**Агрегация**

Рассмотрим механизм агрегации в MongoDB. Агрегация – это группировка значений многих документов. Операции агрегирования позволяют манипулировать такими сгруппированными данными (например, подсчёт – COUNT(\*)). В MySQL аналогом агрегации является команда group by.

**Метод aggregate()**

В MongoDB для агрегации используется метод **aggregate()**. Данный метод имеет следующий синтаксис:

db.ИМЯ\_КОЛЛЕКЦИИ.aggregate(ОПЕРАЦИЯ\_АГРЕГАЦИИ)

Для понимания того, как это работает на практике, рассмотрим простой пример. Предположим, что у нас есть коллекция **developers**, которая содержит следующие документы:

> db.developers.find().pretty()

{

"\_id" : ObjectId("5815f4e735ee883c37ac8b4d"),

"title" : "Eugene Suleimanov",

"specialty" : "Java",

"skills" : [

"Java",

"Hibernate",

"Spring"

],

"salary" : 3500

}

{

"\_id" : ObjectId("5815f4ec35ee883c37ac8b4e"),

"title" : "Ivan Ivanov",

"specialty" : "Java",

"skills" : [

"Java",

"Hibernate",

"Spring"

],

"salary" : 3500

}

{

"\_id" : ObjectId("5815f4f535ee883c37ac8b4f"),

"title" : "Nikolay Nikolaev",

"specialty" : "C++",

"skills" : [

"C++",

"STL",

"SQL"

],

"salary" : 3500

}

Допустим, что нам необходимо вычислить общую зарплату всех разработчиков в коллекции. Для этого мы можем использовать следующий запрос:

> db.developers.aggregate( [ {$group :{ \_id : "Developers", total\_salary: { $sum : "$salary" }}} ] )

{ "\_id" : "Developers", "total\_salary" : 10500 }

Ниже приведён список операций для агрегации документов:

|  |  |  |
| --- | --- | --- |
| **Выраженеие** | **Описание** | **Пример** |
| $sum | Суммирует указанные значения всех документов в коллекции. | db.ИМЯ\_КОЛЛЕКЦИИ.aggregate([{$group : {\_id : “$title”, total\_salary : {$sum : “$salary”}}}]) |
| $avg | Рассчитывает среднее значение указанного поля поля для всех документов коллекции. | db.ИМЯ\_КОЛЛЕКЦИИ.aggregate([{$group : {\_id : “$title”, avg\_salary : {$avg : “$salary”}}}]) |
| $min | Получает минимальное значение указанного поля документа в коллекции | db.ИМЯ\_КОЛЛЕКЦИИ.aggregate([{$group : {\_id : “$title”, min\_salary : {$min : “$salary”}}}]) |
| $max | Получает максимальное значение указанного поля документа в коллекции | db.ИМЯ\_КОЛЛЕКЦИИ.aggregate([{$group : {\_id : “$title”, max\_salary : {$max : “$salary”}}}]) |
| $push | Вставляет значение в массив в результирующем документе. | db.ИМЯ\_КОЛЛЕКЦИИ.aggregate([{$group : {\_id : “$title”, skills : {$push: “$skills”}}}]) |
| $addToSet | Вставляет значение в массив в результирующем документе, но не создаёт дубликаты. | db.ИМЯ\_КОЛЛЕКЦИИ.aggregate([{$group : {\_id : “$title”, skills : {$addToSet : “$skills”}}}]) |
| $first | Получает первый документ из сгруппированных. Обычно используется вместе с сортировкой. | db.ИМЯ\_КОЛЛЕКЦИИ.aggregate([{$group : {\_id : “$title”, first\_skill : {$first : “$skills”}}}]) |
| $last | Получает крайний документ из сгруппированных. Обычно используется вместе с сортировкой. | db.ИМЯ\_КОЛЛЕКЦИИ.aggregate([{$group : {\_id : “$title”, last\_skill : {$last : “$skills”}}}]) |

Для более детальной информации смотри:

<https://docs.mongodb.com/manual/aggregation/>

<https://docs.mongodb.com/manual/core/aggregation-pipeline/#std-label-aggregation-pipeline-example>

https://docs.mongodb.com/manual/reference/operator/aggregation-pipeline/#std-label-aggregation-pipeline-operator-reference

Отдельно стоит рассмотреть команды группировки: count, distinct.

**Число элементов в коллекции**

С помощью функции count() можно получить число элементов в коллекции:

1 > db.users.count()

Можно группировать параметры поиска и функцию count, чтобы подсчитать, сколько определенных документов, например, у которых name=Tom:

1 > db.users.find({name: "Tom"}).count()

Более того мы можем создавать цепочки функций, чтобы конкретизировать условия подсчета:

1 > db.users.find({name: "Tom"}).skip(2).count(true)

Здесь надо отметить, что по умолчанию функция count не используется с функциями limit и skip. Чтобы их использовать, как в примере выше, в функцию count надо передать булевое значение true

**Функция distinct**

В коллекции могут быть документы, которые содержат одинаковые значения для одного или нескольких полей. Например, в нескольких документах определено name: "Tom". И нам надо найти только уникальные различающиеся значения для одного из полей документа. Для этого мы можем воспользоваться функцией distinct:

1 > db.users.distinct("name")

2 ["Tom", "Bill", "Bob"]

Здесь я запрашиваю только уникальные значения для поля name. И на следующей строке консоль выводит в виде массива найденные уникальные значения.