Aggregation operator

Aggregation in MongoDB is a powerful feature that allows for complex data transformations and computations on collections of documents. It enables users to group, filter, and manipulate data to produce summarized results.

The essential concepts and syntax of aggregation, equipping you with the knowledge to perform powerful data analysis using MongoDB.

We'll explore various operators like $avg, $min, $max, $push, and $addToSet, demonstrating how they can be used to group and summarize data in meaningful ways.

INTRODUCTION

Aggregations are operations that process data records and return computed results. MongoDB provides a rich set of aggregation operations that examine and perform calculations on the data sets. Running data aggregation on the mongod instance simplifies application code and limits resource requirements.

SYNTAX

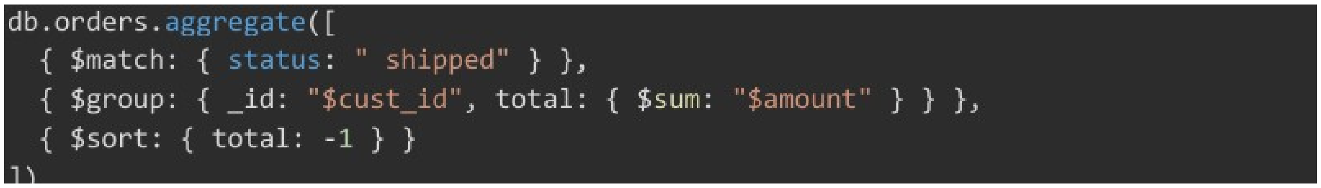
The fundamental syntax for performing aggregation in MongoDB is as follows:

db.collection.aggregate(<AGGREGATE OPERATION>)

‘db’ is the database object.

‘collection’ is the name of the collection on which to perform the aggregation.

Example:



```
db.orders.aggregate([
  { $match: { status: " shipped" } },
  { $group: { _id: "$cust_id", total: { $sum: "$amount" } } },
  { $sort: { total: -1 } }
])
```

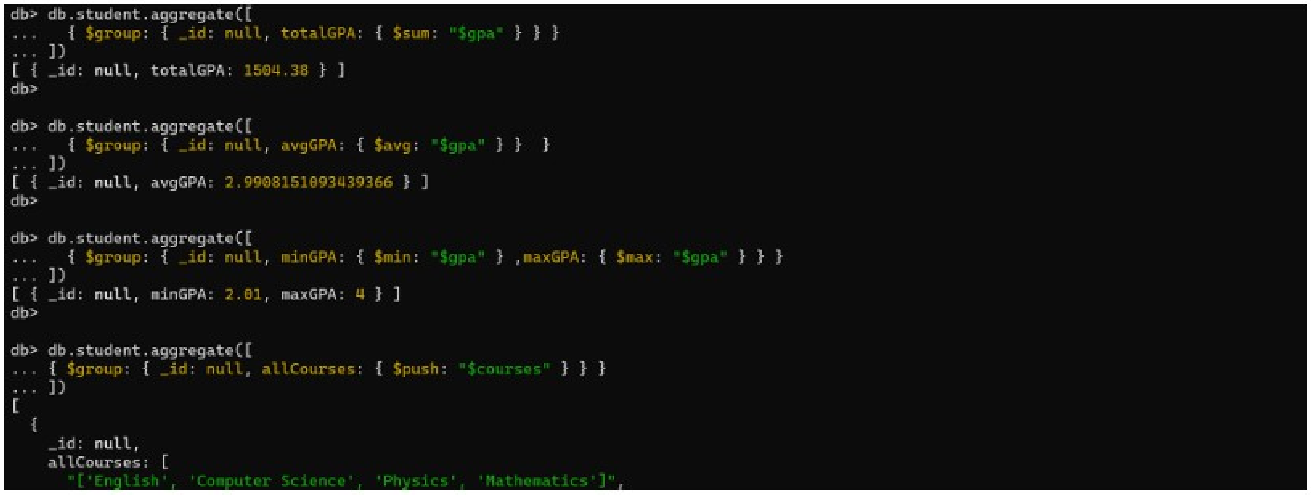
$match-Filters the documents to pass only the documents that match the specified condition(s) to the next pipeline stage.

$match-The $group stage separates documents into groups according to a "group key". The output is one document for each unique group key.

