PRÁCTICA MONGODB

Hacer una prueba de rendimiento con y sin índices, justificando el resultado de:

1. **Operaciones de búsqueda.**
2. **Operaciones de agregación.**
3. **Operaciones de actualización.**
4. **Operaciones de borrado.**

Descripción de qué es un índice, cuándo utilizarlo, etc.

Selección de un dataset en el repositorio de GitHub y carga en MongoDB.

Imf > db.practicafinal.insertMany([{"_id":0,"name":"aimee Zank","scores":[{"score":1.463179736705023,"type":"exam"},{"score":11.78273309957772,"type":"quiz"},{"score":35.8740349954354,"type":"homework"}]},

... {"_id":1,"name":"Aurelia Menendez","scores":[{"score":60.06045071030959,"type":"exam"},{"score":52.79790691903873,"type":"quiz"},{"score":71.76133439165544,"type":"homework"}]},

... {"_id":2,"name":"Corliss Zuk","scores":[{"score":67.03077096065002,"type":"exam"},{"score":6.301851677835235,"type":"quiz"},{"score":66.28344683278382,"type":"homework"}]},

... {"_id":3,"name":"Bao Ziglar","scores":[{"score":71.64343899778332,"type":"exam"},{"score":24.80221293650313,"type":"quiz"},{"score":42.26147058804812,"type":"homework"}]},

... {"_id":4,"name":"Zachary Langlais","scores":[{"score":78.68385091304332,"type":"exam"},{"score":90.2963101368042,"type":"quiz"},{"score":34.41620148042529,"type":"homework"}]},

... {"_id":5,"name":"Wilburn Spiess","scores":[{"score":44.87186330181261,"type":"exam"},{"score":25.72395114668016,"type":"quiz"},{"score":63.42288310628662,"type":"homework"}]},

... {"_id":6,"name":"Jenette Flanders","scores":[{"score":37.32285459166097,"type":"exam"},{"score":28.32634976913737,"type":"quiz"},{"score":81.57115318686338,"type":"homework"}]},

... {"_id":7,"name":"Salena Olmos","scores":[{"score":90.37826509157176,"type":"exam"},{"score":42.48780666956811,"type":"quiz"},{"score":96.52986171633331,"type":"homework"}]},

... {"_id":8,"name":"Daphne Zheng","scores":[{"score":22.13583712862635,"type":"exam"},{"score":14.63969941335069,"type":"quiz"},{"score":75.94123677556644,"type":"homework"}]},

... {"_id":9,"name":"Sanda Ryba","scores":[{"score":97.00509953654694,"type":"exam"},{"score":97.80449632538915,"type":"quiz"},{"score":25.27368532432955,"type":"homework"}]},

... {"_id":10,"name":"Denisha Cast","scores":[{"score":45.61876862259409,"type":"exam"},{"score":98.35723209418343,"type":"quiz"},{"score":55.90835657173456,"type":"homework"}]},

... {"_id":11,"name":"Marcus Blohm","scores":[{"score":78.42617835651868,"type":"exam"},{"score":82.58372817930675,"type":"quiz"},{"score":87.49924733328717,"type":"homework"}]},

... {"_id":12,"name":"Quincy Danaher","scores":[{"score":54.29841278520669,"type":"exam"},{"score":85.61270164694737,"type":"quiz"},{"score":80.40732356118075,"type":"homework"}]},

... {"_id":13,"name":"Jessika Dagenais","scores":[{"score":90.47179954427436,"type":"exam"},{"score":90.30014024468489,"type":"quiz"},{"score":95.17753772405909,"type":"homework"}]},

... {"_id":14,"name":"Alix Sherrill","scores":[{"score":25.15924151998215,"type":"exam"},{"score":68.64484047692098,"type":"quiz"},{"score":24.68462152686763,"type":"homework"}]},

... {"_id":15,"name":"Tambra Mercure","scores":[{"score":69.1565022533158,"type":"exam"},{"score":3.311794422000724,"type":"quiz"},{"score":45.03178973642521,"type":"homework"}]},

... {"_id":16,"name":"Dodie Staller","scores":[{"score":7.772386442858281,"type":"exam"},{"score":31.84300235104542,"type":"quiz"},{"score":80.52136407989194,"type":"homework"}]},

... {"_id":17,"name":"Fletcher Mcconnell","scores":[{"score":39.41011069729274,"type":"exam"},{"score":81.13270307809924,"type":"quiz"},{"score":97.70116640402922,"type":"homework"}]},

... {"_id":18,"name":"Verdell Sowinski","scores":[{"score":62.12870233109035,"type":"exam"},{"score":84.74586220889356,"type":"quiz"},{"score":81.58947824932574,"type":"homework"}]},

... {"_id":19,"name":"Gisela Levin","scores":[{"score":44.51211101958831,"type":"exam"},{"score":0.6578497966368002,"type":"quiz"},{"score":93.36341655949683,"type":"homework"}]},

