

# Banco de dados não relacionais



# mongoDB®

## Aggregations

### Estágio \$group

**Prof. Me. Ricardo Resende de Mendonça**  
[ricardo.mendonca@online.uscs.edu.br](mailto:ricardo.mendonca@online.uscs.edu.br)

**Prof. Me. Renato Carioca Duarte**  
[renato.duarte@online.uscs.edu.br](mailto:renato.duarte@online.uscs.edu.br)

# Introdução

## Aggregations

O pipeline de agregação é uma sequência de estágios que processam os documentos de entrada de acordo com operações específicas, passando os resultados de um estágio para o próximo. Cada estágio do pipeline pode executar uma variedade de operações, como filtragem, projeção, agrupamento, ordenação ou junções.



Input → \$match → \$lookup → \$project → \$group → \$sort → \$limit → Output

**\$match:** Filtra os documentos na coleção com base em critérios específicos, semelhante à operação find(). Reduzindo o número de documentos que serão processados nos estágios subsequentes.

**\$project:** Projeta os campos dos documentos de entrada, permitindo que você inclua, exclua ou renomeie campos, bem como crie novos campos calculados usando expressões de agregação.

**\$lookup:** Executa uma operação de junção (join) entre documentos de várias coleções, combinando documentos com base em campos correspondentes e incluindo os documentos correspondentes em um array aninhado.

**\$group:** Agrupa os documentos com base em uma chave específica e executa operações de agregação nos grupos resultantes. Permitindo o cálculo estatístico como soma, média, mínimo, máximo e contabilização por grupo.

**\$unwind:** Desconstrói um campo de array em vários documentos, criando um documento separado para cada elemento do array. Isso é útil quando você precisa operar em elementos individuais de um array.

**\$sort:** Realiza a ordenação do resultado.

**\$limit:** Limita o número de documentos que serão processados ou retornados na consulta. Isso é útil para restringir o tamanho dos resultados ou para otimizar consultas.

Para acompanhar os exemplos e testar os comandos você deve considerar as seguintes collections:

