

AGGREGAION PIPELINE

The MongoDB **Aggregation pipeline** is a framework for data aggregation modeled on the concept of data processing pipelines. Documents enter a **multi-stage pipeline** that transforms the documents into aggregated results.

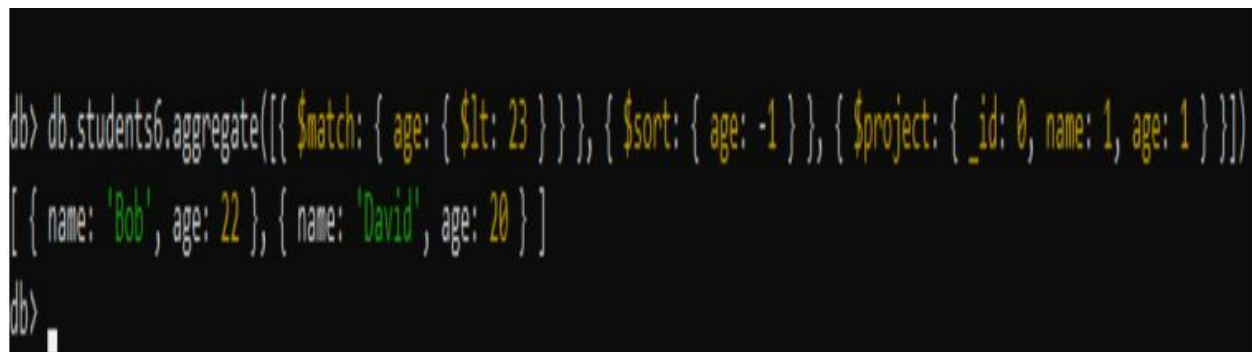
Each stage performs an operation on the input documents and passes the results to the next stage. The stages can filter, group, and modify the documents in various ways.

It encourage to execute several queries to demonstrate various **Aggregation operators**

\$match,\$sort:

Now to find students with age less than 23 it could be sorted by descending order to obtain only name and age we use a command .

```
db.students6.aggregate([{$match:{age:{$lt:23}}},{ $sort:{age:-1}},{$project:{_id:0,name:1,age:1}}])
```



```
db> db.students6.aggregate([ { $match: { age: { $lt: 23 } } }, { $sort: { age: -1 } }, { $project: { _id: 0, name: 1, age: 1 } } ] )
[ { name: 'Bob', age: 22 }, { name: 'David', age: 20 } ]
db>
```

\$group:

Now to group students by major to calculate average age and total number of students in each major using sum:2 we use a command

db.students6.aggregate([{\$group:{_id:"\$major",averageAge:{\$avg:"\$age"},totalStudents:{\$sum:2}}])

```
db> db.students6.aggregate([ { $group: { _id: "$major", averageAge: { $avg: "$age" }, totalStudents: { $sum: 2 } } }])

{ _id: 'Computer Science', averageAge: 22.5, totalStudents: 4 },
{ _id: 'English', averageAge: 28, totalStudents: 2 },
{ _id: 'Mathematics', averageAge: 22, totalStudents: 2 },
{ _id: 'Biology', averageAge: 23, totalStudents: 2 }
```

Now to group students by major to calculate average age and total number of students in each major using sum:1 we use a command

db.students6.aggregate([{\$group:{_id:"\$major",averageAge:{\$avg:"\$age"},totalStudents:{\$sum:1}}])

```
db> db.students6.aggregate([
... { $group: { _id: "$major", averageAge: { $avg: "$age" }, totalStudents: { $sum: 1 } } }])
[
  { _id: 'English', averageAge: 28, totalStudents: 1 },
  { _id: 'Computer Science', averageAge: 22.5, totalStudents: 2 },
  { _id: 'Mathematics', averageAge: 22, totalStudents: 1 },
  { _id: 'Biology', averageAge: 23, totalStudents: 1 }
]
```

Now to group students by minor to calculate average age and total number of students in each major using sum:1 we use a command

db.students6.aggregate([{\$group:{_id:"\$minor",averageAge:{\$avg:"\$age"},totalStudents:{\$sum:1}}])

```

]
db> db.students6.aggregate([ { $group: { _id: "$minor", averageAge: { $avg: "$age" }, totalStudents: { $sum: 1 } } } ] )
[ { _id: null, averageAge: 23.6, totalStudents: 5 } ]

```

\$project,\$skip:

Here to find students with an average score (from scores array) above 85 and skip the first document to do this so have to use a command is

```

db.students6.aggregate([{$project:{_id:0,name:1,averageScore:{$avg:"$scores"}}},{$match:{averageScore:{$gt:85}}},{$skip:1}])

```

```

db> db.students6.aggregate([
... {$project:{_id:0,name:1,averageScore:{$avg:"$scores"}}},{$match:{averageScore:{$gt:85}}},{$skip:1}])
[ { name: 'David', averageScore: 93.33333333333333 } ]

```

Again now to find students with an average score (from scores array) below 86 and skip the first two document to do this so have to use a command is

```

db.students6.aggregate([{$project:{_id:0,name:1,averageScore:{$avg:"$scores"}}},{$match:{averageScore:{$lt:86}}},{$skip:2}])

```

```

db> db.students6.aggregate([{$project:{_id:0,name:1,averageScore:{$avg:"$scores"}}},{$match:{averageScore:{$lt:86}}},{$skip:2}]);
[ { name: 'Eve', averageScore: 83.33333333333333 } ]

```

Here to find students name with an average score (from scores array) above 95 and skip the first one document to do this so have to use a command is

```
db.students6.aggregate([{$project:{name:1,averageScore:{$avg:"$scores"}},{$match:{averageScore:{$lt:95}}},{$skip:1}])
```

```
b> db.students6.aggregate([ { $project: { name: 1, averageScore: { $avg: "$scores" } } }, { $match: { averageScore: { $lt: 95 } } }, { $skip: 1 } ]])

{ _id: 2, name: 'Bob', averageScore: 91 },
{ _id: 3, name: 'Charlie', averageScore: 82 },
{ _id: 4, name: 'David', averageScore: 93.33333333333333 },
{ _id: 5, name: 'Eve', averageScore: 83.33333333333333 }
```