

Bases de Datos 2 2021 -TP3

Bases de Datos NoSQL / Práctica con MongoDB

entrega: 31/5

Parte 1: Bases de Datos NoSQL y Relacionales

► Si bien las BBDD NoSQL tienen diferencias fundamentales con los sistemas de BBDD Relacionales o RDBMS, algunos conceptos comunes se pueden relacionar. Responda las siguientes preguntas, considerando MongoDB en particular como Base de Datos NoSQL.

1. ¿Cuáles de los siguientes conceptos de RDBMS existen en MongoDB? En caso de no existir, ¿hay alguna alternativa? ¿Cuál es?

- Base de Datos: Existe en MongoDB.
- Tabla / Relación: Existe en MongoDB. Por convención, una tabla en Mongo, es denominada 'collection' (colección)
- Fila / Tupla: Existe en MongoDB, aunque con el nombre de 'document' (documento)
- Columna: En Mongo, se lo puede encontrar con el nombre de 'field' (campo).

2. MongoDB tiene soporte para transacciones, pero no es igual que el de los RDBMS. ¿Cuál es el alcance de una transacción en MongoDB?

Cuando se habla de transacciones dentro de una base de datos, se hace referencia a la capacidad de poder realizar operaciones entre múltiples tablas, colecciones o registros y que se garantice que si una de estas operaciones falla, el efecto de todas las operaciones sea revertido. Esto garantiza la consistencia en los datos y es la manera en que las bases de datos, mantienen la integridad de los datos cuando se realizan operaciones entre múltiples registros.

En el caso de MongoDB, previo a la versión 4.0, se garantizaba transacciones ACID (*Atomicity, Consistency, Isolation, Durability*) a nivel de documento (más conocido como BASE -Basically Available, Soft state, Eventually consistent). Esto quiere decir que, las operaciones realizadas en los subdocumentos dentro de un documento cumplían con la condición de que si un error ocurría, la base de datos se encargaba de hacer rollback al estado anterior al inicio de la operación.

Sin embargo existen casos de uso en los cuales, es absolutamente necesario contar con transaccionalidad entre documentos. Cuando las aplicaciones se encontraban empleando MongoDB, lo que debían hacer (previo a la versión 4.0) era implementar la lógica transaccional dentro del código, lo que generaba un aumento en la complejidad y en algunos casos, la latencia de la ejecución de las operaciones.

Debido a esto, en su versión 4.0, MongoDB implementa las transacciones ACID entre documentos y hace una interfaz agradable para implementar esta funcionalidad dentro de aplicaciones existentes.

3. Para acelerar las consultas, MongoDB tiene soporte para índices. ¿Qué tipos de índices soporta?

Los índices en MongoDB se generan en forma de Árbol-B o B-Tree. Es decir, que los datos se guardan en forma de árbol, pero manteniendo los nodos balanceados. Esto incrementa la velocidad a la hora de buscar y también a la hora de devolver resultados ya ordenados.

Los tipos de índices que soporta MongoDB, son:

- Índices simples o de un solo campo: Estos índices se aplican a un solo campo de la colección. Para declarar un índice de este tipo, se debe utilizar el siguiente comando:

```
db.users.ensureIndex( { "user_id" : 1 } )
```

Este tipo de índices, facilita el ordenamiento de los documentos.

- Índices compuestos: Permite crear índices sobre varios campos. Para declarar un índice compuesto, se debe ejecutar el siguiente comando:

```
db.users.ensureIndex( { "user_name" : 1, "age":-1 } )
```

El índice que se genera con la instrucción anterior, agrupa los datos primero por el campo user_name y luego por el campo age. Es decir, se obtendría la siguiente salida:

```
"Antonio", 35
"Antonio", 18
"María", 56
"María", 30
"María", 21
"Pedro", 19
"Unai", 34
"Unai", 27
```

Lo interesante de los índices compuestos, es que permite ser utilizados para consultar uno o varios de los campos, sin que sea necesario incluirlos a todos. En el ejemplo anterior, el índice se puede utilizar siempre que se hagan consultas sobre user_name o sobre user_name y age.

- Índices multikey (multillave): Estos índices, son aplicados a campos que tienen información contenida en arreglos. Al usar estos índices, MongoDB crea una entrada para cada uno de los elementos del arreglo. Si se indexa un campo que contiene un valor de matriz, MongoDB crea entradas de índice independientes para cada elemento de la matriz. Estos índices de varias claves, permiten que las consultas seleccionen documentos que contienen matrices, haciendo coincidir el elemento o elementos de las matrices.
- Índices geoespaciales: Para utilizar este tipo de índices, MongoDB proporciona dos índices especiales: Índices 2d, que usan geometría plana al devolver resultados, e índices 2dsphere que usan geometría esférica para devolver resultados.
- Índices de texto: MongoDB proporciona un texttipo de índice, que admite la búsqueda de un string en una colección. Estos índices de texto no almacenan palabras vacías específicas del idioma (por ejemplo, "el", "a", "o") y derivan las palabras en una colección para almacenar solo palabras raíz.

- Índices hash: MongoDB proporciona un tipo de índice hash , que indexa el hash del valor de un campo. Estos índices que tienen una distribución de valores más aleatoria a lo largo de su rango, solo admiten coincidencias de igualdad y no pueden admitir consultas basadas en rangos.

4. ¿Existen claves foráneas en MongoDB?

No, no existen llaves foráneas en MongoDB. Para relacionar dos documentos, existen dos variantes:

- ❑ **Manual References**: En este caso, se guarda el campo `_id` de un documento como referencia en otro documento. En este modelo, la aplicación debe ejecutar una segunda consulta para devolver los datos relacionados.
- ❑ **DBRefs**: Son referencias de un documento a otro, utilizando el valor del campo del primer documento `_id`, nombre de la colección (y opcionalmente el nombre de base de datos). Con esto los DBRefs permiten vincular documentos de varias colecciones para linkearse en una sola colección. Los DBRefs proporcionan, en esencia, una semántica común para la representación de los vínculos entre documentos. Los DBRefs también requieren consultas adicionales para devolver los documentos de referencia. Por otro lado, la mayoría de los drivers ofrecen métodos de utilidad para hacer la query de la DBRef de forma automática.

Parte 2: Primeros pasos con MongoDB

► Descargue la última versión de MongoDB desde el sitio oficial. Ingrese al cliente de línea de comando para realizar los siguientes ejercicios.

5. Cree una nueva base de datos llamada ecommerce, y una colección llamada products. En esa colección inserte un nuevo documento (un producto) con los siguientes atributos:

{name:'Caldera Caldaia Duo', price:140000}

recupere la información del producto usando el comando `db.products.find()` (puede agregar la función `.pretty()` al final de la expresión para ver los datos indentados). Notará que no se encuentran exactamente los atributos que insertó. ¿Cuál es la diferencia?

► Una característica fundamental de MongoDB y otras bases NoSQL es que los documentos no tienen una estructura definida, como puede ser una tabla en un RDBMS. En una misma colección pueden convivir documentos con diferentes atributos, e incluso atributos de múltiples valores y documentos embebidos.

Se inicia el cliente de mongo:

```
>> sudo service mongod start
```

```
>> mongo
```

Para crear la base de datos, se ejecuta el comando:

```
>> use ecommerce
```

Para crear la colección e insertar un documento, se ejecuta el comando:

```
>> db.products.insertOne({name:'Caldera Caldaia Duo', price:140000})
```

Al ejecutar el comando `db.products.find().pretty()`, se obtiene la siguiente salida:

```
bd2@bd2-VirtualBox: ~  
> db.products.find().pretty()  
{  
  "_id" : ObjectId("60a4770b9bf0c31d8536dd4e"),  
  "name" : "Caldera Caldaia Duo",  
  "price" : 140000  
}  
> □
```

Además de los atributos insertados, MongoDB agrega un identificador al documento ingresado. El identificador se reconoce con el nombre de “_id”, junto con un hash asociado.

