MongoDB. Map-reduce & aggregation framework

Развернем ВМ mongo 4.4.9

GCP

```
gcloud beta compute --project=celtic-house-266612 instances create mongo --
zone=us-central1-a --machine-type=e2-medium --subnet=default --network-
tier=PREMIUM --maintenance-policy=MIGRATE --service-account=933982307116-
compute@developer.gserviceaccount.com --
scopes=https://www.googleapis.com/auth/cloud-platform --image=ubuntu-2004-
focal-v20210223 --image-project=ubuntu-os-cloud --boot-disk-size=10GB --
boot-disk-type=pd-ssd --boot-disk-device-name=mongo --no-shielded-secure-
boot --shielded-vtpm --shielded-integrity-monitoring --reservation-
affinity=any

gcloud compute ssh mongo
```

Установим монго

```
wget -qO - https://www.mongodb.org/static/pgp/server-4.4.asc | sudo apt-key
add - && echo "deb [ arch=amd64,arm64 ] https://repo.mongodb.org/apt/ubuntu
focal/mongodb-org/4.4 multiverse" | sudo tee
/etc/apt/sources.list.d/mongodb-org-4.4.list && sudo apt-get update && sudo
apt-get install -y mongodb-org

sudo mkdir /home/mongo &&  sudo mkdir /home/mongo/db1 && sudo chmod 777
/home/mongo/db1

mongod --dbpath /home/mongo/db1 --port 27001 --fork --logpath
/home/mongo/db1/db1.log --pidfilepath /home/mongo/db1/db1.pid

mongo --port 27001
```

Запуск MongoDB локально в Docker

```
docker run --name mongodb-otus -d mongo:4.4

docker exec -it mongodb-otus mongo
```

DEMO

```
db.orders.insertMany([
    {custID:"10001",amount:500,status:"A"},
    {custID:"10001",amount:250,status:"A"},
    {custID:"10002",amount:200,status:"A"},
    {custID:"10001",amount: 300, status:"D"}])
```

Проверим:

```
db.orders.find()
```

Сумма `amount` в разрезе `custID`:

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

Аналог SQL:

```
SELECT custID, sum(amount)
FROM orders
WHERE status = 'A'
GROUP BY custID
```

См. слайды.

Добавим коллекцию `inventory` (с массивом и составным объектом):

```
db.inventory.insert([
{ _id: 2, item: { name: "cd", code: "123" }, qty: 20, tags: [ "B" ] },
{ _id: 3, item: { name: "ij", code: "456" }, qty: 25, tags: [ "A", "B" ] },
{ _id: 4, item: { name: "xy", code: "456" }, qty: 30, tags: [ "B", "A" ] },
{ _id: 5, item: { name: "mn", code: "000" }, qty: 20, tags: [ [ "A", "B" ],
"C" ] },
{ _id: 6, item: { name: "mn", code: "222" }, qty: 2, tags: [ [ "A", "B" ],
"C" ] },
{ _id: 1, item: { name: "ab", code: "123" }, qty: 15, tags: [ "A", "B", "C"
] }])
```

Поиск по массиву:

> Сколько строк будет в результате?

```
db.inventory.aggregate([
   { $match: { tags: ["B"] } },
   { $group: {
      _id: "$item.name",
      total: { $sum: "$qty" } } }
])
```

Поиск по элементам массива:

```
db.inventory.aggregate([
   { $match: { tags: "B" } },
   { $group: {
      _id: "$item.name",
      total: { $sum: "$qty" } } }
])
```

Поиск в подмассиве:

```
db.inventory.aggregate([
   { $match: { "tags": { $elemMatch: { $elemMatch:{ $in :["B"] } } } } },
   { $group: {
      _id: "$item.name",
      total: { $sum: "$qty" } } }
])
```

Если не нужна агрегация:

```
db.inventory.aggregate([
   { $match: { "tags": { $elemMatch: { $elemMatch:{ $in :["B"] } } } } }
])
```

Логическое ИЛИ:

```
db.inventory.aggregate( [
  { $match: { $or: [
        { "tags": { $elemMatch: { $elemMatch:{ $in :["B"] } } } },
        { "tags": "B" }]
    }}
])
```

Про запросы к массивам: https://docs.mongodb.com/manual/tutorial/query-arrays/

Посчитать мин, макс, среднее по всей коллекции:

```
db.inventory.aggregate([
    { $group: {
                _id: "$item.name",
                total: { $sum: "$qty"},
                avg: { $avg: "$qty"},
                min: { $min: "$qty"},
                max: { $max: "$qty"}
              }
    }
])
```

Лайфхак. Группируем по константе %) -- или _id : null.

