Introducción a Pipeline de Agregación

Bases de Datos NoSQL

Bases de Datos, 2023

SQL a MongoDB: Consultas básicas (repaso)

MySQL	MongoDB
`SELECT * FROM users`	`db.users.find()`

`SELECT name, email`
`FROM users`
	`db.users.find(`
	`  { },`
	`  { "name": 1, "email": 1, "_id": 0 }`
	`)`

`SELECT name, email`
`FROM users`
`WHERE name = "Ned Stark"`
	`db.users.find(`
	`  { "name": "Ned Stark" },`
	`  { "name": 1, "email": 1, "_id": 0 }`
	`)`

SQL a MongoDB: Operaciones de agregación

MySQL	MongoDB

`SELECT COUNT(*), MAX(imdb_rating)`
`FROM movies`
`WHERE year = 2019`
	

`SELECT year, COUNT(*)`
`FROM movies`
`WHERE year >= 2000`
`GROUP BY year`
`HAVING COUNT(*) > 100`
`ORDER BY year DESC`
	

SQL a MongoDB: Operaciones de agregación

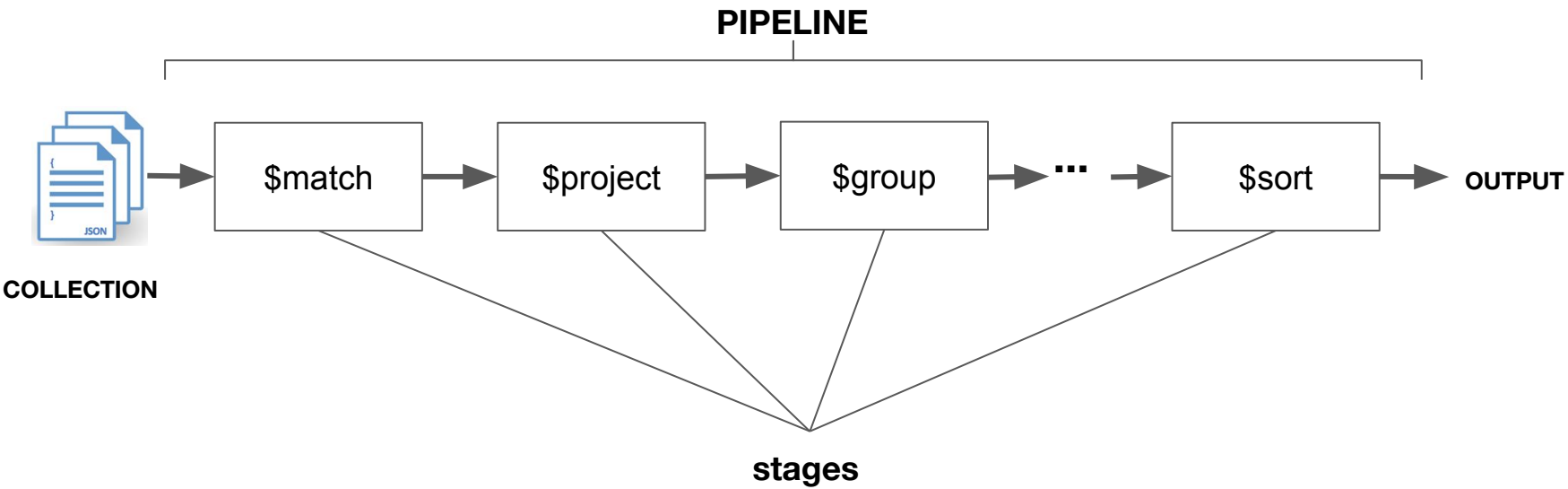
MySQL	MongoDB

`SELECT COUNT(*), MAX(imdb_rating)`
`FROM movies`
`WHERE year = 2019`
	Aggregation Pipeline

`SELECT year, COUNT(*)`
`FROM movies`
`WHERE year >= 2000`
`GROUP BY year`
`HAVING COUNT(*) > 100`
`ORDER BY year DESC`
	Aggregation Pipeline

