http://cultofdigits.com

Изучение MongoDB

Aggregation framework

План

- Агрегация данных
- Стадии агрегации
- Объединение стадий в потоки
- Операторы агрегации

http://cultofdigits.com

Агрегация данных

```
db.posts.find({}, {tags: true, views: true})


{ "_id" : 1, "tags" : [ "mongo", "donate" ], "views" : 1021 }
{ "_id" : 2, "tags" : [ "mongo", "video", "youtube", "lecture" ], "views" : 723 }
{ "_id" : 3, "tags" : [ "dart", "webGL", "game" ], "views" : 234 }
{ "_id" : 4, "tags" : [ "dart", "webGL", "JavaScript", "frontend" ], "views" : 691 }
{ "_id" : 5, "tags" : [ "Django", "CMS", "Plugins" ], "views" : 741 }
{ "_id" : 6, "tags" : [ "Django", "CMS", "Plugins" ], "views" : 1726 }
{ "_id" : 7, "tags" : [ "dart", "Polymer", "Plugins", "HTML", "frontend" ], "views" : 2761 }
{ "_id" : 8, "tags" : [ "famo.us", "JS", "JavaScript", "animation", "frontend" ], "views" : 1061 }
{ "_id" : 9, "tags" : [ "famo.us", "JS", "JavaScript", "intro", "frontend" ], "views" : 206 }
{ "_id" : 10, "tags" : [ "dart", "IndexedDB", "databases", "frontend" ], "views" : 972 }
{ "_id" : 11, "tags" : [ "sciense", "youtube", "fun" ], "views" : 627 }
```

http://cultofdigits.com

Агрегация данных

```
{ "_id" : 1, "tags" : [ "mongo", "donate" ], "views" : 1021 }
{ "_id" : 2, "tags" : [ "mongo", "video", "youtube", "lecture" ], "views" : 723 }
{ "_id" : 3, "tags" : [ "dart", "webGL", "game" ], "views" : 234 }
{ "_id" : 4, "tags" : [ "dart", "webGL", JavaScript", "frontend" ], "views" : 691 }
{ "_id" : 5, "tags" : [ "Django", "CMS", "Plugins" ], "views" : 741 }
{ "_id" : 6, "tags" : [ "Django", "CMS", "Plugins" ], "views" : 1726 }
{ "_id" : 7, "tags" : [ "dart", "Polymer", "Plugins", "HTML", "frontend" ], "views" : 2761 }
{ "_id" : 8, "tags" : [ "famo.us", "JS", JavaScript", "animation", "frontend" ], "views" : 1061 }
{ "_id" : 9, "tags" : [ "famo.us", "JS", JavaScript", "intro", "frontend" ], "views" : 206 }
{ "_id" : 10, "tags" : [ "dart", "IndexedDB", "databases", "frontend" ], "views" : 972 }
{ "_id" : 11, "tags" : [ "sciense", "youtube", "fun" ], "views" : 627 }
```

http://cultofdigits.com

Агрегация данных

```
{ "_id" : 4, "tags" : [ "dart", "webGL", JavaScript", "frontend" ], "views" : 691 }
{ "_id" : 8, "tags" : [ "famo.us", "JS", JavaScript", "animation", "frontend" ], "views" : 1061 }
{ "_id" : 9, "tags" : [ "famo.us", "JS", JavaScript", "intro", "frontend" ], "views" : 206 }
```

Агрегация данных

http://cultofdigits.com

```
{ "views" : 1958 }
```

Агрегация в SQL

```
SELECT column_name, aggregate_function(column_name)
FROM table_name
WHERE column_name operator value
GROUP BY column_name;
```

http://cultofdigits.com

Потоковое преобразование данных

Поток стадий

```
db.collection.aggregation([
    {stage1: options1},
    {stage2: options2},
    ...
    {stageN: optionsN},
])
```

Стадии агрегации

$match	Фильтрует документы в потоке. После этой стадии остаются только документы удовлетворяющие заданному условию. Для условия используются стандартный синтаксис монго.
$project	Преобразует каждый документ в потоке (добавляет илу удаляет поля из документа)
$group	Группирует документы в соответствии с выражением.
$unwind	Преобразует поле с типом массива в несколько документов.
$sort **$limit** **$skip**	Сортировка лимит и смещение документов. Синтаксис и поведение аналогично стандартному поведению монго.

