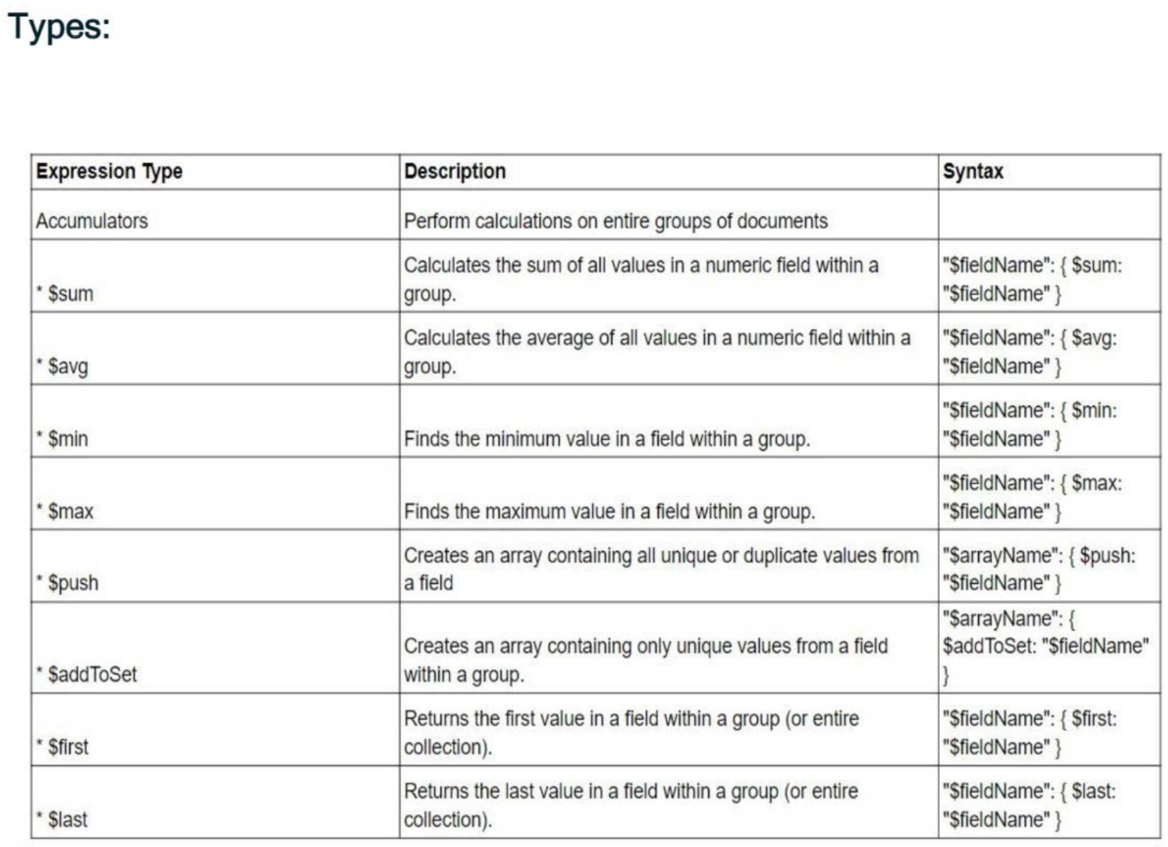
Class 6: Aggregation Operators

Aggregation operations process multiple documents and return computed results. You can use aggregation operations to:

- Group values from multiple documents together.
- Perform operations on the grouped data to return a single result.
- Analyze data changes over time.

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" }

Types:

The $project takes a document that can specify the inclusion of fields, the suppression of the _id field, the addition of new fields, and the resetting the values of existing fields. The specifications have the following forms:

Syntax	Description
<field>: <1 or true>	Specify the inclusion of a field.
_id: <0 or false>	Specify the suppression of the _id field.
<field>: <expression>	Add a new field or reset the value of an existing field.

$group

The $group stage is used to group documents based on one or more fields and perform aggregation operations on the grouped data. It allows you to:

• Group documents by one or more fields

• Perform aggregation operations on the grouped data, such as sum, average, count, etc.

• Create new fields that represent the aggregated values

The $group stage takes an object as its argument, where each key is the name of a field and the value is an expression that defines the aggregation operation.

$project

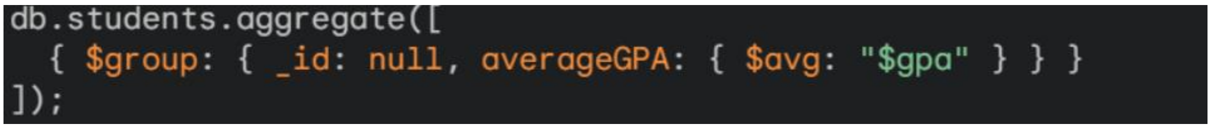
The $project stage is used to transform and reshape the data in the pipeline. It allows you to:

- Add new fields to the documents
- Rename existing fields
- Remove fields
- Perform calculations and transformations on fields
- Create new arrays or objects

The $project stage takes an object as its argument, where each key is the name of a field and the value is an expression that defines the transformation.