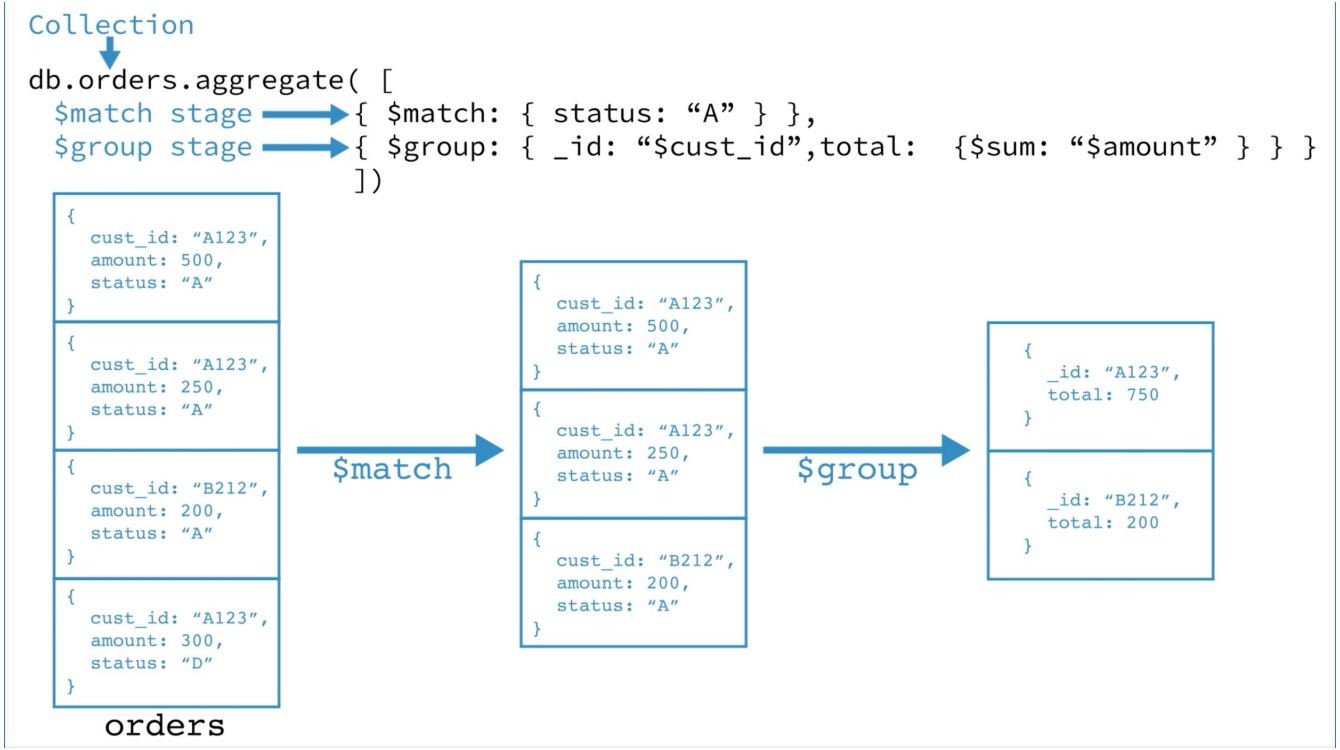
¿Qué es el Pipeline de agregación?

Pipeline de agregación es una composición de N etapas, donde cada etapa transforma los documentos de entrada en resultados agregados.



https://docs.mongodb.com/manual/reference/operator/aggregation-pipeline/

¿Qué es el Pipeline de agregación?



https://docs.mongodb.com/manual/core/aggregation-pipeline/

Pipeline de agregación

- Sintaxis general:

```
db.<collection>.aggregate( [{stage_1}, {stage_2}, ... , {stage_N}], {options})
```

Pipeline de agregación: $match

- **$match** filtra los documentos. Sintaxis de $match:

```
db.<collection>.aggregate( [ { $match: {<query>} } ] )
```

