

\$unwind (aggregation)

Definition

\$unwind

Deconstructs an array field from the input documents to output a document for *each* element. Each output document is the input document with the value of the array field replaced by the element.

Syntax

You can pass a field path operand or a document operand to unwind an array field.

Field Path Operand

You can pass the array field path to [\\$unwind](#). When using this syntax, [\\$unwind](#) does not output a document if the field value is null, missing, or an empty array.

```
{ $unwind: <field path> }
```

When you specify the field path, prefix the field name with a dollar sign \$ and enclose in quotes.

```
{
  $unwind:
  {
    path: <field path>,
    includeArrayIndex: <string>,
    preserveNullAndEmptyArrays: <boolean>
  }
}
```

Field	Type	Description
path	string	Field path to an array field. To specify a field path, prefix the field name with a dollar sign \$ and enclose in quotes.

includeArrayIndex	string	Optional. The name of a new field to hold the array index of the element. The name cannot start with a dollar sign \$.
preserveNullAndEmptyArrays	boolean	Optional. <ul style="list-style-type: none"> If true, if the path is null, missing, or an empty array, \$unwind outputs the document. If false, if path is null, missing, or an empty array, \$unwind does not output a document. <p>The default value is false.</p>

Examples

Unwind Array

```

C:\Program Files\MongoDB\Server\4.2\bin> mongo.exe
MongoDB Shell
> use school
switched to db school
> db.school.aggregate([{$unwind:"$marks"}])
{"_id" : ObjectId("5ec233f75ddaf5609c6cb180"), "stu_name" : "vinod", "class" : "2nd", "marks" : 60, "fees" : { "tuition" : 6000, "admission" : 600, "books" : 700 }}
{"_id" : ObjectId("5ec233f75ddaf5609c6cb181"), "stu_name" : "amit", "class" : "2nd", "marks" : 55, "fees" : { "tuition" : 4000, "admission" : 300, "books" : 300 }}
{"_id" : ObjectId("5ec236255ddaf5609c6cb183"), "stu_name" : "bipin", "class" : "2nd", "marks" : 45, "fees" : { "tuition" : 2000, "admission" : 500, "books" : 700 }}
{"_id" : ObjectId("5ec236255ddaf5609c6cb184"), "stu_name" : "anil", "class" : "3rd", "marks" : 50, "fees" : { "tuition" : 5000, "admission" : 300, "books" : 900 }}
{"_id" : ObjectId("5ec236255ddaf5609c6cb185"), "stu_name" : "ajay", "class" : "3rd", "marks" : 50, "fees" : { "tuition" : 4500, "admission" : 600, "books" : 200 }}
{"_id" : ObjectId("5ec38c34c2f383b00725e114"), "stud_id" : "A02", "stu_name" : "suraj", "class" : "2nd", "marks" : 45 }
{"_id" : ObjectId("5ec38c34c2f383b00725e114"), "stud_id" : "A02", "stu_name" : "suraj", "class" : "2nd", "marks" : 52 }
{"_id" : ObjectId("5ec38c34c2f383b00725e114"), "stud_id" : "A02", "stu_name" : "suraj", "class" : "2nd", "marks" : 62 }

```

\$sort (aggregation)

On this page

- [Definition](#)
- [Examples](#)
- [\\$sort Operator and Memory](#)
- [\\$sort Operator and Performance](#)

Definition

\$sort

Sorts all input documents and returns them to the pipeline in sorted order.

The `$sort` stage has the following prototype form:

```
{ $sort: { <field1>: <sort order>, <field2>: <sort order> ... } }
```

`$sort` takes a document that specifies the field(s) to sort by and the respective sort order. `<sort order>` can have one of the following values:

Value	Description
1	Sort ascending.
-1	Sort descending.
{ \$meta: "textScore" }	Sort by the computed <code>textScore</code> metadata in descending order. See Metadata Sort for an example.

If sorting on multiple fields, sort order is evaluated from left to right. For example, in the form above, documents are first sorted by `<field1>`. Then documents with the same `<field1>` values are further sorted by `<field2>`.

