Министерство науки и образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский государственный технический университет имени

Н.Э. Баумана

(национальный исследовательский университет)»

(МГТУ им. Н.Э. Баумана)



**УЕБНО-МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ЛАБОРАТОРНЫМ РАБОТАМ ПО**

**КУРСУ «БАЗЫ ДАННЫХ NoSQL»**

**Лабораторная работа №3**

**«Основы MongoDB. Часть 2»**

Авторы:

Кудрявцев А.П., [kudryavtsevap@bmstu.ru](mailto:kudryavtsevap@bmstu.ru)

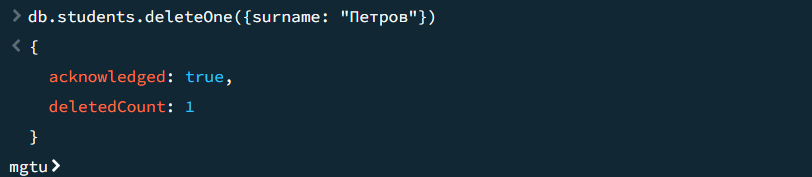
Фомин М.М.

Москва, 2024

1. Удаление документов

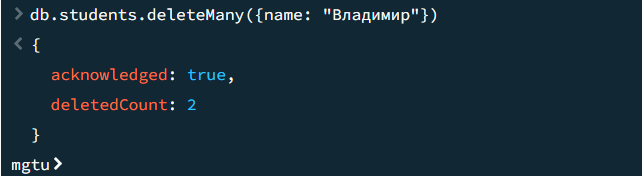
Для удаления документов в MongoDB предусмотрены функции deleteOne() - удаляет один документ и deleteMany() - позволяет удалить несколько документов. В качестве параметра в эти функции передается фильтр удаляемых документов.

Например, удалим документ, в котором surname="Петров":



В итоге первый найденный документ с surname="Петров" будет удален. После удаления консоль отображает нам объект, в котором параметр deletedCount указывает на количество удаленных документов.

Для удаления всех документов, которые соответствуют фильтру, применяется функция deleteMany():



Чтобы удалить разом все документы из коллекции, надо оставить пустым параметр запроса:

db.students.deleteMany({})

2. Удаление коллекций и баз данных

Мы можем удалять не только документы, но и коллекции и базы данных. Для удаления коллекций используется функция drop:

db.students.drop()

Чтобы удалить всю базу данных, надо воспользоваться функцией dropDatabase():

db.dropDatabase()

3. Изменение данных

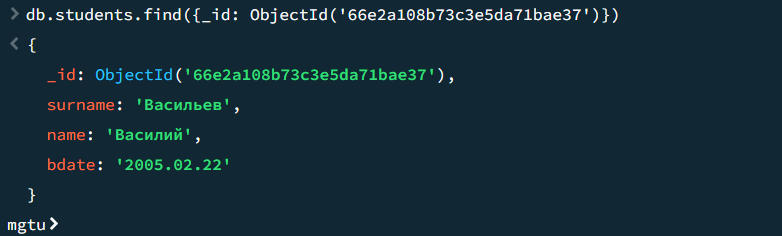
После сохранения документа в базе данных его можно изменить с по-

мощью одного из нескольких методов обновления: updateOne, updateMany

и replaceOne.

3.1. Замена документа

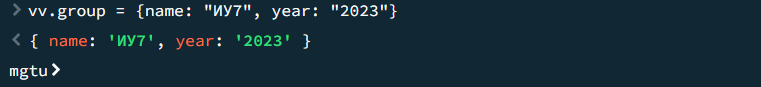
Метод replaceOne полностью заменяет соответствующий документ новым. Рассмотрим пример. Был заведен документ в коллекции students, у которой не указана группа и отчество:



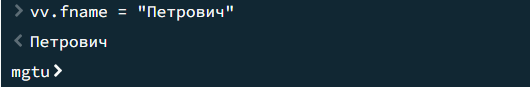
Получаем документ в переменной vv:



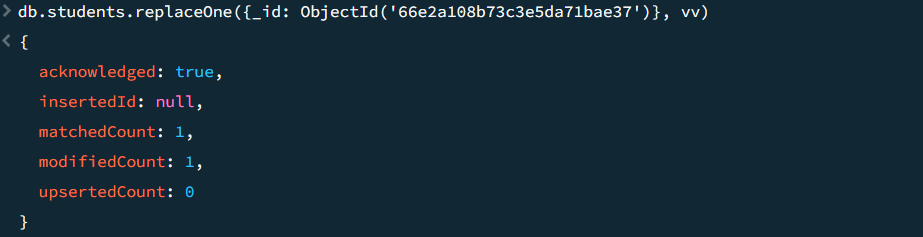
У объекта vv создадим вложенный документ и занесем данные:



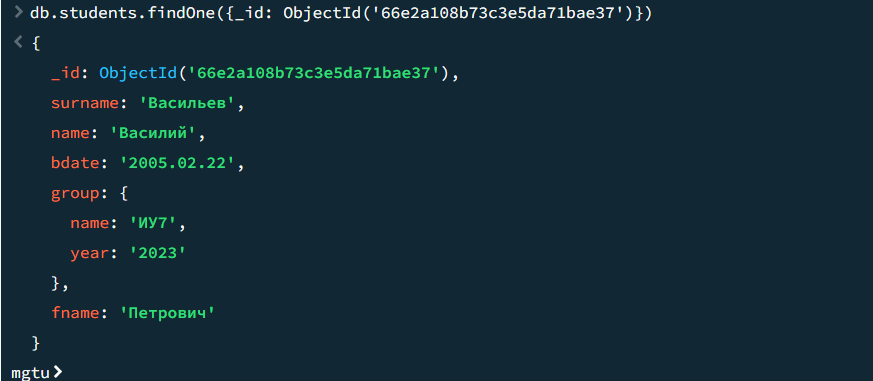
Добавим отчество:



Таким образом, мы в vv сформировали требуемый документ и осталось только заменить документ в коллекции:



На выходе получаем отчет о выполненной работе: модифицирована одна запись. Проверим:



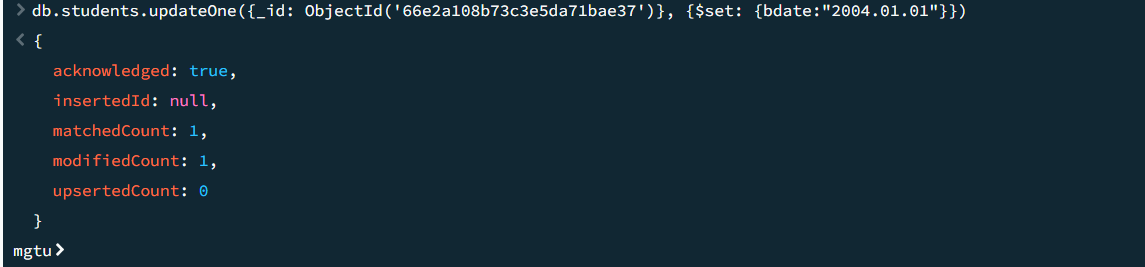
У метода replaceOne имеется третий параметр, основной опцией которого является параметр upsert. Если параметр upsert имеет значение true, что MongoDB будет обновлять документ, если он найден, и создавать новый, если такого документа нет. Если же он имеет значение false, то MongoDB не будет создавать новый документ, если запрос на выборку не найдет ни одного документа.