... {"_id":20,"name":"Tressa Schwing","scores":[{"score":42.17439799514388,"type":"exam"},{"score":71.99314840599558,"type":"quiz"},{"score":81.23972632069464,"type":"homework"}]},

... {"_id":21,"name":"Rosana Vales","scores":[{"score":46.2289476258328,"type":"exam"},{"score":98.34164225207036,"type":"quiz"},{"score":36.18769746805938,"type":"homework"}]},

... {"_id":22,"name":"Margart Vitello","scores":[{"score":75.04996547553947,"type":"exam"},{"score":10.23046475899236,"type":"quiz"},{"score":96.72520512117761,"type":"homework"}]},

... {"_id":23,"name":"Tamika Schildgen","scores":[{"score":45.65432764125526,"type":"exam"},{"score":64.32927049658846,"type":"quiz"},{"score":83.53933351660562,"type":"homework"}]},

... {"_id":24,"name":"Jesusa Rickenbacker","scores":[{"score":86.0319702155683,"type":"exam"},{"score":1.967495200433389,"type":"quiz"},{"score":61.10861071547914,"type":"homework"}]}

Así hasta _id=199

1. ANÁLISIS DE LA ESTRUCTURA DEL DATASET SELECCIONADO

MongoDB almacena los documentos en un formato llamado BSON (o JSON binario), siendo cada documento descriptivo, en el que incluye su longitud y los tipos de datos.

La estructura de datos está compuesta por colecciones y documentos, como una sucesión de campos en forma de clave:valor

MongoDB ha almacenado los documentos en colecciones. Cada documento almacenado en una colección, tiene un campo _id único que identifica inmutablemente a dicho documento, actuando como clave primaria (ej.: _id:0, _id: 1 …. _id":199), y a este campo le asignamos el valor (ej.: "name":"Aimee Zank", “name":"Aurelia Menendez"…"name":"Rae Kohout").

El campo también tiene un subconjunto de objetos, como es el caso del campo "scores", que es a su vez un array, formado por una serie de diccionarios.

```
        ]
}
> db.practicafinal.find({name:"Nobuko Linzey"}).pretty()
{
        "_id" : 110,
        "name" : "Nobuko Linzey",
        "scores" : [
                {
                        "score" : 67.40792606687442,
                        "type" : "exam"
                },
                {
                        "score" : 58.58331128403415,
                        "type" : "quiz"
                },
                {
                        "score" : 47.44831568815929,
                        "type" : "homework"
                }
        ]
}
> 
```

2. PRUEBA DE RENDIMIENTO

Primero veremos el rendimiento sin índices:

3.1. **Operaciones de búsqueda.**

SIN INDICE

```
imf> db.practicafinal.find({name:"Nobuko Linzey"},{_id:0})
[
  {
    name: 'Nobuko Linzey',
    scores: [
      { score: 80.5, type: 'exam' },
      { score: 58.58331128403415, type: 'quiz' },
      { score: 47.44831568815929, type: 'homework' }
    ]
  }
]
```

Vemos ahora el rendimiento con **explain("executionStats")**

```
imf> db.practicafinal.find({name:"Nobuko Linzey"},{_id:0}).explain("executionSta
{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'imf.practicafinal',
    indexFilterSet: false,
    parsedQuery: { name: { '$eq': 'Nobuko Linzey' } },
    queryHash: '64908032',
    planCacheKey: '64908032',
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,
    winningPlan: {
      stage: 'PROJECTION_DEFAULT',
      transformBy: { _id: 0 },
      inputStage: {
        stage: 'COLLSCAN',
        filter: { name: { '$eq': 'Nobuko Linzey' } },
        direction: 'forward'
      }
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 1,
    executionTimeMillis: 0,
    totalKeysExamined: 0,
    totalDocsExamined: 200,
    executionStages: {
      stage: 'PROJECTION_DEFAULT',
      nReturned: 1,
      executionTimeMillisEstimate: 0,
      works: 202,
      advanced: 1,
      needTime: 200,
      needYield: 0,
      saveState: 0,
      restoreState: 0,
      isEOF: 1,
      transformBy: { _id: 0 },
      inputStage: {
        stage: 'COLLSCAN',
        filter: { name: { '$eq': 'Nobuko Linzey' } },
        nReturned: 1,
        executionTimeMillisEstimate: 0,
        works: 202,
        advanced: 1,
        needTime: 200,
        needYield: 0,
        saveState: 0,
        restoreState: 0,
        isEOF: 1,
        direction: 'forward',
        docsExamined: 200
      }
    }
  },
  command: {
    find: 'practicafinal',
    filter: { name: 'Nobuko Linzey' },
    projection: { _id: 0 },
    '$db': 'imf'
  },
  serverInfo: {
    host: '13c6561c38af',
    port: 27017,
    version: '6.0.1',
    gitVersion: '32f0f9c88dc44a2c8073a5bd47cf779d4bfdee6b'
  },
  serverParameters: {
    internalQueryFacetBufferSizeBytes: 104857600,
    internalQueryFacetMaxOutputDocSizeBytes: 104857600,
    internalLookupStageIntermediateDocumentMaxSizeBytes: 104857600,
    internalDocumentSourceGroupMaxMemoryBytes: 104857600,
    internalQueryMaxBlockingSortMemoryUsageBytes: 104857600,
    internalQueryProhibitBlockingMergeOnMongoS: 0,
    internalQueryMaxAddToSetBytes: 104857600,
    internalDocumentSourceSetWindowFieldsMaxMemoryBytes: 104857600
  },
  ok: 1
}
```

