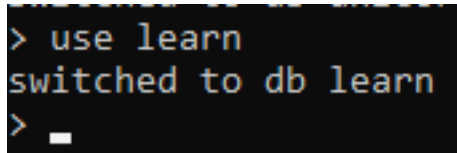
Овладеть практическими навыками работы с CRUD-операциями, с вложенными объектами в коллекции базы данных MongoDB, агрегации и изменения данных, со ссылками и индексами в базе данных MongoDB.

2. Практическое задание

Выполнить указанные в описании лабораторной работы команды.

3. Выполнение

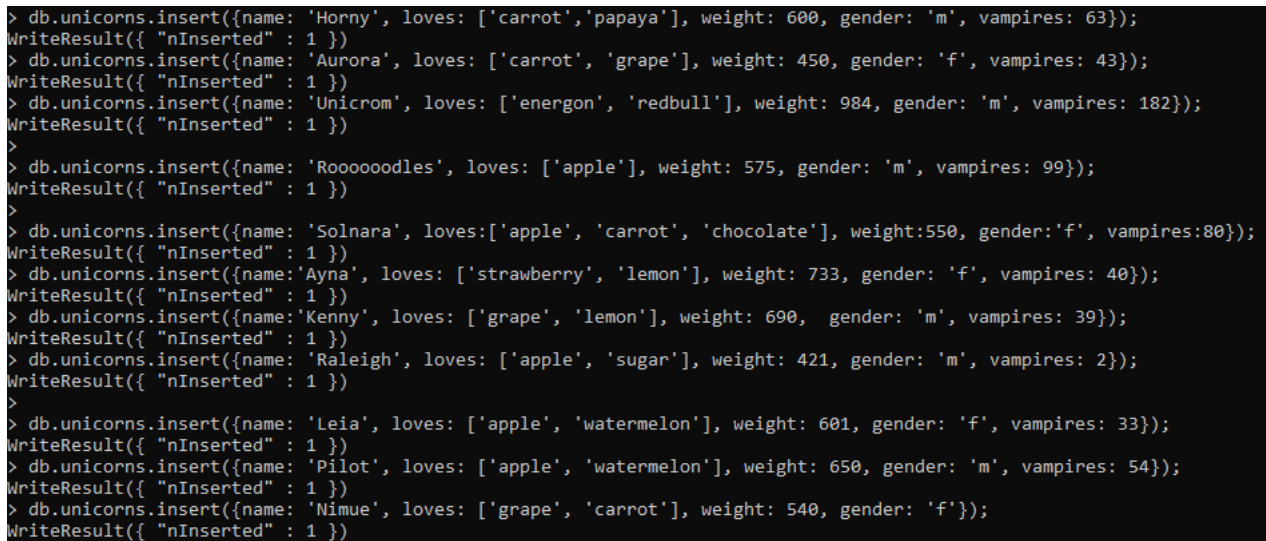
3.1. Создадим базу данных learn



```
> use learn
switched to db learn
> _
```

Рисунок 3.1. База данных learn

3.2. Заполним коллекцию единорогов unicorns



```
> db.unicorns.insert({name: 'Horny', loves: ['carrot', 'papaya'], weight: 600, gender: 'm', vampires: 63});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Aurora', loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Unicrom', loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
WriteResult({ "nInserted" : 1 })
>
> db.unicorns.insert({name: 'Rooodooles', loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
WriteResult({ "nInserted" : 1 })
>
> db.unicorns.insert({name: 'Solnara', loves: ['apple', 'carrot', 'chocolate'], weight: 550, gender: 'f', vampires: 80});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Kenny', loves: ['grape', 'lemon'], weight: 690, gender: 'm', vampires: 39});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
WriteResult({ "nInserted" : 1 })
>
> db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
WriteResult({ "nInserted" : 1 })
```

Рисунок 3.2. Заполним коллекцию unicorns

3.3. Используя второй способ, вставим в коллекцию единорогов документ

```
{name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165}
```

```
> document = {name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165}
{
  "name" : "Dunx",
  "loves" : [
    "grape",
    "watermelon"
  ],
  "weight" : 704,
  "gender" : "m",
  "vampires" : 165
}
> db.unicorns.insert(document)
WriteResult({ "nInserted" : 1 })
>
```