3.2. Операторы обновления

Часто не требуется обновлять весь документ, а только значение одного или нескольких его свойств. Для этого применяются функции updateOne() (обновляет только один документ) и updateMany() (обновляет множество документов).

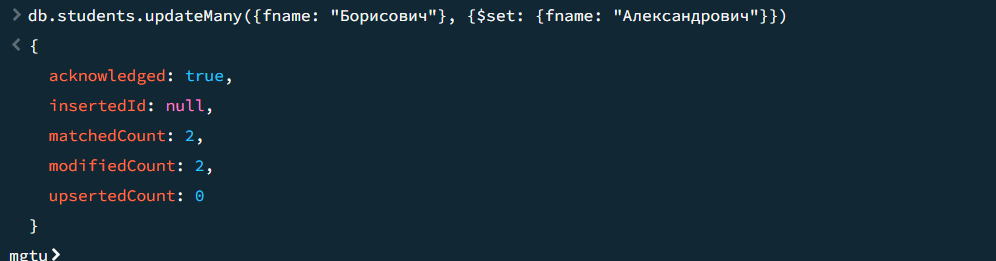
Для обновления отдельных полей в этих функциях применяется оператор $set. Если документ не содержит обновляемое поле, то оно создается.

Изменим дату у документа с заданным \_id:

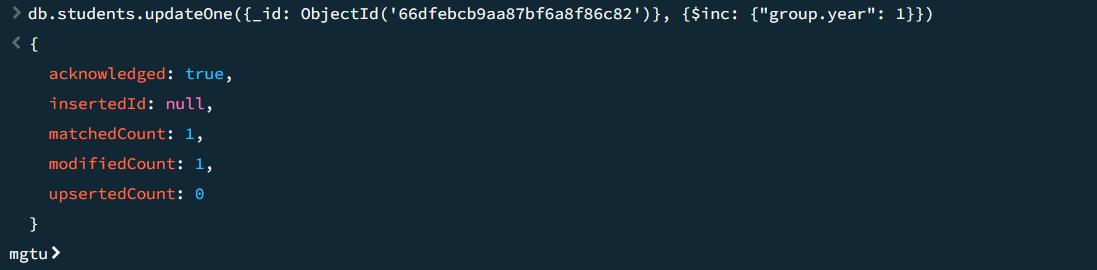


Выполняется поиск документа и заменятся bdate на новое значение.

Если необходимо обновить все документы, соответствующие некоторому критерию, то применяется функция updateMany():



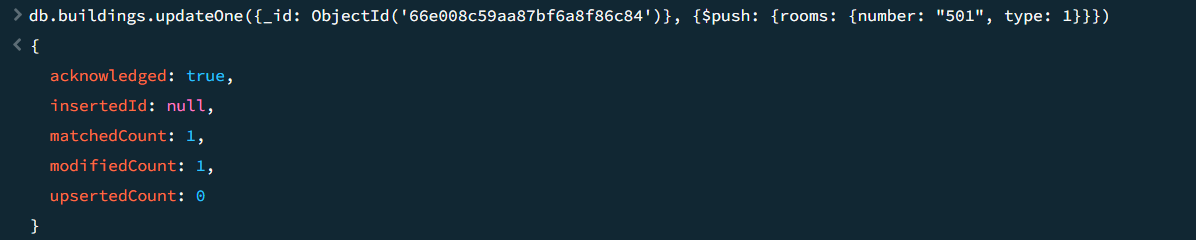
Для простого увеличения значения числового поля на определенное количество единиц применяется оператор $inc. Если документ не содержит обновляемое поле, то оно создается. Данный оператор применим только к числовым значениям.



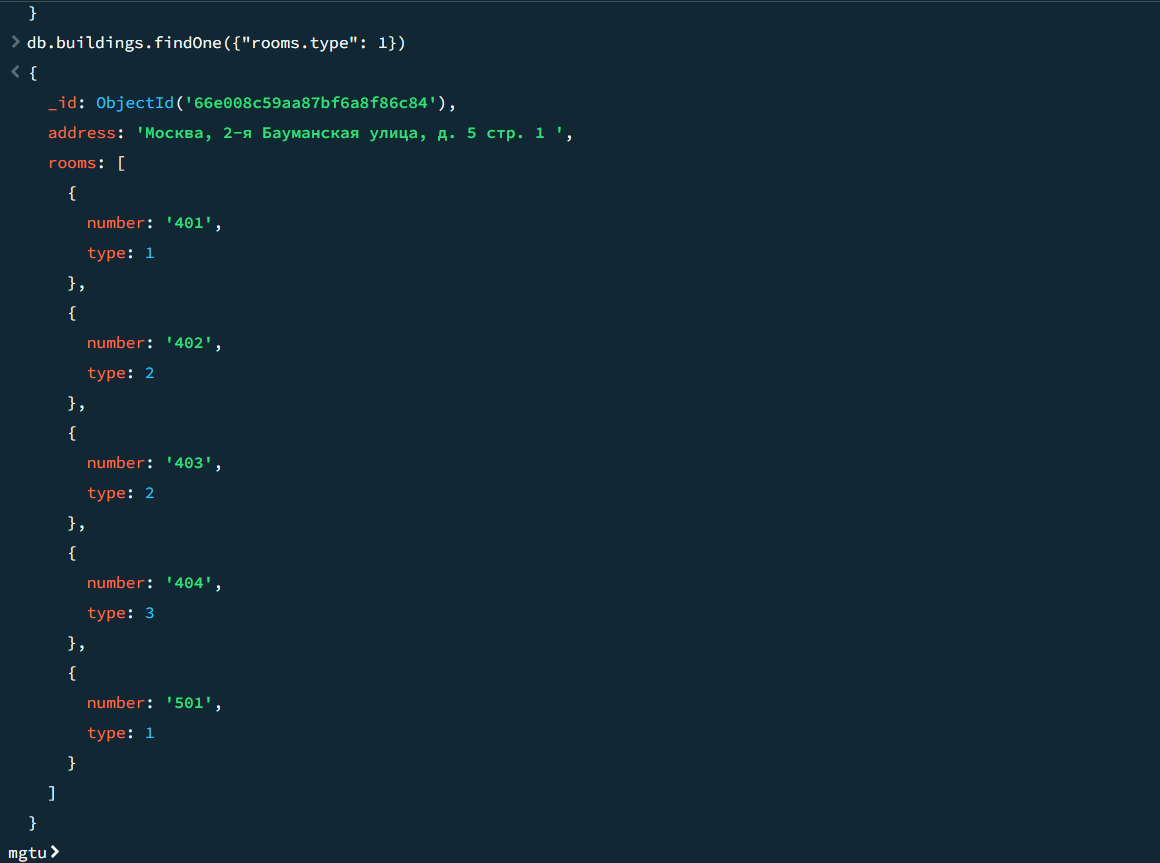
3.3. Операторы массива

Для манипулирования массивами существует обширный класс операторов обновления. Массивы представляют собой распространенные и мощные структуры данных.