Lo importante aquí:

nReturned: 1 (devuelve una salida)

TotalKeysExamined: 0 (no usa índice)

totalDocsExamined: 200 (examinó 200 documentos y devolvió 1)

CON INDICE

imf> db.practicafinal.createIndex({ name: 1 })

name_1

 Hemos creado índice en **name**

Vemos rendimiento:


saveState: 0,
restoreState: 0,
isEOF: 1,
transformBy: { _id: 0 },
inputStage: {
stage: 'FETCH',
nReturned: 1,
executionTimeMillisEstimate: 0,
works: 2,
advanced: 1,
needTime: 0,
needYield: 0,
saveState: 0,
restoreState: 0,
isEOF: 1,
docsExamined: 1,
alreadyHasObj: 0,
inputStage: {
stage: 'IXSCAN',
nReturned: 1,
executionTimeMillisEstimate: 0,
works: 2,
advanced: 1,
needTime: 0,
needYield: 0,
saveState: 0,
restoreState: 0,
isEOF: 1,
keyPattern: { name: 1 },
indexName: 'name_1',
isMultiKey: false,
multiKeyPaths: { name: [] },
isUnique: false,
isSparse: false,
isPartial: false,
indexVersion: 2,
direction: 'forward',
indexBounds: { name: ['["Nobuko Linzey", "Nobuko Linzey"]'] },
keysExamined: 1,
seeks: 1,
dupsTested: 0,
dupsDropped: 0
}
}
}
},
command: {
find: 'practicafinal',


},
command: {
find: 'practicafinal',
filter: { name: 'Nobuko Linzey' },
projection: { _id: 0 },
'$db': 'imf'
},
serverInfo: {
host: '13e6561c38af',
port: 27017,
version: '6.0.1',
gitVersion: '32f0f9c88dc44a2c8073a5bd47cf779d4bfdee6b'
},
serverParameters: {
internalQueryFacetBufferSizeBytes: 104857600,
internalQueryFacetMaxOutputDocSizeBytes: 104857600,
internalLookupStageIntermediateDocumentMaxSizeBytes: 104857600,
internalDocumentSourceGroupMaxMemoryBytes: 104857600,
internalQueryMaxBlockingSortMemoryUsageBytes: 104857600,
internalQueryProhibitBlockingMergeOnMongoS: 0,
internalQueryMaxAddToSetBytes: 104857600,
internalDocumentSourceSetWindowFieldsMaxMemoryBytes: 104857600
},
ok: 1
}

Lo importante:

nReturned: 1 (devuelve una salida)

TotalKeysExamined: 1 (1 índice)

totalDocsExamined: 1 (examinó 1 documentos y devolvió 1)

Como podemos ver el rendimiento ha mejorado lo máximo, antes revisó 200 documentos para devolver 1 y ahora solo ha tenido que revisar uno y devolver uno. Hemos maximizado el rendimiento con el índice.

SIN INDICE

- Otra búsqueda diferente

```
imf> db.practicafinal.find({"name":"Cody Strouth"},{"scores":{$elemMatch: {"type":"quiz"}}},{"scores.score":1})
[
  { _id: 69, scores: [ { score: 99.80348240553108, type: 'quiz' } ] },
  { _id: 183, scores: [ { score: 78.61720316992681, type: 'quiz' } ] }
]
```

Calculamos rendimiento:

```
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 2,
    executionTimeMillis: 0,
    totalKeysExamined: 0,
    totalDocsExamined: 200,
    executionStages: {
      stage: 'PROJECTION_DEFAULT',
      nReturned: 2,
      executionTimeMillisEstimate: 0,
      works: 202,
      advanced: 2,
      needTime: 199,
      needYield: 0,
      saveState: 0,
      restoreState: 0,
      isEOF: 1,
      transformBy: { scores: { '$elemMatch': { type: 'quiz' } } },
      inputStage: {
        stage: 'COLLSCAN',
        filter: { name: { '$eq': 'Cody Strouth' } },
        nReturned: 2,
        executionTimeMillisEstimate: 0,
        works: 202,
        advanced: 2,
        needTime: 199,
        needYield: 0,
        saveState: 0,
        restoreState: 0,
        isEOF: 1,
        direction: 'forward',
        docsExamined: 200
      }
    }
```