Examples

Ascending/Descending Sort

For the field or fields to sort by, set the sort order to 1 or -1 to specify an ascending or descending sort respectively, as in the following example:

```
db.school.aggregate(  
  [  
    { $sort : { stu_name: 1 } }  
  ]  
)
```

This operation sorts the documents in the `users` collection, in descending order according by the `age` field and then in ascending order according to the value in the `posts` field.

```
C:\Program Files\MongoDB\Server\4.2\bin\mongo.exe
"errmsg" : "FieldPath field names may not start with '$'.",
"code" : 16410,
"codeName" : "Location16410"
} : aggregate failed :
_getErrorWithCode@src/mongo/shell/utils.js:25:13
doassert@src/mongo/shell/assert.js:18:14
_assertCommandWorked@src/mongo/shell/assert.js:583:17
assert.commandWorked@src/mongo/shell/assert.js:673:16
DB.prototype._runAggregate@src/mongo/shell/db.js:266:5
DBCollection.prototype.aggregate@src/mongo/shell/collection.js:1012:12
@(shell):1:1
> db.school.aggregate([{$sort:{stu_name:1}}])
{ "_id" : ObjectId("5ec233f75ddaf5609c6cb182"), "Stu_name" : "arun" }
{ "_id" : ObjectId("5ec236255ddaf5609c6cb185"), "stu_name" : "ajay", "class" : "3rd", "marks" : 50, "fees" : { "tuti
on" : 4500, "admission" : 600, "books" : 200 } }
{ "_id" : ObjectId("5ec233f75ddaf5609c6cb181"), "stu_name" : "amit", "class" : "2nd", "marks" : 55, "fees" : { "tuti
on" : 4000, "admission" : 300, "books" : 300 } }
{ "_id" : ObjectId("5ec236255ddaf5609c6cb184"), "stu_name" : "anil", "class" : "3rd", "marks" : 50, "fees" : { "tuti
on" : 5000, "admission" : 300, "books" : 900 } }
{ "_id" : ObjectId("5ec236255ddaf5609c6cb183"), "stu_name" : "bipin", "class" : "2nd", "marks" : 45, "fees" : { "tut
ion" : 2000, "admission" : 500, "books" : 700 } }
{ "_id" : ObjectId("5ec38c34c2f383b00725e114"), "stud_id" : "A02", "stu_name" : "suraj", "class" : "2nd", "marks" :
[ 45, 52, 62 ] }
{ "_id" : ObjectId("5ec233f75ddaf5609c6cb180"), "stu_name" : "vinod", "class" : "2nd", "marks" : 60, "fees" : { "tut
ion" : 6000, "admission" : 600, "books" : 700 } }
>
```

\$skip (aggregation)

Definition

\$skip

Skips over the specified number of [documents](#) that pass into the stage and passes the remaining documents to the next stage in the [pipeline](#).

The `$skip` stage has the following prototype form:

```
{ $skip: <positive integer> }
```

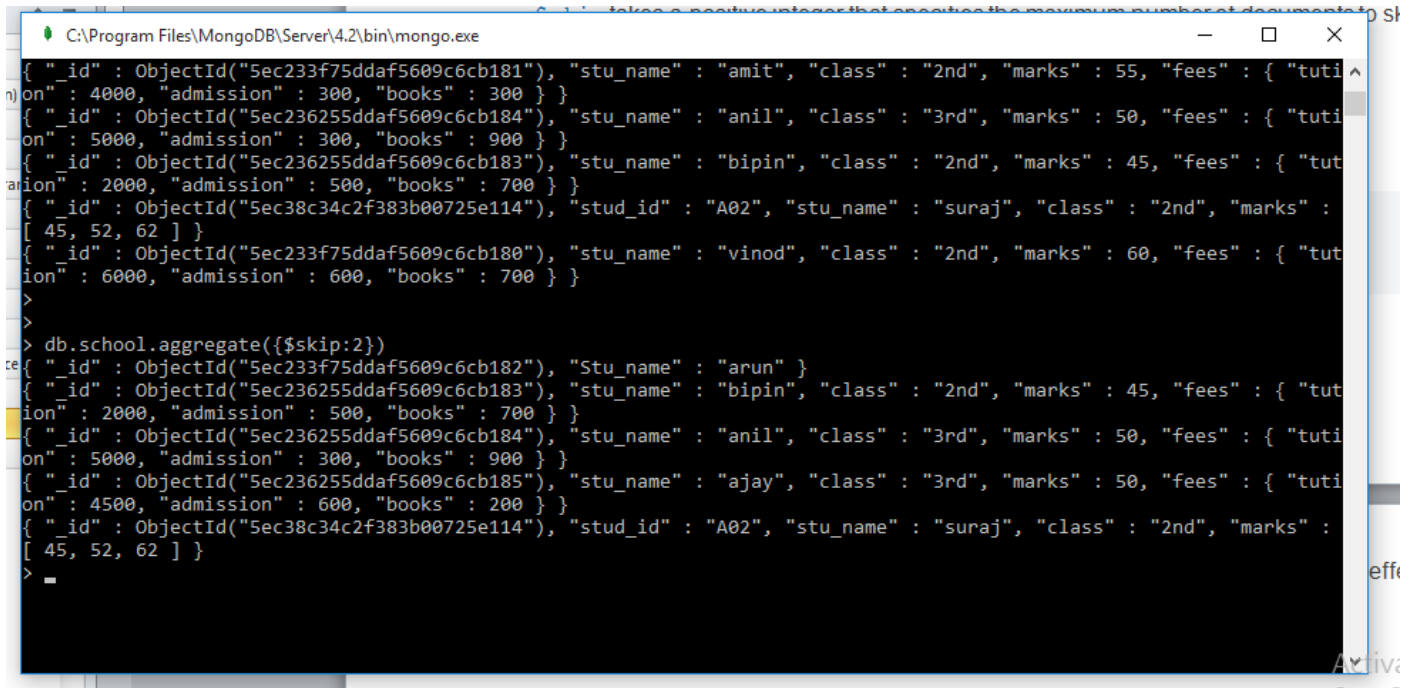
`$skip` takes a positive integer that specifies the maximum number of documents to skip.