Оператор $push добавляет элементы в конец массива, если массив существует, и создает новый массив, если его нет. Добавим помещение в коллекцию buildings:

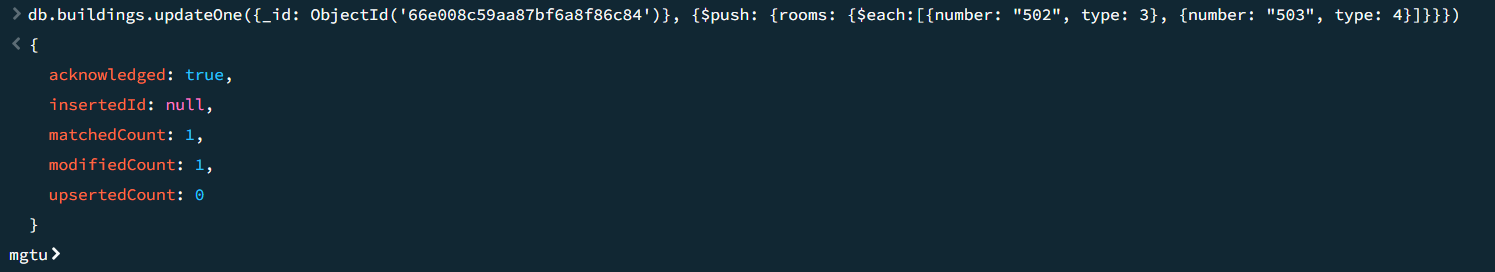


Проверим:



Используя оператор $each, можно добавить сразу несколько значений:

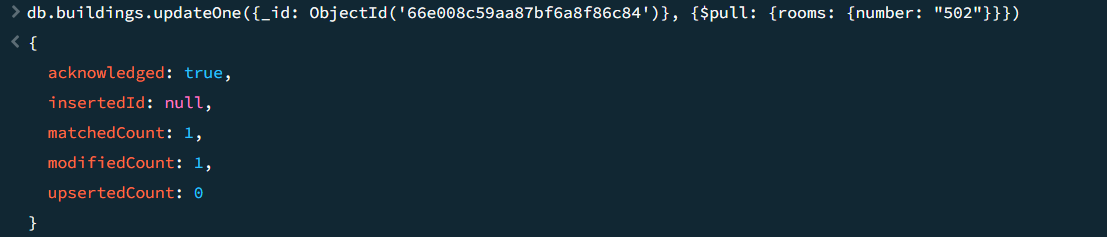
db.buildings.updateOne({\_id: ObjectId('66e008c59aa87bf6a8f86c84')}, {$push: {rooms: {$each:[{number: "502", type: 3}, {number: "503", type: 4}]}}})



Оператор $addToSet подобно оператору $push добавляет объекты в массив. Отличие состоит в том, что $addToSet добавляет данные, если их еще нет в массиве (при добавлении через $push данные дублируются, если добавляются элементы, которые уже есть в массиве).

Существует несколько способов удалить элементы из массива. Если вы хотите рассматривать массив как очередь или стек, можно использовать оператор $pop, который может удалять элементы с любого конца.   
{$pop: {key: 1}} удаляет элемент из конца массива. {$pop: {key: -1}} удаляет его с начала.

Иногда элемент должен быть удален на основе определенных критериев, а не своего положения в массиве. Оператор $pull используется для удаления элементов массива, соответствующих заданным критериям. Удалим помещение 502:



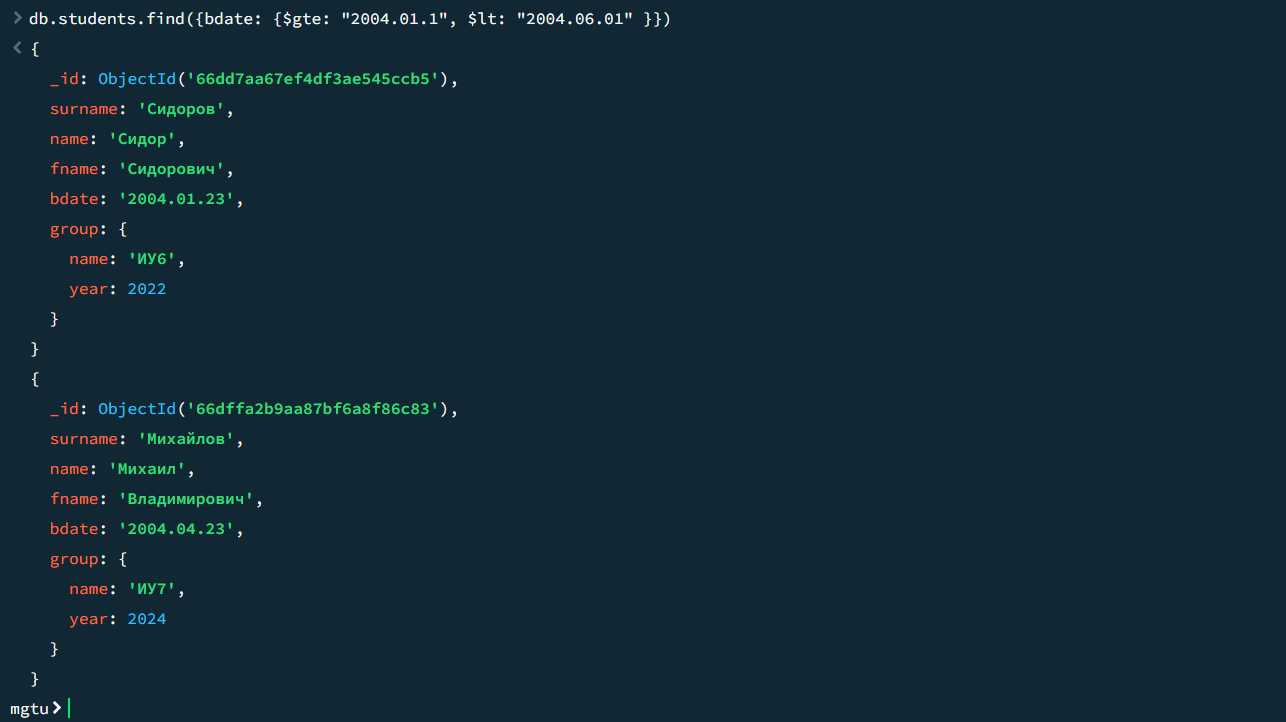
А если мы хотим удалить не одно значение, а сразу несколько, тогда мы можем применить оператор $pullAll.

4. Критерии запроса

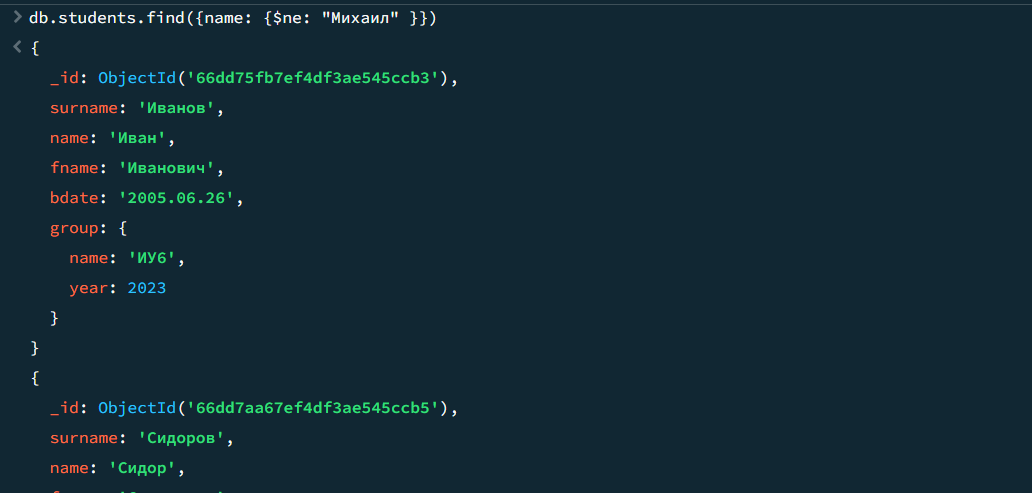
Запросы могут выходить за рамки точного соответствия, они могут соответствовать более сложным критериям, таким как диапазоны, операторы OR и отрицание.