Lo importante aquí:

nReturned: 2 (devuelve dos salidas)

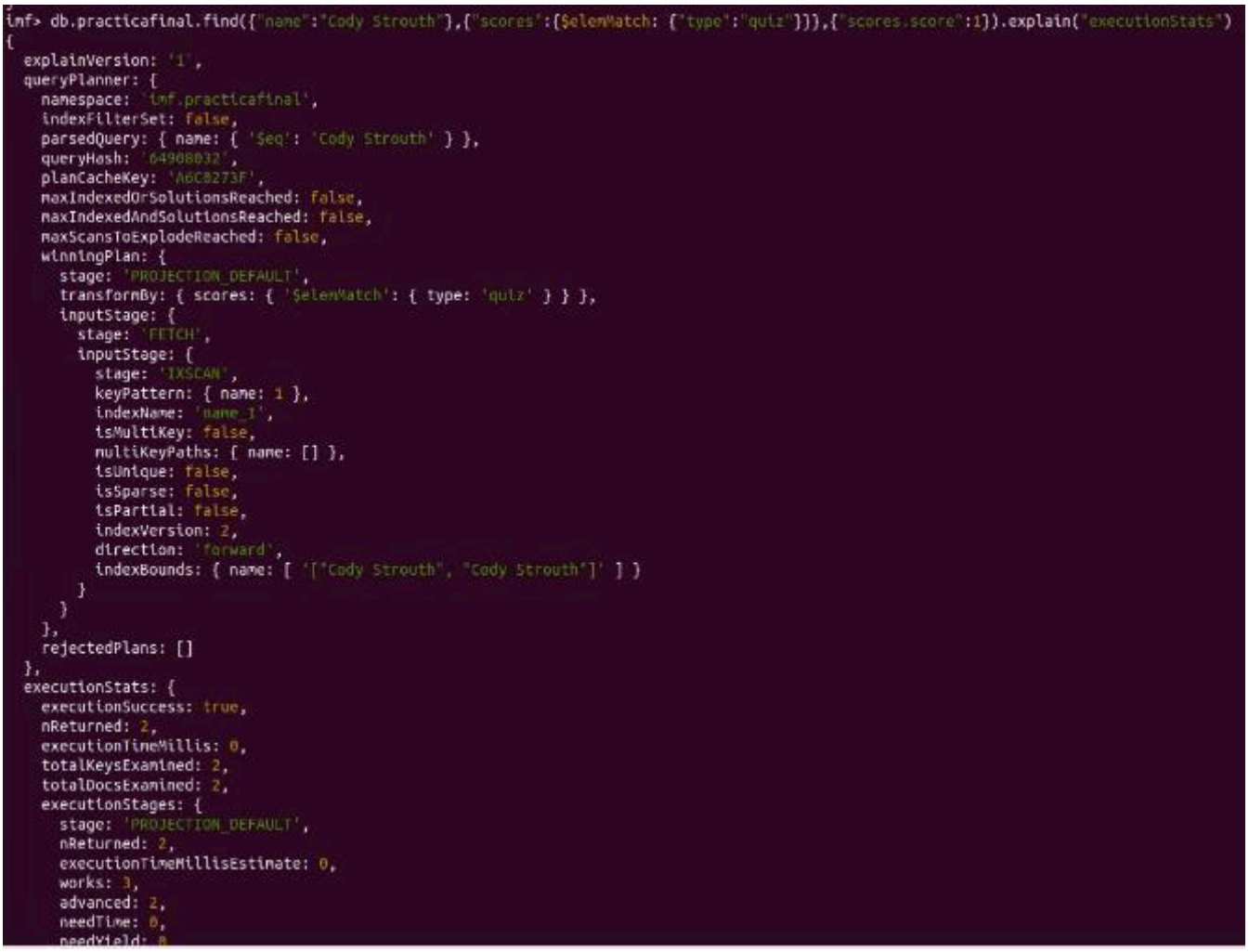
TotalKeysExamined: 0 (no usa índice)

totalDocsExamined: 200 (examinó 200 documentos y devolvió 2)

