ntic master
revolucionamos la comunicación

UNIVERSIDAD COMPLUTENSE DE MADRID

Índices y Agregaciones

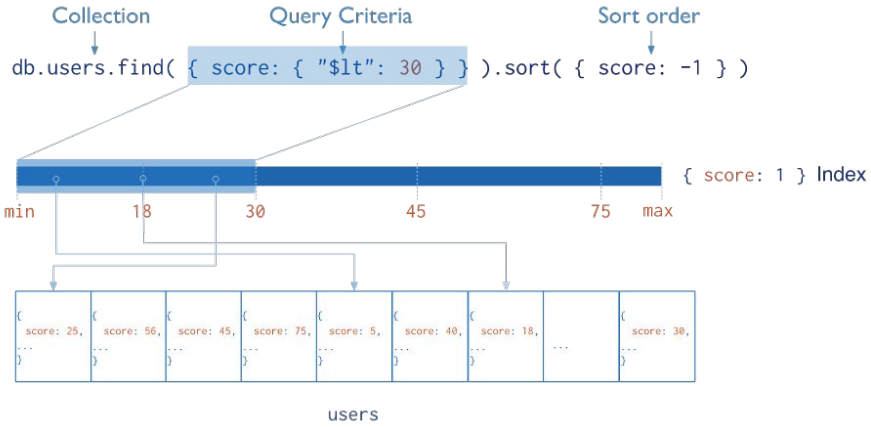
Alvaro Bravo

Índice

1. Indices

Sirven para soportar eficazmente las consultas.

Almacenan una pequeña parte del conjunto de datos de la colección en una forma fácil de recorrer (ordenada)



https://docs.mongodb.com/manual/indexes/

Índices más usados:

- **Simple**

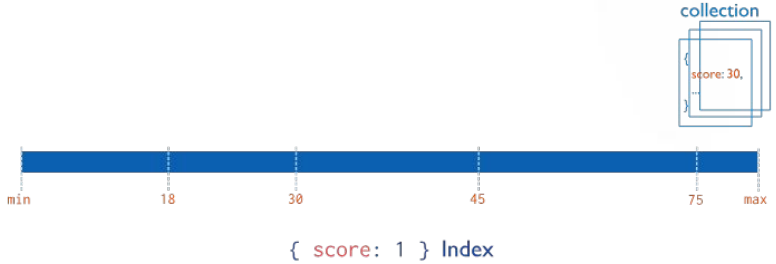
  ```
  db.collection.createIndex( { score: 1 } )
  ```

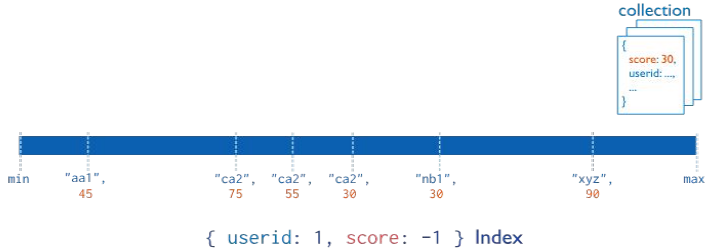
- **Compuesto**

  ```
  db.products.createIndex( { "item": 1, "stock": 1 } )
  ```

- **Multiclave**

  ```
  db.products.createIndex( { "item": 1, "stock": 1 } )
  ```





Otros:

- **Text, Wilcard, 2dsphere, 2d Indexes, geoHaystack, hashed, ...**

Propiedades:

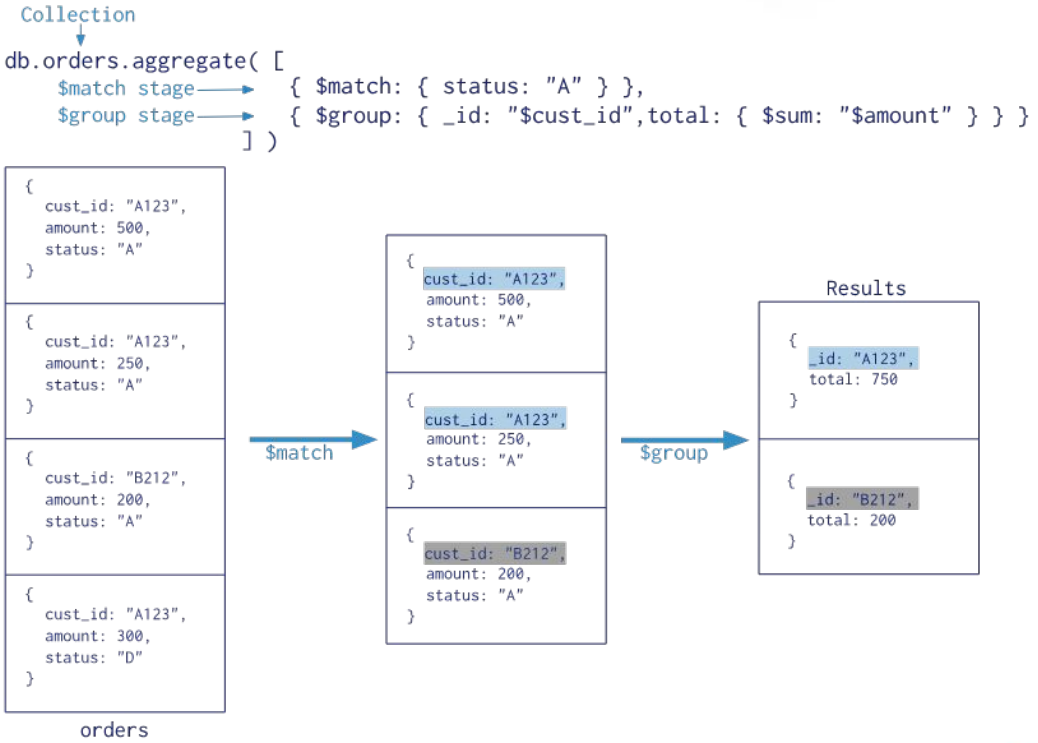
- TTL's
- Unico
- Parcial(filtro)
- Sensitivos (o no).
- Hidden . No es visible por query planner y no puede ser usado para soportar query's. Se utiliza para saber el impacto de borrar el índice.

Estrategias:

- Crea índices que soportan las querys.
- Un índice admite una consulta cuando el índice contiene todos los campos analizados por la consulta. La creación de índices que admiten consultas da como resultado un rendimiento de consulta enormemente mayor.
- Usa índices para ordenar resultados
- Asegurar que se dispone de espacio en RAM para los índices
- Que el índice es selectivo (pocos valores repetidos)

2. Agregaciones

- Framework para agregación de datos basados en **pipelines** de **etapas**
- Cada **etapa** transforma los documentos a medida que pasan a través del **pipeline**.



Etapas

- **$project**. Se utiliza para seleccionar algunos campos específicos de una colección.
- **$match**. Operación de filtrado. Puede reducir la cantidad de documentos que se entregan como entrada en la siguiente etapa.
- **$group**. Es la propia agregación.
- **$sort**. Ordenar documentos.
- **$skip**. Para saltar documentos.
- **$limit**. Limita el número de documentos de salida.
- **$set**. Para añadir nuevos campos.
- **$lookup**. Para realizar left outer join con colecciones de la misma database.
- **$replaceWith**. Reemplaza documento original por otro embebido.
- **$unwind**. Para transformar arrays en nuevos documentos
- **$out**. Para guardar los datos transformados en una colección

SQL Terms, Functions, and Concepts	MongoDB Aggregation Operators
WHERE	`$match`
GROUP BY	`$group`
HAVING	`$match`
SELECT	`$project`
ORDER BY	`$sort`
LIMIT	`$limit`
SUM()	`$sum`
COUNT()	`$sum` `$sortByCount`
join	`$lookup`

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

Operadores

- Aritméticos. $add, $multiply, $subtract, $divide
- Array. $arrayElemAt, $arrayToObject, $concatArrays, $filter, $indexOfArray, $objectToArray, $reduce, $in, $size, $zip
- Booleanos. $and,$not,$or
- Comparacion. $cmp, $eq, $gt, $gte, $lt, $lte, $ne
- Condicionales. $switch, $cond, $ifNull
- Fechas. $dateToString, $dateToParts, $dateFromParts, $dateFromStrings, $month, $year
- Cadenas. $concat, $ltrim, $split, $toLower, $strLenCP
- Tipos. $toString, $toDate
- Acumuladores. $avg, $first, $min, $max, $push, $sum,$mergeObjects