4.1. Условные операторы

$lt, $lte, $gt и $gte – все это операторы сравнения, соответствующие <, <=, > и >= соответственно. Их можно комбинировать для поиска диапазона значений. Например, чтобы найти студентов, родившихся в диапазоне [2004.01.01, 2004.06.01):



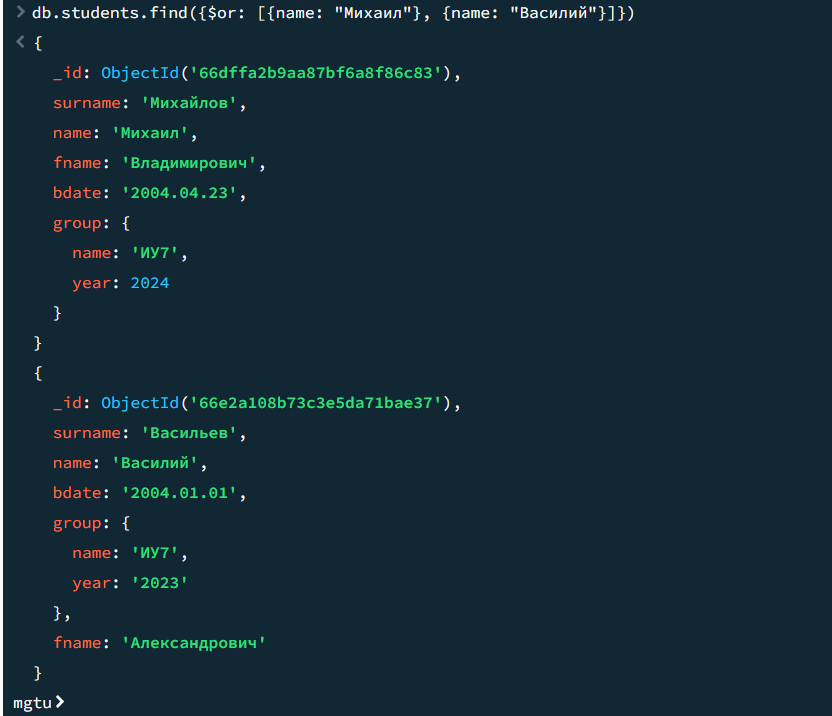
Чтобы запросить документы, в которых значение ключа не равно определенному значению, нужно использовать другой условный оператор $ne, который означает «не равно». Если вы хотите найти всех студентов, у которых нет имени Михаил, можно запросить их таким образом:



Оператор $ne можно использовать с любым типом.

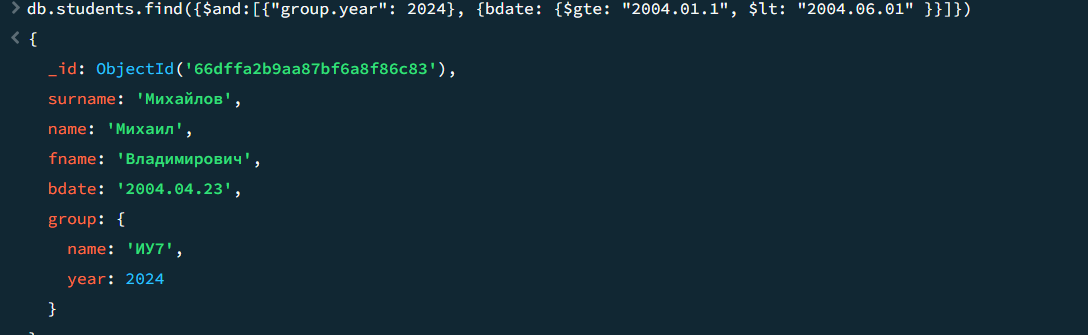
4.2. Оператор $or

Оператор $or представляет логическую операцию ИЛИ и определяет набор пар ключ-значение, которые должны иметься в документе. И, если документ имеет хоть одну такую пару ключ-значение, то он соответствует данному запросу и извлекается из БД. Извлечем тех студентов, которые имеют имена Михаил и Василий:



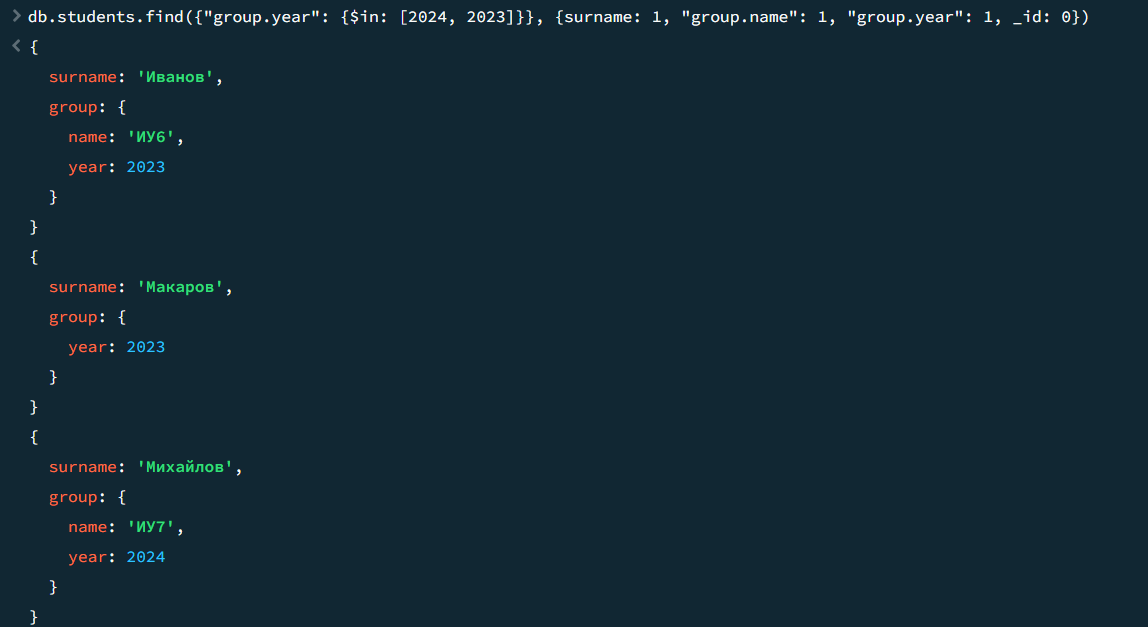
4.3. Оператор $and

Оператор $and представляет логическую операцию И (логическое умножение) и определяет набор критериев, которым обязательно должен соответствовать документ. В отличие от оператора $or документ должен соответствовать всем указанным критериям. Найдем студентов заданного курса (2024 г.), рожденных в диапазоне [2004.01.01, 2004.06.01):