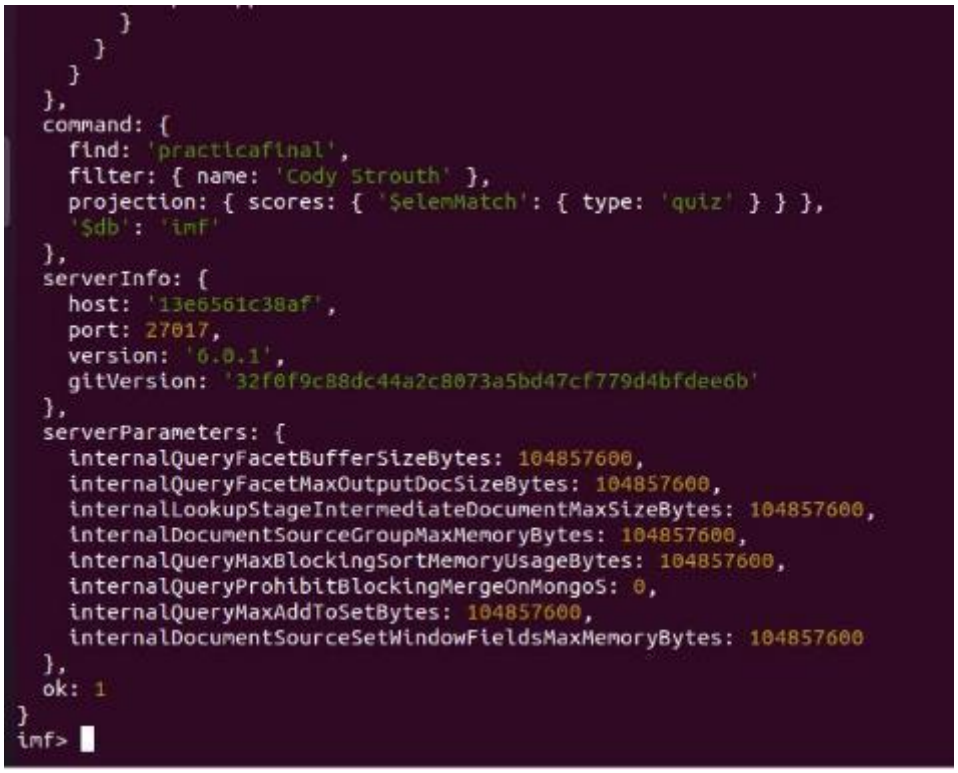
CON INDICE

Seguimos con el name como índice

imf> db.practicafinal.find({"name":"Cody Strouth"},{"scores":{$elemMatch: {"type":"quiz"}}},{"scores.score":1}).explain("executionStats")



```
    transformBy: { scores: { '$elemMatch': { type: 'quiz' } } },
    inputStage: {
      stage: 'FETCH',
      nReturned: 2,
      executionTimeMillisEstimate: 0,
      works: 3,
      advanced: 2,
      needTime: 0,
      needYield: 0,
      saveState: 0,
      restoreState: 0,
      isEOF: 1,
      docsExamined: 2,
      alreadyHasObj: 0,
      inputStage: {
        stage: 'IXSCAN',
        nReturned: 2,
        executionTimeMillisEstimate: 0,
        works: 3,
        advanced: 2,
        needTime: 0,
        needYield: 0,
        saveState: 0,
        restoreState: 0,
        isEOF: 1,
        keyPattern: { name: 1 },
        indexName: 'name_1',
        isMultiKey: false,
        multiKeyPaths: { name: [] },
        isUnique: false,
        isSparse: false,
        isPartial: false,
        indexVersion: 2,
        direction: 'forward',
        indexBounds: { name: [ '["Cody Strouth", "Cody Strouth"]' ] },
        keysExamined: 2,
        seeks: 1,
        dupsTested: 0,
        dupsDropped: 0
      }
    }
  }
},
command: {
  find: 'practicafinal',
  filter: { name: 'Cody Strouth' },
  projection: { scores: { '$elemMatch': { type: 'quiz' } } },
  '$db': 'imf'
},
```



```
      }
    }
  }
},
command: {
  find: 'practicafinal',
  filter: { name: 'Cody Strouth' },
  projection: { scores: { '$elemMatch': { type: 'quiz' } } },
  '$db': 'imf'
},
serverInfo: {
  host: '13e6561c38af',
  port: 27017,
  version: '6.0.1',
  gitVersion: '32f0f9c88dc44a2c8073a5bd47cf779d4bfdee6b'
},
serverParameters: {
  internalQueryFacetBufferSizeBytes: 104857600,
  internalQueryFacetMaxOutputDocSizeBytes: 104857600,
  internalLookupStageIntermediateDocumentMaxSizeBytes: 104857600,
  internalDocumentSourceGroupMaxMemoryBytes: 104857600,
  internalQueryMaxBlockingSortMemoryUsageBytes: 104857600,
  internalQueryProhibitBlockingMergeOnMongoS: 0,
  internalQueryMaxAddToSetBytes: 104857600,
  internalDocumentSourceSetWindowFieldsMaxMemoryBytes: 104857600
},
ok: 1
}
imf>
```

Lo importante aquí:

nReturned: 2 (devuelve dos salidas)

TotalKeysExamined: 2 (dos índices name, uno por cada salida)

totalDocsExamined: 2 (examinó 2 documentos)

> Como podemos ver el rendimiento ha mejorado lo máximo, antes revisó 200 documentos para devolver 1 y ahora solo ha tenido que revisar uno y devolver uno. Hemos maximizado el rendimiento con el índice.