Example

Consider the following example:

```
db.article.aggregate (
  { $skip : 5 }
);
```

This operation skips the first 5 documents passed to it by the pipeline. `$skip` has no effect on the content of the documents it passes along the pipeline.



```
C:\Program Files\MongoDB\Server\4.2\bin\mongo.exe
{ "_id" : ObjectId("5ec233f75ddaf5609c6cb181"), "stu_name" : "amit", "class" : "2nd", "marks" : 55, "fees" : { "tution" : 4000, "admission" : 300, "books" : 300 } }
{ "_id" : ObjectId("5ec236255ddaf5609c6cb184"), "stu_name" : "anil", "class" : "3rd", "marks" : 50, "fees" : { "tution" : 5000, "admission" : 300, "books" : 900 } }
{ "_id" : ObjectId("5ec236255ddaf5609c6cb183"), "stu_name" : "bipin", "class" : "2nd", "marks" : 45, "fees" : { "tution" : 2000, "admission" : 500, "books" : 700 } }
{ "_id" : ObjectId("5ec38c34c2f383b00725e114"), "stud_id" : "A02", "stu_name" : "suraj", "class" : "2nd", "marks" : [ 45, 52, 62 ] }
{ "_id" : ObjectId("5ec233f75ddaf5609c6cb180"), "stu_name" : "vinod", "class" : "2nd", "marks" : 60, "fees" : { "tution" : 6000, "admission" : 600, "books" : 700 } }
>
>
> db.school.aggregate({$skip:2})
{ "_id" : ObjectId("5ec233f75ddaf5609c6cb182"), "Stu_name" : "arun" }
{ "_id" : ObjectId("5ec236255ddaf5609c6cb183"), "stu_name" : "bipin", "class" : "2nd", "marks" : 45, "fees" : { "tution" : 2000, "admission" : 500, "books" : 700 } }
{ "_id" : ObjectId("5ec236255ddaf5609c6cb184"), "stu_name" : "anil", "class" : "3rd", "marks" : 50, "fees" : { "tution" : 5000, "admission" : 300, "books" : 900 } }
{ "_id" : ObjectId("5ec236255ddaf5609c6cb185"), "stu_name" : "ajay", "class" : "3rd", "marks" : 50, "fees" : { "tution" : 4500, "admission" : 600, "books" : 200 } }
{ "_id" : ObjectId("5ec38c34c2f383b00725e114"), "stud_id" : "A02", "stu_name" : "suraj", "class" : "2nd", "marks" : [ 45, 52, 62 ] }
>
=
```

\$limit (aggregation)

Definition

`$limit`

Limits the number of documents passed to the next stage in the [pipeline](#).

The `$limit` stage has the following prototype form:

```
{ $limit: <positive integer> }
```

`$limit` takes a positive integer that specifies the maximum number of documents to pass along.