Рисунок 3.3. Вставка документа в коллекцию

3.4. Проверим содержимое коллекции с помощью метода find

```
> db.unicorns.find()
{ "_id" : ObjectId("62bdb685cd3f831260d66e69"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("62bdb68ccd3f831260d66e6a"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("62bdb68fcd3f831260d66e6b"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("62bdb69ccd3f831260d66e6c"), "name" : "Rooooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("62bdb6a2cd3f831260d66e6d"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("62bdb6b0cd3f831260d66e6e"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("62bdb6b5cd3f831260d66e6f"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("62bdb6b9cd3f831260d66e70"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("62bdb6c4cd3f831260d66e71"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("62bdb6c6cd3f831260d66e72"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("62bdb6c8cd3f831260d66e73"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("62bdb6cccd3f831260d66e74"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("62bdb6cccd3f831260d66e75"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("62bdb6cccd3f831260d66e76"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("62bdb7dcd3f831260d66e77"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
```

Рисунок 3.4. Содержимое коллекции с помощью find

3.5. Сформируем запросы для вывода списков самцов и самок единорогов. Ограничим список самок первыми тремя особями. Отсортируем списки по имени

```
> db.unicorns.find({gender: 'f'}).sort({name: 1}).limit(3)
{ "_id" : ObjectId("62bdb68ccd3f831260d66e6a"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("62bdb6b0cd3f831260d66e6e"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("62bdb6c4cd3f831260d66e71"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
```

Рисунок 3.5. Запрос списка самок единорогов

```
> db.unicorns.find({gender: 'f'}).sort({name: 1}).limit(3)
{ "_id" : ObjectId("62bdb68ccd3f831260d66e6a"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("62bdb6b0cd3f831260d66e6e"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("62bdb6c4cd3f831260d66e71"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
> db.unicorns.find({gender: 'm'}).sort({name: 1})
{ "_id" : ObjectId("62bdb7dcd3f831260d66e77"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("62bdb685cd3f831260d66e69"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("62bdb685cd3f831260d66e6f"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("62bdb6c4cd3f831260d66e72"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("62bdb6cccd3f831260d66e75"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("62bdb6b9cd3f831260d66e70"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("62bdb69ccd3f831260d66e6c"), "name" : "Rooooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("62bdb68fcd3f831260d66e6b"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
```

Рисунок 3.6. Запрос списка самцов единорогов

3.6. Найдём всех самок, которые любят carrot. Ограничим этот список первой особью с помощью функций findOne и limit

```
> db.unicorns.find({gender: "f", loves: "carrot"}).limit(1)
{ "_id" : ObjectId("62bdb68ccd3f831260d66e6a"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
>
```

Рисунок 3.7. Запрос первой самки, любящей carrot

3.7. Модифицируйте запрос для вывода списков самцов единорогов, исключив из результата информацию о предпочтениях и поле

```
> db.unicorns.find({gender: "m"}, {loves: 0, gender: 0})
{ "_id" : ObjectId("62bdb685cd3f831260d66e69"), "name" : "Horny", "weight" : 600, "vampires" : 63 }
{ "_id" : ObjectId("62bdb68fcd3f831260d66e6b"), "name" : "Unicrom", "weight" : 984, "vampires" : 182 }
{ "_id" : ObjectId("62bdb69ccd3f831260d66e6c"), "name" : "Rooooooodles", "weight" : 575, "vampires" : 99 }
{ "_id" : ObjectId("62bdb6b5cd3f831260d66e6f"), "name" : "Kenny", "weight" : 690, "vampires" : 39 }
{ "_id" : ObjectId("62bdb6b9cd3f831260d66e70"), "name" : "Raleigh", "weight" : 421, "vampires" : 2 }
{ "_id" : ObjectId("62bdb6c4cd3f831260d66e72"), "name" : "Pilot", "weight" : 650, "vampires" : 54 }
{ "_id" : ObjectId("62bdb6cccd3f831260d66e75"), "name" : "Pilot", "weight" : 650, "vampires" : 54 }
{ "_id" : ObjectId("62bdb7dccd3f831260d66e77"), "name" : "Dunx", "weight" : 704, "vampires" : 165 }
>
```