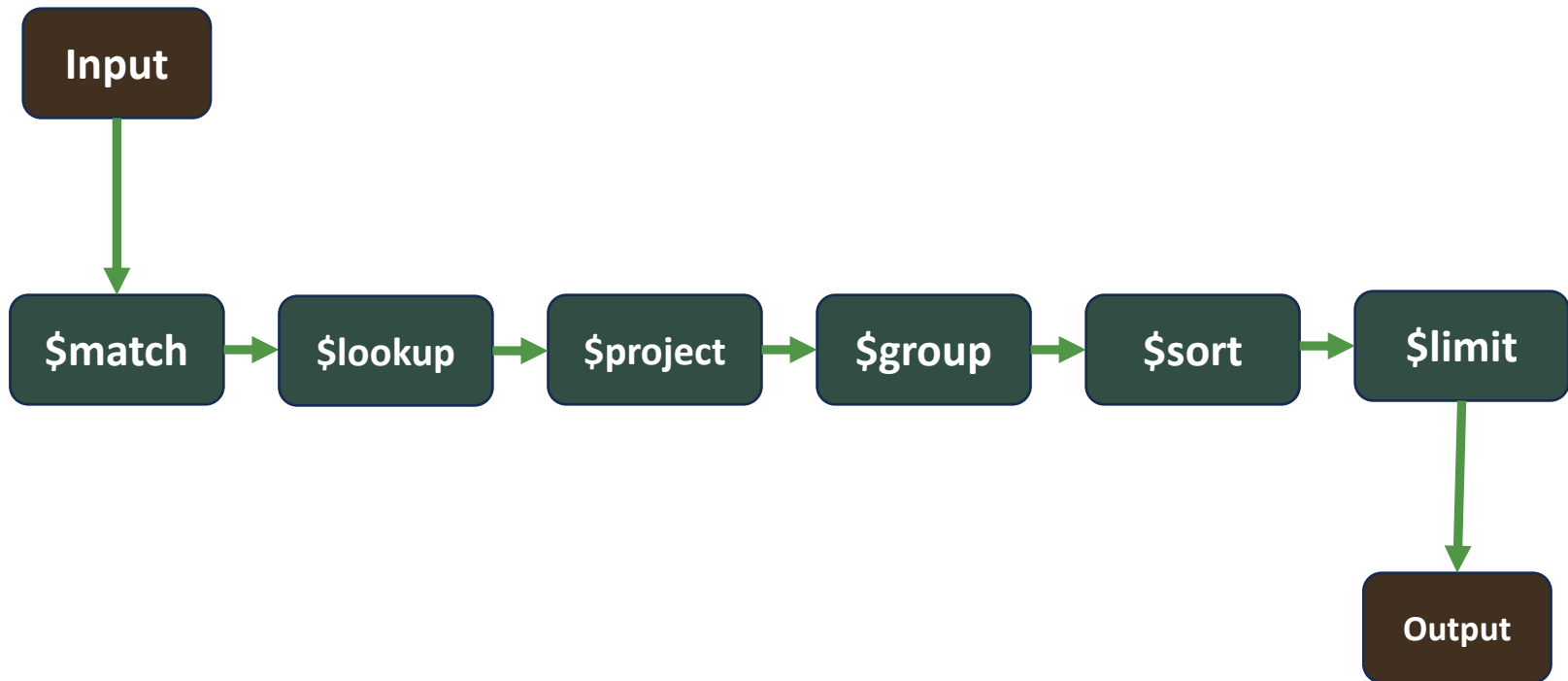
```
db.alunos.insertMany([
  { _id: 1, nome: "João", media_final: 8.5, sexo: "M", turma: "3AN" },
  { _id: 2, nome: "Maria", media_final: 7.9, sexo: "F", turma: "3AN" },
  { _id: 3, nome: "Pedro", media_final: 9.2, sexo: "M", turma: "3AN" },
  { _id: 4, nome: "Ana", media_final: 6.8, sexo: "F", turma: "3AN" },
  { _id: 5, nome: "Carlos", media_final: 8.0, sexo: "M", turma: "3AN" },
  { _id: 6, nome: "Julia", media_final: 7.3, sexo: "F", turma: "3AN" },
  { _id: 7, nome: "Mariana", media_final: 8.9, sexo: "F", turma: "3AN" },
  { _id: 8, nome: "Lucas", media_final: 9.5, sexo: "M", turma: "3AN" },
  { _id: 9, nome: "Fernando", media_final: 6.5, sexo: "M", turma: "3BN" },
  { _id: 10, nome: "Gabriel", media_final: 7.7, sexo: "M", turma: "3AN" },
  { _id: 11, nome: "Amanda", media_final: 8.3, sexo: "F", turma: "3AN" },
  { _id: 12, nome: "Rafael", media_final: 9.0, sexo: "M", turma: "3AN" },
  { _id: 13, nome: "Isabela", media_final: 7.1, sexo: "F", turma: "3BN" },
  { _id: 14, nome: "Mateus", media_final: 8.7, sexo: "M", turma: "3AN" },
  { _id: 15, nome: "Letícia", media_final: 9.8, sexo: "F", turma: "3AN" },
  { _id: 16, nome: "Daniel", media_final: 5.2, sexo: "M", turma: "3AN" },
  { _id: 17, nome: "Patrícia", media_final: 4.9, sexo: "F", turma: "3BN" },
  { _id: 18, nome: "Rodrigo", media_final: 5.8, sexo: "M", turma: "3AN" },
  { _id: 19, nome: "Laura", media_final: 6.3, sexo: "F", turma: "3BN" },
  { _id: 20, nome: "Gustavo", media_final: 7.2, sexo: "M", turma: "3AN" },
  { _id: 21, nome: "Camila", media_final: 6.1, sexo: "F", turma: "3BN" },
  { _id: 22, nome: "Vinícius", media_final: 6.7, sexo: "M", turma: "3AN" },
  { _id: 23, nome: "Carolina", media_final: 5.5, sexo: "F", turma: "3BN" },
  { _id: 24, nome: "Diego", media_final: 7.6, sexo: "M", turma: "3AN" },
  { _id: 25, nome: "Tatiane", media_final: 4.3, sexo: "F", turma: "3BN" },
  { _id: 26, nome: "Ricardo", media_final: 8.4, sexo: "M", turma: "3AN" },
  { _id: 27, nome: "Patrícia", media_final: 6.9, sexo: "F", turma: "3BN" },
  { _id: 28, nome: "Paulo", media_final: 9.1, sexo: "M", turma: "3AN" },
  { _id: 29, nome: "Fernanda", media_final: 5.7, sexo: "F", turma: "3BN" },
  { _id: 30, nome: "Luciana", media_final: 7.8, sexo: "F", turma: "3AN" }
])
```