- Ejemplo,

```
> db.movies.aggregate([
    {
        "$match": { "runtime": { $gte: 30, $lt: 40 } }
    }
])
```

https://docs.mongodb.com/manual/reference/operator/aggregation/match/

Pipeline de agregación: $project

- **$project** proyecta los campos especificados existentes/derivados. Sintaxis:

db.<collection>.aggregate([{ $project: {<query>} }])

- Ejemplo,

```
> db.movies.aggregate([
    {
       $match: { "runtime": { $gte: 120, $lt: 180 } }
    },
    {
       $project: {
          "title": 1, "runtime": 1, "_id": 0,
          "runtime_in_hours": { $divide: ["$runtime", 60] }  // campo derivado
       }
    }
])
```

https://docs.mongodb.com/manual/reference/operator/aggregation/project/

Pipeline de agregación: $project

- Desde MongoDB 4.4 se pueden usar expresiones de agregación en las proyecciones de una operación **find**

```
> db.movies.aggregate([
    {
        $match: { "runtime": { $gte: 120, $lt: 180 } }
    },
    {
        $project: {
            "title": 1, "runtime": 1, "_id": 0,
            "runtime_in_hours": { $divide: ["$runtime", 60] }
        }
    }
])
```

```
> db.movies.find(
    { "runtime": { $gte: 120, $lt: 180 } }
    { "title": 1, "runtime": 1, "_id": 0,
      "runtime_in_hours": { $divide: [ "$runtime", 60 ] }
    }
)
```

https://docs.mongodb.com/manual/meta/aggregation-quick-reference/#aggregation-expressions

Stages similares a los métodos de cursor

- Sintaxis:

```
db.<collection>.aggregate( [ { $limit: <integer> } ] )
```

```
db.<collection>.aggregate( [ { $skip: <integer> } ] )
```

```
db.<collection>.aggregate( [ { $count: <string> } ] )
```

```
db.<collection>.aggregate( [
        {
                $sort: {<field_1>: <order>, ... ,<field_N>: <order>}
        }
] )
```

Stages similares a los métodos de cursor

- Ejemplo,

```
> db.movies.aggregate([
    {
        $match: { "runtime": { $gte: 120, $lt: 180 } }
    },
    {
        $project: {
            "title": 1, "year": 1, "runtime": 1, "_id": 0
        }
    },
    { $sort: { "year": 1 } },
    { $limit: 10 }
])
```

```
{ "title" : "Cabiria", "year" : 1914, "runtime" : 148 }
{ "title" : "The Birth of a Nation", "year" : 1915, "runtime" : 165 }
...
```

```
> db.movies.aggregate([
    {
        $match: { "runtime": { $gt: 180 } }
    },
    {
        $count: "long_movies"
    }
])
```

```
{
    "long_movies" : 512
}
```

Pipeline de agregación: $group

- **$group** agrupa los documentos por alguna expresión específica. Sintaxis:

```
db.<collection>.aggregate( [
    {
        $group:
        {
            "_id": <expression>,
            <field1>: { <accumulator1>: <expr1>},

            <fieldN>: { <accumulatorN>: <exprN>},
        }
    }
] )
```

Acumulador

$avg
$max
$min
$sum
$stdDevPop
$first
$last
$addToSet

Pipeline de agregación: $group

- Ejemplo,

```
> db.movies.aggregate([
    {
        $match: { "year": { $gte: 2000 } }
    },
    {
        $group: {
            "_id": "$year",
            "avg_runtime": { $avg: "$runtime" }
         }
    },
    { $sort: { "_id": 1 } }
])

{ "_id" : 2000, "avg_runtime" : 101.662749706228 }
{ "_id" : 2001, "avg_runtime" : 101.814938684504 }
{ "_id" : 2002,  "avg_runtime" : 101.939618644068 }
...
```

Pipeline de agregación: $group

- Ejemplos con “_id” = null

```
> db.movies.aggregate([
    {
        $group: {
            "_id": null,
            "numbers_of_films": { "$sum": 1 }
        }
    }
])
```