Рисунок 3.8. Список единорогов без информации о предпочтениях и поле

3.8. Вывести список единорогов в обратном порядке добавления

```
> db.unicorns.find({gender: "m"}, {loves: 0, gender: 0})
{ "_id" : ObjectId("62bdb685cd3f831260d66e69"), "name" : "Horny", "weight" : 600, "vampires" : 63 }
{ "_id" : ObjectId("62bdb68fcd3f831260d66e6b"), "name" : "Unicrom", "weight" : 984, "vampires" : 182 }
{ "_id" : ObjectId("62bdb69ccd3f831260d66e6c"), "name" : "Rooooooodles", "weight" : 575, "vampires" : 99 }
{ "_id" : ObjectId("62bdb6b5cd3f831260d66e6f"), "name" : "Kenny", "weight" : 690, "vampires" : 39 }
{ "_id" : ObjectId("62bdb6b9cd3f831260d66e70"), "name" : "Raleigh", "weight" : 421, "vampires" : 2 }
{ "_id" : ObjectId("62bdb6c4cd3f831260d66e72"), "name" : "Pilot", "weight" : 650, "vampires" : 54 }
{ "_id" : ObjectId("62bdb6cccd3f831260d66e75"), "name" : "Pilot", "weight" : 650, "vampires" : 54 }
{ "_id" : ObjectId("62bdb7dccd3f831260d66e77"), "name" : "Dunx", "weight" : 704, "vampires" : 165 }
> db.unicorns.find().sort({$natural: -1})
{ "_id" : ObjectId("62bdb7dccd3f831260d66e77"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("62bdb6cccd3f831260d66e75"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("62bdb6cccd3f831260d66e75"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("62bdb6cccd3f831260d66e74"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("62bdb6c4cd3f831260d66e73"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("62bdb6c4cd3f831260d66e72"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("62bdb6c4cd3f831260d66e71"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("62bdb6b9cd3f831260d66e70"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("62bdb6b5cd3f831260d66e6f"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("62bdb6b0cd3f831260d66e6e"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("62bdb6a2cd3f831260d66e6d"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("62bdb69ccd3f831260d66e6c"), "name" : "Rooooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("62bdb68fcd3f831260d66e6b"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("62bdb68ccd3f831260d66e6a"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("62bdb685cd3f831260d66e69"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
>
```

Рисунок 3.9. Список единорогов в обратном порядке добавления

3.9. Вывести список единорогов с названием первого любимого предпочтения, исключив идентификатор

```
> db.unicorns.find({}, {loves: {$slice: 1}, _id:0})
{ "name" : "Horny", "loves" : [ "carrot" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "name" : "Aurora", "loves" : [ "carrot" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "name" : "Unicrom", "loves" : [ "energon" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "name" : "Rooooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "name" : "Solnara", "loves" : [ "apple" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "name" : "Ayna", "loves" : [ "strawberry" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "name" : "Kenny", "loves" : [ "grape" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "name" : "Raleigh", "loves" : [ "apple" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "name" : "Leia", "loves" : [ "apple" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "name" : "Pilot", "loves" : [ "apple" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "name" : "Nimue", "loves" : [ "grape" ], "weight" : 540, "gender" : "f" }
{ "name" : "Leia", "loves" : [ "apple" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "name" : "Pilot", "loves" : [ "apple" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "name" : "Nimue", "loves" : [ "grape" ], "weight" : 540, "gender" : "f" }
{ "name" : "Dunx", "loves" : [ "grape" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
>
```

Рисунок 3.10

3.10. Вывести список самок единорогов весом от полутонны до 700 кг, исключив вывод идентификатора.

```
> db.unicorns.find({gender: "f", weight: {$gt: 500, $lt: 700}}, {_id: 0})
{ "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
>
```

Рисунок 3.11

3.11. Вывести список самцов единорогов весом от полутонны и предпочитающих grape и lemon, исключив вывод идентификатора

```
> db.unicorns.find({gender: 'm', weight: {$gt: 500}, loves: {$all: ['grape', 'lemon']}}, {_id: 0})
{ "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
>
```

Рисунок 3.12

3.12. Найти всех единорогов, не имеющих ключ vampires

```
> db.unicorns.find({vampires: {$exists: false}})
{ "_id" : ObjectId("62bdb6c4cd3f831260d66e73"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
```

Рисунок 3.13

3.13. Вывести список упорядоченный список имен самцов единорогов с информацией об их первом предпочтении

```
> db.unicorns.find({vampires: {$exists:false}})
{ "_id" : ObjectId("62bdb6c4cd3f831260d66e73"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("62bdb6cccd3f831260d66e76"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
> db.unicorns.find({gender: 'm'}, {loves: {$slice:1}}).sort({name:1})
{ "_id" : ObjectId("62bdb7dccc3f831260d66e77"), "name" : "Dunx", "loves" : [ "grape" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("62bdb685cd3f831260d66e69"), "name" : "Horny", "loves" : [ "carrot" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("62bdb6b5cd3f831260d66e6f"), "name" : "Kenny", "loves" : [ "grape" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("62bdb6c4cd3f831260d66e72"), "name" : "Pilot", "loves" : [ "apple" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("62bdb6cccd3f831260d66e75"), "name" : "Pilot", "loves" : [ "apple" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("62bdb6b9cd3f831260d66e70"), "name" : "Raleigh", "loves" : [ "apple" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("62bdb69ccd3f831260d66e6c"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("62bdb68fcd3f831260d66e6b"), "name" : "Unicrom", "loves" : [ "energon" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
```