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

Types

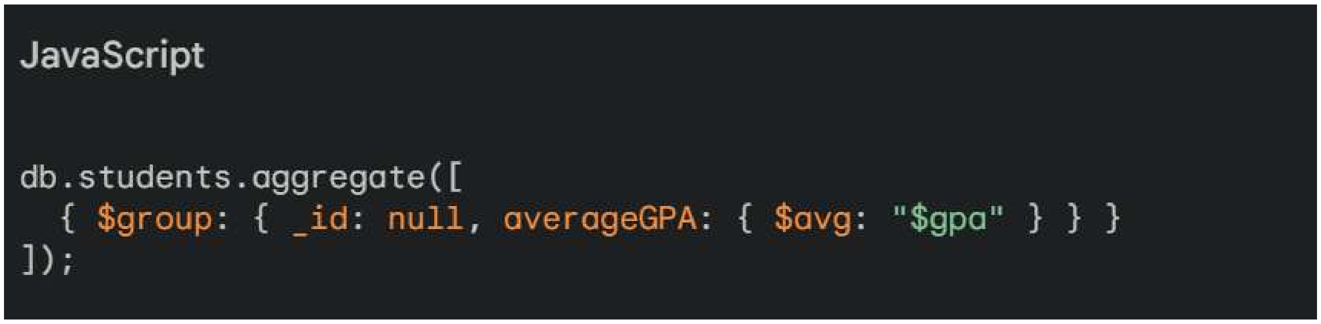
Expression Type	Description	Syntax
Accumulators	Perform calculations on entire groups of documents	
* $sum	Calculates the sum of all values in a numeric field within a group.	"$fieldName": { $sum: "$fieldName" }
* $avg	Calculates the average of all values in a numeric field within a group.	"$fieldName": { $avg: "$fieldName" }
* $min	Finds the minimum value in a field within a group.	"$fieldName": { $min: "$fieldName" }
* $max	Finds the maximum value in a field within a group.	"$fieldName": { $max: "$fieldName" }
* $push	Creates an array containing all unique or duplicate values from a field	"$arrayName": { $push: "$fieldName" }
* $addToSet	Creates an array containing only unique values from a field within a group.	"$arrayName": { $addToSet: "$fieldName" }
* $first	Returns the first value in a field within a group (or entire collection).	"$fieldName": { $first: "$fieldName" }
* $last	Returns the last value in a field within a group (or entire collection).	"$fieldName": { $last: "$fieldName" }



```
db> db.student.aggregate([
...   { $group: { _id: null, totalGPA: { $sum: "$gpa" } } }
... ])
[ { _id: null, totalGPA: 1504.38 } ]
db>

db> db.student.aggregate([
...   { $group: { _id: null, avgGPA: { $avg: "$gpa" } }  }
... ])
[ { _id: null, avgGPA: 2.9908151093439366 } ]
db>

db> db.student.aggregate([
...   { $group: { _id: null, minGPA: { $min: "$gpa" } ,maxGPA: { $max: "$gpa" } } }
... ])
[ { _id: null, minGPA: 2.01, maxGPA: 4 } ]
db>

db> db.student.aggregate([
... { $group: { _id: null, allCourses: { $push: "$courses" } } }
... ])
[
  {
    _id: null,
    allCourses: [
      "['English', 'Computer Science', 'Physics', 'Mathematics']",
```

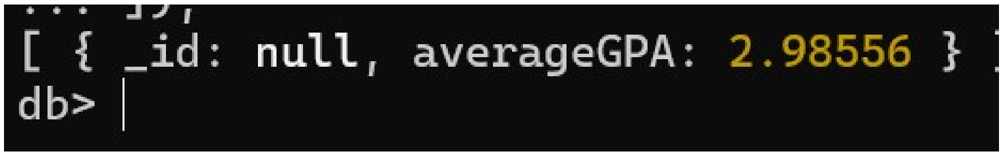
The code is written in MongoDB aggregation framework. It uses the $group operator to group.

Total GPA Calculation:
It calculates the total GPA by summing up the individual GPAs (stored in the field “Sgpa”).
Average GPA:
This code calculates the average GPA of all students. It uses the $avg operator to calculate the average of all GPA values in the collection.
Minimum and Maximum GPA:
This code finds the minimum and maximum GPA values in the collection using the $min and $max operators.



```
JavaScript

db.students.aggregate([
  { $group: { _id: null, averageGPA: { $avg: "$gpa" } } }
]);
```

The output of this program give the gpaof all students

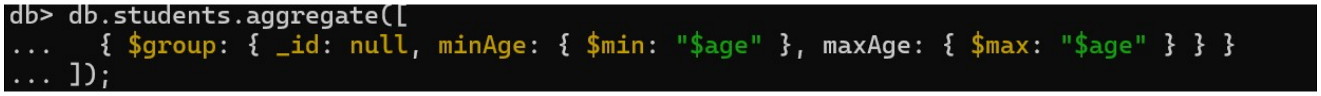


```
... ]);
[ { _id: null, averageGPA: 2.98556 } ]
db> 
```

Explanation:

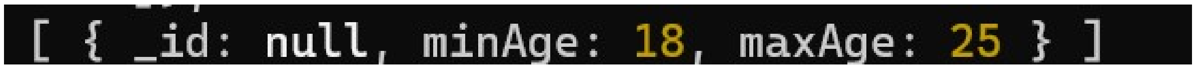
$group: Groups all documents together.
_id: null: Sets the group identifier to null (optional, as there's only one group in thiscase).
averageGPA: Calculates the average value of the "gpa" field using the $avg operator.