4.4. Оператор $in

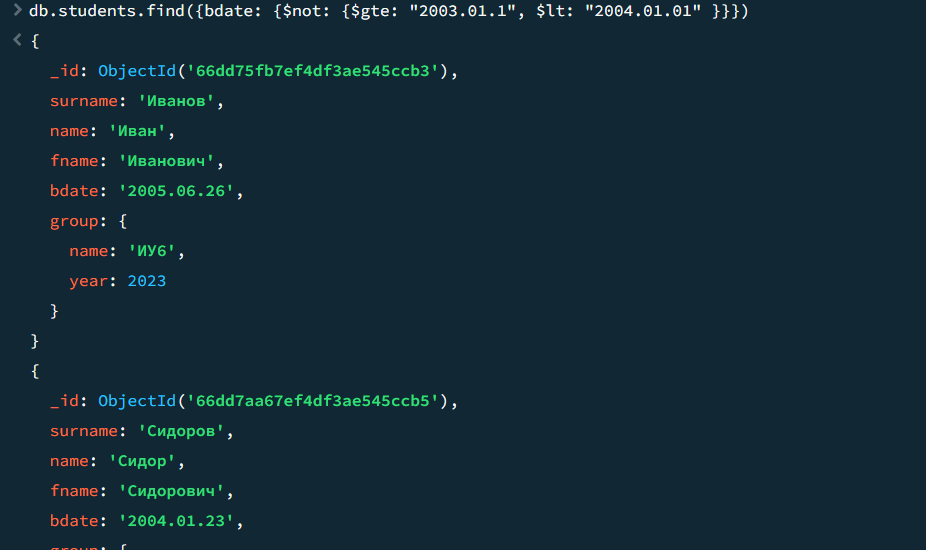
Оператор $in может использоваться для запроса различных значений для одного ключа. Выведем студентов двух курсов:



Противоположным образом действует оператор $nin - он определяет массив возможных выражений и ищет те ключи, значение которых отсутствует в этом массиве.

4.5. Оператор $not

Документ должен не соответствовать условию, его можно применять поверх любых других критериев. Найдем студентов, родившихся раньше 2004.01.01 или позже 2004.06.01 (отрицание условия п. 4.2.):



4.6. Оператор $nor

Оператор $nor соединяет два условия, и документ должен не соответствовать обоим условиям.

5. Запросы элементов массива

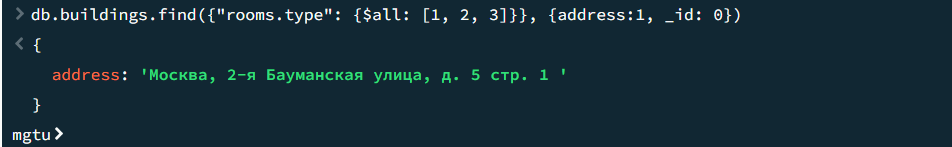
Ряд операторов предназначены для работы с массивами:

* $all: определяет набор значений, которые должны иметься в массиве;
* $size: определяет количество элементов, которые должны быть в массиве;
* $elemMatch: определяет условие, которому должны соответствовать элементы в массиве.

5.1. $all

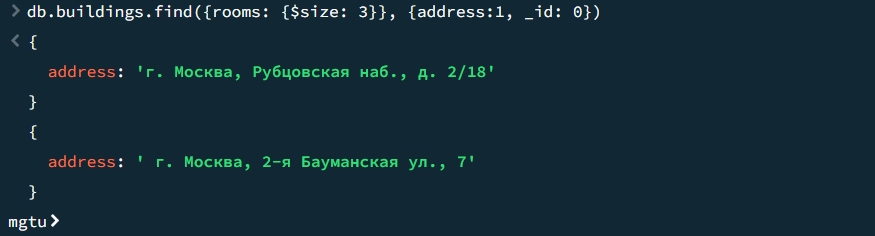
Оператор $all определяет массив возможных выражений и требует, чтобы документы имели весь определяемый набор выражений. Соответственно он применяется к массиву.

Например, выведем все здания, содержащих аудитории трех типов – 1, 2, 3:



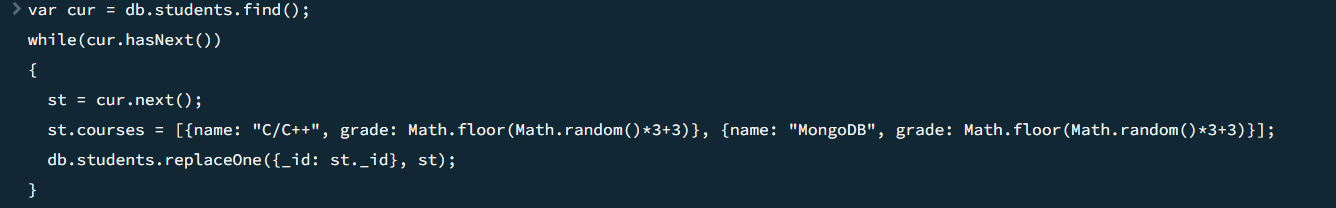
5.2. $size

Оператор $size используется для нахождения документов, в которых массивы имеют количество элементов, равное значению $size. Например, извлечем все документы, в которых в массиве rooms три элемента:

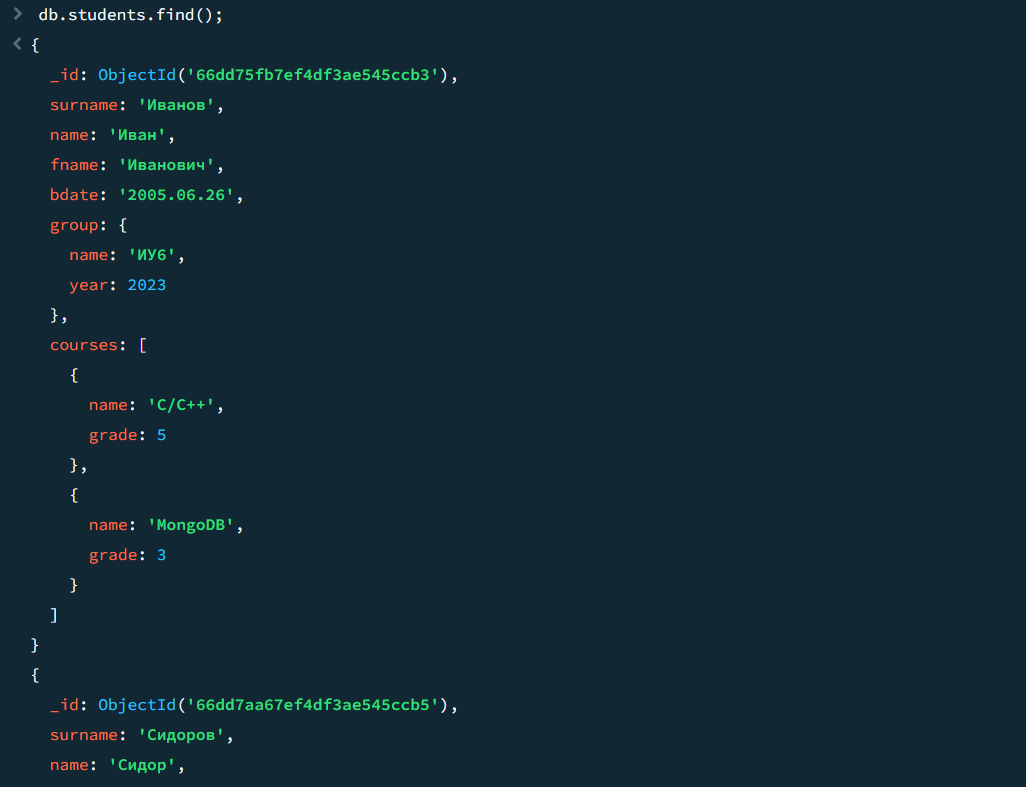


5.3. $elemMatch

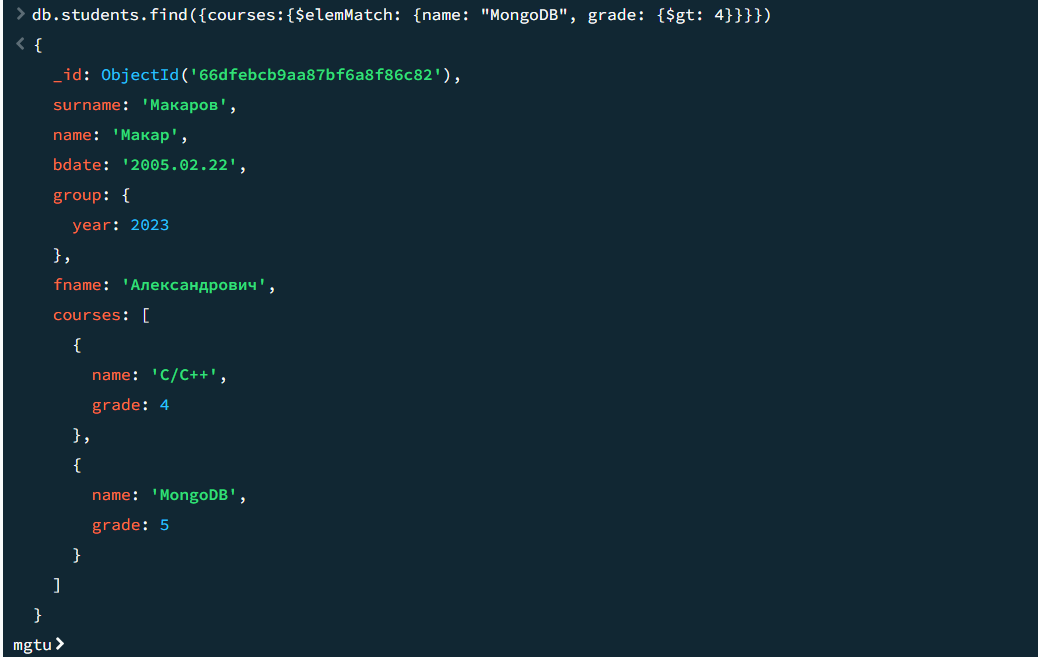
Оператор $elemMatch позволяет выбрать документы, в которых массивы содержат элементы, попадающие под определенные условия. Перед тем, как воспользоваться оператором $elemMatch, добавим в коллекцию students, массив курсов и полученные оценки:



Здесь использовали генератор случайных чисел Math.random() для проставления оценок. Посмотрим на результат:

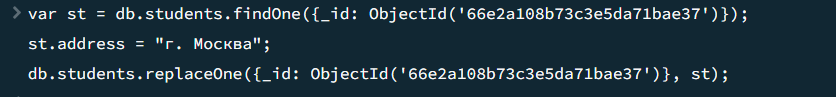


Теперь найдем студентов, которые для курса MongoDB имеют оценку выше 4:

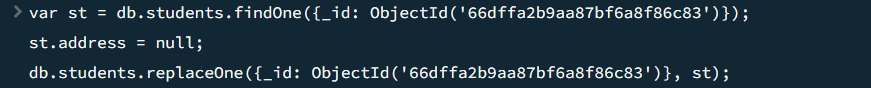


6. Оператор $exists

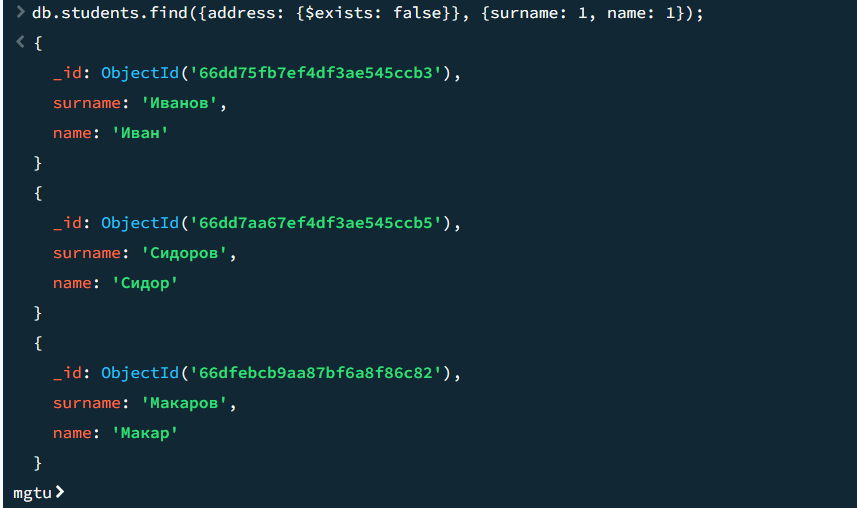
Оператор $exists позволяет извлечь только те документы, в которых определенный ключ присутствует или отсутствует. Добавим атрибут address в два документа. В адрес первый занесем данные:



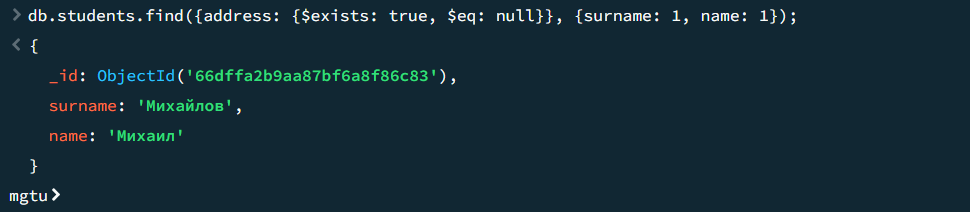
Второй адрес – оставим пустым:



Получим список тех документов, у которых нет атрибута address:

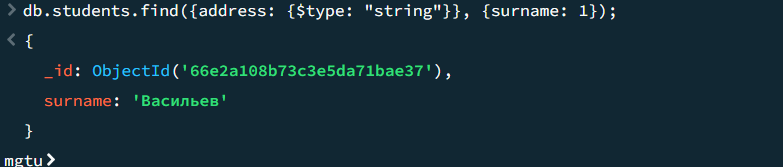


А теперь, выведем список документов, у которых есть атрибут address, но данные не внесены:

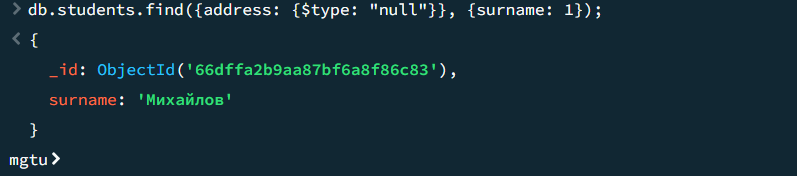


7. Оператор $type

Оператор $type извлекает только те документы, в которых определенный ключ имеет значение определенного типа, например, строку число, массив и т.д.:



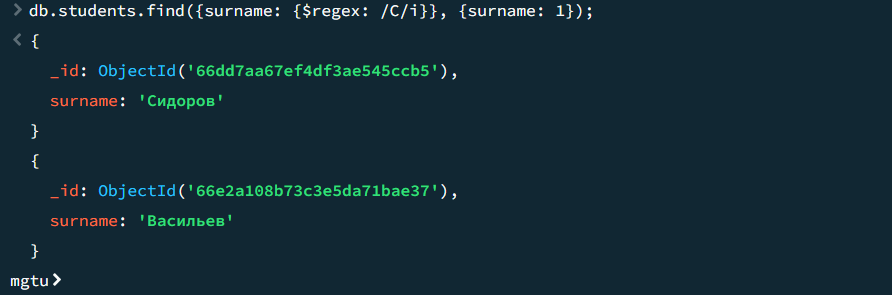
Только в одном документе заведен address, в остальных он отсутствует, либо тип не определен (null). Найдем, документ, у которого address не определен:



8. Регулярные выражения

Регулярные выражения присутствуют во всех языках программирования. Особенно они полезны, если приходится работать с текстом. MongoDB – этот тот случай, когда регулярные выражения будут весьма востребованы.