Рисунок 3.14

3.14. Создайте коллекцию towns, включающую следующие документы

```
> db.towns.insert({name: "Punxsutawney ",
... populatiuon: 6200,
... last_sensus: ISODate("2008-01-31"),
... famous_for: [""],
... mayor: {
...   name: "Jim Wehrle"
... }})
WriteResult({ "nInserted" : 1 })
> db.towns.insert({name: "New York",
... populatiuon: 22200000,
... last_sensus: ISODate("2009-07-31"),
... famous_for: ["status of liberty", "food"],
... mayor: {
...   name: "Michael Bloomberg",
...   party: "I"}})
WriteResult({ "nInserted" : 1 })
> db.towns.insert({name: "Portland",
... populatiuon: 528000,
... last_sensus: ISODate("2009-07-20"),
... famous_for: ["beer", "food"],
... mayor: {
...   name: "Sam Adams",
...   party: "D"}})
WriteResult({ "nInserted" : 1 })
>
```

Рисунок 3.15

3.15. Сформировать запрос, который возвращает список городов с независимыми мэрами (party="I"). Вывести только название города и информацию о мэре

```
> db.towns.find({"mayor.party": "I"}, {"name":1, "mayor":1, "_id":0})
{ "name" : "New York", "mayor" : { "name" : "Michael Bloomberg", "party" : "I" } }
>
```

Рисунок 3.16

3.16. Сформировать запрос, который возвращает список беспартийных мэров (party отсутствует). Вывести только название города и информацию о мэре

```
> db.towns.find({"mayor.party": {$exists:false}}, {"name":1, "mayor":1, "_id":0})
{ "name" : "Punxsutawney ", "mayor" : { "name" : "Jim Wehrle" } }
```

Рисунок 3.17

3.17. Сформировать функцию для вывода списка самцов единорогов

```
> fn = function() { return this.gender=='m'; }
function() { return this.gender=='m'; }
> db.unicorns.find(fn)
{ "_id" : ObjectId("62bdb685cd3f831260d66e69"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("62bdb68fcd3f831260d66e6b"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("62bdb69ccd3f831260d66e6c"), "name" : "Rooooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("62bdb6b5cd3f831260d66e6f"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("62bdb6b9cd3f831260d66e70"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("62bdb6c4cd3f831260d66e72"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("62bdb7dccc3f831260d66e77"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
```

Рисунок 3.18

3.18. Создать курсор для этого списка из первых двух особей с сортировкой в лексикографическом порядке

```
> var cursor = db.unicorns.find({gender: "m"});null;
null
> cursor.sort({name: 1}).limit(2);null;
null
>
```

Рисунок 3.19

3.19. Вывести результат, используя forEach

```
> cursor.forEach(function(fn) { print (fn.name); })
Dunx
Horny
```

Рисунок 3.20

3.20. Вывести количество самок единорогов весом от полутонны до 600 кг

```
> db.unicorns.find({gender: "f", weight: {$gt:500, $lt:600}}).count()
2
```

Рисунок 3.21

3.21. Вывести список предпочтений

```
> db.unicorns.distinct("loves")
[
  "apple",
  "carrot",
  "chocolate",
  "energon",
  "grape",
  "lemon",
  "papaya",
  "redbull",
  "strawberry",
  "sugar",
  "watermelon"
]
```

Рисунок 3.22

3.22. Посчитать количество особей единорогов обоих полов. с учётом Barney из следующих заданий

```
> db.unicorns.aggregate({"$group": {"_id": "$gender", "count": {"$sum": 1}}})
{ "_id" : "f", "count" : 5 }
{ "_id" : "m", "count" : 7 }
```

Рисунок 3.23

3.23. Выполнить команду

```
> db.unicorns.save({name: 'Barney', loves: ['grape'],
... weight: 340, gender: 'm'})
writeResult({ "ninserted" : 1 })
```

Рисунок 3.24

3.24. Проверить содержимое коллекции unicorns

```
> db.unicorns.find()
{ "_id" : ObjectId("62bdb685cd3f831260d66e69"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("62bdb68ccd3f831260d66e6a"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("62bdb68fcd3f831260d66e6b"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("62bdb69ccd3f831260d66e6c"), "name" : "Rooooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("62bdb6a2cd3f831260d66e6d"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("62bdb6b0cd3f831260d66e6e"), "name" : "Ayna", "loves" : [ "strawberry", "lemon" ], "weight" : 733, "gender" : "f", "vampires" : 40 }
{ "_id" : ObjectId("62bdb6b5cd3f831260d66e6f"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("62bdb6b9cd3f831260d66e70"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("62bdb6c4cd3f831260d66e71"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("62bdb6c4cd3f831260d66e72"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("62bdb6c4cd3f831260d66e73"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("62bdb7dcd3f831260d66e77"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("62bdde7290ac4880590ae804"), "name" : "Barney", "loves" : [ "grape" ], "weight" : 340, "gender" : "m" }
```