Exemplo de um documento da collection Alunos:

```
1 {  
2   "_id" : 1,  
3   "nome" : "João",  
4   "media_final" : 8.5,  
5   "sexo" : "M",  
6   "turma" : "3AN"  
7 }
```

# Atenção!!!

Os estágios vistos anteriormente são válidos e devem ser utilizados sempre que necessário.



# .countDocuments()

A função `countDocuments` é uma função usada para contar o número de documentos em uma coleção.

Contabiliza todos os documentos  
da collection alunos

```
35 db.alunos.countDocuments( )
```

```
36
```

```
37
```

```
38 db.alunos.countDocuments({sexo: "F", turma: "3BN"})
```

```
39
```

```
40
```

```
41 db.alunos.count( )
```

Contabiliza todos os  
documentos da collection  
alunos que são do sexo  
feminino e da turma 3BN

Função depreciada do MongoDB 4  
em diante



# \$group

O estágio \$group agrupa os documentos com base em uma chave específica e executa operações de agregação nos grupos resultantes. Permitindo o cálculo estatístico como soma, média, mínimo, máximo e contabilização por grupo.

```
1 db.alunos.aggregate([
2   {
3     $group: {
4       _id: "$sexo",
5       media_final_media: { $avg: "$media_final" }
6     }
7   }
8 ])
```



```
1 /* 1 */
2 {
3   "_id" : "F",
4   "media_final_media" : 6.906666666666666
5 },
6
7 /* 2 */
8 {
9   "_id" : "M",
10  "media_final_media" : 7.806666666666667
11 }
```



# \$project

O estágio \$project permite não somente a definição de quais atributos serão exibidos, também sendo possível renomear no momento da projeção o título ou *label* do atributo.

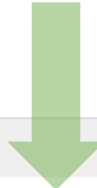
```
10 db.alunos.aggregate([
11   {
12     $group: {
13       _id: "$sexo",
14       media_final_media: { $avg: "$media_final" }
15     }
16   },
17   {$project: {_id: true, media_final_media: true}}
18 ])
```



Essa é a forma que estamos acostumados a utilizar o estágio de projeção

O estágio \$project permite não somente a definição de quais atributos serão exibidos, também sendo possível renomear no momento da projeção o título ou *label* do atributo.

```
21- db.alunos.aggregate([
22-   {
23-     $group: {
24-       _id: "$sexo",
25-       media_final_media: { $avg: "$media_final" }
26-     }
27-   },
28-   {$project: {_id: false, sexo: "$_id", media: "$media_final_media"}}
29- ])
```