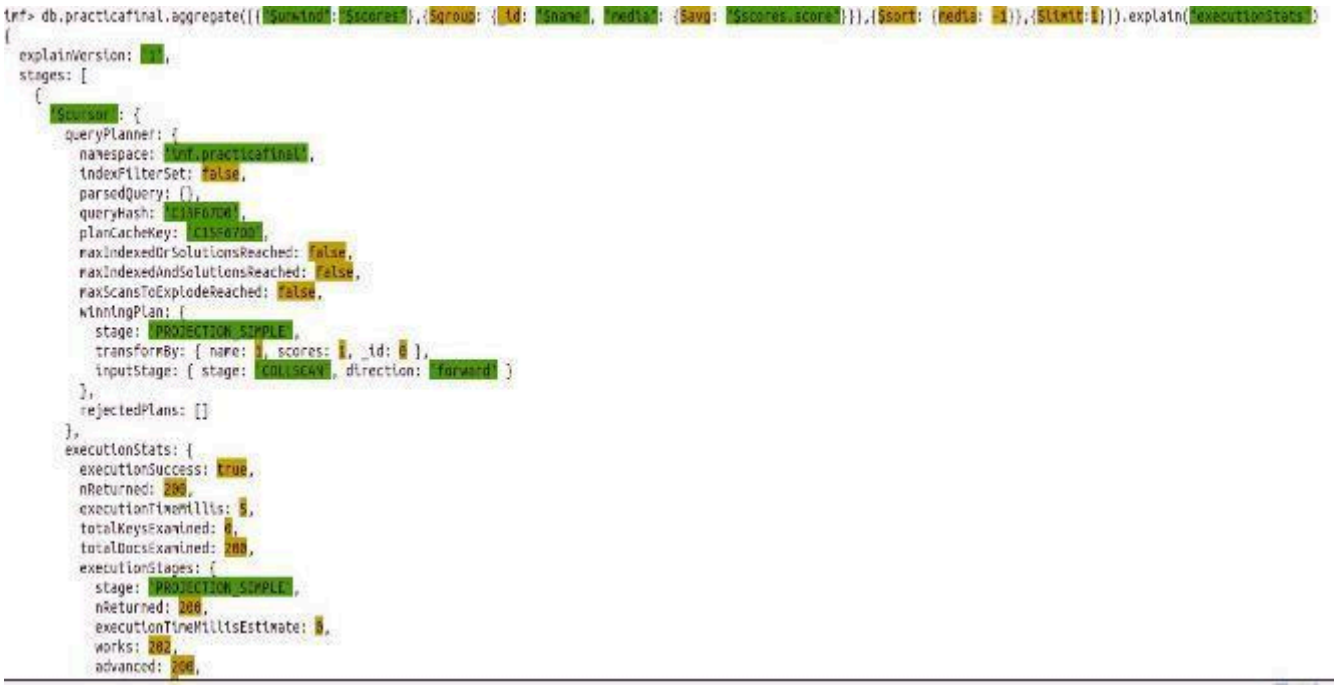
3.2. Operaciones de agregación

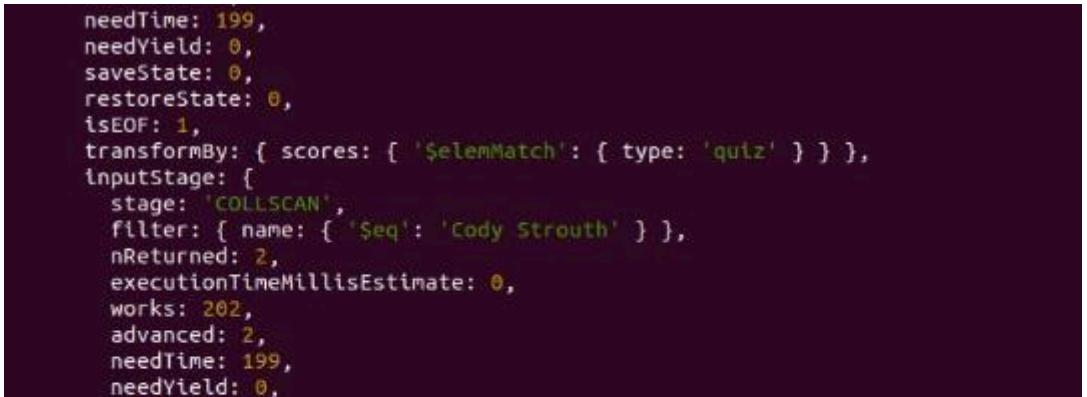
SIN INDICE

imf>db.practicafinal.aggregate([{"$unwind":"$scores"},{$group: {_id: "$name", "media": {$avg: "$scores.score"}}},{$sort: {media: -1}},{$limit:1}])