<u>Minimum and maximum age:</u>



```
db> db.students.aggregate([
...   { $group: { _id: null, minAge: { $min: "$age" }, maxAge: { $max: "$age" } } }
... ]);
```

The output give the minimum and maximum age of the students.



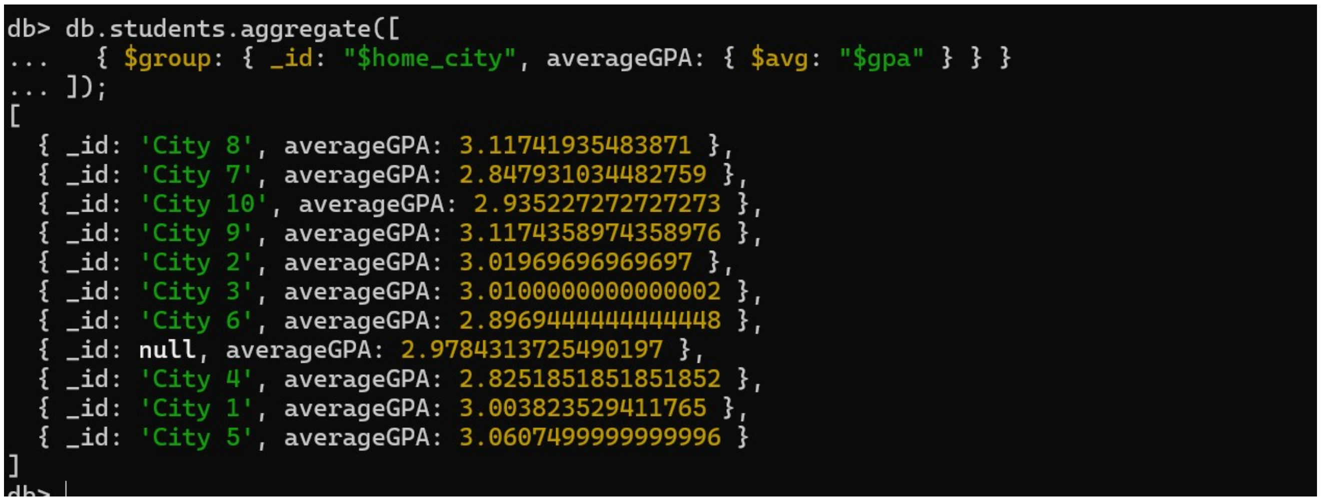
```
[ { _id: null, minAge: 18, maxAge: 25 } ]
```

minAge: Uses the $min operator to find the minimum value in the "agefield.

maxAge: Uses the $max operator to find the maximum value in the "age" field.

Here is the example for finding average gpa for all home cities

db.students.aggregate([{$group:{_id:"$home_city",averageGPA:{$avg:{$gpa}}}]);

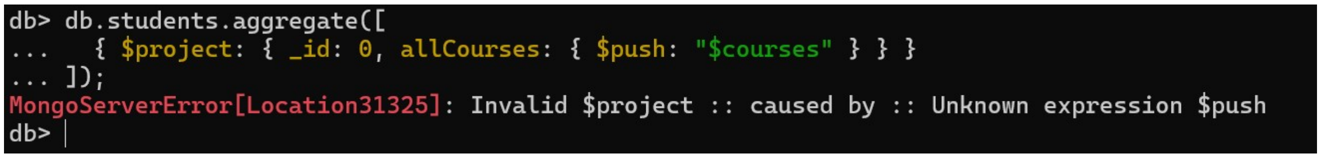


```
db> db.students.aggregate([
...   { $group: { _id: "$home_city", averageGPA: { $avg: "$gpa" } } }
... ]);
[
  { _id: 'City 8', averageGPA: 3.11741935483871 },
  { _id: 'City 7', averageGPA: 2.847931034482759 },
  { _id: 'City 10', averageGPA: 2.935227272727273 },
  { _id: 'City 9', averageGPA: 3.1174358974358976 },
  { _id: 'City 2', averageGPA: 3.01969696969697 },
  { _id: 'City 3', averageGPA: 3.0100000000000002 },
  { _id: 'City 6', averageGPA: 2.8969444444444448 },
  { _id: null, averageGPA: 2.9784313725490197 },
  { _id: 'City 4', averageGPA: 2.8251851851851852 },
  { _id: 'City 1', averageGPA: 3.003823529411765 },
  { _id: 'City 5', averageGPA: 3.0607499999999996 }
]
db> 
```