Рисунок 3.25

3.25. Для самки единорога Айна внести изменения в БД: теперь ее вес 800, она убила 51 вампира

```
> db.unicorns.update({name: "Ayna"}, {weight: 800, vampires: 51})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>
```

Рисунок 3.26

3.26. Проверить содержимое коллекции unicorns

```
db.unicorns.find()
{ "_id" : ObjectId("62bdb685cd3f831260d66e69"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 63 }
{ "_id" : ObjectId("62bdb68ccd3f831260d66e6a"), "name" : "Aurora", "loves" : [ "carrot", "grape" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
{ "_id" : ObjectId("62bdb68fcd3f831260d66e6b"), "name" : "Unicorn", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 182 }
{ "_id" : ObjectId("62bdb69ccd3f831260d66e6c"), "name" : "Rooooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 99 }
{ "_id" : ObjectId("62bdb6a2cd3f831260d66e6d"), "name" : "Solnara", "loves" : [ "apple", "carrot", "chocolate" ], "weight" : 550, "gender" : "f", "vampires" : 80 }
{ "_id" : ObjectId("62bdb6b0cd3f831260d66e6e"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 800, "vampires" : 51 }
{ "_id" : ObjectId("62bdb6b5cd3f831260d66e6f"), "name" : "Raleigh", "loves" : [ "apple", "sugar" ], "weight" : 690, "gender" : "m", "vampires" : 39 }
{ "_id" : ObjectId("62bdb6b9cd3f831260d66e70"), "name" : "Leia", "loves" : [ "apple", "watermelon" ], "weight" : 421, "gender" : "m", "vampires" : 2 }
{ "_id" : ObjectId("62bdb6c4cd3f831260d66e71"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 601, "gender" : "f", "vampires" : 33 }
{ "_id" : ObjectId("62bdb6c4cd3f831260d66e72"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 650, "gender" : "m", "vampires" : 54 }
{ "_id" : ObjectId("62bdb6c4cd3f831260d66e73"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 540, "gender" : "f" }
{ "_id" : ObjectId("62bdb7dcd3f831260d66e77"), "name" : "Barney", "loves" : [ "grape" ], "weight" : 704, "gender" : "m", "vampires" : 165 }
{ "_id" : ObjectId("62bde7290ac4880590ae804"), "name" : "Barney", "loves" : [ "grape" ], "weight" : 340, "gender" : "m" }
```

Рисунок 3.27

3.27. Для самца единорога Raleigh внести изменения в БД: теперь он любит рэдбул

```
> db.unicorns.update({name: "Raleigh", gender: "m"}, {$set: {loves: "redbull"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

Рисунок 3.28

3.28. Проверить содержимое коллекции unicorns

```
db.unicorns.find({name: "Raleigh"})
{ "_id" : ObjectId("62bdb6b9cd3f831260d66e70"), "name" : "Raleigh", "loves" : "redbull", "weight" : 421, "gender" : "m", "vampires" : 2 }
```

Рисунок 3.29

3.29. Всем самцам единорогов увеличить количество убитых вампиров на 5

```
> db.unicorns.update({gender: "m"}, {$inc: {vampires: 5}}, {multi: true})
WriteResult({ "nMatched" : 8, "nUpserted" : 0, "nModified" : 8 })
```

Рисунок 3.30

3.30. Проверить содержимое коллекции unicorns

