Обновление документов

Обновление документов

После сохранения документа в базе данных его можно изменить с помощью одного из нескольких методов обновления: updateOne, updateMany и replaceOne. Методы updateOne и updateMany принимают документ фильтра в качестве первого параметра и документ модификатора, описывающий изменения, которые необходимо внести, в качестве второго параметра.

Метод replaceOne также принимает фильтр в качестве первого параметра, а в качестве второго параметра ожидает документ, которым он заменит документ, соответствующий фильтру.

Метод replaceOne

Метод replaceOne полностью заменяет соответствующий документ новым.

```
db.collection.replaceOne(filter, replacement, options)
```

- filter: принимает запрос на выборку документа, который надо обновить
- replacement: представляет новый документ, который заместит старый при обновлении
- options: определяет дополнительные параметры при обновлении документов, основным из которых является параметр upsert.

Метод replaceOne

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.

Например:

```
db.users.replaceOne({name: "Bob"}, {name: "Bob", age: 25})
```

В данном случае находим документ, в котором name ="Bob", и заменяем его документом {name: "Bob", age: 25}

Использование операторов обновления (updateOne и updateMany)

Операторы обновления – это специальные ключи, которые можно применять для указания сложных операций обновления, таких как изменение, добавление или удаление ключей, и даже манипулирование массивами и встраиваемыми документами.

Предположим, мы храним аналитику веб-сайта в коллекции и хотим, чтобы счетчик увеличивался каждый раз, когда кто-то посещает страницу. Мы можем использовать операторы обновления, чтобы делать это увеличение атомарно.

Использование операторов обновления (updateOne и updateMany)

Операторы обновления – это специальные ключи, которые можно применять для указания сложных операций обновления, таких как изменение, добавление или удаление ключей, и даже манипулирование массивами и встраиваемыми документами.

Предположим, мы храним аналитику веб-сайта в коллекции и хотим, чтобы счетчик увеличивался каждый раз, когда кто-то посещает страницу. Мы можем использовать операторы обновления, чтобы делать это увеличение атомарно.

Использование операторов обновления (updateOne и updateMany). Модификатор $inc

Каждый URL-адрес и количество просмотров страниц хранятся в документе, который выглядит следующим образом:

```
{
"_id" : ObjectId("4b253b067525f35f94b60a31"),
"url" : "www.example.com",
"pageviews" : 52
}
```

Каждый раз, когда кто-то заходит на страницу, мы можем найти эту страницу по URL-адресу и использовать модификатор **$inc**, чтобы увеличить значение ключа pageviews

```
db.analytics.updateOne({url : "www.example.com"},
  {$inc: {pageviews: 1}})
```

Использование операторов обновления (updateOne и updateMany). Модификатор $set

Модификатор "**$set**" устанавливает значение поля. Если поле еще не существует, оно будет создано. Это может быть удобно для обновления схем или добавления пользовательских ключей. Например, предположим, что у вас есть простой профиль пользователя, сохраненный в виде документа, который выглядит

```
{
"_id" : ObjectId("4b253b067525f35f94b60a31"),
"name" : "joe",
"age" : 30,
"sex" : "male",
"location" : "Wisconsin"
}
```

Использование операторов обновления (updateOne и updateMany). Модификатор $set

Это довольно простой профиль. Если пользователь хочет сохранить свою любимую книгу в своем профиле, он может добавить ее, используя модификатор "$set":

```
db.users.updateOne({_id : ObjectId("4b253b067525f35f94b60a31")},
    {$set : {"favorite book" : "War and Peace"}})
```

Если пользователь решает, что ему по-настоящему нравится другая книга, можно снова использовать модификатор "$set" для изменения значения:

```
db.users.updateOne({name : "joe"}, {$set : {favorite book :
"Green Eggs and Ham"}})
```

$set может даже изменить тип ключа, который он модифицирует. Например, если наш непостоянный пользователь решит, что ему на самом деле нравится всего несколько книг, он может изменить значение ключа favorite book в массив:

```
db.users.updateOne({name : "joe"}, {$set : {favorite book :
["Cat's Cradle", "Foundation Trilogy", "Ender's Game"]}})
```

Использование операторов обновления (updateOne и updateMany). Модификатор $set

Если пользователь понимает, что ему на самом деле не нравится чтение, он может полностью удалить ключ с помощью **$unset**:

```
db.users.updateOne({"name" : "joe"}, {"$unset" : {"favorite book" : 1}})
```

Модификатор "$set" также можно использовать для доступа к вложенным документам и изменять их:

{
"_id" : ObjectId("4b253b067525f35f94b60a31"),
"title" : "A Blog Post",
"content" : "...",
"author" : {
"name" : "joe",
"email" : "joe@example.com"
}
}

Использование операторов обновления (updateOne и updateMany). Модификатор $set

Если пользователь понимает, что ему на самом деле не нравится чтение, он может полностью удалить ключ с помощью **$unset**:

```
db.users.updateOne({"name" : "joe"}, {"$unset" :
{"favorite book" : 1}})
```

Всегда нужно использовать $-модификатор для добавления, изменения или удаления ключей.