Пример

```
db.posts.aggregate([
    {$match: {tags: "JavaScript"}},
    {$group: {_id: null, views: {$sum: "$views"}}}
])
```

$match

```
db.collection.aggregation([
    { $match: {tags: "JavaScript"} }
])
```

$project

```
db.collection.aggregation([{
    $project: {
        field1: true,
        field2: $field, fromObj: "$obj.field",
        field3: projectFunction(expression)
    }
}])
```

Операции преобразования

Для Булевых выражений

Для чисел

Для дат

Для строк

Для массивов

Для множеств

https://docs.mongodb.org/master/reference/operator/aggregation/

Операции преобразования для чисел

$add	`{$add: [<число>,...]}`	Складывает числа и возвращает суммы. (может работать для дат).
$subtract	`{$substract: [<число 1>, <число 2>]}`	Вычитает числа и возвращает разность.
$multiply	`{$multiply: [<число>,...]}`	Перемножает числа и возвращает произведение.
$divide	`{$divide: [<число 1>, <число 2>]}`	Делит числа и возвращает результат деления.
$mod	`{$mod: [<число 1>, <число 2>]}`	Делит числа и возвращает остаток от деления.

Перевод температуры из Цельсия в Фарингейт.

Формула перевода tF = tC * 1.8 + 32

```
 db.weather.aggregate([
        {
                $project: {
                        tF: {$add: [ {$multiply: ["$temperature", 1.8]}, 32]}
                }
        }
])
```

Перевод температуры из Цельсия в Фарингейт.

Формула перевода tF = tC * 1.8 + 32

```
db.weather.aggregate([
        {$project: {temperature: {$multiply: ["$temperature", 1.8]}}},
        {$project: {tF: {$add: ["$temperature", 32]}}}
])
```

Операции преобразования для строк

$concat	`{$concat: [<строка>...]}`	Складывает строки

$substr	`{ $substr:`
	`  [ <строка>, <старт>, <длинна> ]`
	`}`
		Возвращает подстроку в строке

$toLower	`{$toLower: <строка>}`	Преобразует строку к нижнему регистру
$toUpper	`{$toUpper: <строка>}`	Преобразует строку к верхнему регистру

$strcasecmp	`{ $strcasecmp:`
	`  [ <выражение1>, <выражение2> ]`
	`}`
		Сравнивает 2 строки и возвращает:
		-1 если первая строка меньше второй
		0 если строки равны
		1 если первая строка больше

Операции преобразования для дат

$dayOfYear	`{ $dayOfYear: <expression> }`	Возвращает номер дня в году (1..366)
$dayOfMonth	`{ $dayOfMonth: <expression> }`	Возвращает номер дня в месяце (1..31)
$dayOfWeek	`{ $dayOfWeek: <expression> }`	Возвращает день недели (1..7)
$year	`{ $year: <expression> }`	Возвращает год
$month	`{ $month: <expression> }`	Возвращает месяц
$week	`{ $week: <expression> }`	Возвращает неделю
$hour	`{ $hour: <expression> }`	Возвращает часы
$minute	`{ $minute: <expression> }`	Возвращает минуты
$second	`{ $second: <expression> }`	Возвращает секунды
$millisecond	`{ $millisecond: <expression> }`	Возвращает милисекунды
$dateToString	`{ $dateToString: { format: <Формат>, date: <Дата> } }`	Преобразует дату в строку в определенном формате https://docs.mongodb.org/master/reference/operator/aggregation/dateToString/#format-specifiers

Операции преобразования для массивов

$size	{$size: <массив>}	Возвращает количество элементов в массиве

Операции преобразования Булевых выражений

$and	`{ $and:`
	`  [ <Выражение1>, <Выражение2>, ... ]`
	`}`
		Возвращает true если все элементы == true

$or	`{ $or:`
	`  [ <Выражение1>, <Выражение2>, ... ]`
	`}`
		Возвращает true если хотябы один элемент == true

