

### \*\*\* Data Processing with Aggregation Operators \*\*\*

1. Make sure you have MongoDB Database Tools installed (<https://docs.mongodb.com/database-tools/installation/installation/>).
2. Make sure you have a local MongoDB instance running (see lab #2) or use the remote MongoDB Atlas instance (see lab #1).

- ### 3. Download "Books" sample data
- <https://github.com/ozlerhakan/mongodb-json-files>

4. Use mongoimport tool to import the sample dataset  
Syntax: mongoimport <options> <connection-string> <file>  
mongoimport --db=books --collection=books  
mongodb://mongoadmin:secret@localhost:27888/?authSource=admin books.json

```
PS C:\Program Files\MongoDB\Server\7.0\bin> mongoimport --db books --collection books C:\Users\razva\Documents\Master_Poli\Baze_date\books.json
2024-05-19T01:21:41.056+0300    connected to: mongodb://localhost/
2024-05-19T01:21:41.104+0300    431 document(s) imported successfully. 0 document(s) failed to import.
PS C:\Program Files\MongoDB\Server\7.0\bin>
```

5. Connect to MongoDB using shell  
mongo mongodb://mongoadmin:secret@localhost:27888/?authSource=admin

- ```
6. See some sample documents, to get a taste of the sample data
> use books
> db.books.find({}).limit(3).pretty()
```

```

Stacks use books
watched to do books
books db.books.find({}).limit(1).pretty()
{
  {
    id: 9,
    title: "Specification By Example",
    isbn: "0978068801",
    pageCount: 432,
    publishedDate: "2011-06-01T08:00:00Z",
    thumbnailImageLink: "http://ecx.images-amazon.com/images/I/51jwvWQDq.jpg?book-thumb-image/falsh.jpg",
    status: "PUBLISHED",
    authors: [ "Steve Yegorov" ],
    categories: [ "Software Engineering" ]
  },
  {
    id: 9,
    title: "Zend Framework in Action",
    isbn: "0978068838",
    pageCount: 432,
    publishedDate: "2008-12-01T08:00:00Z",
    thumbnailImageLink: "http://ecx.images-amazon.com/images/I/51jwvWQDq.jpg?book-thumb-image/falsh.jpg",
    shortDescription: "Zend Framework in Action is a comprehensive tutorial that shows how to use the Zend Framework to create web-based applications and web services. This book takes you on an over-the-shoulder tour of the components of the Zend Framework as you build your own web application.",
    longDescription: "From rather humble beginnings as the Personal Home Page scripting language, PHP has found its way into almost every server, corporation, and dev shop in the world. On an average day, somewhere between 500,000 and 2 million orders do something in PHP. Even when you use a well-understood language like PHP, building a modern web application requires tools that decrease development time and cost while improving code quality. Frameworks such as Ruby-on-Rails and Django have been getting a lot of attention as a result of their ease of use and ability to speed up development. The Zend Framework has the backing of Zend Technologies Inc., the driving force behind the new programming language in which it is written. The first production release of the Zend Framework became available in July of 2007. Zend Framework in Action is a comprehensive tutorial that shows how to use the Zend framework to create web-based applications and web services. This book takes you on an over-the-shoulder tour of the components of the Zend Framework as you build a high-quality, real-world web application. This book is organized around the techniques you'll use every day as a web developer: data handling, forms, authentication, and so forth. As you follow the running example, you'll learn to build interactive Ajax-driven features into your application without sacrificing nut-and-bolt considerations like security and performance. This book is aimed at the intermediate PHP developer who wants to master framework-driven web development. Zend Framework in Action goes beyond the docs but still provides quick access to the most common topics encountered in the development of web applications.",
    status: "PUBLISHED",
    authors: [ "Josh Allen", "Nick Iotz", "Steven Brown" ],
    categories: [ "Web Development" ]
  },
  {
    id: 6,
    title: "Flac 3 in Action",
    isbn: "0978068790",
    pageCount: 176,
    publishedDate: "2009-02-01T08:00:00Z",
    thumbnailImageLink: "http://ecx.images-amazon.com/images/I/51jwvWQDq.jpg?book-thumb-image/flash.jpg",
    longDescription: "New web applications require engaging user-friendly interfaces, and the cooler, the better. With Flac 3, web developers of all skill levels can create high-quality, effective, and interactive Rich Internet Applications (RIAs) quickly and easily. Using the powerful browser from RIA development by offering sophisticated tools and a straightforward programming language in which you can focus on what you want to do instead of how to do it, and use that the major community of Flac are free and open-source, the cool barrier is gone, as well! Flac 3 in Action is an easy-to-follow, hands-on Flac tutorial. Check-out-of examples, this book goes beyond feature coverage and helps you put Flac to work in real day-to-day tasks. You'll quickly master the Flac API and learn to use the Flac runtime and validation. The last part of the book, working with databases, you get the information you need to integrate Flac with existing database systems. Many Flac books are overcrowding us with users. Focusing on the complexities of the languages and the super-specialized subjects in the Flac ecosystem, Flac 3 in Action fills out the nuts and bolts to help you get down to business with Flac 3, fast.",
    status: "PUBLISHED",
    authors: [ "Brian Alton", "Tom Hirsch", "Faisal Abid" ],
    categories: [ "Database" ]
  }
]
}