Рассмотрим пример применения $regex, задающего регулярное выражение. Поставим перед собой задачу – найти всех студентов, фамилия которых содержит буквы «С»:



Здесь использовали флаг i для игнорирования регистра.

9. Курсоры

Результат выборки, получаемой с помощью функции find, называется курсором. При необходимости мы можем передать курсор в отдельную переменную. Курсоры инкапсулируют в себе наборы получаемых из БД объектов. Используя синтаксис языка JavaScript и методы курсоров, мы можем вывести полученные документы на экран и как-то их обработать. Пример использования курсора есть в п. 5.3.

Методы курсора приведены в таблице[[1]](#footnote-1).

| **Имя** | **Описание** |
| --- | --- |
| [cursor.addOption()](https://www.mongodb.com/docs/manual/reference/method/cursor.addOption/#mongodb-method-cursor.addOption) | Добавляет специальные флаги протокола проводной связи, которые изменяют поведение запроса. |
| [cursor.allowDiskUse()](https://www.mongodb.com/docs/manual/reference/method/cursor.allowDiskUse/#mongodb-method-cursor.allowDiskUse) | Позволяет MongoDB использовать временные файлы на диске для хранения данных, превышающих ограничение системной памяти в 100 мегабайт, при обработке операции блокирующей сортировки. |
| [cursor.allowPartialResults()](https://www.mongodb.com/docs/manual/reference/method/cursor.allowPartialResults/#mongodb-method-cursor.allowPartialResults) | Позволяет операциям [db.collection.find()](https://www.mongodb.com/docs/manual/reference/method/db.collection.find/#mongodb-method-db.collection.find) с сегментированной коллекцией возвращать частичные результаты, а не ошибку, если один или несколько запрошенных сегментов недоступны. |
| [cursor.batchSize()](https://www.mongodb.com/docs/manual/reference/method/cursor.batchSize/#mongodb-method-cursor.batchSize) | Управляет количеством документов, которые MongoDB вернет клиенту в одном сетевом сообщении. |
| [cursor.close()](https://www.mongodb.com/docs/manual/reference/method/cursor.close/#mongodb-method-cursor.close) | Закройте курсор и освободите связанные с ним ресурсы сервера. |
| [cursor.isClosed()](https://www.mongodb.com/docs/manual/reference/method/cursor.isClosed/#mongodb-method-cursor.isClosed) | Возвращает true, если курсор закрыт. |
| [cursor.collation()](https://www.mongodb.com/docs/manual/reference/method/cursor.collation/#mongodb-method-cursor.collation) | Задает параметры сортировки для курсора, возвращаемого [db.collection.find().](https://www.mongodb.com/docs/manual/reference/method/db.collection.find/#mongodb-method-db.collection.find) |
| [cursor.comment()](https://www.mongodb.com/docs/manual/reference/method/cursor.comment/#mongodb-method-cursor.comment) | Добавляет комментарий к запросу, чтобы обеспечить отслеживаемость в журналах и коллекции system.profile. |
| [cursor.count()](https://www.mongodb.com/docs/manual/reference/method/cursor.count/#mongodb-method-cursor.count) | Изменяет курсор так, чтобы он возвращал количество документов в результирующем наборе, а не сами документы. |
| [cursor.explain()](https://www.mongodb.com/docs/manual/reference/method/cursor.explain/#mongodb-method-cursor.explain) | Отчеты о плане выполнения запроса для курсора. |
| [cursor.forEach()](https://www.mongodb.com/docs/manual/reference/method/cursor.forEach/#mongodb-method-cursor.forEach) | Применяет функцию JavaScript к каждому документу в курсоре. |
| [cursor.hasNext()](https://www.mongodb.com/docs/manual/reference/method/cursor.hasNext/#mongodb-method-cursor.hasNext) | Возвращает true, если курсор содержит документы и может быть итерирован. |
| [cursor.hint()](https://www.mongodb.com/docs/manual/reference/method/cursor.hint/#mongodb-method-cursor.hint) | Заставляет MongoDB использовать определенный индекс для запроса. |
| [cursor.isExhausted()](https://www.mongodb.com/docs/manual/reference/method/cursor.isExhausted/#mongodb-method-cursor.isExhausted) | Возвращает true, если курсор закрыт ив пакете не осталось объектов. |
| [cursor.itcount()](https://www.mongodb.com/docs/manual/reference/method/cursor.itcount/#mongodb-method-cursor.itcount) | Вычисляет общее количество документов в курсоре на стороне клиента путем выборки и итерации набора результатов. |
| [cursor.limit()](https://www.mongodb.com/docs/manual/reference/method/cursor.limit/#mongodb-method-cursor.limit) | Ограничивает размер результирующего набора курсора. |
| [cursor.map()](https://www.mongodb.com/docs/manual/reference/method/cursor.map/#mongodb-method-cursor.map) | Применяет функцию к каждому документу в курсоре и собирает возвращаемые значения в массив. |
| [cursor.max()](https://www.mongodb.com/docs/manual/reference/method/cursor.max/#mongodb-method-cursor.max) | Указывает исключительную верхнюю границу индекса для курсора. Для использования с[cursor.hint()](https://www.mongodb.com/docs/manual/reference/method/cursor.hint/#mongodb-method-cursor.hint) |
| [cursor.maxAwaitTimeMS()](https://www.mongodb.com/docs/manual/reference/method/cursor.maxAwaitTimeMS/#mongodb-method-cursor.maxAwaitTimeMS) | Указывает предел в миллисекундах для ожидания следующего обновления результата запроса. |
| [cursor.maxTimeMS()](https://www.mongodb.com/docs/manual/reference/method/cursor.maxTimeMS/#mongodb-method-cursor.maxTimeMS) | Задает кумулятивный лимит времени в миллисекундах для обработки операций над курсором. |
| [cursor.min()](https://www.mongodb.com/docs/manual/reference/method/cursor.min/#mongodb-method-cursor.min) | Указывает инклюзивную нижнюю границу индекса для курсора. Для использования с[cursor.hint()](https://www.mongodb.com/docs/manual/reference/method/cursor.hint/#mongodb-method-cursor.hint) |
| [cursor.next()](https://www.mongodb.com/docs/manual/reference/method/cursor.next/#mongodb-method-cursor.next) | Возвращает следующий документ в курсоре. |
| [cursor.noCursorTimeout()](https://www.mongodb.com/docs/manual/reference/method/cursor.noCursorTimeout/#mongodb-method-cursor.noCursorTimeout) | Дает указание серверу не закрывать курсор автоматически после периода бездействия. |
| [cursor.objsLeftInBatch()](https://www.mongodb.com/docs/manual/reference/method/cursor.objsLeftInBatch/#mongodb-method-cursor.objsLeftInBatch) | Возвращает количество документов, оставшихся в текущем пакете курсора. |
| [cursor.pretty()](https://www.mongodb.com/docs/manual/reference/method/cursor.pretty/#mongodb-method-cursor.pretty) | Настраивает курсор для отображения результатов в удобном для чтения формате. |
| [cursor.readConcern()](https://www.mongodb.com/docs/manual/reference/method/cursor.readConcern/#mongodb-method-cursor.readConcern) | Указывает [проблему чтения](https://www.mongodb.com/docs/manual/reference/glossary/#std-term-read-concern) для [find()](https://www.mongodb.com/docs/manual/reference/method/db.collection.find/#mongodb-method-db.collection.find) операции. |
| [cursor.readPref()](https://www.mongodb.com/docs/manual/reference/method/cursor.readPref/#mongodb-method-cursor.readPref) | Задает [предпочтение чтения](https://www.mongodb.com/docs/manual/reference/glossary/#std-term-read-preference) курсору для управления тем, как клиент направляет запросы в [набор реплик.](https://www.mongodb.com/docs/manual/reference/glossary/#std-term-replica-set) |
| [cursor.returnKey()](https://www.mongodb.com/docs/manual/reference/method/cursor.returnKey/#mongodb-method-cursor.returnKey) | Изменяет курсор так, чтобы он возвращал ключи индекса, а не документы. |
| [cursor.showRecordId()](https://www.mongodb.com/docs/manual/reference/method/cursor.showRecordId/#mongodb-method-cursor.showRecordId) | Добавляет поле идентификатора внутреннего хранилища к каждому документу, возвращаемому курсором. |
| [cursor.size()](https://www.mongodb.com/docs/manual/reference/method/cursor.size/#mongodb-method-cursor.size) | Возвращает количество документов в курсоре после применения методов [skip()](https://www.mongodb.com/docs/manual/reference/method/cursor.skip/#mongodb-method-cursor.skip) и [limit()](https://www.mongodb.com/docs/manual/reference/method/cursor.limit/#mongodb-method-cursor.limit). |
| [cursor.skip()](https://www.mongodb.com/docs/manual/reference/method/cursor.skip/#mongodb-method-cursor.skip) | Возвращает курсор, который начинает возвращать результаты только после прохождения или пропуска ряда документов. |
| [cursor.sort()](https://www.mongodb.com/docs/manual/reference/method/cursor.sort/#mongodb-method-cursor.sort) | Возвращает результаты, упорядоченные в соответствии со спецификацией сортировки. |
| [cursor.tailable()](https://www.mongodb.com/docs/manual/reference/method/cursor.tailable/#mongodb-method-cursor.tailable) | Отмечает курсор как tailable. Действительно только для курсоров над ограниченными коллекциями. |
| [cursor.toArray()](https://www.mongodb.com/docs/manual/reference/method/cursor.toArray/#mongodb-method-cursor.toArray) | Возвращает массив, содержащий все документы, возвращенные курсором. |
| [cursor.tryNext()](https://www.mongodb.com/docs/manual/reference/method/cursor.tryNext/#mongodb-method-cursor.tryNext) | Возвращает следующий элемент в итерации, если он доступен, иначе — null. |