6. Agregue los siguientes documentos a la colección de productos:

```
{name:'Caldera Orbis 230', price:77000, tags: ['gas', 'digital']}  
{name:'Caldera Ariston Clas', price:127000, tags: ['gas envasado', 'termostato']}  
{name:'Caldera Caldaia S30', price:133000}  
{name:'Caldera Mural Orbis 225cto', price:100000, tags: ['gas', 'digital',  
'termostato']}
```

Y busque los productos:

- de \$100.000 o menos
- que tengan la etiqueta (tag) “digital”
- que no tengan etiquetas (es decir, que el atributo esté ausente)
- que incluyan la palabra ‘Orbis’ en su nombre
- con la palabra ‘Orbis’ en su nombre y menores de \$100.000

vuelva a realizar la última consulta pero proyecte sólo el nombre del producto en los resultados, omitiendo incluso el atributo _id de la proyección.

Para crear los documentos indicados, se ejecutaron los siguientes comandos:

```
>> db.products.insertOne({name:'Caldera Orbis 230', price:77000, tags:['gas',  
'digital']})  
  
>> db.products.insertOne({name:'Caldera Ariston Clas', price:127000, tags:['gas  
envasado', 'termostato']})  
  
>> db.products.insertOne({name:'Caldera Caldaia S30', price:133000})  
  
>> db.products.insertOne({name:'Caldera Mural Orbis 225cto', price:100000,  
tags:['gas','digital','termostato']})
```

```
bd2@bd2-VirtualBox: ~
> db.products.insertOne({name:'Caldera Orbis 230',price:77000,tags:['gas','digital']})
{
  "acknowledged" : true,
  "insertedId" : ObjectId("60a5caa5a0f13ab185eb0de3")
}
> db.products.insertOne({name:'Caldera Ariston Clas',price:127000,tags:['gas envasado','termostato']})
{
  "acknowledged" : true,
  "insertedId" : ObjectId("60a5cb33a0f13ab185eb0de4")
}
> db.products.insertOne({name:'Caldera Caldaia S30',price:133000})
{
  "acknowledged" : true,
  "insertedId" : ObjectId("60a5ccb8a0f13ab185eb0de5")
}
> db.products.insertOne({name:'Caldera Mural Orbis 225cto',price:100000,tags:['gas','digital','termostato']})
{
  "acknowledged" : true,
  "insertedId" : ObjectId("60a5ccb2a0f13ab185eb0de6")
}
```

Consultas sobre la colección de 'products':

- ❑ Buscar aquellos productos de \$100.000 o menos:

>> `db.products.find({price:{$lte:100000}})`

```
bd2@bd2-VirtualBox: ~
> db.products.find({price:{$lte:100000}})
{ "_id" : ObjectId("60a5caa5a0f13ab185eb0de3"), "name" : "Caldera Orbis 230", "price" : 77000, "tags" : [ "gas", "digital" ] }
{ "_id" : ObjectId("60a5ccb2a0f13ab185eb0de6"), "name" : "Caldera Mural Orbis 225cto", "price" : 100000, "tags" : [ "gas", "digital", "termostato" ] }
```

- ❑ Buscar aquellos productos que tengan la etiqueta (tag) "digital"

>> `db.products.find({"tags":"digital"})`

```
bd2@bd2-VirtualBox: ~
> db.products.find({"tags":"digital"})
{ "_id" : ObjectId("60a5caa5a0f13ab185eb0de3"), "name" : "Caldera Orbis 230", "price" : 77000, "tags" : [ "gas", "digital" ] }
{ "_id" : ObjectId("60a5ccb2a0f13ab185eb0de6"), "name" : "Caldera Mural Orbis 225cto", "price" : 100000, "tags" : [ "gas", "digital", "termostato" ] }
```

- ❑ Buscar aquellos productos que no tengan etiquetas (es decir, que el atributo esté ausente)

>> `db.products.find({tags:{$exists:false}})`

```
bd2@bd2-VirtualBox: ~
> db.products.find({tags:{$exists:false}})
{ "_id" : ObjectId("60a4770b9bf0c31d8536dd4e"), "name" : "Caldera Caldaia Duo", "price" : 140000 }
{ "_id" : ObjectId("60a5ccb8a0f13ab185eb0de5"), "name" : "Caldera Caldaia S30", "price" : 133000 }
```

- ❑ Buscar aquellos productos que incluyan la palabra 'Orbis' en su nombre

```
>> db.products.find({ "name": /. *Orbis. */})
```

```
bd2@bd2-VirtualBox: ~  
> db.products.find({"name": /. *Orbis. */})  
{ "_id" : ObjectId("60a5caa5a0f13ab185eb0de3"), "name" : "Caldera Orbis 230", "price"  
  : 77000, "tags" : [ "gas", "digital" ] }  
{ "_id" : ObjectId("60a5ccb2a0f13ab185eb0de6"), "name" : "Caldera Mural Orbis 225cto  
  ", "price" : 100000, "tags" : [ "gas", "digital", "termostato" ] }  
>
```

- ❑ Buscar aquellos productos con la palabra 'Orbis' en su nombre y menores de \$100.000

```
>> db.products.find({price:{$lt:100000}}, {"name": /. *Orbis. */})
```

```
bd2@bd2-VirtualBox: ~  
> db.products.find({price:{$lt:100000}}, {"name": /. *Orbis. */}).pretty()  
{ "_id" : ObjectId("60a5caa5a0f13ab185eb0de3"), "name" : /. *Orbis. */ }  
>  
> db.products.find("60a5caa5a0f13ab185eb0de3").pretty()  
{  
  " _id" : ObjectId("60a5caa5a0f13ab185eb0de3"),  
  "name" : "Caldera Orbis 230",  
  "price" : 77000,  
  "tags" : [  
    "gas",  
    "digital"  
  ]  
}
```

- ❑ Buscar aquellos productos con la palabra 'Orbis' en su nombre y menores de \$100.000, proyectando sólo el nombre del producto en los resultados, omitiendo incluso el atributo _id de la proyección.

```
>> db.products.find({"name": /. *Orbis. */ , "price":{$lt:100000}}, {name:1, _id:0})
```

```
bd2@bd2-VirtualBox: ~  
> db.products.find({"name": /. *Orbis. */ , "price":{$lt:100000}}, {name:1, _id:0})  
{ "name" : "Caldera Orbis 230" }  
>
```

► En MongoDB hay diferentes maneras de realizar actualizaciones, de acuerdo a las necesidades del esquema flexible de documentos.

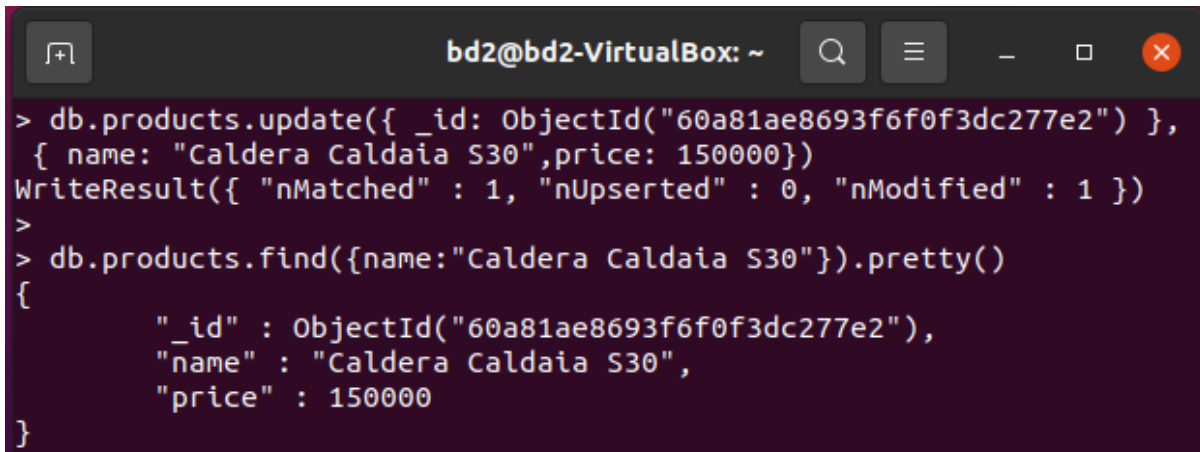
- ❑ 7. Actualice la "Caldera Caldaia S30" cambiándole el precio a \$150.000.