```

- ```
7. Let's find out the number of books in each individual state
> db.books.aggregate([
  {$group : { _id: '$status',
    count: {$sum:1} }}
]);
```

```
books> db.books.aggregate([
... {$group : { _id: '$status',
... count: {$sum:1} }}
... ]);
[ { _id: 'MEAP', count: 68 }, { _id: 'PUBLISH', count: 363 } ]
books>
```

8. Now determine the number of books published each year

```
> db.books.aggregate([
  {$group: { _id: {year : {$year : '$publishedDate'}}, count: {$sum:1}} }
]);
```

```
books> db.books.aggregate([
... {$group : { _id: '$status',
... count: {$sum:1} }}
... ]);
[ { _id: 'MEAP', count: 68 }, { _id: 'PUBLISH', count: 363 } ]
books> db.books.aggregate([
... { $group: { _id: {year : {$year : '$publishedDate'}}, count: {$sum:1}} }
... ]);
[
  { _id: { year: 2005 }, count: 23 },
  { _id: { year: 2014 }, count: 16 },
  { _id: { year: 1997 }, count: 13 },
  { _id: { year: 1996 }, count: 9 },
  { _id: { year: 1998 }, count: 12 },
  { _id: { year: 2004 }, count: 13 },
  { _id: { year: 2008 }, count: 19 },
  { _id: { year: 2001 }, count: 5 },
  { _id: { year: 2012 }, count: 31 },
  { _id: { year: 2000 }, count: 10 },
  { _id: { year: 2007 }, count: 14 },
  { _id: { year: 1993 }, count: 1 },
  { _id: { year: 2002 }, count: 23 },
  { _id: { year: 2011 }, count: 38 },
  { _id: { year: 1995 }, count: 7 },
  { _id: { year: 2006 }, count: 11 },
  { _id: { year: 1999 }, count: 14 },
  { _id: { year: 2009 }, count: 27 },
  { _id: { year: null }, count: 78 },
  { _id: { year: 2013 }, count: 31 }
]
Type "it" for more
books>
```

9. Nice, but we want the result sorted by year

```
> db.books.aggregate([
  {$group: { _id: {year : {$year : '$publishedDate'}}, count: {$sum:1}} },
  {$sort: {count:-1}}
]);
```

```
books> db.books.aggregate([
...   {$group: {_id: {year: {$year: '$publishedDate'}}}, count: {$sum:1}} },
...   {$sort: {count:-1}}
... ]);
[
  { _id: { year: null }, count: 78 },
  { _id: { year: 2011 }, count: 38 },
  { _id: { year: 2012 }, count: 31 },
  { _id: { year: 2013 }, count: 31 },
  { _id: { year: 2009 }, count: 27 },
  { _id: { year: 2005 }, count: 23 },
  { _id: { year: 2002 }, count: 23 },
  { _id: { year: 2010 }, count: 21 },
  { _id: { year: 2008 }, count: 19 },
  { _id: { year: 2014 }, count: 16 },
  { _id: { year: 2003 }, count: 15 },
  { _id: { year: 2007 }, count: 14 },
  { _id: { year: 1999 }, count: 14 },
  { _id: { year: 1997 }, count: 13 },
  { _id: { year: 2004 }, count: 13 },
  { _id: { year: 1998 }, count: 12 },
  { _id: { year: 2006 }, count: 11 },
  { _id: { year: 2000 }, count: 10 },
  { _id: { year: 1996 }, count: 9 },
  { _id: { year: 1995 }, count: 7 }
]
Type "it" for more
books>
```