[{ id: 'Jessika Dagenais', media: 89.89140723964316 }]

Para ver el rendimiento





Lo importante aquí:

nReturned: 200 (devuelve)

TotalKeysExamined: 0 (no usa índice)

totalDocsExamined: 200 (examinó 200 documentos)

CON INDICE

Uso como índice scores

imf> db.inventory.createIndex({ scores: 1 })

scores_1

imf> db.practicafinal.aggregate([{"$unwind":"$scores"},{$group: {_id: "$name", "media": {$avg: "$scores.score"}}},{$sort: {media: -1}},{$limit:1}]).explain("executionStats")


{
'$unwind': { path: '$scores' },
nReturned: Long("600"),
executionTimeMillisEstimate: Long("0")
},
{
'$group': { _id: '$name', media: { '$avg': '$scores.score' } },
maxAccumulatorMemoryUsageBytes: { media: Long("10032") },
totalOutputDataSizeBytes: Long("28180"),
usedDisk: false,
spills: Long("0"),
nReturned: Long("114"),
executionTimeMillisEstimate: Long("0")
},
{
'$sort': { sortKey: { media: -1 }, limit: Long("1") },
totalDataSizeSortedBytesEstimate: Long("0"),
usedDisk: false,
spills: Long("0"),
nReturned: Long("1"),
executionTimeMillisEstimate: Long("0")
}
],
serverInfo: {
host: '13e6561c38af',
port: 27017,
version: '6.0.1',
gitVersion: '32f0f9c88dc44a2c8073a5bd47cf779d4bfdee6b'
},
serverParameters: {
internalQueryFacetBufferSizeBytes: 104857600,
internalQueryFacetMaxOutputDocSizeBytes: 104857600,
internalLookupStageIntermediateDocumentMaxSizeBytes: 104857600,
internalDocumentSourceGroupMaxMemoryBytes: 104857600,
internalQueryMaxBlockingSortMemoryUsageBytes: 104857600,
internalQueryProhibitBlockingMergeOnMongoS: 0,
internalQueryMaxAddToSetBytes: 104857600,
internalDocumentSourceSetWindowFieldsMaxMemoryBytes: 104857600
},
command: {
aggregate: 'practicafinal',
pipeline: [
{ '$unwind': '$scores' },
{
'$group': { _id: '$name', media: { '$avg': '$scores.score' } }
},
{ '$sort': { media: -1 } },
{ '$limit': 1 }

Lo importante aquí:

nReturned: 200 (devuelve)

TotalKeysExamined: 0 (no usa el índice)

totalDocsExamined: 200 (examinó 200 documentos)

> Como podemos ver el rendimiento es igual en aggregate, con o sin índice

Y si sólo uno como índice name se llega al mismo resultado

Y si usamos los dos índices a la vez, no reconoce la función con aggregate:

imf> db.practicafinal.aggregate([{"$unwind":"$scores"},{$group: {_id: "$name", "media": {$avg: "$scores.score"}}},{$sort: {media: -1}},{$limit:1}]).hint({ name: 1, scores: 1 }).explain("executionStats")

TypeError: db.practicafi ... t:1}]).hint is not a function

Como podemos ver el rendimiento es igual en aggregate, con o sin índice

Vemos otro ejemplo de agregación:

SIN INDICE

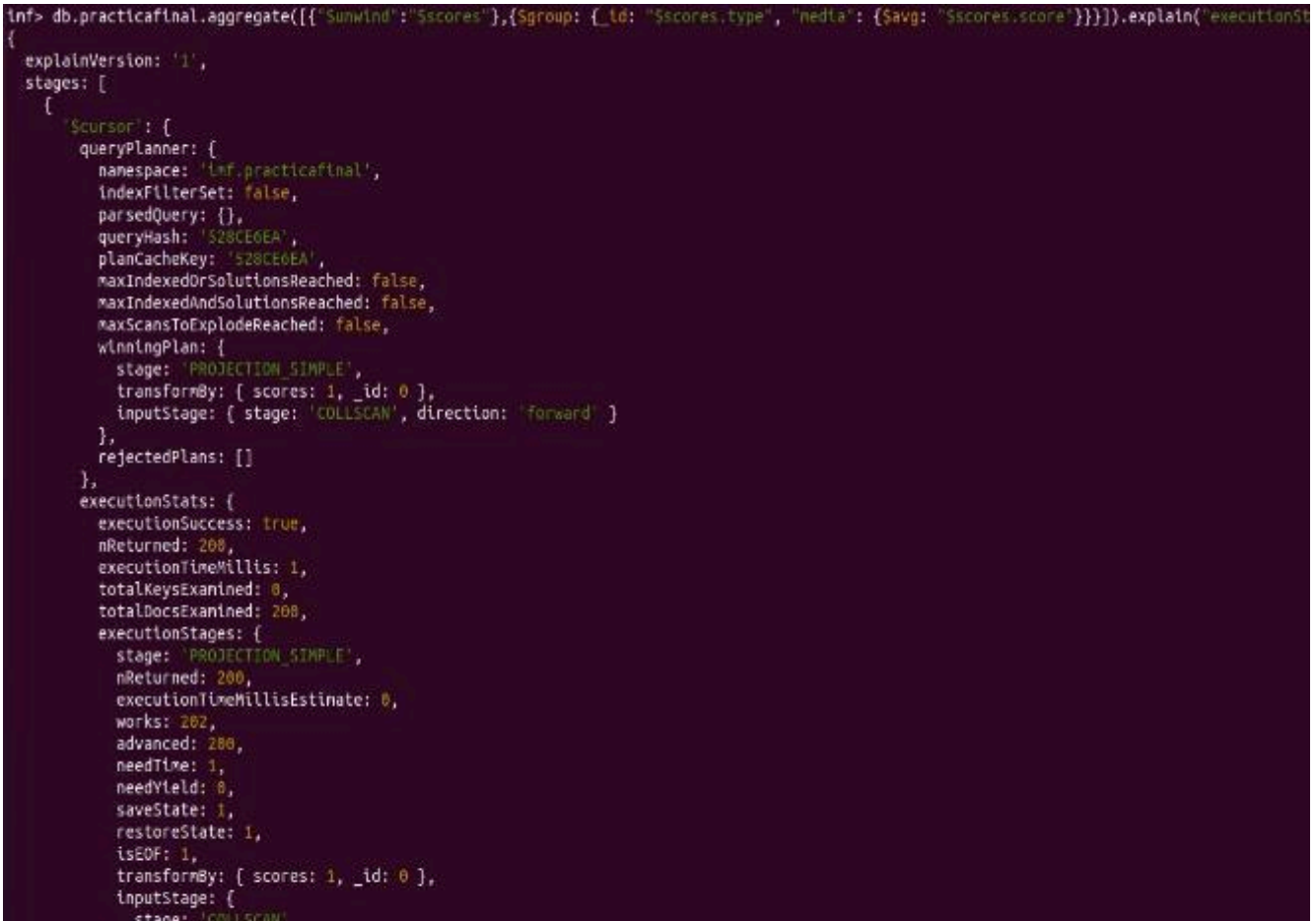
imf> db.practicafinal.aggregate([{"$unwind":"$scores"},{$group:{_id:"$scores.type", "media":{$avg: "$scores.score"}}}])

[

{ _id: 'exam', media: 48.67367075950175 },

{ _id: 'quiz', media: 48.99672319430254 },

{ _id: 'homework', media: 67.81869620661149 }



Lo importante aquí:

nReturned: 200 (devuelve)

TotalKeysExamined: 0 (sin índice)

totalDocsExamined: 200 (examinó 200 documentos)

CON INDICE

imf> db.practicafinal.aggregate([{"$unwind":"$scores"},{$group:{_id:"$scores.type", "media":{$avg: "$scores.score"}}}])

Como podemos ver el rendimiento es igual en aggregate, con o sin índice

3.3.Operaciones de actualización

CON/SIN INDICE

```
imf>db.practicafinal.updateOne({"name":"Nobuko Linzey"},{ $set: { "scores.$[elem].score" : 80.5 } },{ arrayFilters: [ { "elem.type": "exam" } ] })

imf> db.practicafinal.find({"name":"Nobuko Linzey"})

[ {   _id: 110,
   name: 'Nobuko Linzey',
   scores: [
    { score: 21.83, type: 'exam' },
    { score: 58.58331128403415, type: 'quiz' },
    { score: 47.44831568815929, type: 'homework' }
   ]
 }
]

imf> db.practicafinal.updateOne({"name":"Nobuko Linzey"},{ $set: { "scores.$[elem].score" : 21.83 } },{ arrayFilters: [ { "elem.type": "exam" } ] }).explain("executionStats")
```

TypeError: db.practicafi ... }] }).explain is not a function

No se aplica la función explain en update. No podemos ver el rendimiento así.

3. 4.Operaciones de Borrado

CON/SIN INDICE

```
db.practicafinal.deleteOne({'name':'Efrain Claw'})
```

imf> db.practicafinal.deleteOne({'name':'Efrain Claw'})

{ acknowledged: true, deletedCount: 1 }

imf> db.practicafinal.deleteOne({'name':'Efrain Claw'}).explain("executionStats")

TypeError: db.practicafi ... law'}).explain is not a function

No podemos aplicar explain tampoco en funciones de borrado.