Aggregation en MongoDB

Miguel Ángel Pulido Olmo

1º DAM

Índice:

1. ¿Para qué sirve?

Las operaciones de agregación procesan varios documentos y devuelven resultados calculados. Puede utilizar operaciones de agregación para:

- Agrupe los valores de varios documentos juntos.
- Realice operaciones en los datos agrupados para devolver un único resultado.
- Analice los cambios de datos a lo largo del tiempo.

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. $sort

Ordena todos los documentos de entrada y los devuelve a la canalización en orden.

Tiene la siguiente forma de prototipo:

```
{ $sort: { <field1>: <sort order>, <field2>: <sort order> ... } }
```

Toma un documento que especifica el (los) campo (s) por ordenar y el orden de clasificación respectivo. <sort order>puede tener uno de los siguientes valores:

1: Orden ascendente.

-1: Orden descendente.

5. $limit

Limita el número de documentos se pasan a la siguiente etapa en la tubería .

Tiene la siguiente forma de prototipo:

```
{ $limit: <positive 64-bit integer> }
```

Toma un número entero positivo que especifica el número máximo de documentos que se pueden transmitir.

6. $round

Redondea un número a un entero o a un lugar decimal especificado.

Tiene la siguiente sintaxis:

```
{ $round : [ <number>, <place> ] }
```

<place> es opcional.

Para que round funcione, tiene que estar dentro de un Project.

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. $divide

Divide un número por otro y devuelve el resultado.

Tiene la siguiente sintaxis:

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

4. $avg

Devuelve el valor medio de los valores numéricos.

Tiene la siguiente sintaxis:

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

5. $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> }
```

6. $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> }
```

7. $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> ] }
```

8. $expr

Permite el uso de expresiones de agregación dentro del lenguaje de consulta.

Tiene la siguiente sintaxis:

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

$expr los hemos utilizado para hacer búsquedas en fechas

9. $year

Devuelve la parte del año de una fecha.

Tiene la siguiente sintaxis de expresión de operador:

```
{ $year: <dateExpression> }
```

La siguiente agregación utiliza los `$year` operadores de fecha y otros para desglosar el `date` campo:

```
db.sales.aggregate(
   [
     {
       $project:
         {
           year: { $year: "$date" },
           month: { $month: "$date" },
           day: { $dayOfMonth: "$date" },
           hour: { $hour: "$date" },
           minutes: { $minute: "$date" },
           seconds: { $second: "$date" },
           milliseconds: { $millisecond: "$date" },
           dayOfYear: { $dayOfYear: "$date" },
           dayOfWeek: { $dayOfWeek: "$date" },
           week: { $week: "$date" }
         }
     }
   ]
)
```