{ "_id" : null, "numbers_of_films" : 45993.0 }

```
> db.movies.aggregate([
    {
        $group: {
            "_id": null,
            "numbers_of_films": { "$sum": 1 }
        }
    },
    {
        $project: { "_id": 0, "numbers_of_films": 1 }
    }
])
```

{ "numbers_of_films" : 45993.0 }

Pipeline de agregación: $group

Más ejemplos de $group

- ZIP Code Data
- User Preferences Data

Pipeline de agregación: Acumuladores en $project

- Las acumuladores en $project operan sobre un array en el doc. actual

```
db.<collection>.aggregate( [
    {
        $project:
        {
            <field1>: { <accumulator1>: <expr1>},

            <fieldN>: { <accumulatorN>: <exprN>},
        }
    }
] )
```

Acumulador

$avg
$max
$min
$sum
$stdDevPop
$stdDevSam

Pipeline de agregación: Acumuladores en $project

- Ejemplo,

```
> db.movies.aggregate([
    {
        $match: { "languages": { $size: 2 } }
    },
    {
        $project: {
            "title": 1, "languages": 1, "_id": 0,
            "min_language": { $min: "$languages" }  // alphabetical order
        }
    }
])
```

{ "title" : "The Student of Prague", "languages" : ["German","English"], "min_language" : "English" }
...

Pipeline de agregación: $lookup

- **$lookup** realiza un left outer join a una colección en la misma db y permite una única condición de Join. Sintaxis:

```
db.<collection>.aggregate( [
    {
        $lookup:
        {
            from: <collection to join>,
            localField: <field from the input documents>,
            foreignField: <field from the documents of the "from" collection>,
            as: <output array field>
        }
    }
])
```

Pipeline de agregación: $lookup

- Ejemplo,

```
db.movies.aggregate([
    {
        $lookup: {
            from: "comments",
            localField: "_id",
            foreignField: "movie_id",
            as: "movie_comments"
        }
    },
    {
        $match: {
            "movie_comments": { $size: 2 }
        }
    }
])
```

```
{
    "_id" : ObjectId("573a1390f29313caabcd41b1"),
    "title" : "The Bewitched Inn",
    "year" : 1897,
    "movie_comments" : [
        {
            "_id" : ObjectId("5a9427648b0beebeb69579cf"),
            "name" : "Greg Powell",
            "movie_id" : ObjectId("573a1390f29313caabcd41b1"),
            "text" : "Tenetur dolorum molestiae ea. Eligendi…",
            "date" : ISODate("1987-02-10T00:29:36.000Z")
        },
        {
            "_id" : ObjectId("5a9427648b0beebeb69579d0"),
            "name" : "Talisa Maegyr",
            "movie_id" : ObjectId("573a1390f29313caabcd41b1"),
            "text" : "Rem itaque ad sit rem voluptatibus. Ad fugiat...",
            "date" : ISODate("1998-08-22T11:45:03.000Z")
        }
    ]
}
...
```

Pipeline de agregación: $lookup

- Ejemplo,

```
> db.orders.insert([
   { "_id": 1, "item": "almonds", "price": 12, "quantity": 2 },
   { "_id": 2, "item": "pecans", "price": 20, "quantity": 1 },
   { "_id": 3  }
])

> db.inventory.insert([
   { "_id": 1, "sku": "almonds", description: "product 1", "instock" : 120 },
   { "_id" : 2, "sku" : "bread", description: "product 2", "instock" : 80 },
   { "_id" : 3, "sku" : "cashews", description: "product 3", "instock" : 60 },
   { "_id" : 4, "sku" : "pecans", description: "product 4", "instock" : 70 },
   { "_id" : 5, "sku": null, description: "Incomplete" },
   { "_id" : 6 }
])
```

```
> db.orders.aggregate( [
   {
       $lookup:
       {
          from: "inventory",
          localField: "item",
          foreignField: "sku",
          as: "inventory_docs"
       }
    }
] )
```

