

Class 6: Aggregation Operators

Execute Aggregation operations (\$avg, \$min,\$max, \$push, \$addToSet etc.). students encourage to execute several queries to demonstrate various aggregation operators)

What is aggregation?

Aggregation in MongoDB is a powerful framework for data aggregation operations, used to process data records and return computed results. It can be compared to the SQL GROUP BY clause but offers much more flexibility and capability. The aggregation framework operates through a pipeline approach, where multiple stages are used to transform and filter documents.

Syntax:

```
db.collection.aggregate(<AGGREGATE OPERATION>)
```

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" }

Average GPA of All Students

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

The provided MongoDB aggregation query calculates the average GPA of all students in the `students` collection.

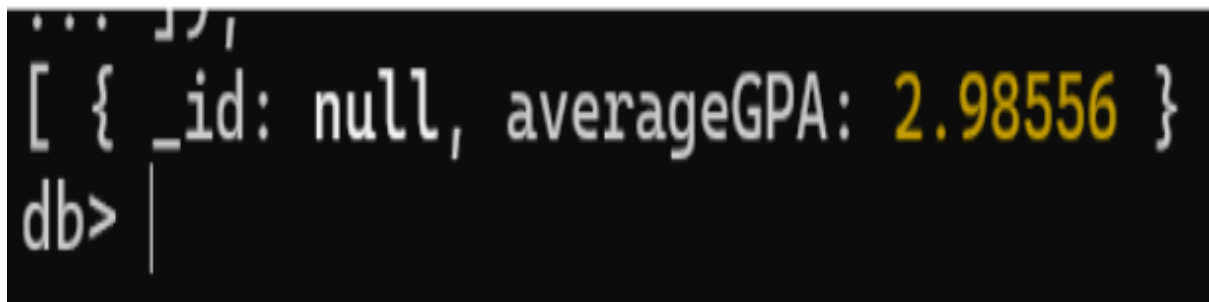
- **Aggregation Framework:**

- `db.students.aggregate()` is the entry point for the aggregation framework on the `students` collection.

- **\$group Stage:**

- `{ $group: { _id: null, averageGPA: { $avg: "$gpa" } } }` is a grouping stage in the aggregation pipeline.

Output:



```
... ],
[ { _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 this case).
 - `averageGPA`: Calculates the average value of the "gpa" field using the `$avg` operator.

Minimum and Maximum Age:

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

The provided MongoDB aggregation query calculates the minimum and maximum ages of students in the `students` collection.

- `db.students.aggregate()` is the entry point for the aggregation framework on the `students` collection.
- `{ $group: { _id: null, minAge: { $min: "$age" }, maxAge: { $max: "$age" } } }` is a grouping stage in the aggregation pipeline.

- `_id: null:`
 - This specifies that all documents will be grouped into a single group. The `null` value for `_id` means that there is no specific grouping key, and all documents are considered as a single group.
- `minAge: { $min: "$age" }:`
 - This creates a new field named `minAge` in the output document.
 - `$min: "$age"` calculates the minimum value of the `age` field across all documents in the `students` collection.
- `maxAge: { $max: "$age" }:`
 - This creates a new field named `maxAge` in the output document.
 - `$max: "$age"` calculates the maximum value of the `age` field across all documents in the `students` collection.

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

Explanation:

- Similar to the previous example, it uses `$group` to group all documents.
- `minAge`: Uses the `$min` operator to find the minimum value in the "age" field.
- `maxAge`: Uses the `$max` operator to find the maximum value in the "age" field.

How to get Average GPA for all home cities?

```
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 }
]
```

Pushing All Courses into a Single Array

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

- **Aggregation Framework:**

- `db.students.aggregate()` is the entry point for the aggregation framework on the `students` collection.

- **\$project Stage:**

- `{ $project: { _id: 0, allCourses: { $push: "$courses" } } }` is a projection stage in the aggregation pipeline.
- `_id: 0:`
 - This excludes the `_id` field from the output documents.
- `allCourses: { $push: "$courses" }:`
 - The `$push` operator is used to append values to an array in the resulting documents.
 - `"$courses"` specifies the field whose values will be pushed into the `allCourses` array.

Explanation:

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

Result:

This will return a list of documents, each with an `allCourses` array containing all unique courses offered (assuming courses might be duplicated across students).

BUT:

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

This is because our Array is incorrect:)

Collect Unique Courses Offered (Using \$addToSet);

```
db.candidates.aggregate([
  { $unwind: "$courses" },
  { $group: { _id: null, uniqueCourses: { $addToSet: "$courses" } } }
]);
```

The provided MongoDB aggregation query extracts a unique list of courses from the candidates collection.

- **Aggregation Framework:**
 - `db.students.aggregate()` is the entry point for the aggregation framework on the `students` collection.
- **\$group Stage:**
 - `{ $group: { _id: null, minAge: { $min: "$age" }, maxAge: { $max: "$age" } } }` is a grouping stage in the aggregation pipeline.

Result:

The result of this aggregation query will be a single document containing the minimum and maximum ages of all students:

What does it do?

```
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'
    ]
  }
]
```