Se actualiza el documento:

```
db.products.update({ _id: ObjectId("60a81ae8693f6f0f3dc277e2") }, { name: "Caldera Caldaia S30", price: 150000})
```

Se chequea que el mismo se haya actualizado:

```
db.products.find({name:"Caldera Caldaia S30"}).pretty()
```

A terminal window titled 'bd2@bd2-VirtualBox: ~' with search, menu, and window control icons. It shows the execution of two MongoDB commands. The first command updates a document with _id '60a81ae8693f6f0f3dc277e2' to have name 'Caldera Caldaia S30' and price 150000. The second command finds the document by name and pretty-prints it, showing the updated fields.

```
> db.products.update({ _id: ObjectId("60a81ae8693f6f0f3dc277e2") }, { name: "Caldera Caldaia S30", price: 150000})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>
> db.products.find({name:"Caldera Caldaia S30"}).pretty()
{
  "_id" : ObjectId("60a81ae8693f6f0f3dc277e2"),
  "name" : "Caldera Caldaia S30",
  "price" : 150000
}
```

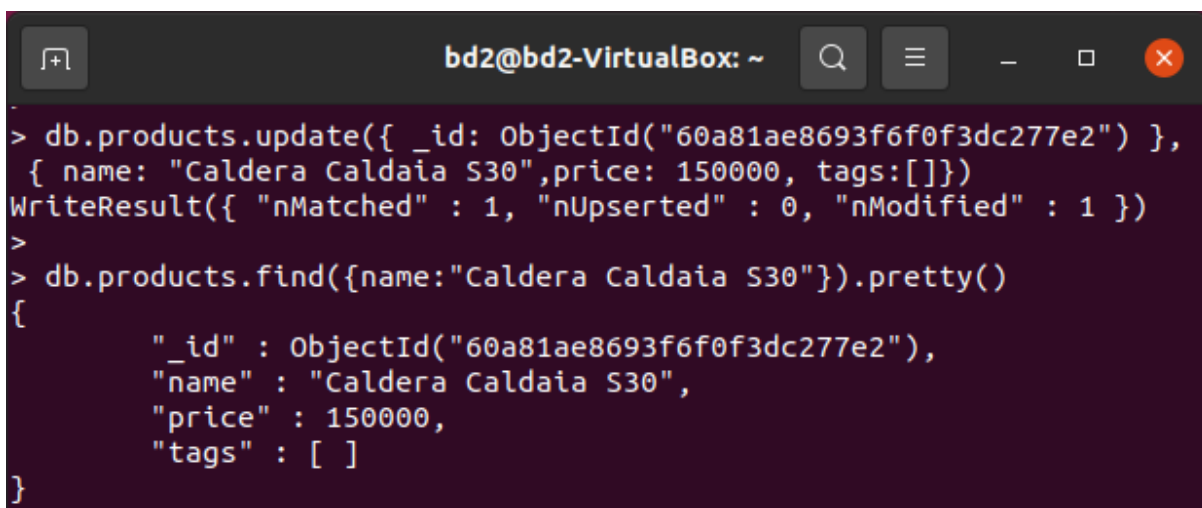
- 8. Cree el array de etiquetas (tags) para la "Caldera Caldaia S30".

Se actualiza el documento:

```
db.products.update({ _id: ObjectId("60a81ae8693f6f0f3dc277e2") }, { name: "Caldera Caldaia S30", price: 150000, tags:[]})
```

Se chequea que el mismo se haya actualizado:

```
db.products.find({name:"Caldera Caldaia S30"}).pretty()
```

A terminal window titled 'bd2@bd2-VirtualBox: ~' with search, menu, and window control icons. It shows the execution of two MongoDB commands. The first command updates a document with _id '60a81ae8693f6f0f3dc277e2' to have name 'Caldera Caldaia S30', price 150000, and an empty tags array. The second command finds the document by name and pretty-prints it, showing the updated fields including the tags array.

```
> db.products.update({ _id: ObjectId("60a81ae8693f6f0f3dc277e2") }, { name: "Caldera Caldaia S30", price: 150000, tags:[]})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>
> db.products.find({name:"Caldera Caldaia S30"}).pretty()
{
  "_id" : ObjectId("60a81ae8693f6f0f3dc277e2"),
  "name" : "Caldera Caldaia S30",
  "price" : 150000,
  "tags" : [ ]
}
```

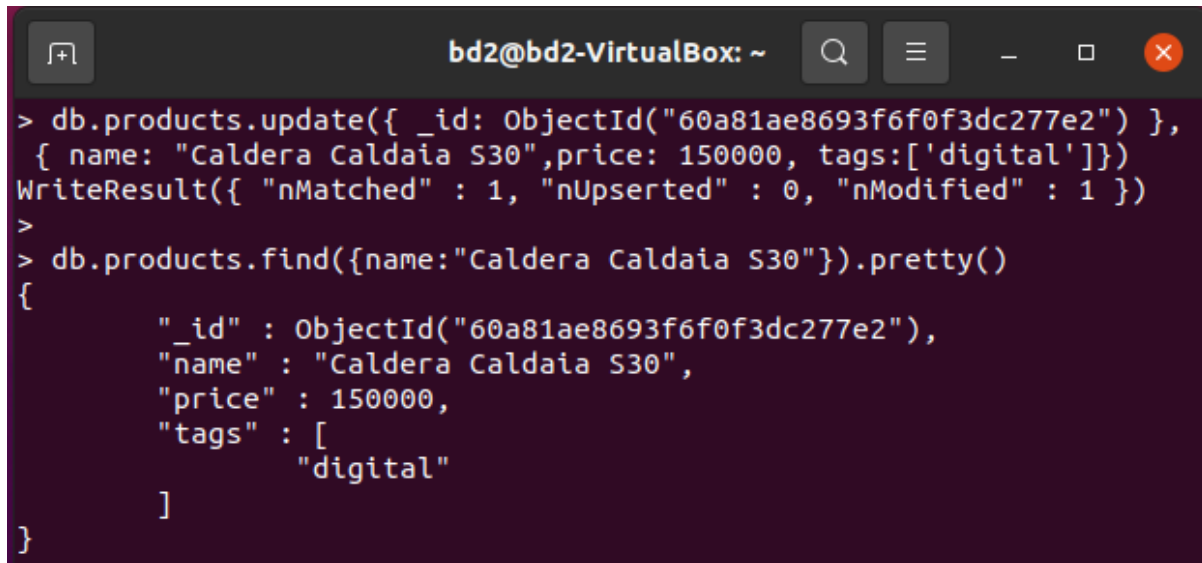
- 9. Agregue “digital” a las etiquetas de la “Caldera Caldaia S30”.