10. Notice that the highest number of books have been published in an unknown year (78 book is a null year). We want that year removed from our statistics.

```
> db.books.aggregate([
  {$match: { 'publishedDate' : { "$exists": true } } },
  {$group: {_id: {year: {$year: '$publishedDate'}}}, count: {$sum:1}} },
  {$sort: {count:-1}}
]);
```

```
books> db.books.aggregate([
... {$match: { 'publishedDate' : {"$exists": true } }},
... {$group: {_id: {year : {$year : '$publishedDate'}}}, count: {$sum:1}} },
... {$sort: {count:-1}}
... ]);
[
  { _id: { year: 2011 }, count: 38 },
  { _id: { year: 2012 }, count: 31 },
  { _id: { year: 2013 }, count: 31 },
  { _id: { year: 2009 }, count: 27 },
  { _id: { year: 2005 }, count: 23 },
  { _id: { year: 2002 }, count: 23 },
  { _id: { year: 2010 }, count: 21 },
  { _id: { year: 2008 }, count: 19 },
  { _id: { year: 2014 }, count: 16 },
  { _id: { year: 2003 }, count: 15 },
  { _id: { year: 2007 }, count: 14 },
  { _id: { year: 1999 }, count: 14 },
  { _id: { year: 1997 }, count: 13 },
  { _id: { year: 2004 }, count: 13 },
  { _id: { year: 1998 }, count: 12 },
  { _id: { year: 2006 }, count: 11 },
  { _id: { year: 2000 }, count: 10 },
  { _id: { year: 1996 }, count: 9 },
  { _id: { year: 1995 }, count: 7 },
  { _id: { year: 2001 }, count: 5 }
]
Type "it" for more
books>
```

11. We want to get the number of books written by each of the authors

```
> db.books.aggregate([
  {$group: {_id: '$authors', count: {$sum:1}}},
  {$sort: {count:-1}}
]);
```

```
books> db.books.aggregate([
... {$group: {_id: '$authors', count: {$sum:1}} },
... {$sort: {count:-1}}
... ]);
[
  { _id: [], count: 37 },
  { _id: [ 'Vikram Goyal' ], count: 12 },
  { _id: [ 'Richard Siddaway' ], count: 3 },
  { _id: [ 'Ted Neward' ], count: 3 },
  { _id: [ 'David A. Black' ], count: 3 },
  { _id: [ 'Jon Skeet' ], count: 3 },
  { _id: [ 'Christian Bauer', 'Gavin King' ], count: 3 },
  { _id: [ 'Tim Hatton' ], count: 3 },
  { _id: [ 'Mala Gupta' ], count: 2 },
  { _id: [ 'Chris Buckett' ], count: 2 },
  { _id: [ 'Steven J. Gutz' ], count: 2 },
  { _id: [ 'Pete Brown' ], count: 2 },
  { _id: [ 'Amit Rathore' ], count: 2 },
  { _id: [ 'Ramnivas Laddad' ], count: 2 },
  { _id: [ 'Peter Armstrong' ], count: 2 },
  { _id: [ 'Alex Holmes' ], count: 2 },
  { _id: [ 'Benjamin G. Sullins', 'Mark B. Whipple' ], count: 2 },
  { _id: [ 'Don Jones' ], count: 2 },
  { _id: [ 'Bear Bibeault', 'Yehuda Katz' ], count: 2 },
  { _id: [ 'Craig Walls' ], count: 2 }
]
Type "it" for more
books>
```