`avg`, `min`, `max` по всей коллекции без группировки:

```
db.inventory.aggregate([
    { $group: {
                _id: 1,
                total: { $sum: "$qty"},
                avg: { $avg: "$qty"},
                min: { $min: "$qty"},
                max: { $max: "$qty"}
              }
    }
])
```

Выбрать все поддокументы:

```
db.inventory.find({},{"item" : 1 })
```

`$addFields` - добавляет поля к результирующему документу
https://www.mongodb.com/docs/manual/reference/operator/aggregation/addFields/#mongodb-pipeline-pipe.-addFields

Выбрать информацию из поддокументов:

```
db.inventory.aggregate([
  {$addFields : {
    itemName : "$item.name",
    itemCode : "$item.code"}}])
```

Отображаем только `itemName` и `itemCode`:

```
db.inventory.aggregate([
  {$addFields :
    { itemName : "$item.name",
      itemCode : "$item.code"}},
    {$project : {
      itemName : 1,
      itemCode : 1} }])
```

Без вывода `_id`, только `name` и `code`

```
db.inventory.aggregate([
  {$addFields :
    { itemName : "$item.name",
      itemCode : "$item.code"}},
    {$project : {
      itemName : 1,
      itemCode : 1,
      _id: 0 } }])
```

JOIN ($lookup)

Дополнительный материал
https://docs.mongodb.com/manual/reference/operator/aggregation/lookup/#pipe._S_lookup

Создадим коллекцию `orders2`:

```
db.orders2.insert([
   { "_id" : 1, "item" : "almonds", "price" : 12, "quantity" : 2 },
   { "_id" : 2, "item" : "pecans", "price" : 20, "quantity" : 1 },
   { "_id" : 3  }
])
```

и коллекцию `inventory2`:

```
db.inventory2.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 }
])
```

Left join

```
db.orders2.find()
db.inventory2.find()
```

`orders2.item == inventory2.sku`

```
db.orders2.aggregate([
   {
     $lookup:
       {
         from: "inventory2",
         localField: "item",
         foreignField: "sku",
         as: "inventory_docs"
       }
  }
])
```

Обратите внимание на то, что `null == null`.

Можно сохранить результат в другую коллекцию `$out`:

```
db.orders2.aggregate([
   {
     $lookup:
       {
         from: "inventory2",
         localField: "item",
         foreignField: "sku",
         as: "inventory_docs"
       }
  },
    { $out : "results" }
])

db.results.find()
```

---

single purpose aggragation

См. слайды.

distinct

```
db.inventory3.insertMany([
{ "_id": 1, "dept": "A", "item": { "sku": "111", "color": "red" }, "sizes":
[ "S", "M" ] },
{ "_id": 2, "dept": "A", "item": { "sku": "111", "color": "blue" },
"sizes": [ "M", "L" ] },
{ "_id": 3, "dept": "B", "item": { "sku": "222", "color": "blue" },
"sizes": "S" },
{ "_id": 4, "dept": "A", "item": { "sku": "333", "color": "black" },
"sizes": [ "S" ] }])

db.inventory3.distinct( "dept" )

db.inventory3.distinct( "item.sku", { dept: "A" } )
```

См. слайды.

Map Reduce

Хотим узнать количество интересов у группы людей

```
db.users.insertMany([
    {name : "John2",age : 23,interests : ["football", "IT", "cooking",
"Postgres"]},
    {name : "Daw2",age : 23,interests : ["football", "IT", "MongoDB"]},
    {name : "Jane2",age : 33,interests : ["cooking",  "tvshow"]}]);
```

map

```
function map(){
    for(var i in this.interests) {
        emit(this.interests[i], 1);
    } //this.interests.forEach(function(interest){ emit(interest, 1); });
}
```

Пример вектора, получающегося в map():

```
MongoDB: [ 1, 1, 1, 1, 1 ]
```

reduce

```
function reduce(key, values) {
    var sum = 0;
    for(var i in values) {
        sum += values[i];
```

```
    }
    return sum; //return Array.sum(values)
}
```

mapReduce

```
db.users.mapReduce(map, reduce,{out:"interests"})
db.interests.find()
```

Хотим узнать среднее количество интересов у людей разных возрастов. Записываем документ, вместо единичного значения.

```
db.users.find()
```

```
function map(){
    emit(this.age, { interests_count: this.interests.length, count: 1 });
}

function reduce(key, values) {
    var sum = 0;
    var count = 0;
    for(var i in values){
        count += values[i].count;
        sum += values[i].interests_count;
    }
    return {interests_count: sum, count: count};
}

function finalize(key, reducedValue) {
  return reducedValue.interests_count / reducedValue.count;
}

db.users.mapReduce(map, reduce, {finalize: finalize,
out:"interests_by_age"})

db.interests_by_age.find()
```

Если создали инстанс Mongo в GCP, то можно прибраться:

```
gcloud compute instances delete mongo
```