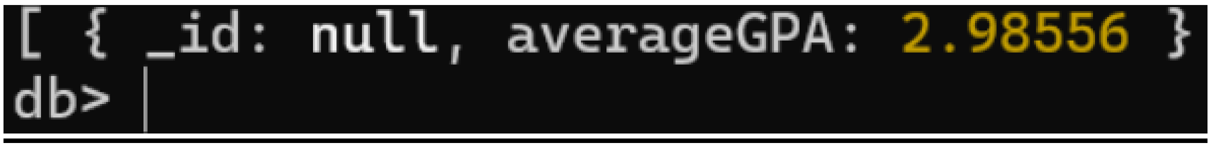
Example:

To find the average GPA of all students:



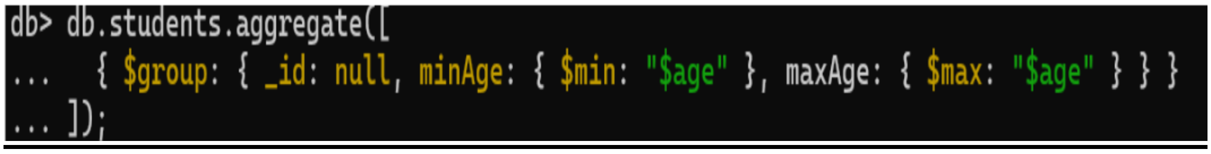
```
db.students.aggregate([
  { $group: { _id: null, averageGPA: { $avg: "$gpa" } } }
]);
```

Answer:



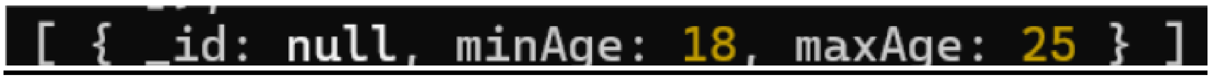
```
[ { _id: null, averageGPA: 2.98556 }
db> 
```

To find the Minimum and maximum age:



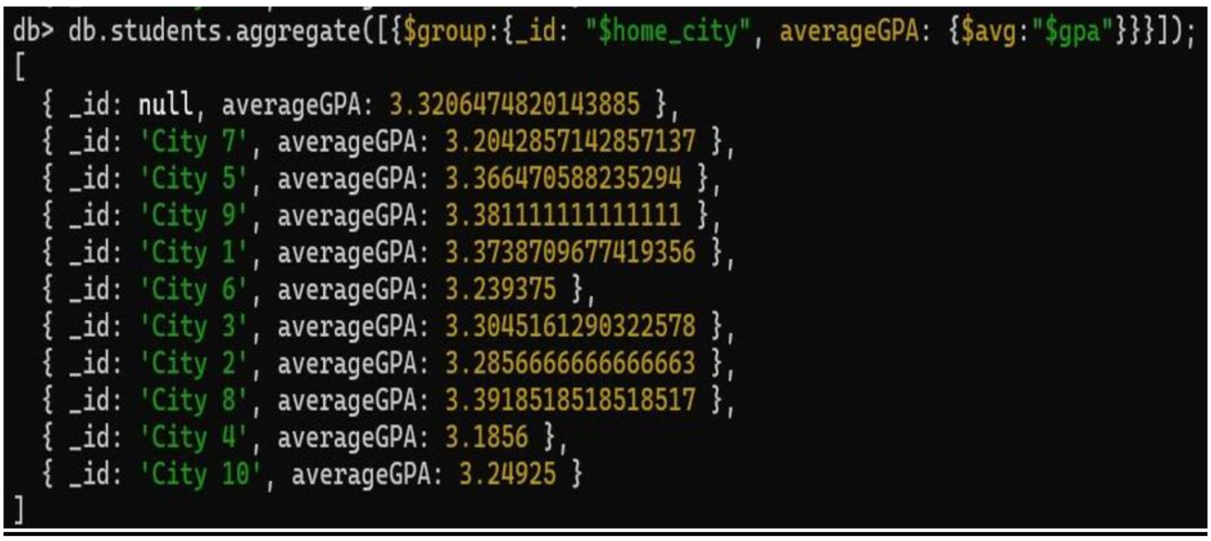
```
db> db.students.aggregate([
...   { $group: { _id: null, minAge: { $min: "$age" }, maxAge: { $max: "$age" } } }
... ]);
```

Answers:



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

To get average GPA for all the home cities:



```
db> db.students.aggregate([{$group:{_id: "$home_city", averageGPA: {$avg:"$gpa"}}}]);
[
  { _id: null, averageGPA: 3.3206474820143885 },
  { _id: 'City 7', averageGPA: 3.2042857142857137 },
  { _id: 'City 5', averageGPA: 3.366470588235294 },
  { _id: 'City 9', averageGPA: 3.381111111111111 },
  { _id: 'City 1', averageGPA: 3.3738709677419356 },
  { _id: 'City 6', averageGPA: 3.239375 },
  { _id: 'City 3', averageGPA: 3.3045161290322578 },
  { _id: 'City 2', averageGPA: 3.2856666666666663 },
  { _id: 'City 8', averageGPA: 3.3918518518518517 },
  { _id: 'City 4', averageGPA: 3.1856 },
  { _id: 'City 10', averageGPA: 3.24925 }
]
```