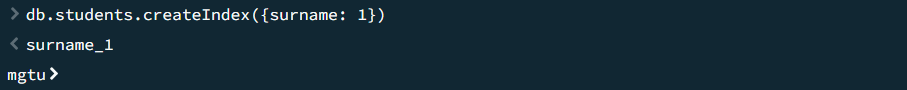
10. Индексы

При поиске документов в небольших коллекциях мы не испытаем особых проблем. Однако, когда коллекции содержат миллионы документов, а нам надо сделать выборку по определенному полю, то поиск нужных данных может занять некоторое время, которое может оказаться критичным для нашей задачи. В этом случае нам могут помочь индексы.

Индексы позволяют упорядочить данные по определенному полю, что впоследствии ускорит поиск. Например, если мы в своем приложении или задаче, как правило, выполняем поиск по некоторому атрибуту, то мы можем индексировать коллекцию по этому атрибуту.

10.1. Создание индекса

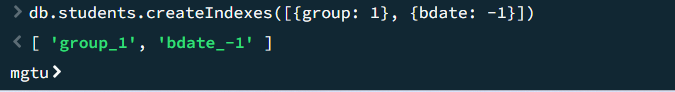
Для создания индекса применяется функция createIndex(), в которую передается объект с указанием полей, для которых создается индекс. Например, создание индекса по полю surname:



Здесь «1» указывает направление сортировки по возрастанию, «-1» - по убыванию. В примере выше был создан индекс с именем surname\_1 по полю surname по возрастанию. MongoDB позволяет установить до 64 индексов на одну коллекцию. При создании индекса можно задать имя индекса:

db.students.createIndex({surname: 1}, {name: "last\_name"})

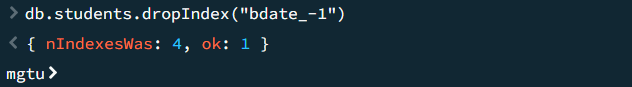
Для создания нескольких индексов применяется функция createIndexes() - в нее передается массив объектов, которые устанавливают поля для индексов:



В приведенном примере создали два индекса: по вложенному документу group и по bdate дате рождения по убыванию, где убывание указывается «-1».

10.2. Удаление индексов

Для удаления индексов применяется функция dropIndex(), в которую передается имя индекса. Например, удалим выше определенный индекс bdate\_-1:



10.3. Опции индекса

Функция создания индекса содержит несколько опций. Одну опцию, задающую имя индекса, мы рассмотрели. Другая важная опция – это задание уникального индекса (попутно мы задали имя индекса):

db.students.createIndex({surname: 1}, {name: "last\_name", unique: true})

Теперь, если мы попытаемся добавить в коллекцию два документа с одним и тем же значением surname, то мы получим ошибку.

В тоже время тут есть свои тонкости. Так, документ может не иметь ключа surname. В этом случае для добавляемого документа автоматически создается ключ surname со значением null. Поэтому при добавлении второго документа, в котором не определен ключ surname, будет выброшено исключение, так как ключ surname со значением null уже присутствует в коллекции.

Другие опции индекса оставляются для самостоятельного изучения.

10.4. Управление индексами

Все индексы базы данных хранятся в системной коллекции indexes. Для обращения к ней мы можем использовать функцию getIndexes, например, чтобы вывести всю информацию об индексах для конкретной коллекции:



11. Практическая часть

1. Создать индексы, в том числе и уникальные индексы, там, где необходимо.
2. Скрипты добавления, изменения, удаления документов из коллекций с учетом имеющихся связей, для всех коллекций в БД.
3. Скрипты поиска и выдачи данных по целевым сущностям с использованием сортировки, проекции данных и регулярных выражений.

12. Список использованных источников

1. MongoDB: <https://www.mongodb.com/>
2. Шеннон Брэдшоу, Йон Брэзил, Кристина Ходоров  
   MongoDB: полное руководство. Мощная и масштабируемая система управления базами данных / пер. с англ. Д. А. Беликова – М.: ДМК Пресс, 2020. – 540 с.: ил.

1. <https://www.mongodb.com/docs/manual/reference/method/js-cursor/> [↑](#footnote-ref-1)