Se actualiza el documento:

```
db.products.update({ _id: ObjectId("60a81ae8693f6f0f3dc277e2") }, { name: "Caldera Caldaia S30", price: 150000, tags: ['digital'] })
```

Se chequea que el mismo se haya actualizado:

```
db.products.find({name:"Caldera Caldaia S30"}).pretty()
```

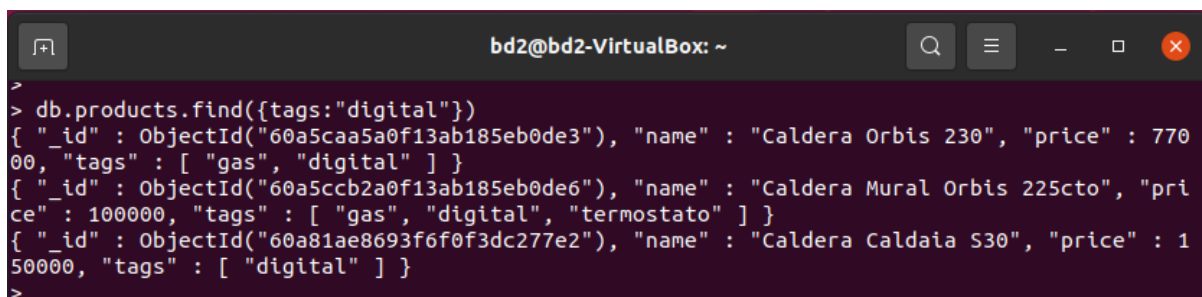


```
bd2@bd2-VirtualBox: ~  
> db.products.update({ _id: ObjectId("60a81ae8693f6f0f3dc277e2") },  
  { name: "Caldera Caldaia S30", price: 150000, tags: ['digital'] })  
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })  
>  
> db.products.find({name:"Caldera Caldaia S30"}).pretty()  
{  
  "_id" : ObjectId("60a81ae8693f6f0f3dc277e2"),  
  "name" : "Caldera Caldaia S30",  
  "price" : 150000,  
  "tags" : [  
    "digital"  
  ]  
}
```

- 10. Incremente en un 10% los precios de todas las calderas digitales.

Para validar el correcto incremento en los precios, se busca los productos en donde en el tag sea ‘digital’. Para esto, se ejecuta la siguiente consulta:

```
db.products.find({tags:"digital"})
```



```
bd2@bd2-VirtualBox: ~  
>  
> db.products.find({tags:"digital"})  
{ "_id" : ObjectId("60a5caa5a0f13ab185eb0de3"), "name" : "Caldera Orbis 230", "price" : 77000, "tags" : [ "gas", "digital" ] }  
{ "_id" : ObjectId("60a5ccb2a0f13ab185eb0de6"), "name" : "Caldera Mural Orbis 225cto", "price" : 100000, "tags" : [ "gas", "digital", "termostato" ] }  
{ "_id" : ObjectId("60a81ae8693f6f0f3dc277e2"), "name" : "Caldera Caldaia S30", "price" : 150000, "tags" : [ "digital" ] }
```

Luego, se ejecuta la actualización correspondiente, por sobre el precio de dichos productos.

```
db.products.update({tags:"digital"}, {$mul: {price:1.1}}, {multi:true})
```



```
bd2@bd2-VirtualBox: ~
> db.products.update({tags:"digital"}, {$mul: {price:1.1}}, {multi:true})
WriteResult({ "nMatched" : 3, "nUpserted" : 0, "nModified" : 3 })
>
```

Por último, se vuelve a ejecutar la query inicial, para validar que los precios hayan variado.

```
db.products.find({tags:"digital"})
```

```
bd2@bd2-VirtualBox: ~
> db.products.find({tags:"digital"})
{ "_id" : ObjectId("60a5caa5a0f13ab185eb0de3"), "name" : "Caldera Orbis 230", "price" : 84700, "tags" : [ "gas", "digital" ] }
{ "_id" : ObjectId("60a5ccb2a0f13ab185eb0de6"), "name" : "Caldera Mural Orbis 225cto", "price" : 110000.000000000001, "tags" : [ "gas", "digital", "termostato" ] }
{ "_id" : ObjectId("60a81ae8693f6f0f3dc277e2"), "name" : "Caldera Caldaia S30", "price" : 165000, "tags" : [ "digital" ] }
>
```

Parte 3: Índices

► Elimine todos los productos de la colección. Guarde en un archivo llamado 'generador.js' el siguiente código JavaScript y ejecútelo con: `load(<ruta del archivo 'generador.js'>)`. Si utiliza un cliente que lo permita (ej. Robo3T), se puede ejecutar directamente en el espacio de consultas.

11. Busque en la colección de compras (purchases) si existe algún índice definido.

- Para buscar en la colección de compras si existe algún índice se ejecuto el comando:

```
db.purchases.getIndexes()
```

```
bd2@bd2-VirtualBox: ~
> db.purchases.getIndexes()
[ { "v" : 2, "key" : { "_id" : 1 }, "name" : "_id_" } ]
```

12. Cree un índice para el campo productName. Busque los las compras que tengan en el nombre del producto el string "11" y utilice el método `explain("executionStats")` al final de la consulta, para comparar la cantidad de documentos examinados y el tiempo en milisegundos de la consulta con y sin índice.

- Se consulta las compras que contengan en el nombre de producto "11":

```
db.purchases.find({"productName":"/11/"}).explain("executionStats")
```

Resultado sin índice:

```
bd2@bd2-VirtualBox: ~
> db.purchases.find({"productName": /11/}).explain("executionStats")
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "ecommerce.purchases",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "productName" : {
        "$regex" : "11"
      }
    },
    "winningPlan" : {
      "stage" : "COLLSCAN",
      "filter" : {
        "productName" : {
          "$regex" : "11"
        }
      },
      "direction" : "forward"
    },
    "rejectedPlans" : [ ]
  },
  "executionStats" : {
    "executionSuccess" : true,
    "nReturned" : 2125,
    "executionTimeMillis" : 153,
    "totalKeysExamined" : 0,
    "totalDocsExamined" : 45831,
    "executionStages" : {
      "stage" : "COLLSCAN",
      "filter" : {
        "productName" : {
          "$regex" : "11"
        }
      },
      "nReturned" : 2125,
      "executionTimeMillisEstimate" : 36,
      "works" : 45833,
      "advanced" : 2125,
      "needTime" : 43707,
      "needYield" : 0,
      "saveState" : 45,
      "restoreState" : 45,
      "isEOF" : 1,
      "direction" : "forward",
      "docsExamined" : 45831
    }
  },
  "serverInfo" : {
    "host" : "bd2-VirtualBox",
    "port" : 27017,
    "version" : "4.4.5",
    "gitVersion" : "ff5cb77101b052fa02da43b8538093486cf9b3f7"
  },
  "ok" : 1
}
```

executionTimeMillis: 153

totalKeysExamined: 0

totalDocsExamined: 45831

- Se crea el índice para el campo productName, con el comando:

```
db.purchases.createIndex({productName:1})
```

```
bd2@bd2-VirtualBox: ~
> db.purchases.createIndex({productName:1})
{
  "createdCollectionAutomatically" : false,
  "numIndexesBefore" : 1,
  "numIndexesAfter" : 2,
  "ok" : 1
}
```

Resultado con índice:

```
bd2@bd2-VirtualBox: ~  
} db.purchases.find({"productName": /11/}).explain("executionStats")  
{  
  "queryPlanner" : {  
    "planerVersion" : 1,  
    "namespace" : "ecommerce.purchases",  
    "indexFilterSet" : false,  
    "parsedQuery" : {  
      "productName" : {  
        "$regex" : "11"  
      }  
    },  
    "winningPlan" : {  
      "stage" : "FETCH",  
      "inputStage" : {  
        "stage" : "IXSCAN",  
        "filter" : {  
          "productName" : {  
            "$regex" : "11"  
          }  
        },  
        "keyPattern" : {  
          "productName" : 1  
        },  
        "indexName" : "productName_1",  
        "isMultiKey" : false,  
        "multiKeyPaths" : {  
          "productName" : [ ]  
        },  
        "isUnique" : false,  
        "isSparse" : false,  
        "isPartial" : false,  
        "indexVersion" : 2,  
        "direction" : "forward",  
        "indexBounds" : {  
          "productName" : [  
            "[\\V, 0)",  
            "[11/, 11/]"  
          ]  
        }  
      },  
      "rejectedPlans" : [ ]  
    },  
    "executionStats" : {  
      "executionSuccess" : true,  
      "nReturned" : 2125,  
      "executionTimeMillis" : 229,  
      "totalKeysExamined" : 45831,  
      "totalDocsExamined" : 2125,  
      "executionStages" : {  
        "stage" : "FETCH",  
        "nReturned" : 2125,  
        "executionTimeMillisEstimate" : 68,  
        "works" : 45832,  
        "advanced" : 2125,  
        "needTime" : 43706,  
        "needYield" : 0,  
        "saveState" : 45,  
        "restoreState" : 45,  
        "isEOF" : 1,  
        "docsExamined" : 2125,  
        "alreadyHasObj" : 0,  
        "inputStage" : {  
          "stage" : "IXSCAN",  
          "filter" : {  
            "productName" : {  
              "$regex" : "11"  
            }  
          },  
          "nReturned" : 2125,  
          "executionTimeMillisEstimate" : 56,  
          "works" : 45832,  
          "advanced" : 2125,  
          "needTime" : 43706,  
          "needYield" : 0,  
          "saveState" : 45,  
          "restoreState" : 45,  
          "isEOF" : 1,  
          "keyPattern" : {  
            "productName" : 1  
          },  
          "indexName" : "productName_1",  
          "isMultiKey" : false,  
          "multiKeyPaths" : {  
            "productName" : [ ]  
          },  
          "isUnique" : false,  
          "isSparse" : false,  
          "isPartial" : false,  
          "indexVersion" : 2,  
          "direction" : "forward",  
          "indexBounds" : {  
            "productName" : [  
              "[\\V, 0)",  
              "[11/, 11/]"  
            ]  
          },  
          "keysExamined" : 45831,  
          "seeks" : 1,  
          "dupsTested" : 0,  
          "dupsDropped" : 0  
        }  
      }  
    },  
    "serverInfo" : {  
      "host" : "bd2-VirtualBox",  
      "port" : 27017,  
      "version" : "4.4.5",  
      "gitVersion" : "ff5cb77101b052fa02da43b0538093486cf9b3f7"  
    },  
    "ok" : 1  
  }  
}
```