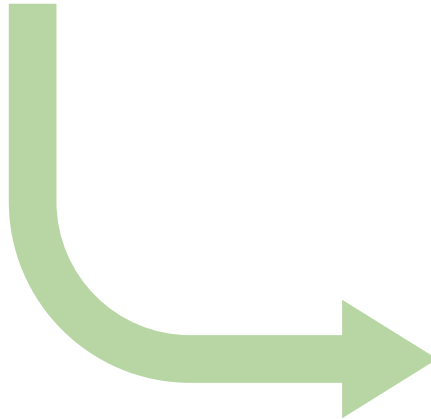


```
1  /* 1 */
2  {
3    "sexo" : "F",
4    "media" : 6.906666666666666
5  },
6
7  /* 2 */
8  {
9    "sexo" : "M",
10   "media" : 7.806666666666667
11 }
```

# \$round

O operador \$round é um operador de agregação no MongoDB que arredonda um número para o inteiro mais próximo ou para um número específico de casas decimais.

```
32 db.alunos.aggregate([
33   {
34     $group: {
35       _id: "$sexo",
36       media_final_media: { $avg: "$media_final" }
37     }
38   },
39   {
40     $project: {
41       _id: true,
42       media_final_arredondada: { $round: ["$media_final_media", 2] }
43     }
44   }
45 ])
```



```
1  /* 1 */
2  {
3    "_id" : "F",
4    "media_final_arredondada" : 6.91
5  },
6
7  /* 2 */
8  {
9    "_id" : "M",
10   "media_final_arredondada" : 7.81
11 }
```

# \$group – Múltiplos agrupamentos

O estágio \$group permite múltiplos agrupamentos, para isso o valor da chave `_id` deve ser um documento com múltiplos atributos. .

```
132 db.alunos.aggregate([
133   { $group:
134     { _id: { sexo: "$sexo", turma: "$turma" },
135       media: { $avg: "$media_final" }
136     },
137   },
138   {
139     $project: { _id: 0,
140                 sexo: "$_id.sexo",
141                 turma: "$_id.turma",
142                 media_arredondada: { $round: ["$media", 2] }
143               }
144   }
145 ])
146 ])
```

```
1 /* 1 */
2 {
3   "sexo" : "F",
4   "turma" : "3BN",
5   "media_arredondada" : 5.85
6 },
7
8 /* 2 */
9 {
10   "sexo" : "F",
11   "turma" : "3AN",
12   "media_arredondada" : 8.11
13 },
```



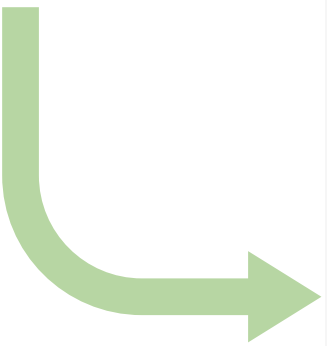
Os principais operadores para cálculos do estágio \$group são:

- **\$sum**: Calcula a soma dos valores de um campo específico em todos os documentos do grupo.
- **\$avg**: Calcula a média dos valores de um campo específico em todos os documentos do grupo.
- **\$min**: Encontra o valor mínimo de um campo específico em todos os documentos do grupo.
- **\$max**: Encontra o valor máximo de um campo específico em todos os documentos do grupo.
- **\$first**: Retorna o primeiro valor de um campo específico encontrado nos documentos do grupo.
- **\$last**: Retorna o último valor de um campo específico encontrado nos documentos do grupo.

# \$group & \$sum

**\$sum:** Calcula a soma dos valores de um campo específico em todos os documentos do grupo.

```
171- db.alunos.aggregate([
172-   {$group:
173-     { _id: {sexo: "$sexo", turma: "$turma"},
174-       soma: {$sum: "$media_final"},
175-     }
176-   },
177-   {
178-     $project: { _id: 0,
179-                 sexo: "$_id.sexo",
180-                 turma: "$_id.turma",
181-                 soma: true,
182-               }
183-   },
184- ])
```



```
2- {
3-   "soma" : 110.6,
4-   "sexo" : "M",
5-   "turma" : "3AN"
6- },
7-
8- /* 2 */
9- {
10-   "soma" : 56.8000000000000004,
11-   "sexo" : "F",
12-   "turma" : "3AN"
13- },
```



