T4:3Agg MongoDB

lunes, 12 de mayo de 2025 18:07

Es la forma en la que Mongo permite procesar datos, de forma ordenada, siguiendo pasos para, filtrar, transformar, agrupar, ordenar, hacer cálculos...

El pipeline se entiende como una cadena de montaje (llamados ***pipelines de agregación***) en donde hay *stages*, que vendrán definidos por un "$"

La forma principal, sencilla y directa de hacer consultas en MongoDB es con *find()*, sin embargo, es también una forma limitada, ya que solo permite filtrado y proyección de resultados...
La *aggregate()* es la query para procedimientos más complejos.

Los Stages mas usados/básicos:

🧪 Juntos en un pipeline:

```
js                                                    Copiar
db.listingsAndReviews.aggregate([
  { $match: { amenities: "Wifi" } },        // ✅ Se quedan solo los qu
  { $project: { name: 1, price: 1, _id: 0 } } // ✅ De esos, solo mues
])
```

✅ Conclusión rápida:

- `$match` = **¿Qué documentos paso?**
- `$project` = **¿Qué campos de esos documentos muestro o modifico?**

```
db.listingsAndReviews.aggregate([
  {
    $group: {
      _id: "$address.country",        // Agrupa por país
      totalAlojamientos: { $sum: 1 } // Cuenta cuántos documentos ha
    }
  }
])
```

🔍 ¿Qué devuelve?

```
js                                                    Copiar   Editar
{ _id: "Brazil", totalAlojamientos: 154 }
{ _id: "United States", totalAlojamientos: 1987 }
{ _id: "Spain", totalAlojamientos: 380 }
...
```

🔵 `_id` : es **por lo que estás agrupando** (como el "GROUP BY" de SQL)
🔵 `totalAlojamientos` : es el **cálculo que haces dentro del grupo** (en este caso, contar).

🔷 Etapas más comunes del pipeline de agregación:

Etapa	Qué hace
`$match`	Filtra documentos (como `find()`)
`$group`	Agrupa por un campo y permite aplicar funciones de agregación como `$sum`, `$avg`, `$count`...
`$sort`	Ordena los resultados
`$project`	Elige qué campos mostrar y cómo (incluso puedes renombrar, calcular, crear nuevos)
`$limit`	Limita el número de resultados
`$skip`	Salta una cantidad de resultados (como paginación)
`$unwind`	Descompone arrays para analizarlos elemento por elemento
`$lookup`	Une documentos de otra colección (como un JOIN en SQL)

1. En `sample_airbnb.listingsAndReviews`, ¿qué "room types" existen?

```
sample_airbnb> db.listingsAndReviews.aggregate({"$group": { "_id": "$room types"}})
[ { _id: null } ]
sample_airbnb> db.listingsAndReviews.aggregate({"$group": { "_id": "$room_type"}})
[
  { _id: 'Private room' },
  { _id: 'Shared room' },
  { _id: 'Entire home/apt' }
]
sample_airbnb> db.listingsAndReviews.distinct("room_type")
[ 'Entire home/apt', 'Private room', 'Shared room' ]
sample_airbnb>
```

2. En `sample_training.companies`, haga una query que devuelva el nombre y el año en el que se fundaron las 5 compañías más antiguas.

```
sample_training> db.companies.find({},{"name": 1, "founded_year": 1, "_id": 0}).sort("founded_year").limit(5)
[
  { name: 'there', founded_year: null },
  { name: 'spinvox', founded_year: null },
  { name: 'Edgeio', founded_year: null },
  { name: 'Info', founded_year: null },
  { name: 'PlaceBlogger', founded_year: null }
]
sample_training> db.companies.aggregate([
...   { $project: { name: 1, founded_year: 1, _id: 0 } },
...   { $sort: { founded_year: 1 } },
...   { $limit: 5 }
... ])
[
  { name: 'there', founded_year: null },
  { name: 'spinvox', founded_year: null },
  { name: 'Edgeio', founded_year: null },
  { name: 'Info', founded_year: null },
  { name: 'PlaceBlogger', founded_year: null }
]
```

```
sample_training> db.companies.find({"founded_year": {"$ne": null}},{"name": 1, "founded_year": 1, "_id": 0}).sort("founded_year").limit(5)
[
  { name: 'US Army', founded_year: 1800 },
  { name: 'Alstrasoft', founded_year: 1800 },
  { name: 'DuPont', founded_year: 1802 },
  { name: 'Bachmann Industries', founded_year: 1833 },
  { name: 'Bertelsmann', founded_year: 1835 }
]
sample_training>
```

```
sample_training> db.companies.aggregate([
... { "$match": { "founded_year": { "$ne": null}}},
... { "$project": {"name": 1, "founded_year":1, "_id":0}},
... { "$sort": {"founded_year":1}},
... { "$limit": 5}
... ])
[
  { name: 'US Army', founded_year: 1800 },
  { name: 'AlstrasoftA', founded_year: 1800 },
  { name: 'DuPont', founded_year: 1802 },
  { name: 'Bachmann Industries', founded_year: 1833 },
  { name: 'Bertelsmann', founded_year: 1835 }
]
sample_training>
```

3. En `sample_training.trips`, ¿en qué año nació el ciclista más joven? (sol. 1999)

```
sample_training> db.trips.find({"birth year": {"$type": "number"}}, {"birth year": 1, "_id":0}).sort({"birth year":-1}).limit(1)
[ { 'birth year': 1999 } ]
sample_training>
```

```
sample_training> db.trips.aggregate([
...   { $match: { "birth year": { $type: "number" } } },
...   { $sort: { "birth year": -1 } },
...   { $limit: 1 },
...   { $project: { _id: 0, "birth year": 1 } }
... ])
[ { 'birth year': 1999 } ]
```