executionTimeMillis: 229
totalKeysExamined: 45831
totalDocsExamined: 2125

Se concluye que las consultas con índice demoran más tiempo en ser resueltas.

13. Busque las compras enviadas dentro de la ciudad de Buenos Aires. Para esto, puede definir una variable en la terminal y asignarle como valor el polígono del archivo provisto caba.geojson (copiando y pegando directamente). Cree un índice geoespacial de tipo 2dsphere para el campo location de la colección purchases y, de la misma forma que en el punto 12, compare la performance de la consulta con y sin dicho índice.

- Se genera la variable caba que contiene los datos del archivo .geojson
- Se consulta las compras que hayan sido enviadas dentro de Buenos Aires:

```
db.purchases.find({location:{$geoIntersects: {$geometry: caba}}})
.explain("executionStats")
```

Resultado sin índice:

```
bd2@bd2-VirtualBox: ~
> db.purchases.find({location:{$geoIntersects:{$geometry: caba}}}).explain("executionStats")
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "ecommerce.purchases",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "location" : {
        "$geoIntersects" : {
          "$geometry" : {
            "type" : "MultiPolygon",
            "coordinates" : [
              [
                [
                  [
                    -58.46385847167969,
                    -34.53456889748654
                  ],
                  [
                    -58.49979488634765,
                    -34.54983198845187
                  ],
                  [
                    -58.532886345214844,
                    -34.614561581688186
                  ],
                  [
                    -58.528633117675774,
                    -34.6538278014492
                  ],
                  [
                    -58.48674774169922,
                    -34.68742794931483
                  ],
                  [
                    -58.47988128621804,
                    -34.6828648848744
                  ],
                  [
                    -58.46855163574218,
                    -34.65297974261105
                  ],
                  [
                    -58.465118488283225,
                    -34.64733122084415
                  ],
                  [
                    -58.4585952758789,
                    -34.63988735682951
                  ],
                  [
                    -58.45344543457832,
                    -34.63683274732642
                  ],
                  [
                    -58.45344543457832,
                    -34.63683274732642
                  ],
                  [
                    -58.447265625,
                    -34.63573026886882
                  ],
                  [
                    -58.438339233308445,
                    -34.63838297923296
                  ],
                  [
                    -58.38188433349689,
                    -34.62162597825766
                  ],
                  [
                    -58.38237762451172,
                    -34.5925196889388
                  ],
                  [
                    -58.378944396972656,
                    -34.5843238246475
                  ],
                  [
                    -58.46385847167969,
                    -34.53456889748654
                  ],
                  [
                    -58.46385847167969,
                    -34.53456889748654
                  ],
                  [
                    -58.49979488634765,
                    -34.54983198845187
                  ],
                  [
                    -58.49979488634765,
                    -34.54983198845187
                  ]
                ]
              ]
            ]
          }
        }
      }
    }
  },
  "winningPlan" : {
    "stage" : "COLLSCAN",
    "filter" : {
      "location" : {
        "$geoIntersects" : {
          "$geometry" : {
            "type" : "MultiPolygon",
            "coordinates" : [
              [
                [
                  [
                    -58.46385847167969,
                    -34.53456889748654
                  ],
                  [
                    -58.49979488634765,
                    -34.54983198845187
                  ],
                  [
                    -58.532886345214844,
                    -34.614561581688186
                  ],
                  [
                    -58.528633117675774,
                    -34.6538278014492
                  ],
                  [
                    -58.48674774169922,
                    -34.68742794931483
                  ],
                  [
                    -58.47988128621804,
                    -34.6828648848744
                  ],
                  [
                    -58.46855163574218,
                    -34.65297974261105
                  ],
                  [
                    -58.465118488283225,
                    -34.64733122084415
                  ],
                  [
                    -58.4585952758789,
                    -34.63988735682951
                  ],
                  [
                    -58.45344543457832,
                    -34.63683274732642
                  ],
                  [
                    -58.45344543457832,
                    -34.63683274732642
                  ],
                  [
                    -58.447265625,
                    -34.63573026886882
                  ],
                  [
                    -58.438339233308445,
                    -34.63838297923296
                  ],
                  [
                    -58.38188433349689,
                    -34.62162597825766
                  ],
                  [
                    -58.38237762451172,
                    -34.5925196889388
                  ],
                  [
                    -58.378944396972656,
                    -34.5843238246475
                  ],
                  [
                    -58.46385847167969,
                    -34.53456889748654
                  ],
                  [
                    -58.46385847167969,
                    -34.53456889748654
                  ],
                  [
                    -58.49979488634765,
                    -34.54983198845187
                  ],
                  [
                    -58.49979488634765,
                    -34.54983198845187
                  ]
                ]
              ]
            ]
          }
        }
      }
    }
  }
}
```

```

    ],
    "direction": "forward",
    "rejectedPlans": [ ]
  },
  "executionStats": {
    "executionSuccess": true,
    "nReturned": 18125,
    "executionTimeMillis": 696,
    "totalKeysExamined": 0,
    "totalDocsExamined": 45831,
    "executionStages": [ {
      "stage": "COLLSCAN",
      "filter": {
        "location": {
          "$geoIntersects": {
            "$geometry": {
              "type": "MultiPolygon",
              "coordinates": [
                [
                  [
                    [
                      -58.46305847167969,
                      -34.53456889748654
                    ],
                    [
                      -58.49979400634765,
                      -34.54983198845187
                    ],
                    [
                      -58.532066345214844,
                      -34.614561581688186
                    ],
                    [
                      -58.528633117675774,
                      -34.6538270014492
                    ],
                    [
                      -58.48674774169922,
                      -34.68742794931483
                    ],
                    [
                      -58.532066345214844,
                      -34.614561581688186
                    ],
                    [
                      -58.528633117675774,
                      -34.6538270014492
                    ],
                    [
                      -58.48674774169922,
                      -34.68742794931483
                    ],
                    [
                      -58.479881286621894,
                      -34.68206400648744
                    ],
                    [
                      -58.46855163574218,
                      -34.65297974261185
                    ],
                    [
                      -58.465118488283125,
                      -34.64733112904415
                    ],
                    [
                      -58.4585952758789,
                      -34.63998735602951
                    ],
                    [
                      -58.45344543457832,
                      -34.63683274732642
                    ],
                    [
                      -58.447265625,
                      -34.6357582686882
                    ],
                    [
                      -58.438339233398445,
                      -34.63838297923296
                    ],
                    [
                      -58.38100433349689,
                      -34.62162387826766
                    ],
                    [
                      -58.38237762451171,
                      -34.59251968889388
                    ],
                    [
                      -58.378044396972656,
                      -34.5843238246475
                    ],
                    [
                      -58.479881286621894,
                      -34.68206400648744
                    ],
                    [
                      -58.46855163574218,
                      -34.65297974261185
                    ],
                    [
                      -58.465118488283125,
                      -34.64733112904415
                    ],
                    [
                      -58.4585952758789,
                      -34.63998735602951
                    ],
                    [
                      -58.45344543457832,
                      -34.63683274732642
                    ],
                    [
                      -58.447265625,
                      -34.6357582686882
                    ],
                    [
                      -58.438339233398445,
                      -34.63838297923296
                    ],
                    [
                      -58.38100433349689,
                      -34.62162387826766
                    ],
                    [
                      -58.38237762451171,
                      -34.59251968889388
                    ],
                    [
                      -58.378044396972656,
                      -34.5843238246475
                    ],
                    [
                      -58.46305847167969,
                      -34.53456889748654
                    ]
                  ]
                ]
              ]
            }
          }
        ]
      }
    ]
  },
  "nReturned": 18125,
  "executionTimeMillis": 696,
  "totalKeysExamined": 0,
  "totalDocsExamined": 45831
}