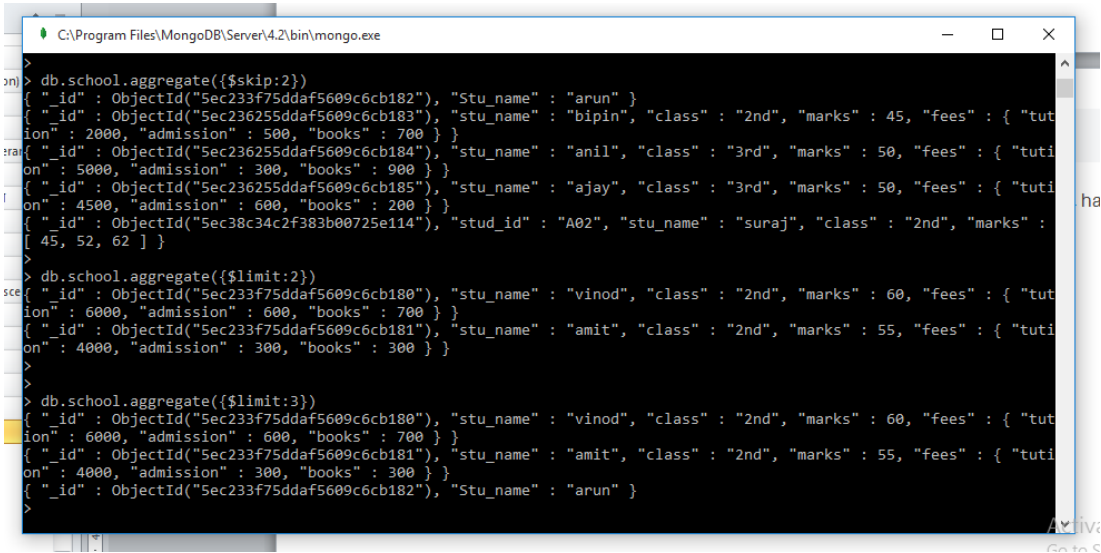
Example

Consider the following example:

```
db.article.aggregate (
```

```
{ $limit : 5 }  
);
```

This operation returns only the first 5 documents passed to it by the pipeline. `$limit` has no effect on the content of the documents it passes.



```
C:\Program Files\MongoDB\Server\4.2\bin\mongo.exe  
> db.school.aggregate({$skip:2})  
{ "_id" : ObjectId("5ec233f75ddaf5609c6cb182"), "Stu_name" : "arun" }  
{ "_id" : ObjectId("5ec236255ddaf5609c6cb183"), "stu_name" : "bipin", "class" : "2nd", "marks" : 45, "fees" : { "tu  
ion" : 2000, "admission" : 500, "books" : 700 } }  
{ "_id" : ObjectId("5ec236255ddaf5609c6cb184"), "stu_name" : "anil", "class" : "3rd", "marks" : 50, "fees" : { "tuti  
on" : 5000, "admission" : 300, "books" : 900 } }  
{ "_id" : ObjectId("5ec236255ddaf5609c6cb185"), "stu_name" : "ajay", "class" : "3rd", "marks" : 50, "fees" : { "tuti  
on" : 4500, "admission" : 600, "books" : 200 } }  
{ "_id" : ObjectId("5ec38c34c2f383b00725e114"), "stud_id" : "A02", "stu_name" : "suraj", "class" : "2nd", "marks" :  
[ 45, 52, 62 ] }  
>  
> db.school.aggregate({$limit:2})  
{ "_id" : ObjectId("5ec233f75ddaf5609c6cb180"), "stu_name" : "vinod", "class" : "2nd", "marks" : 60, "fees" : { "tut  
ion" : 6000, "admission" : 600, "books" : 700 } }  
{ "_id" : ObjectId("5ec233f75ddaf5609c6cb181"), "stu_name" : "amit", "class" : "2nd", "marks" : 55, "fees" : { "tuti  
on" : 4000, "admission" : 300, "books" : 300 } }  
>  
> db.school.aggregate({$limit:3})  
{ "_id" : ObjectId("5ec233f75ddaf5609c6cb180"), "stu_name" : "vinod", "class" : "2nd", "marks" : 60, "fees" : { "tut  
ion" : 6000, "admission" : 600, "books" : 700 } }  
{ "_id" : ObjectId("5ec233f75ddaf5609c6cb181"), "stu_name" : "amit", "class" : "2nd", "marks" : 55, "fees" : { "tuti  
on" : 4000, "admission" : 300, "books" : 300 } }  
{ "_id" : ObjectId("5ec233f75ddaf5609c6cb182"), "Stu_name" : "arun" }  
>
```