# \$group & \$sum (*Utilizando para contabilização*)

Unidade 3

**\$sum:** Calcula a soma dos valores de um campo específico em todos os documentos do grupo.

```
171 db.alunos.aggregate([
172   {$group:
173     { _id: {sexo: "$sexo", turma: "$turma"},
174       total: {$sum: 1},
175     }
176   },
177   {
178     $project: { _id: 0,
179                 sexo: "$_id.sexo",
180                 turma: "$_id.turma" ,
181                 total: true,
182               }
183   },
184 ])
```

```
2 {
3   "total" : 8,
4   "sexo" : "F",
5   "turma" : "3BN"
6 },
7
8 /* 2 */
9 {
10  "total" : 7,
11  "sexo" : "F",
12  "turma" : "3AN"
13 },
14
15 /* 3 */
16 {
17  "total" : 1,
18  "sexo" : "M",
19  "turma" : "3BN"
20 },
21
22 /* 4 */
23 {
24  "total" : 14,
25  "sexo" : "M",
26  "turma" : "3AN"
27 }
```

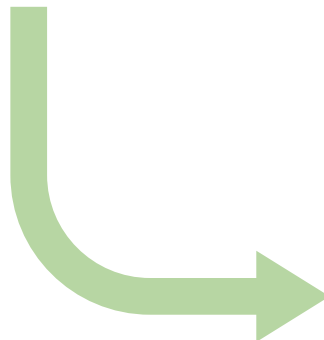


mongoDB

# \$group & \$avg

**\$avg**: Calcula a média dos valores de um campo específico em todos os documentos do grupo.

```
171- db.alunos.aggregate([
172-   {$group:
173-     { _id: {sexo: "$sexo", turma: "$turma"},
174-       media: {$avg: "$media_final"},
175-     }
176-   },
177-   {
178-     $project: { _id: 0,
179-                 sexo: "$_id.sexo",
180-                 turma: "$_id.turma",
181-                 media_arredondada: {$round: ["$media", 2]},
182-               }
183-   },
184- ])
```



```
2- {
3-   "sexo" : "F",
4-   "turma" : "3BN",
5-   "media_arredondada" : 5.85
6- },
7-
8- /* 2 */
9- {
10-   "sexo" : "F",
11-   "turma" : "3AN",
12-   "media_arredondada" : 8.11
13- },
```





# \$group & \$min

**\$min:** Encontra o valor mínimo de um campo específico em todos os documentos do grupo.

```
171 db.alunos.aggregate([
172   {$group:
173     { _id: {sexo: "$sexo", turma: "$turma"},
174       menor: {$min: "$media_final"},
175     }
176   },
177   {
178     $project: { _id: 0,
179                 sexo: "$_id.sexo",
180                 turma: "$_id.turma" ,
181                 menor: true,
182               }
183   },
184 ])
```

```
2 {
3   "menor" : 4.3,
4   "sexo" : "F",
5   "turma" : "3BN"
6 },
7
8 /* 2 */
9 {
10   "menor" : 6.8,
11   "sexo" : "F",
12   "turma" : "3AN"
13 },
```



# \$group & \$max

**\$max:** Encontra o valor máximo de um campo específico em todos os documentos do grupo.

```
171 db.alunos.aggregate([
172   {$group:
173     { _id: {sexo: "$sexo", turma: "$turma"},
174       maior: {$max: "$media_final"},
175     }
176   },
177   {
178     $project: { _id: 0,
179                 sexo: "$_id.sexo",
180                 turma: "$_id.turma",
181                 maior: true,
182               }
183   },
184 ])
```

```
2 {
3   "maior" : 6.5,
4   "sexo" : "M",
5   "turma" : "3BN"
6 },
7
8 /* 2 */
9 {
10   "maior" : 9.8,
11   "sexo" : "F",
12   "turma" : "3AN"
13 },
```