$not	`{ $not: [ <Выражение> ] }`	Возвращает true если выражение == false

Операции преобразования для множеств

$setEquals	`{ $setEquals: [ <Массив1>, <Массив2>, ... ] }`	Возвращает true если все массивы содержат одинаковые значения не зависимо от порядка и повторений. И false в остальных случаях
$setIntersection	`{ $setIntersection: [ <Массив1>, <Массив2>, ... ] }`	Возвращает массив, содержащий элементы которые есть в каждом из переданных массивах
$setUnion	`{ $setUnion: [ <Массив1>, <Массив2>, ... ] }`	Возвращает массив элементов которые есть хотя бы в одном из массивов.
$setDifference	`{ $setDifference: [ <Массив1>, <Массив2>, ... ] }`	Возвращает массив элементов которые не повторяются в остальных массивах.
$setIsSubset	`{ $setIsSubset: [ <Массив1>, <Массив2>] }`	Принимает два массива и возвращает true если все элементы первого массива, есть во втором массиве.
$anyElementTrue	`{ $anyElementTrue: [ <Массив1>, <Массив2>, ... ] }`	Возвращает true если хотя-бы один из элементов == true
$allElementsTrue	`{ $allElementsTrue: [ <Массив1>, <Массив2>, ... ] }`	Возвращает true если все элементы == true

Операции сравнения

$cmp	`{ $cmp: [ <Выражение1>, <Выражение2> ] }`	Сравнивает 2 выражения и возвращает: -1 если первое выражение меньше второго 0 если они равны 1 если первое выражение больше второго
$eq	`{ $eq: [ <Выражение1>, <Выражение2> ] }`	Возвращает true если выражения равны
$gt	`{ $gt: [ <Выражение1>, <Выражение2> ] }`	Возвращает true если первое выражение больше второго
$gte	`{ $gte: [ <Выражение1>, <Выражение2> ] }`	Возвращает true если первое выражение больше или равно второго
$lt	`{ $lt: [ <Выражение1>, <Выражение2> ] }`	Возвращает true если первое выражение меньше второго
$lte	`{ $lte: [ <Выражение1>, <Выражение2> ] }`	Возвращает true если первое выражение меньше или равно второго
$ne	`{ $ne: [ <Выражение1>, <Выражение2> ] }`	Возвращает true если выражения не равны

Группировка

```
db.posts.aggregate([
    {$group: {
        _id: "$field1", //or {"$f1", "$f2"}
        aggegated_field: {accumulator: "$field"}}
        ...
    }
])
```

Группировка

```
> db.posts.aggregate([
        {$match: {comments: {$exists: true}}},
        {$project: {comments: true}},
        {$unwind: "$comments"},
        {$group: {_id: "$_id", minRating: {$min: "$comments.rating"}}}
])

{ "_id" : 10, "minRating" : 4 }
{ "_id" : 2, "minRating" : -5 }
```

Операторы накапливания

$sum	Возвращает сумму элементов в группе. Значения с не числовым типом игнорируются.
$avg	Возвращает средне-арифметическое элементов в группе. Значения с не числовым типом игнорируются.
$max	Возвращает максимальное значение элемента в группе.
$min	Возвращает максимальное значение элемента в группе.
$push	Возвращает массив со значением выражения для каждой группы.
$addToSet	Возвращает массив со значением выражения для каждой группы. Значения не повторяются, порядок элементов не определен.

$first	Возвращает значение первого документа каждой группы. Порядок записей такой же как и в коллекции.
$last	Возвращает значение последнего документа каждой группы.

$sort $limit $skip

```
db.collection.aggregate({
    {$sort: {field: -1}},
})
```

$unwind


```
{
        "_id" : 2,
        "comments" : [
          { "name" : "Anonymous", "comment" : "wow great news", "date" : "2015-02-01", "rating" : 12 },
          { "name" : "Alex", "comment" : "Thanks", "date" : "2015-02-01", "rating" : 43 },
          { "name" : "Anon", "comment" : "+1", "date" : "2015-02-01", "rating" : -5 }
        ]
}
```

$unwind


```
{
  "_id" : 2,
  "comments" : { "name" : "Alex", "comment" : "Thanks", "date" : "2015-02-01", "rating" : 43 }
}
{
  "_id" : 2,
  "comments" : { "name" : "Anonymous", "comment" : "wow great news", "date" : "2015-02-01", "rating" : 12 }
}
{
  "_id" : 2,
  "comments" : { "name" : "Anon", "comment" : "+1", "date" : "2015-02-01", "rating" : -5 }
}
```

$unwind

```
db.collection.aggregation([
    { $unwind: "$comments"}
])
```

Пример $unwind

```
db.posts.aggregate([
    {$project: {tags: true}},
    {$unwind: "$tags"},
    {$group: {_id: "$tags", count: {$sum: 1}}},
    {$sort: {count: 1}},
    {$group: {_id: false, tag: {$last: "$_id"}, count: {$last: "$count"}}}
])
```

http://cultofdigits.com

Спасибо!