Нереляционные базы данных

Лабораторная работа 3

Курсоры

В лабораторной работе 1 мы вкратце рассмотрели команду `find`. Однако, `find` — это не только селекторы. Как уже упоминалось, результатом `find` является курсор. Пришло время рассмотреть это детальнее.

Выбор полей

Прежде чем переходить к курсорам, следует знать, что `find` принимает второй необязательный параметр. Это — список полей, которые мы хотим получить. Например, мы можем получить все имена единорогов следующим запросом:

```
db.unicorns.find(null, {name: 1});
```

Поле `_id` по умолчанию возвращается всегда. Мы можем явным способом исключить его, указав `{name:1, _id: 0}`.

За исключением поля `_id`, нельзя смешивать включения и исключения полей. Задумавшись, можно понять, зачем так сделано. Можно или хотеть включить или хотеть наоборот — исключить определённые поля явным образом.

Сортировка

Я уже несколько раз упомянул, что `find` возвращает курсор, который исполняется отложенно — по мере необходимости. Однако, вы уже без сомнения могли видеть, что`find` исполняется мгновенно. Такое поведение характерно только для консоли. Можно пронаблюдать за истинным поведением курсоров, взглянув на любой из методов, который мы можем присоединить к `find`. Первым из них будет `sort`. Синтаксис `sort` примерно такой же, как у выбора полей, который мы видели в предыдущем разделе. Мы указываем поля, по которым надо сортировать, используя `1` для сортировки по возрастанию и `-1`для сортировки по убыванию. Например:

```
//сортируем по весу — от тяжёлых к лёгким единорогам
db.unicorns.find().sort({weight: -1})

//по имени единорога, затем по числу убитых вампиров:
db.unicorns.find().sort({name: 1, vampires: -1})
```

Подобно реляционной базе данных, MongoDB может использовать индексы для сортировки. Детальнее мы рассмотрим индексы несколько позже. Однако следует знать, что без индекса MongoDB ограничивает размер сортируемых данных. Если вы попытаетесь отсортировать большой объем данных, не используя индекс, вы получите ошибку. Некоторые считают это ограничением.

Разбиение на страницы

Разбиение на страницы может быть осуществлено с помощью методов `limit` и `skip`. Чтобы получить второго и третьего по весу единорога, можно выполнить:

```
db.unicorns.find().sort({weight: -1}).limit(2).skip(1)
```

Используя `limit` вместе с `sort` можно избежать проблем с сортировкой по неиндексированным полям.

Count

Консоль позволяет выполнить `count` прямо над коллекцией:

```
db.unicorns.count({vampires: {$gt: 50}})
```

На практике же `count` — это метод курсора, консоль просто обеспечивает удобное сокращение. С драйверами, не поддерживающим подобного сокращения, нужно писать что-то вроде этого (конечно, и в консоли тоже так можно):

```
db.unicorns.find({vampires: {$gt: 50}}).count()
```

`forEach`

Функция курсора forEach позволяет выполнить заданную функцию для каждого документа выборки.

```
db.users.find().forEach( function(myDoc) { print( myDoc.name + " is here"); } );
```

`Выводы к лабораторной работе`

Довольно просто пользоваться `find` и курсорами. Есть еще несколько дополнительных функций курсоров, которые вы можете освоить самостоятельно, но теперь вы должны уже освоиться в работе с консолью `mongo` и пониманием основных принципов MongoDB.

Требования к сдаче лабораторной работы

1. Отчет к лабораторной работе, содержащий пошаговое исполнение заданий и промежуточные результаты

Задания к лабораторной работе

1. Проведите 4 выборки данных, используя включение, исключение полей и индексов. Убедитесь, что только индекс может быть исключен при включении полей.
2. Проведите 2 выборки данных, используя сортировку по убыванию и по возрастанию

3. Проведите 3 выборки данных, используя различные комбинации сортировки по полям.
4. Проведите следующие выборки для коллекции из ка минимум шести элементов: первые пять элементов; все элементы, начиная с четвертого; элементы с третьего по пятый.
5. Посчитайте число элементов в каждой выборке задания 4. Убедитесь, что записи count(условие) и find(условие).count() равнозначны.
6. Создайте коллекцию с 5-10 документами вида {number: <положительное, отрицательное число или ноль>}, с помощью функции forEach посчитайте сумму положительных элементов, количество четных элементов и произведение ненулевых элементов.