Pipeline de agregación: $lookup

- *Avanzado:* Desde Mongo 3.6 **$lookup** acepta una sintaxis más expresiva que permite, entre otras cosas, definir más de una condición de Join.

```
db.<collection>.aggregate( [
   {
      $lookup: {
         from: <collection to join>,
         let: { <var_1>: <expression>, …, <var_n>: <expression> },
         pipeline: [ <pipeline to execute on the collection to join> ],
         as: <output array field>
      }
   }
])
```

Pipeline de agregación: $lookup

- Ejemplo,

```
> db.orders.insert([
  { "_id" : 1, "item" : "almonds", "price" : 12, "ordered" : 2 },
  { "_id" : 2, "item" : "pecans", "price" : 20, "ordered" : 1 },
  { "_id" : 3, "item" : "cookies", "price" : 10, "ordered" : 60 }
])

> db.warehouses.insert([
  { "_id" : 1, "stock_item" : "almonds", warehouse: "A", "instock" : 120 },
  { "_id" : 2, "stock_item" : "pecans", warehouse: "A", "instock" : 80 },
  { "_id" : 3, "stock_item" : "almonds", warehouse: "B", "instock" : 60 },
  { "_id" : 4, "stock_item" : "cookies", warehouse: "B", "instock" : 40 },
  { "_id" : 5, "stock_item" : "cookies", warehouse: "A", "instock" : 80 }
])
```

Pipeline de agregación: $lookup

```
> db.orders.aggregate([
   {
      $lookup:
         {
           from: "warehouses",
           let: { order_item: "$item", order_qty: "$ordered" },
           pipeline: [
              { $match:
                 { $expr:
                    { $and:
                       [
                         { $eq: [ "$stock_item",  "$$order_item" ] },
                         { $gte: [ "$instock", "$$order_qty" ] }
                       ]
                    }
                 }
              },
              { $project: { stock_item: 0, _id: 0 } }
           ],
           as: "stockdata"
         }
    }
])
```

Pipeline de agregación: db.createView()

- Crea una vista a partir de aplicar un pipeline de agregación a una colección/vista (*source*). Las vistas son read-only y se computan on-demand durante cada operación de lectura.

db.createView(<view>, <source>, <pipeline> [, <options>])

Parámetros:

- *view*: El nombre de la vista a crear.
- *source:* El nombre de la colección o vista desde la cual se creará la vista.
- *pipeline:* El pipeline de agregación que se aplicará sobre *source* para generar la vista.
- *options:* Parámetro opcional. Opciones adicionales del método.

Pipeline de agregación: db.createView()

- Ejemplo,

```
> db.createView(
    "movies90s",
    "movies",
    [
        {
            $match: { "year": { $gte: 1990, $lt: 2000 } }
        },
        {
            $project: { "title": 1, "year": 1 }
        }
    ]
)

> db.movies90s.find()
```

```
{
    "_id" : ObjectId("573a1396f29313caabce3d17"),
    "title" : "Larks on a String",
    "year" : 1990
}
{
    "_id" : ObjectId("573a1397f29313caabce799b"),
    "title" : "Me and the Kid",
    "year" : 1993
}
{
    "_id" : ObjectId("573a1398f29313caabce9dbb"),
    "title" : "Halfaouine: Boy of the Terraces",
    "year" : 1990
}
...
```

Más Stages

MongoDB provee más stages (ver la documentación oficial para mayor información)

$addFields	$bucket	$bucketAuto	$collStats
$count	$currentOp	$facet	$geoNear
$graphLookup	$indexStats	$out	$redact
$sample	$sortByCount	$unwind	$merge
$unionWith			

https://docs.mongodb.com/manual/reference/operator/aggregation-pipeline/

Referencias

- SQL to Aggregation Mapping Chart.

  https://docs.mongodb.com/manual/reference/sql-aggregation-comparison/
- Aggregation Pipeline. https://docs.mongodb.com/manual/core/aggregation-pipeline/
- MongoDB Aggregation Example. https://examples.javacodegeeks.com/software-development/mongodb/mongodb-aggregation-example/
- MongoDB Aggregation Framework Stages and Pipelining. https://severalnines.com/resources/whitepapers/mongodb-aggregation-framework-stages-and-pipelining