The output give the average GPA for all the home cities.

```
db.students.aggregate([
  { $project: { _id: 0, allCourses: { $push: "$courses" } } }
]);
```

`$project`: Transforms the input documents.
`_id: 0`: Excludes the `_id` field from the output documents.
`allCourses`: Uses the `$push` operator to create an array. It pushes all elements from the "courses" field of each student document into the `allCourses` array.



```
db> db.students.aggregate([
...   { $project: { _id: 0, allCourses: { $push: "$courses" } } }
... ]);
MongoServerError[Location31325]: Invalid $project :: caused by :: Unknown expression $push
db> 
```

The output will give as a Invalid $project because our array is incorrect.

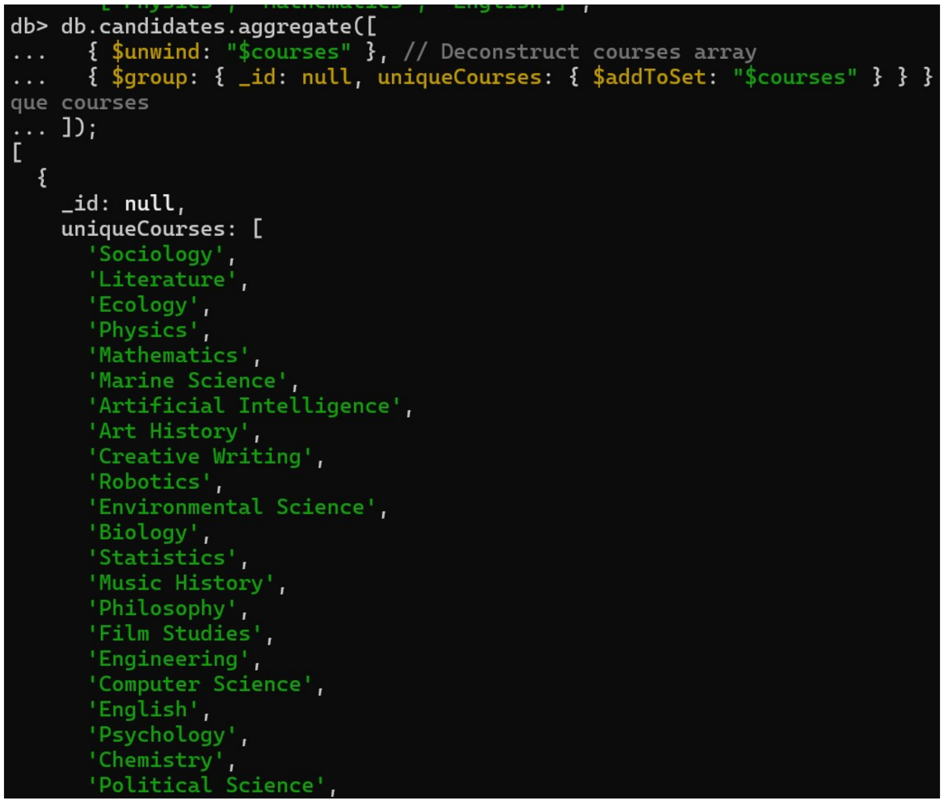
Collect Unique Courses Offered (Using $addToSet):

db.candidates.aggregate([

 { $unwind: "$courses" },

 { $group: { _id: null, uniqueCourses: { $addToSet: "$courses" } } }

]);



```
      [ 'Physics', 'Mathematics', 'English' ],
db> db.candidates.aggregate([
...   { $unwind: "$courses" }, // Deconstruct courses array
...   { $group: { _id: null, uniqueCourses: { $addToSet: "$courses" } } }
que courses
... ]);
[
  {
    _id: null,
    uniqueCourses: [
      'Sociology',
      'Literature',
      'Ecology',
      'Physics',
      'Mathematics',
      'Marine Science',
      'Artificial Intelligence',
      'Art History',
      'Creative Writing',
      'Robotics',
      'Environmental Science',
      'Biology',
      'Statistics',
      'Music History',
      'Philosophy',
      'Film Studies',
      'Engineering',
      'Computer Science',
      'English',
      'Psychology',
      'Chemistry',
      'Political Science',
```

The output give all the courses offered to a students.