```
> db.unicorns.find({gender: "m"})
{ "_id" : ObjectId("62bdb685cd3f831260d66e69"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 68 }
{ "_id" : ObjectId("62bdb68fcd3f831260d66e6b"), "name" : "Unicrom", "loves" : [ "energon", "redbull" ], "weight" : 984, "gender" : "m", "vampires" : 187 }
{ "_id" : ObjectId("62bdb69ccd3f831260d66e6c"), "name" : "Roooooodles", "loves" : [ "apple" ], "weight" : 575, "gender" : "m", "vampires" : 104 }
{ "_id" : ObjectId("62bdb6b5cd3f831260d66e6f"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 44 }
{ "_id" : ObjectId("62bdb6b9cd3f831260d66e70"), "name" : "Raleigh", "loves" : "redbull", "weight" : 421, "gender" : "m", "vampires" : 7 }
{ "_id" : ObjectId("62bdb6c4cd3f831260d66e72"), "name" : "Pilot", "loves" : [ "apple", "watermelon" ], "weight" : 650, "gender" : "m", "vampires" : 59 }
{ "_id" : ObjectId("62bdb7dccc3f831260d66e77"), "name" : "Dunx", "loves" : [ "grape", "watermelon" ], "weight" : 704, "gender" : "m", "vampires" : 170 }
{ "_id" : ObjectId("62bdde7290ac4880590ae804"), "name" : "Barney", "loves" : [ "grape" ], "weight" : 340, "gender" : "m", "vampires" : 5 }
```

Рисунок 3.31

3.31. Изменить информацию о городе Портланд: мэр этого города теперь беспартийный

```
> db.towns.update({name: "Portland"}, {$unset: {"mayor.party": "D"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>
```

Рисунок 3.32

3.32. Проверить содержимое коллекции towns

```
> db.towns.find({name: "Portland"})
{ "_id" : ObjectId("62bdd572cd3f831260d66e7b"), "name" : "Portland", "populatioun" : 528000, "last_sensus" : ISODate("2009-07-20T00:00:00Z"), "famous_for" : [ "beer", "food" ], "mayor" : { "name" : "Sam Adams" } }
```

Рисунок 3.33

3.33. Изменить информацию о самце единорога Pilot: теперь он любит и шоколад

```
> db.unicorns.update({name: "Pilot", gender: "m"}, {$push: {loves: "chocolate"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>
```

Рисунок 3.34

3.34. Проверить содержимое коллекции unicorns

```
> db.unicorns.find({name: "Pilot"})
{ "_id" : ObjectId("62bdb6c4cd3f831260d66e72"), "name" : "Pilot", "loves" : [ "apple", "watermelon", "chocolate" ], "weight" : 650, "gender" : "m", "vampires" : 59 }
```

Рисунок 3.35

3.35. Изменить информацию о самке единорога Aurora: теперь она любит еще и сахар, и лимоны

```
> db.unicorns.update({name: "Aurora", gender: "f"}, {$addToSet: {loves: {$each: ["sugar", "lemon"]}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

Рисунок 3.36

3.36. Проверить содержимое коллекции unicorns

```
...
> db.unicorns.find((name: "Aurora"))
{ "_id" : ObjectId("62bdb68ccd3f831260d66e6a"), "name" : "Aurora", "loves" : [ "carrot", "grape", "sugar", "lemon" ], "weight" : 450, "gender" : "f", "vampires" : 43 }
```

Рисунок 3.37

3.37. Создайте коллекцию towns, включающую следующие документы

```
> db.towns.insert({name: "Punxsutawney ",
... popujatiuon: 6200,
... last_sensus: ISODate("2008-01-31"),
... famous_for: ["phil the groundhog"],
... mayor: {
...   name: "Jim Wehrle"
... }})
WriteResult({ "nInserted" : 1 })
> db.towns.insert({name: "New York",
... popujatiuon: 22200000,
... last_sensus: ISODate("2009-07-31"),
... famous_for: ["status of liberty", "food"],
... mayor: {
...   name: "Michael Bloomberg",
...   party: "I"}})
WriteResult({ "nInserted" : 1 })
> db.towns.insert({name: "Portland",
... popujatiuon: 528000,
... last_sensus: ISODate("2009-07-20"),
... famous_for: ["beer", "food"],
... mayor: {
...   name: "Sam Adams",
...   party: "D"}})
WriteResult({ "nInserted" : 1 })
>
```

Рисунок 3.38

3.38. Удалите документы с беспартийными мэрами

```
> db.towns.remove({"mayor.party": {$exists: false}})
WriteResult({ "nRemoved" : 3 })
>
```

Рисунок 3.39

3.39. Проверьте содержание коллекции

```
> db.towns.find({"mayor.party": {$exists:false}})
>
```

Рисунок 3.40

3.40. Очистите коллекцию

```
> db.towns.remove({})
WriteResult({ "nRemoved" : 3 })
>
```

Рисунок 3.41

3.41. Просмотрите список доступных коллекций

```
> show collections
towns
unicorns
>
```

Рисунок 3.42

3.42. Создайте коллекцию зон обитания единорогов, указав в качестве идентификатора кратко название зоны, далее включив полное название и описание

```
> db.zones.insert({_id: 'tdr', short_name: 'tdr', full_name: 'tundra', description: 'very cold and lonely'})
WriteResult({ "nInserted" : 1 })
>
```