# \$group & ( \$first & \$last)

**\$first:** Retorna o primeiro valor de um campo específico encontrado nos documentos do grupo.

**\$last:** Retorna o último valor de um campo específico encontrado nos documentos do grupo.

```
107 db.alunos.aggregate([
108   { $sort: {nome: 1} },
109   {$group:
110     { _id: {sexo: "$sexo", turma: "$turma"},
111       primeiro: {$first: "$nome"},
112       ultimo: {$last: "$nome"},
113       total: {$sum: 1}
114     }
115   },
116   { $sort: {turma: 1, sexo: 1} }
117 ])
```

# \$group & \$lookup

O estágio de **\$group** pode ser utilizado em conjunto com os demais estágios. Para exemplificar a utilização do **\$group** e **\$lookup** sendo utilizados em conjunto você deve considerar as seguintes collections:

```
db.funcionarios.insertMany([
  { _id: 1, nome: "João", nascimento: ISODate("2000-03-15"), cargo_id: 10, departamento_id: 91, salario: 4200 },
  { _id: 2, nome: "Maria", nascimento: ISODate("2002-07-25"), cargo_id: 20, departamento_id: 91, salario: 8500 },
  { _id: 3, nome: "Pedro", nascimento: ISODate("2005-11-10"), cargo_id: 30, departamento_id: 92, salario: 6450 }
])
```

```
db.cargos.insertMany([
  { _id: 10, nome: "Vendedor", salario_base: 4000 },
  { _id: 20, nome: "Gerente de Vendas", salario_base: 8000 },
  { _id: 30, nome: "Analista de Marketing", salario_base: 6000 }
])
```

```
db.departamentos.insertMany([
  { _id: 91, nome: "Vendas", telefones: ["95984-7894", "3651-4512"], andar: 2, sala: "Sala 201" },
  { _id: 92, nome: "Marketing", telefones: ["95986-3649", "2461-3894"], andar: 3, sala: "Sala 301" }
])
```

# \$group & \$lookup

O estágio de **\$group** pode ser utilizado em conjunto com os demais estágios. Para exemplificar a utilização do **\$group** e **\$lookup** sendo utilizados em conjunto você deve considerar as seguintes collections:

## Versão resumida dos Documentos existentes na *collection* funcionarios

```
{ _id: 1, nome: "João", cargo_id: 10, departamento_id: 91, salario: 4200 },  
{ _id: 2, nome: "Maria", cargo_id: 20, departamento_id: 91, salario: 8500 },  
{ _id: 3, nome: "Pedro", cargo_id: 30, departamento_id: 92, salario: 6450 }
```

# \$group & \$lookup

O estágio de **\$group** pode ser utilizado em conjunto com os demais estágios. Para exemplificar a utilização do \$group e \$lookup sendo utilizados em conjunto você deve considerar as seguintes collections:

```
20 db.funcionarios.aggregate([
21   { $lookup: {
22     from: "departamentos",
23     localField: "departamento_id",
24     foreignField: "_id",
25     as: "departamento"
26   }
27 },
28 { $unwind: "$departamento" },
29 { $group: {
30   _id: "$departamento._id",
31   nome_departamento: { $first: "$departamento.nome" },
32   total_funcionarios: { $sum: 1 },
33   salario_medio: { $avg: "$salario" }
34 }
35 },
36 { $project: {
37   _id: 0, nome_departamento: 1, total_funcionarios: 1,
38   salario_medio: { $round: ["$salario_medio", 2] }
39 }
40 }
41 ]);
```

```
1  /* 1 */
2  {
3    "nome_departamento" : "Marketing",
4    "total_funcionarios" : 1,
5    "salario_medio" : Double("6450")
6  },
7
8  /* 2 */
9  {
10   "nome_departamento" : "Vendas",
11   "total_funcionarios" : 2,
12   "salario_medio" : Double("6350")
13 }
```



# Dúvidas?