```

```

    ],
    "direction": "forward",
    "rejectedPlans": [ ]
  },
  "executionStats": {
    "executionSuccess": true,
    "nReturned": 18125,
    "executionTimeMillis": 696,
    "totalKeysExamined": 0,
    "totalDocsExamined": 45831,
    "executionStages": [ {
      "stage": "COLLSCAN",
      "filter": {
        "location": {
          "$geoIntersects": {
            "$geometry": {
              "type": "MultiPolygon",
              "coordinates": [
                [
                  [
                    [
                      -58.46305847167969,
                      -34.53456889748654
                    ],
                    [
                      -58.49979400634765,
                      -34.54983198845187
                    ],
                    [
                      -58.532066345214844,
                      -34.614561581688186
                    ],
                    [
                      -58.528633117675774,
                      -34.6538270014492
                    ],
                    [
                      -58.48674774169922,
                      -34.68742794931483
                    ],
                    [
                      -58.532066345214844,
                      -34.614561581688186
                    ],
                    [
                      -58.528633117675774,
                      -34.6538270014492
                    ],
                    [
                      -58.48674774169922,
                      -34.68742794931483
                    ],
                    [
                      -58.479881286621894,
                      -34.68206400648744
                    ],
                    [
                      -58.46855163574218,
                      -34.65297974261185
                    ],
                    [
                      -58.465118488283125,
                      -34.64733112904415
                    ],
                    [
                      -58.4585952758789,
                      -34.63998735602951
                    ],
                    [
                      -58.45344543457832,
                      -34.63683274732642
                    ],
                    [
                      -58.447265625,
                      -34.6357582686882
                    ],
                    [
                      -58.438339233398445,
                      -34.63838297923296
                    ],
                    [
                      -58.38100433349689,
                      -34.62162387826766
                    ],
                    [
                      -58.38237762451171,
                      -34.59251968889388
                    ],
                    [
                      -58.378044396972656,
                      -34.5843238246475
                    ],
                    [
                      -58.479881286621894,
                      -34.68206400648744
                    ],
                    [
                      -58.46855163574218,
                      -34.65297974261185
                    ],
                    [
                      -58.465118488283125,
                      -34.64733112904415
                    ],
                    [
                      -58.4585952758789,
                      -34.63998735602951
                    ],
                    [
                      -58.45344543457832,
                      -34.63683274732642
                    ],
                    [
                      -58.447265625,
                      -34.6357582686882
                    ],
                    [
                      -58.438339233398445,
                      -34.63838297923296
                    ],
                    [
                      -58.38100433349689,
                      -34.62162387826766
                    ],
                    [
                      -58.38237762451171,
                      -34.59251968889388
                    ],
                    [
                      -58.378044396972656,
                      -34.5843238246475
                    ],
                    [
                      -58.46305847167969,
                      -34.53456889748654
                    ]
                  ]
                ]
              ]
            }
          }
        ]
      }
    ]
  },
  "nReturned": 18125,
  "executionTimeMillis": 696,
  "totalKeysExamined": 0,
  "totalDocsExamined": 45831
}

```

```

    ],
    "direction": "forward",
    "rejectedPlans": [ ]
  },
  "executionStats": {
    "executionSuccess": true,
    "nReturned": 18125,
    "executionTimeMillis": 696,
    "totalKeysExamined": 0,
    "totalDocsExamined": 45831,
    "executionStages": [ {
      "stage": "COLLSCAN",
      "filter": {
        "location": {
          "$geoIntersects": {
            "$geometry": {
              "type": "MultiPolygon",
              "coordinates": [
                [
                  [
                    [
                      -58.46305847167969,
                      -34.53456889748654
                    ],
                    [
                      -58.49979400634765,
                      -34.54983198845187
                    ],
                    [
                      -58.532066345214844,
                      -34.614561581688186
                    ],
                    [
                      -58.528633117675774,
                      -34.6538270014492
                    ],
                    [
                      -58.48674774169922,
                      -34.68742794931483
                    ],
                    [
                      -58.532066345214844,
                      -34.614561581688186
                    ],
                    [
                      -58.528633117675774,
                      -34.6538270014492
                    ],
                    [
                      -58.48674774169922,
                      -34.68742794931483
                    ],
                    [
                      -58.479881286621894,
                      -34.68206400648744
                    ],
                    [
                      -58.46855163574218,
                      -34.65297974261185
                    ],
                    [
                      -58.465118488283125,
                      -34.64733112904415
                    ],
                    [
                      -58.4585952758789,
                      -34.63998735602951
                    ],
                    [
                      -58.45344543457832,
                      -34.63683274732642
                    ],
                    [
                      -58.447265625,
                      -34.6357582686882
                    ],
                    [
                      -58.438339233398445,
                      -34.63838297923296
                    ],
                    [
                      -58.38100433349689,
                      -34.62162387826766
                    ],
                    [
                      -58.38237762451171,
                      -34.59251968889388
                    ],
                    [
                      -58.378044396972656,
                      -34.5843238246475
                    ],
                    [
                      -58.479881286621894,
                      -34.68206400648744
                    ],
                    [
                      -58.46855163574218,
                      -34.65297974261185
                    ],
                    [
                      -58.465118488283125,
                      -34.64733112904415
                    ],
                    [
                      -58.4585952758789,
                      -34.63998735602951
                    ],
                    [
                      -58.45344543457832,
                      -34.63683274732642
                    ],
                    [
                      -58.447265625,
                      -34.6357582686882
                    ],
                    [
                      -58.438339233398445,
                      -34.63838297923296
                    ],
                    [
                      -58.38100433349689,
                      -34.62162387826766
                    ],
                    [
                      -58.38237762451171,
                      -34.59251968889388
                    ],
                    [
                      -58.378044396972656,
                      -34.5843238246475
                    ],
                    [
                      -58.46305847167969,
                      -34.53456889748654
                    ]
                  ]
                ]
              ]
            }
          }
        ]
      }
    ]
  },
  "nReturned": 18125,
  "executionTimeMillis": 696,
  "totalKeysExamined": 0,
  "totalDocsExamined": 45831
}

```

```

    "executionTimeMillisEstimate" : 479,
    "works" : 45831,
    "advanced" : 18125,
    "needTime" : 35707,
    "needFields" : 0,
    "saveState" : 49,
    "restoreState" : 49,
    "isExp" : 1,
    "direction" : "forward",
    "docsExamined" : 45831
  },
  "serverInfo" : {
    "host" : "bd2-VirtualBox",
    "port" : 27017,
    "version" : "4.4.3",
    "gitVersion" : "f15c2b77191b052fa82da43b8138893486cf9b3ff7"
  },
  "ok" : 1
}