12. Notice that not only individual authors have been counted but also groups of authors. We still want individual authors and for that we need to explode (\$unwind) the authors array.

```
> db.books.aggregate([
  {$unwind: '$authors' },
  {$group: {_id: '$authors', count: {$sum:1}} },
  {$sort: {count:-1}}
]);
```

```
books> db.books.aggregate([
...   {$unwind: '$authors' },
...   {$group: {_id: '$authors', count: {$sum:1}} },
...   {$sort: {count:-1}}
... ]);
[
  { _id: '', count: 59 },
  { _id: 'Vikram Goyal', count: 12 },
  { _id: 'Don Jones', count: 6 },
  { _id: 'Richard Siddaway', count: 6 },
  { _id: 'Christian Bauer', count: 5 },
  { _id: 'Jon Skeet', count: 5 },
  { _id: 'Yehuda Katz', count: 5 },
  { _id: 'Gavin King', count: 5 },
  { _id: 'Erik Hatcher', count: 4 },
  { _id: 'Greg Low', count: 4 },
  { _id: 'Kalen Delaney', count: 4 },
  { _id: 'Craig Walls', count: 4 },
  { _id: 'Matthew Scarpino', count: 3 },
  { _id: 'Paul S. Randal', count: 3 },
  { _id: 'Adam Machanic', count: 3 },
  { _id: 'Jeffery Hicks', count: 3 },
  { _id: 'W. Frank Ableson', count: 3 },
  { _id: 'Shannon Appelcline', count: 3 },
  { _id: 'Robi Sen', count: 3 },
  { _id: 'Gary Gregory', count: 3 }
]
Type "it" for more
books>
```

#### DO IT YOURSELF:

A. Find out the number of books for individual authors, excluding empty ones. Hint: start from the command #12 and match the authors that are not equal to an empty string.

```

Type "it" for more
books> db.books.aggregate([
...   {$unwind: '$authors'},
...   {$match: {authors: {$ne: ""}}},
...   {$group: {_id: '$authors', count: {$sum: 1}}},
...   {$sort: {count: -1}}
... ]);
[
  { _id: 'Vikram Goyal', count: 12 },
  { _id: 'Richard Siddaway', count: 6 },
  { _id: 'Don Jones', count: 6 },
  { _id: 'Gavin King', count: 5 },
  { _id: 'Jon Skeet', count: 5 },
  { _id: 'Yehuda Katz', count: 5 },
  { _id: 'Christian Bauer', count: 5 },
  { _id: 'Craig Walls', count: 4 },
  { _id: 'Greg Low', count: 4 },
  { _id: 'Kalen Delaney', count: 4 },
  { _id: 'Erik Hatcher', count: 4 },
  { _id: 'David A. Black', count: 3 },
  { _id: 'Jeffrey Palermo', count: 3 },
  { _id: 'Ted Neward', count: 3 },
  { _id: 'Christopher Allen', count: 3 },
  { _id: 'Tim Hatton', count: 3 },
  { _id: 'Kimberly L. Tripp', count: 3 },
  { _id: 'Daniel Minoli', count: 3 },
  { _id: 'Dave Crane', count: 3 },
  { _id: 'Jimmy Bogard', count: 3 }
]
Type "it" for more
books>

```

B. For each individual category, find out the total number of published pages from all books.  
Hint: Expand categories then group and sort by sum of pages.

```

books> db.books.aggregate([
...   {$unwind: '$categories'},
...   {$group: {_id: '$categories', totalPages: {$sum: '$pageCount'}}},
...   {$sort: {totalPages: -1}}
... ]);
[
  { _id: 'Java', totalPages: 39573 },
  { _id: 'Internet', totalPages: 18091 },
  { _id: 'Microsoft .NET', totalPages: 14859 },
  { _id: 'Web Development', totalPages: 7505 },
  { _id: 'Client-Server', totalPages: 5676 },
  { _id: 'Business', totalPages: 4850 },
  { _id: 'Software Engineering', totalPages: 4604 },
  { _id: 'Microsoft', totalPages: 4579 },
  { _id: 'Programming', totalPages: 4346 },
  { _id: 'Computer Graphics', totalPages: 3122 },
  { _id: 'PowerBuilder', totalPages: 3036 },
  { _id: 'Theory', totalPages: 2943 },
  { _id: 'Python', totalPages: 2894 },
  { _id: 'Networking', totalPages: 2565 },
  { _id: 'Perl', totalPages: 2312 },
  { _id: 'XML', totalPages: 1928 },
  { _id: 'Mobile Technology', totalPages: 1779 },
  { _id: 'Object-Oriented Programming', totalPages: 1634 },
  { _id: 'Miscellaneous', totalPages: 1412 },
  { _id: 'Open Source', totalPages: 948 }
]
Type "it" for more
books>

```

NEXT:

We will explore time series collections. Familiarize yourself with the topic at <https://docs.mongodb.com/manual/core/timeseries-collections/>