Рисунок 3.43

3.43. Включите для нескольких единорогов в документы ссылку на зону обитания, используя второй способ автоматического связывания

```
> db.unicorns.update({name: "Kenny"}, {$set: {"zone": {$ref: "zones", $id: "tdr"}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.update({name: "Nimue"}, {$set: {"zone": {$ref: "zones", $id: "tdr"}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.unicorns.update({name: "Horny"}, {$set: {"zone": {$ref: "zones", $id: "tdr"}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>
```

Рисунок 3.44

3.44. Проверьте содержание коллекции единорогов

```
> db.unicorns.find({name: "Kenny"})
{ "_id" : ObjectId("62bdb6b5cd3f831260d6e6ef"), "name" : "Kenny", "loves" : [ "grape", "lemon" ], "weight" : 690, "gender" : "m", "vampires" : 44, "zone" : DBRef("zones", "tdr") }
> db.unicorns.find({name: "Nimue"})
{ "_id" : ObjectId("62bdb6c4cd3f831260d6e6e73"), "name" : "Nimue", "loves" : [ "grape", "carrot" ], "weight" : 540, "gender" : "f", "zone" : DBRef("zones", "tdr") }
> db.unicorns.find({name: "Horny"})
{ "_id" : ObjectId("62bdb685cd3f831260d6e6e69"), "name" : "Horny", "loves" : [ "carrot", "papaya" ], "weight" : 600, "gender" : "m", "vampires" : 68, "zone" : DBRef("zones", "tdr") }
```

Рисунок 3.45

3.45. Обновить содержание коллекции единорогов unicorns

```
> db.unicorns.insert({name: 'Solnara', loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Ayna', loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Kenny', loves: ['grape', 'lemon'], weight: 690, gender: 'm', vampires: 39});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Raleigh', loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Leia', loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Pilot', loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert ({name: 'Nimue', loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert {name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165}
SyntaxError: unexpected token: '{' :
@(shell):1:19
> db.unicorns.insert({name: 'Dunx', loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165})
WriteResult({ "nInserted" : 1 })
>
```

Рисунок 3.46

3.46. Проверьте, можно ли задать для коллекции unicorns индекс для ключа name с флагом unique

```
> db.unicorns.createIndex({"name":1}, {"unique": true})
{
  "ok" : 0,
  "errmsg" : "Index build failed: 08c340c3-54c0-40ec-af8f-506d0eff3e8: Collection learn.unicorns ( 99b6f0f6-2fab-4f14-99ca-64305c7adfb ) :: caused by :: E11000 duplicate key error collection: learn.unicorns index: name_1_dup key
  ( name: \"Aurora\" ),
  \"code\" : 11000,
  \"codeName\" : \"DuplicateKey\",
  \"keyPattern\" : {
    \"name\" : 1
  },
  \"keyValue\" : {
    \"name\" : \"Aurora\"
  }
}
```

Рисунок 3.47

3.47. Обновить содержание коллекции единорогов unicorns

```
> db.unicorns.insert({name: 'Horny', dob: new Date(1992,2,13,7,47), loves: ['carrot','papaya'], weight: 600, gender: 'm', vampires: 63});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Aurora', dob: new Date(1991, 0, 24, 13, 0), loves: ['carrot', 'grape'], weight: 450, gender: 'f', vampires: 43});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Unicrom', dob: new Date(1973, 1, 9, 22, 10), loves: ['energon', 'redbull'], weight: 984, gender: 'm', vampires: 182});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Rooodoodles', dob: new Date(1979, 7, 18, 18, 44), loves: ['apple'], weight: 575, gender: 'm', vampires: 99});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Solnara', dob: new Date(1985, 6, 4, 2, 1), loves:['apple', 'carrot', 'chocolate'], weight:550, gender:'f', vampires:80});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Ayna', dob: new Date(1998, 2, 7, 8, 30), loves: ['strawberry', 'lemon'], weight: 733, gender: 'f', vampires: 40});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Kenny', dob: new Date(1997, 6, 1, 10, 42), loves: ['grape', 'lemon'], weight: 690, gender: 'm', vampires: 39});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Raleigh', dob: new Date(2005, 4, 3, 0, 57), loves: ['apple', 'sugar'], weight: 421, gender: 'm', vampires: 2});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Leia', dob: new Date(2001, 9, 8, 14, 53), loves: ['apple', 'watermelon'], weight: 601, gender: 'f', vampires: 33});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Pilot', dob: new Date(1997, 2, 1, 5, 3), loves: ['apple', 'watermelon'], weight: 650, gender: 'm', vampires: 54});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert ({name: 'Nimue', dob: new Date(1999, 11, 20, 16, 15), loves: ['grape', 'carrot'], weight: 540, gender: 'f'});
WriteResult({ "nInserted" : 1 })
> db.unicorns.insert({name: 'Dunx', dob: new Date(1976, 6, 18, 18, 18), loves: ['grape', 'watermelon'], weight: 704, gender: 'm', vampires: 165})
WriteResult({ "nInserted" : 1 })
>
```