```

executionTimeMillis: 686
 totalKeysExamined: 0
 totalDocsExamined: 45831

- Se crea el indice 2dsphere para el campo location, con el comando:

```
db.purchases.createIndex( { "location" : "2dsphere" } )
```

```

bd2@bd2-VirtualBox: ~
> db.purchases.createIndex( { "location" : "2dsphere" } )
{
  "createdCollectionAutomatically" : false,
  "numIndexesBefore" : 2,
  "numIndexesAfter" : 3,
  "ok" : 1
}

```

Resultado con índice:

```

bd2@bd2-VirtualBox: ~
> db.purchases.find((location:{$geoIntersects:{$geometry: caba}})).explain("executionStats")
{
  "queryPlanner" : {
    "plannerVersion" : 1,
    "namespace" : "e-commerce.purchases",
    "indexFilterSet" : false,
    "parsedQuery" : {
      "location" : {
        "$geoIntersects" : {
          "$geometry" : {
            "type" : "MultiPolygon",
            "coordinates" : [
              [
                [
                  [
                    -58.46385847167969,
                    -34.53456889748654
                  ],
                  [
                    -58.49979408634765,
                    -34.54983198845187
                  ],
                  [
                    -58.532866345214844,
                    -34.614561581688186
                  ],
                  [
                    -58.528633117675774,
                    -34.6338278014402
                  ],
                  [
                    -58.48674774169922,
                    -34.68742794931483
                  ],
                  [
                    -58.479881286621894,
                    -34.6828648648144
                  ],
                  [
                    -58.46855163574218,
                    -34.65297974261185
                  ],
                  [
                    -58.465118488283125,
                    -34.64733112984415
                  ],
                  [
                    -58.4185952758789,
                    -34.63988725682951
                  ],
                  [
                    -58.45344543457032,
                    -34.63603274732642
                  ]
                ]
              ]
            ]
          }
        }
      }
    }
  }
}

```

```

    ],
    "winningPlan": {
      "stage": "FETCH",
      "filter": {
        "location": {
          "GeoIntersects": {
            "geometry": {
              "type": "MultiPolygon",
              "coordinates": [
                [
                  [
                    [
                      -58.447265625,
                      -34.63575026866882
                    ],
                    [
                      -58.438339233398445,
                      -34.63838297923296
                    ],
                    [
                      -58.38100433349609,
                      -34.62162507826766
                    ],
                    [
                      -58.38237762451171,
                      -34.5925196889388
                    ],
                    [
                      -58.378044396972656,
                      -34.5843230246475
                    ],
                    [
                      -58.46385847167969,
                      -34.53456889748654
                    ]
                  ]
                ]
              ]
            }
          }
        }
      }
    }
  },
  "winningPlan": {
    "stage": "FETCH",
    "filter": {
      "location": {
        "GeoIntersects": {
          "geometry": {
            "type": "MultiPolygon",
            "coordinates": [
              [
                [
                  [
                    -58.46385847167969,
                    -34.53456889748654
                  ],
                  [
                    -58.49979408634785,
                    -34.54983198845187
                  ],
                  [
                    -58.532066345214844,
                    -34.614561581688186
                  ]
                ]
              ]
            ]
          }
        }
      }
    }
  }
}

```

```

    ],
    "winningPlan": {
      "stage": "FETCH",
      "filter": {
        "location": {
          "GeoIntersects": {
            "geometry": {
              "type": "MultiPolygon",
              "coordinates": [
                [
                  [
                    [
                      -58.528633117675774,
                      -34.6538270014402
                    ],
                    [
                      -58.48074774169922,
                      -34.68742794931483
                    ],
                    [
                      -58.470881286421804,
                      -34.68206490648744
                    ],
                    [
                      -58.46855163574218,
                      -34.65297974261185
                    ],
                    [
                      -58.465118488203125,
                      -34.64733112004415
                    ],
                    [
                      -58.4585932758789,
                      -34.63998735602951
                    ],
                    [
                      -58.45344543457032,
                      -34.63693274732642
                    ],
                    [
                      -58.447265625,
                      -34.63575026866882
                    ],
                    [
                      -58.438339233398445,
                      -34.63838297923296
                    ],
                    [
                      -58.38100433349609,
                      -34.62162507826766
                    ],
                    [
                      -58.38237762451171,
                      -34.5925196889388
                    ],
                    [
                      -58.378044396972656,
                      -34.5843230246475
                    ],
                    [
                      -58.46385847167969,
                      -34.53456889748654
                    ]
                  ]
                ]
              ]
            }
          }
        }
      }
    }
  },
  "winningPlan": {
    "stage": "FETCH",
    "filter": {
      "location": {
        "GeoIntersects": {
          "geometry": {
            "type": "MultiPolygon",
            "coordinates": [
              [
                [
                  [
                    -58.528633117675774,
                    -34.6538270014402
                  ],
                  [
                    -58.48074774169922,
                    -34.68742794931483
                  ],
                  [
                    -58.470881286421804,
                    -34.68206490648744
                  ],
                  [
                    -58.46855163574218,
                    -34.65297974261185
                  ],
                  [
                    -58.465118488203125,
                    -34.64733112004415
                  ],
                  [
                    -58.4585932758789,
                    -34.63998735602951
                  ],
                  [
                    -58.45344543457032,
                    -34.63693274732642
                  ],
                  [
                    -58.447265625,
                    -34.63575026866882
                  ],
                  [
                    -58.438339233398445,
                    -34.63838297923296
                  ],
                  [
                    -58.38100433349609,
                    -34.62162507826766
                  ],
                  [
                    -58.38237762451171,
                    -34.5925196889388
                  ],
                  [
                    -58.378044396972656,
                    -34.5843230246475
                  ],
                  [
                    -58.46385847167969,
                    -34.53456889748654
                  ]
                ]
              ]
            ]
          }
        }
      }
    }
  }
}