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

```
{
  cust_id: "abc123",
  ord_date: ISODate("2012-11-02T17:04:11.102Z"),
  status: 'A',
  price: 50,
  items: [ { sku: "xxx", qty: 25, price: 1 },
           { sku: "yyy", qty: 25, price: 1 } ]
}
```

SQL Example	MongoDB Example	Description
`SELECT COUNT(*) AS count FROM orders`	`db.orders.aggregate( [ { $group: { _id: null, count: { $sum: 1 } } } ] )`	Count all records from `orders`
`SELECT SUM(price) AS total FROM orders`	`db.orders.aggregate( [ { $group: { _id: null, total: { $sum: "$price" } } } ] )`	Sum the `price` field from `orders`
`SELECT cust_id, SUM(price) AS total FROM orders GROUP BY cust_id`	`db.orders.aggregate( [ { $group: { _id: "$cust_id", total: { $sum: "$price" } } } ] )`	For each unique `cust_id`, sum the `price` field.
`SELECT cust_id, SUM(price) AS total FROM orders GROUP BY cust_id ORDER BY total`	`db.orders.aggregate( [ { $group: { _id: "$cust_id", total: { $sum: "$price" } } }, { $sort: { total: 1 } } ] )`	For each unique `cust_id`, sum the `price` field, results sorted by sum.

```
{
  cust_id: "abc123",
  ord_date: ISODate("2012-11-02T17:04:11.102Z"),
  status: 'A',
  price: 50,
  items: [ { sku: "xxx", qty: 25, price: 1 },
           { sku: "yyy", qty: 25, price: 1 } ]
}
```

```
SELECT cust_id,
       ord_date,
       SUM(price) AS total
FROM orders
GROUP BY cust_id,
         ord_date
```

```
db.orders.aggregate( [
   {
     $group: {
        _id: {
           cust_id: "$cust_id",
           ord_date: {
               month: { $month: "$ord_date" },
               day: { $dayOfMonth: "$ord_date" },
               year: { $year: "$ord_date"}
           }
        },
        total: { $sum: "$price" }
     }
   }
] )
```

For each unique `cust_id`, `ord_date` grouping, sum the `price` field. Excludes the time portion of the date.

```
SELECT cust_id,
       count(*)
FROM orders
GROUP BY cust_id
HAVING count(*) > 1
```

```
db.orders.aggregate( [
   {
     $group: {
        _id: "$cust_id",
        count: { $sum: 1 }
     }
   },
   { $match: { count: { $gt: 1 } } }
] )
```

For `cust_id` with multiple records, return the `cust_id` and the corresponding record count.

```
SELECT cust_id,
       ord_date,
       SUM(price) AS total
FROM orders
GROUP BY cust_id,
         ord_date
HAVING total > 250
```

```
db.orders.aggregate( [
   {
     $group: {
        _id: {
           cust_id: "$cust_id",
           ord_date: {
              month: { $month: "$ord_date" },
              day: { $dayOfMonth: "$ord_date" },
              year: { $year: "$ord_date"}
           }
        },
        total: { $sum: "$price" }
     }
   },
   { $match: { total: { $gt: 250 } } }
] )
```

For each unique `cust_id`, `ord_date` grouping, sum the `price` field and return only where the sum is greater than 250. Excludes the time portion of the date.

```
SELECT cust_id,
       SUM(price) as total
FROM orders
WHERE status = 'A'
GROUP BY cust_id
```

```
db.orders.aggregate( [
   { $match: { status: 'A' } },
   {
     $group: {
        _id: "$cust_id",
        total: { $sum: "$price" }
     }
   }
] )
```

For each unique `cust_id` with status A, sum the `price` field.

ntic master
revolucionamos la comunicación

UNIVERSIDAD COMPLUTENSE DE MADRID