Рисунок 3.48

3.48. Получите информацию о всех индексах коллекции unicorns

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

Рисунок 3.49

3.49. Удалите все индексы, кроме индекса для идентификатора

```
> db.unicorns.dropIndexes()
{
  "nIndexesWas" : 1,
  "msg" : "non-_id indexes dropped for collection",
  "ok" : 1
}
```

Рисунок 3.50

3.50. Попробуйте удалить индекс для идентификатора

```
> db.unicorns.dropIndexes({"_id":1})
Error: error dropping indexes : {
  "ok" : 0,
  "errmsg" : "cannot drop _id index",
  "code" : 72,
  "codeName" : "InvalidOptions"
} :
_getErrorWithCode@src/mongo/shell/utils.js:25:13
DBCollection.prototype.dropIndexes@src/mongo/shell/collection.js:704:11
@(shell):1:1
>
```

Рисунок 3.51

3.51. Создайте объемную коллекцию numbers, задействовав курсор

```
> for(i = 0; i < 100000; i++){db.numbers.insert({value: i})}
WriteResult({ "nInserted" : 1 })
>
```

Рисунок 3.52

3.52. Выберите последних четыре документа

```
> db.numbers.find().sort({$natural: -1}).limit(4)
{ "_id" : ObjectId("62bdeecd90ac4880590c6ebe"), "value" : 99999 }
{ "_id" : ObjectId("62bdeecd90ac4880590c6ebd"), "value" : 99998 }
{ "_id" : ObjectId("62bdeecd90ac4880590c6ebc"), "value" : 99997 }
{ "_id" : ObjectId("62bdeecd90ac4880590c6ebb"), "value" : 99996 }
>
```

Рисунок 3.53

3.53. Проанализируйте план выполнения запроса 2. Сколько потребовалось времени на выполнение запроса?

```
},
"executionStats" : {
  "executionSuccess" : true,
  "nReturned" : 4,
  "executionTimeMillis" : 0,
  "totalKeysExamined" : 0,
  "totalDocsExamined" : 4,
  "executionStages" : {
    "stage" : "LIMIT"
```

Рисунок 3.54

3.54. Создайте индекс для ключа value

```
> db.numbers.createIndex({"value": 1})
{
  "numIndexesBefore" : 1,
  "numIndexesAfter" : 2,
  "createdCollectionAutomatically" : false,
  "ok" : 1
}
```

Рисунок 3.55

3.55. Получите информацию о всех индексах коллекции numbers

```
> db.numbers.getIndexes()
[
  {
    "v" : 2,
    "key" : {
      "_id" : 1
    },
    "name" : "_id_"
  },
  {
    "v" : 2,
    "key" : {
      "value" : 1
    },
    "name" : "value_1"
  }
]
```

Рисунок 3.56

3.56. Выполните запрос 2

```
> db.numbers.find().sort({$natural: -1}).limit(4)
{ "_id" : ObjectId("62bdeecd90ac4880590c6ebe"), "value" : 99999 }
{ "_id" : ObjectId("62bdeecd90ac4880590c6ebd"), "value" : 99998 }
{ "_id" : ObjectId("62bdeecd90ac4880590c6ebc"), "value" : 99997 }
{ "_id" : ObjectId("62bdeecd90ac4880590c6ebb"), "value" : 99996 }
```

Рисунок 3.57

3.57. Проанализируйте план выполнения запроса с установленным индексом. Сколько потребовалось времени на выполнение запроса?

```
"executionStats" : {  
  "executionSuccess" : true,  
  "nReturned" : 4,  
  "executionTimeMillis" : 0,  
  "totalKeysExamined" : 0,  
  "totalDocsExamined" : 4,  
  "executionStages" : {  
    "stage" : "LIMIT",
```

Рисунок 3.58

3.58. Сравните время выполнения запросов с индексом и без. Дайте ответ на вопрос: какой запрос более эффективен?

Ответ: из-за малого количества рассматриваемых данных определить эффективность невозможно.

4. Выводы

В ходе работы мы овладели практическими навыками работы с CRUD-операциями, с вложенными объектами в коллекции базы данных MongoDB, агрегации и изменения данных, со ссылками и индексами в базе данных MongoDB.