```

```

    ],
    "winningPlan": {
      "stage": "FETCH",
      "filter": {
        "location": {
          "GeoIntersects": {
            "geometry": {
              "type": "MultiPolygon",
              "coordinates": [
                [
                  [
                    [
                      -7718162562058289152,
                      -7718162562058289152
                    ],
                    [
                      -769862296157213606,
                      -769862296157213606
                    ],
                    [
                      -7657245266436685824,
                      -7657245266436685824
                    ],
                    [
                      -7657851752398197748,
                      -7657851752398197748
                    ],
                    [
                      -7657007354343686144,
                      -7657007354343686144
                    ],
                    [
                      -7657047354343686143,
                      -7657047354343686143
                    ],
                    [
                      -7657046254832058368,
                      -7657046254832058368
                    ],
                    [
                      -76570462204721350909,
                      -76570462204721350909
                    ],
                    [
                      -7657046186112581632,
                      -7657046186112581632
                    ],
                    [
                      -7657045970954151424,
                      -7657045970954151424
                    ],
                    [
                      -76570459112346746488,
                      -76570459112346746488
                    ],
                    [
                      -7657045876874936319,
                      -7657045876874936319
                    ],
                    [
                      -7657045802515197951,
                      -7657045802515197951
                    ],
                    [
                      -7657045705076244479,
                      -7657045705076244479
                    ],
                    [
                      -76570451553204389593,
                      -76570451553204389593
                    ],
                    [
                      -7657044855646167823,
                      -7657044855646167823
                    ],
                    [
                      -7657044855888882816,
                      -7657044855888882816
                    ],
                    [
                      -7657044855888882815,
                      -7657044855888882815
                    ],
                    [
                      -7657043978893324040,
                      -7657043978893324040
                    ],
                    [
                      -7657043788938895872,
                      -7657043788938895872
                    ],
                    [
                      -765704368695388927,
                      -765704368695388927
                    ],
                    [
                      -7657043456264152832,
                      -7657043456264152832
                    ],
                    [
                      -7657025295391653888,
                      -7657025295391653888
                    ],
                    [
                      -7657025240538946336,
                      -7657025240538946336
                    ],
                    [
                      -7657025238967144448,
                      -7657025238967144448
                    ],
                    [
                      -7657025227745918976,
                      -7657025227745918976
                    ],
                    [
                      -7657025226948612688,
                      -7657025226948612688
                    ],
                    [
                      -7657025226739286016,
                      -7657025226739286016
                    ],
                    [
                      -7657025226688954368,
                      -7657025226688954368
                    ],
                    [
                      -7657025226688653759,
                      -7657025226688653759
                    ],
                    [
                      -7657025226672171151,
                      -7657025226672171151
                    ]
                  ]
                ]
              ]
            }
          }
        }
      }
    }
  },
  "winningPlan": {
    "stage": "FETCH",
    "filter": {
      "location": {
        "GeoIntersects": {
          "geometry": {
            "type": "MultiPolygon",
            "coordinates": [
              [
                [
                  [
                    -7718162562058289152,
                    -7718162562058289152
                  ],
                  [
                    -769862296157213606,
                    -769862296157213606
                  ],
                  [
                    -7657245266436685824,
                    -7657245266436685824
                  ],
                  [
                    -7657851752398197748,
                    -7657851752398197748
                  ],
                  [
                    -7657007354343686144,
                    -7657007354343686144
                  ],
                  [
                    -7657047354343686143,
                    -7657047354343686143
                  ],
                  [
                    -7657046254832058368,
                    -7657046254832058368
                  ],
                  [
                    -76570462204721350909,
                    -76570462204721350909
                  ],
                  [
                    -7657046186112581632,
                    -7657046186112581632
                  ],
                  [
                    -7657045970954151424,
                    -7657045970954151424
                  ],
                  [
                    -76570459112346746488,
                    -76570459112346746488
                  ],
                  [
                    -7657045876874936319,
                    -7657045876874936319
                  ],
                  [
                    -7657045802515197951,
                    -7657045802515197951
                  ],
                  [
                    -7657045705076244479,
                    -7657045705076244479
                  ],
                  [
                    -76570451553204389593,
                    -76570451553204389593
                  ],
                  [
                    -7657044855646167823,
                    -7657044855646167823
                  ],
                  [
                    -7657044855888882816,
                    -7657044855888882816
                  ],
                  [
                    -7657044855888882815,
                    -7657044855888882815
                  ],
                  [
                    -7657043978893324040,
                    -7657043978893324040
                  ],
                  [
                    -7657043788938895872,
                    -7657043788938895872
                  ],
                  [
                    -765704368695388927,
                    -765704368695388927
                  ],
                  [
                    -7657043456264152832,
                    -7657043456264152832
                  ],
                  [
                    -7657025295391653888,
                    -7657025295391653888
                  ],
                  [
                    -7657025240538946336,
                    -7657025240538946336
                  ],
                  [
                    -7657025238967144448,
                    -7657025238967144448
                  ],
                  [
                    -7657025227745918976,
                    -7657025227745918976
                  ],
                  [
                    -7657025226948612688,
                    -7657025226948612688
                  ],
                  [
                    -7657025226739286016,
                    -7657025226739286016
                  ],
                  [
                    -7657025226688954368,
                    -7657025226688954368
                  ],
                  [
                    -7657025226688653759,
                    -7657025226688653759
                  ],
                  [
                    -7657025226672171151,
                    -7657025226672171151
                  ]
                ]
              ]
            ]
          }
        }
      }
    }
  }
}

```


[illegible]

executionTimeMillis: 395

totalDocsExamined: 12738

En este caso, a diferencia del punto 12, se considera que la velocidad de respuesta con índices es superior, es decir que demora menos tiempo que la respuesta sin índices

Parte 4: Aggregation Framework

•MongoDB cuenta con un Aggregation Framework que brinda la posibilidad de hacer analítica en tiempo real del estilo OLAP (Online Analytical Processing), de forma similar a otros productos específicos como Hadoop o MapReduce. En los siguientes ejercicios se verán algunos ejemplos de su aplicabilidad.

- Para obtener documentos aleatoriamente se usa la funcion de agregación \$sample
`db.products.aggregate([{ $sample: { size: 5 } }])`

```
bd2@bd2-VirtualBox: ~  
> db.products.aggregate([{$sample: { size: 5 }}])  
{  
  "_id" : ObjectId("60adec2e1ef5cd59c6a79f11"), "name" : "Producto 483", "price" : 301165, "tags" : [ "cuotas", "oferta" ] }  
  "_id" : ObjectId("60adec4c1ef5cd59c6a7f3f3"), "name" : "Producto 22213", "price" : 110069, "tags" : [ ] }  
  "_id" : ObjectId("60adec5a1ef5cd59c6a815ab"), "name" : "Producto 30845", "price" : 49942, "tags" : [ "verificado", "oferta" ] }  
  "_id" : ObjectId("60adec371ef5cd59c6a7b869"), "name" : "Producto 6971", "price" : 169813, "tags" : [ "cuotas", "verificado" ] }  
  "_id" : ObjectId("60adec741ef5cd59c6a8569d"), "name" : "Producto 47471", "price" : 203461, "tags" : [ "oferta", "verificado", "cuotas" ] }
```

15. Usando el framework de agregación, obtenga las compras que se hayan enviado a 1km (o menos) del centro de la ciudad de Buenos Aires ([-58.4586,-34.5968]) y guárdelas en una nueva colección.

```
db.purchases.aggregate([ { $geoNear:
  { near:
    {type: "Point", coordinates:[ -58.4586, -34.5968 ]},
    maxDistance:1000,
    distanceField: "dist.calculated"}}},
  {$out:"aUnoDelCentro"}}])
```

16. Obtenga una colección de los productos que fueron enviados en las compras del punto anterior. Note que sólo es posible ligarlas por el nombre del producto.

► Si la consulta se empieza a tornar difícil de leer, se pueden ir guardando los agregadores en variables, que no son más que objetos en formato JSON.

```
var productsJoinAUnoDelCentro = {$lookup:
  {from:"products" ,
   localField: "productName",
   foreignField: "name",
   as: "PName"}}};

var onlyProducts = {$project: {"PName":1, "_id":0}};

var products = db.aUnoDelCentro.aggregate(productsJoinAUnoDelCentro,
  { $unwind: "$PName"},
  onlyProducts)

db.products.aggregate(
  {$project:{"PName":0}},
  {$out:"productosDelCentro"})
```

17. Usando la colección del punto anterior, obtenga una nueva en la que agrega a cada producto un atributo purchases que consista en un array con todas las compras de cada producto.

```
var productsJoinPurchases = {$lookup:
  {from:"purchases" ,
   localField: "name",
   foreignField: "productName",
   as: "purchasesByProduct"}}};

db.productosDelCentro.aggregate(productsJoinPurchases,{$out:"productsWithPurchases"})
```

18. Obtenga el promedio de costo de envío pagado para cada producto del punto anterior.

```
var promedio = db.productsWithPurchasesT.aggregate([{$project:{ "_id":0, "name":1,
avgShipping: { $avg: "$purchasesByProduct.shippingCost"}} } ])
```

Referencias:

<https://docs.mongodb.com/manual/>

<https://www.genbeta.com/desarrollo/una-introduccion-a-mongodb>

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

<https://www.genbeta.com/desarrollo/mongodb-creacion-y-utilizacion-de-indices>

<https://www.codigofuente.org/introduccion-a-mongodb/>

<https://platzi.com/contributions/introduccion-al-uso-de-indices-en-mongodb/>

<http://gpd.sip.ucm.es/rafa/docencia/nosql/indices.html>

<https://charlascylon.com/2013-08-01-tutorial-mongodb-%C3%ADndices>

<https://unpocodejava.com/2014/07/07/referencias-en-mongodb/>

<https://platzi.com/tutoriales/1533-mongodb/3757-transacciones-en-mongodb-2/>

Alumnos: Mosquera, Matías

Marascutti, Juan Marcos

Solar, Jonatan Matias

Materia: Base de Datos 2

Docente: Benítez, Luciano

Año: 2021