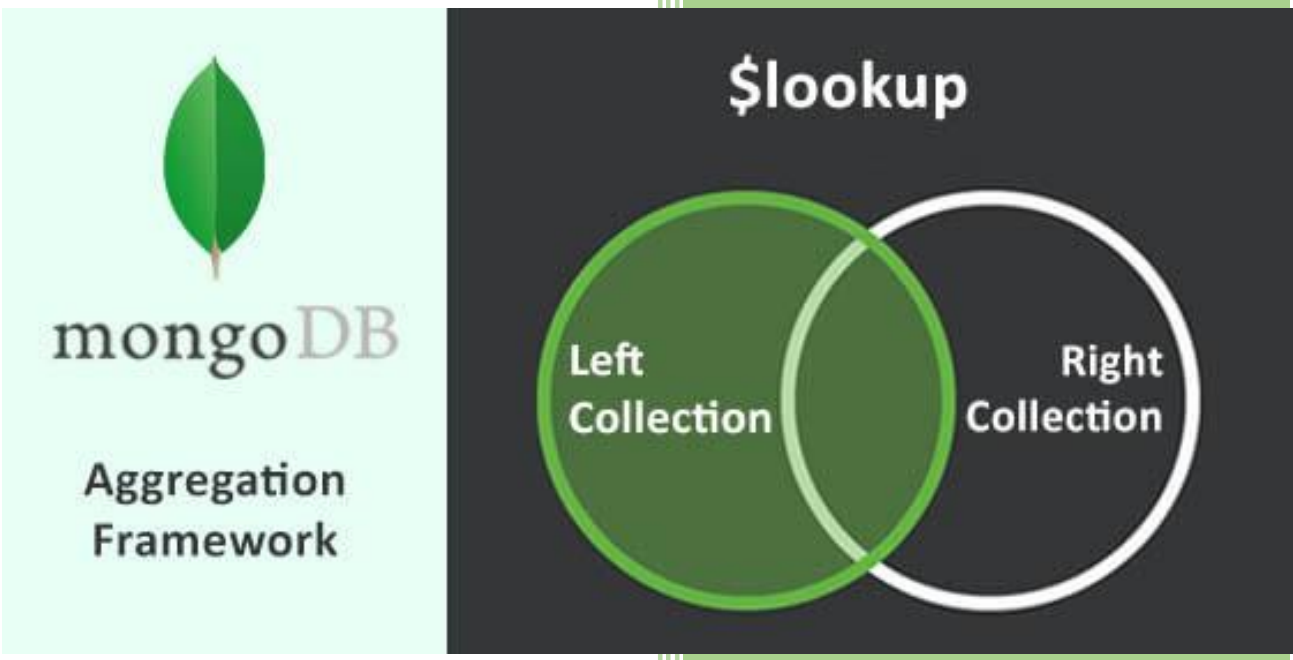
Proyecto Segundo Trimestre



Miguel Ángel Pulido Olmo

Índice:

1. ¿Qué es Aggregation en MongoDB?

Las operaciones de agregaciones procesan registros de datos y devuelven resultados calculados. Las operaciones de agregación agrupan valores de varios documentos y pueden realizar una variedad de operaciones en los datos agrupados para devolver un solo resultado.

2. Operadores que podemos usar en etapas.

1. $match

Filtra los documentos para pasar solo los documentos que coinciden con las condiciones especificadas a la siguiente etapa de canalización.

El $match escenario tiene la siguiente forma de prototipo:

```
{ $match: { <query> } }
```

2. $group

Agrupa los documentos de entrada por la _id expresión especificada y para cada agrupación distinta, genera un documento. El _id campo de cada documento de salida contiene el grupo único por valor. Los documentos de salida también pueden contener campos calculados que contienen los valores de alguna expresión de acumulador.

El $group escenario tiene la siguiente forma de prototipo:

```
g{
  $group:
    {
      _id: <expression>, // Group By Expression
      <field1>: { <accumulator1> : <expression1> },
      ...
    }
 }
```

3. $project

Pasa los documentos con los campos solicitados a la siguiente etapa del proceso. Los campos especificados pueden ser campos existentes de los documentos de entrada o campos recién calculados.

El $project escenario tiene la siguiente forma de prototipo:

```
{ $project: { <specification(s)> } }
```

4. $lookup

Realiza una unión externa izquierda a una colección no fragmentada en la *misma* base de datos para filtrar documentos de la colección "unida" para su procesamiento. A cada documento de entrada, la `$lookup` etapa agrega un nuevo campo de matriz cuyos elementos son los documentos coincidentes de la colección "unida".

Tiene esta sintaxis:

```
{
   $lookup:
     {
       from: <collection to join>,
       localField: <field from the input documents>,
       foreignField: <field from the documents of the "from" collection>,
       as: <output array field>
     }
}
```

5. $unwind

Deconstruye un campo de matriz a partir de los documentos de entrada para generar un documento para *cada* elemento.

Sintaxis:

```
{ $unwind: <field path> }
```

6. $addFields

Agrega nuevos campos a los documentos. `$addFields` genera documentos que contienen todos los campos existentes de los documentos de entrada y los campos recién agregados.

Tiene la siguiente forma:

```
{ $addFields: { <newField>: <expression>, ... } }
```

3. Operadores para usar en cada etapa

1. $sum

Calcula y devuelve la suma colectiva de valores numéricos.

Tiene la siguiente sintaxis:

```
{ $sum: <expression> }
```

2. $multiply

Multiplica números y devuelve el resultado.

Tiene la siguiente sintaxis:

```
{ $multiply: [ <expression1>, <expression2>, ... ] }
```

3. $max

Devuelve el valor máximo. Compara tanto el valor como el tipo, utilizando el orden de comparación BSON especificado para valores de diferentes tipos.

Tiene la siguiente sintaxis:

```
{ $max: <expression> }
```

4. $min

Devuelve el valor mínimo. compara tanto el valor como el tipo, utilizando el orden de comparación BSON especificado para valores de diferentes tipos.

Tiene la siguiente sintaxis:

```
{ $min: <expression> }
```

5. $subtract

Resta dos números para devolver la diferencia, o dos fechas para devolver la diferencia en milisegundos, o una fecha y un número en milisegundos para devolver la fecha resultante.

Tiene la siguiente sintaxis:

```
{ $subtract: [ <expression1>, <expression2> ] }
```

6. $add

Suma números o suma números y una fecha. Si uno de los argumentos es una fecha, `$add` trata los otros argumentos como milisegundos para agregar a la fecha.

La `$add` expresión tiene la siguiente sintaxis:

```
{ $add: [ <expression1>, <expression2>, ... ] }
```