AGGREGATION OPERATORS

In MongoDB, aggregation operators are used within the aggregation pipeline framework to process and transform documents in collections.

Here are some key aggregation operators commonly used in MongoDB:

- **$sum**, **$avg**, **$min**, **$max**:
- Accumulator operators used within $group to perform calculations on grouped documents.

Example:

```
db.collection.aggregate([
  {
    $group: {
      _id: "$category",
      averageAmount: { $avg: "$amount" },
      minAmount: { $min: "$amount" },
      maxAmount: { $max: "$amount" }
    }
  }
])
```

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" }

SYNTAX:db.collection.aggregate(<AGGREGATE OPERATION>

To calculate the average GPA of all students in MongoDB , you would follow these steps:

- **Group by `null`**: This groups all documents into a single group since we want to calculate the average across all documents.
- **Calculate average GPA**: Use the `$avg` aggregation operator to compute the average GPA.

Assuming you have a collection named **`students`** where each document contains a **`gpa`** field,

```
db.students.aggregate([
  {
    $group: {
      _id: null,  // Grouping all documents into a single group
      avgGPA: { $avg: "$gpa" }  // Calculate average GPA
    }
  }
])
```

Example:

If your `students` collection looks like this:

[

```
  { "_id": 1, "name": "Alice", "gpa": 3.5 },
  { "_id": 2, "name": "Bob", "gpa": 3.8 },
  { "_id": 3, "name": "Charlie", "gpa": 3.2 }
]
```

Running the above aggregation would yield a result like this:

```
[
 { "_id": null, "avgGPA": 3.5 }
]
```

"This result indicates that the average GPA of all students in the collection is `3.5`"

EXPLANATION: In reference with the above code

- $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:

To find the minimum and maximum age in a MongoDB collection, you can use the `aggregate` method.
Assuming your MongoDB collection is named `users` and the field for age is `age`-

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

Example:

```
{
  "_id": null,
  "minAge": 18,
  "maxAge": 65
}
```

"This result indicates that the minimum age in the `users` collection is 18 and the maximum age is 65"

Explanation:

- `_id: null`: Group all documents together.
- `minAge: { $min: "$age" }`: Calculate the minimum value of the `age` field.
- `maxAge: { $max: "$age" }`: Calculate the maximum value of the `age` field.

AVERAGE GPA FOR ALL HOME CITIES:

To calculate the average GPA for each home city in a MongoDB collection, you can use the `aggregate` method with the `$group` stage.

Pushing All Courses into a Single Array:

To aggregate and push all courses into a single array in MongoDB, you can use the `$group` and `$push` stages-

```
db.students.aggregate([
 {
  $group: {
    _id: null,
    allCourses: { $push: "$course" }
  }
 }
])
```

Example:

```
{
  "_id": null,
  "allCourses": ["Math", "Science", "History", "Math", "English"]
```

}
“This will return a single document with an array of all courses”

Explanation:

- `_id: null`: Group all documents together.

- `allCourses: { $push: "$course" }`: Push all values of the `course` field into an array.

BUT:

1.Filtering Unwanted Fields: If you want to exclude specific fields while pushing elements, you can leverage the $project stage with exclusion.

This approach uses $objectToArray to convert the document to an array of key-value pairs, excludes _id using projection, and then unwinds and extracts the course data.

2. Conditional Push Based on Field Values: If you want to conditionally push courses based on a specific field value, you can utilize the $cond operator within $project.

The $setDifference stage (optional) removes any null values that might be pushed due to the conditional logic.

To collect unique courses offered in a MongoDB collection, you can use the `$addToSet` operator within the `$group` stage in an aggregation pipeline. This will ensure that each course is added only once to the resulting array, effectively collecting unique courses.

JavaScript

```
db.militaryUnits.aggregate([

 {

$group: {

_id: null, // Group all documents

uniqueForces: { $addToSet: "$forces" } // Collect unique forces offered

}

}

])
```

RESULT: The resulting document will have-

• _id: The value you specified (here, null).

• uniqueForces: An array containing all unique forces offered across all military units in the collection.

EXAMPLE:

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

```
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',
```