Использование операторов обновления (updateOne и updateMany). Модификатор $set

Модификатор $set также можно использовать для доступа к вложенным документам и изменять их:

```
{
"_id" : ObjectId("4b253b067525f35f94b60a31"),
"title" : "A Blog Post",
"content" : "...",
"author" : {
"name" : "joe",
"email" : "joe@example.com"
}
}
```

Обновляем

```
db.blog.posts.updateOne({"author.name" : "joe"}, {"$set" :
{"author.name" : "joe schmoe"}})
```

Инкрементирование и декрементирование

Оператор "$inc" можно использовать для изменения значения существующего ключа или для создания нового ключа, если он еще не существует.

Предположим, мы создаем коллекцию игр, в которой хотим сохранять игры и обновлять оценки по мере их изменения. Когда пользователь начинает играть, скажем, в пейнтбол, мы можем вставить документ, который идентифицирует эту игру по имени и пользователю, который в нее играет:

```
db.games.insertOne({"game" : "pinball", "user" : "joe"})
```

Когда мяч попадает в бампер, счет должен расти. Поскольку очки в пейнтболе начисляются довольно свободно, допустим, что базовая единица очков, которую игрок может заработать, равна 50. Мы можем использовать модификатор "$inc", чтобы добавить 50 к счету игрока:

```
db.games.updateOne({"game" : "pinball", "user" : "joe"},
{"$inc" : {"score" : 50}})
```

Инкрементирование и декрементирование

Если мы посмотрим на документ после этого обновления, то увидим следующее:

```
db.games.findOne()
{
"_id" : ObjectId("4b2d75476cc613d5ee930164"),
"game" : "pinball",
"user" : "joe",
"score" : 50
}
```

Ключа score еще не было, поэтому он был создан с помощью модификатора "$inc", и для него было установлено значение, равное сумме приращения: 50.

Инкрементирование и декрементирование

Модификатор "$inc" похож на "$set", но он предназначен для увеличения (и уменьшения) чисел. Его можно использовать только для значений типа integer, long, double или decimal. Если он используется для любого другого типа значения, это окончится неудачей.

Кроме того, значение ключа "$inc" должно быть числом. Нельзя увеличивать на строку, массив или другое нечисловое значение. В результате появится сообщение об ошибке «Допускается использование модификатора "$inc" только с числами». Чтобы модифицировать другие типы, используйте модификатор "$set" или один из следующих операторов массива.

Операторы массива

Для манипулирования массивами существует обширный класс операторов обновления.

Добавление элементов. Оператор **"$push"** добавляет элементы в конец массива, если массив существует, и создает новый массив, если его нет. Например, предположим, что мы сохраняем посты из блога и хотим добавить ключ "comments", содержащий массив. Мы можем вставить комментарий в несуществующий массив "comments", который создаст массив и добавит комментарий:

```
db.blog.posts.findOne()
{
"_id" : ObjectId("4b2d75476cc613d5ee930164"),
"title" : "A blog post",
"content" : ""
}

> db.blog.posts.updateOne({"title" : "A blog post"},
{"$push" : {"comments" :
{"name" : "joe", "email" : "joe@example.com", "content" : "nice post."}}})
```

Операторы массива

Язык запросов MongoDB предоставляет модификаторы для некоторых опера-торов, включая "$push". Вы можете сдвинуть несколько значений за одну операцию, используя модификатор "$each" для оператора "$push.

Если вы хотите, чтобы массив увеличивался до определенной длины, вы можете использовать модификатор "$slice" с "$push", чтобы предотвратить рост массива выше определенного размера, успешно создавая список элементов.

Обратите внимание, что вы должны использовать модификатор "$each"; нельзя просто использовать модификаторы "$slice" или "$sort" с модификатором "$push" при работе с массивом.

Операторы массива

Возможно, вы захотите рассматривать массив как набор, добавляя только значения, если они отсутствуют. Это можно сделать с помощью модификатора "$ne" в документе запроса.

Например, чтобы вставить автора в список цитат, но только если его там еще нет, используйте следующий код:

```
db.papers.updateOne({authors cited : {$ne : Richie}}, {$push : {authors cited : "Richie"}})
```

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)

```
db.users.updateOne({name : Tom}, {$addToSet: {languages: "russian"}})
```

Удаление элементов

Существует несколько способов удалить элементы из массива. Если вы хотите рассматривать массив как очередь или стек, можно использовать опера-тор "$pop", который может удалять элементы с любого конца. {"$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.

```
db.lists.updateOne({}, {"$pull" : {"todo" :
"laundry"}})
```

Upsert

Upsert (от англ. update (обновлять) + insert (вставить)) – особый тип обновления. Если не найден ни один документ, который соответствует фильтру, будет создан новый документ путем объединения критериев и обновленных документов. Если совпадающий документ найден, он будет обновлен в обычном режиме. Upsert’ы могут быть удобны, потому что могут избавить вас от необходимости «засевать» коллекцию: часто у вас может быть один и тот же код для создания и обновления документов.

Обновление нескольких документов

Чтобы изменить все документы, соответствующие фильтру, используйте метод updateMany.

UpdateMany следует той же семантике, что и updateOne, и принимает те же параметры. Основное различие заключается в количестве документов, которые можно изменить.

UpdateMany предоставляет мощный инструмент для выполнения миграций схемы или развертывания новых функций для определенных пользователей.