Práctica

1) Usando Google Maps encuentra las coordenadas esféricas del Bar Manolo, Calle Princesa 83, y completa su longitud y latitud (los valores que aparecen después de la @ en la url, pero dados la vuelta), las instrucciones de inserción siguiente:

```
db.lugares.drop()
db.lugares.insert({nombre:"Informática-UCM", tipo:"Facultad", pais:"España",
                    posicion: {type:"Point", coordinates:[-3.7353797,40.450305]}})
db.lugares.insert({nombre:"Metropolitano", tipo:"metro", pais:"España",
                    posicion: {type:"Point", coordinates:[-3.7202654,40.4465915]}})
db.lugares.insert({nombre:"Ciudad Universitaria", tipo:"metro", pais:"España",
                   posicion: {type:"Point", coordinates:[-3.7289768,40.4435602]}})
db.lugares.insert({nombre:"Pabellón de Plata", tipo:"templo", pais:"Japón",
                    posicion: {type:"Point", coordinates:[135.7982058,35.0270213]}})
db.lugares.insert({nombre:"Manolo", tipo:"bar", pais:"España",
                    posicion: {type:"Point", coordinates:[????long,????lat]}})
```

Y crear un índice tipo 2d esférico sobre el campo “coord”

Escribir en la solución:
- Comando insert del lugar del que se han buscado las coordenadas (solo de ese)
- Instrucción para crear el índice y respuesta del shell

Solución
- Comando insert del lugar del que se han buscado las coordenadas (solo de ese)

```
db.lugares.insert({nombre:"Manolo", tipo:"bar", pais:"España",
                    posicion: {type:"Point", coordinates:[-3.7201797,40.4326352]}})
```

- Instrucción para crear el índice y respuesta del shell

```
db.lugares.createIndex({posicion:"2dsphere"})
{
      "createdCollectionAutomatically" : false,
      "numIndexesBefore" : 1,
      "numIndexesAfter" : 2,
      "ok" : 1
}
```

2.- Encontrar todos los objetos de la colección lugares que estén a menos de 1000 metros de las coordenadas: lat: 40.4381963,long : -3.7251149 (museo de américa). Para cada lugar se debe mostrar solo el nombre y el tipo

Copia en la solución la consulta.
Solución:
consulta:

```
db.lugares.find( { 'posicion' : { $near : {
                            $geometry: { type:'Point',
                                         coordinates: [-3.7251149,40.4381963]},
                            $maxDistance: 1000 } } }, {nombre:1, tipo:1, _id:0});
```

3.- Borrar el índice del apartado 1, y repetir la consulta 2. ¿Qué se obtiene? Escribir en la solución la instrucción que borra el índice y el resultado que se muestra.

Solucion

Borrar índice: db.lugares.dropIndex("posicion_2dsphere")

Mensaje que se obtiene:

Error: error: {

"waitedMS" : NumberLong(0),

"ok" : 0,

"errmsg" : "error processing query: ns=test.lugaresTree: GEONEAR field=posicion maxdist=1000 isNearSphere=0\nSort: {}\nProj: { nombre: 1.0, tipo: 1.0, _id: 0.0 }\n planner returned error: unable to find index for $geoNear query",

"code" : 2

}

4 Ahora vamos a utilizar puntos en el plano. Para ello basta con definir documentos que tengaun una clave con las dos coodenadas {….,punto: [0,0] ….}.

Introduce en el shell de MongoDB la siguiente instrucción que añade a la colección “plano” unos cuantos puntos:

```
use pract2d

db.plano.drop()

for (var i=0; i<500; i++) { for (var j=0; j<=i; j++)
       { db.plano.insert({desc:"("+i+","+j+")", punto:[i,j]});}}
```

Crear un índice sobre la clave “punto” con nombre “ipunto” (un índice normal, no un geoespacial), y otro sobre la clave “desc” con nombre “idesc” (de nuevo un índice normal).

Comprobar el tamaño en bytes de los dos índices. Copia en la solución las instrucciones para crear los índices, así como la instrucción que nos permite comprobar su tamaño en bytes.

Indica también por qué crees que el de mayor tamaño ocupa más que el otro (una frase debe bastar)

Solución:

Creación de índice: db.plano.createIndex({punto:1},"ipunto")

db.plano.createIndex({desc:1},"idesc")

Comprobar tamaño: db.plano.stats()

¿Cuál de los dos índices tiene mayor tamaño? ipunto

¿Por qué? (una frase debe bastar): porque es un índice multikey, donde tiene que crear una entrada para cada valor del array.

5) Queremos contar el número de elementos que tienen la primera coordenada mayor de 480. Copiarlo en la solución. **Ayuda**: buscar en la parte dedicada a CRUD, dentro de “find en detalle” operadores que actúan sobre arrays.

Solución:

query: db.plano.find({"punto.0":{$gt:480}}).count()

6) Queremos probar la consulta: db.plano.find({punto:{$lt :2}},{desc:1,_id:0})

¿Utilizará el índice? ¿Cuál?

En caso afirmativo, ¿cuántas claves tienen que ser accedidas para obtener el resultado?

¿Y cuantos documentos?

Escribe en la solución estás respuestas, junto con la instrucción que has escrito en el shell para obtener los datos.

Solución:

Instrucción para obtener los datos:

```
db.plano.explain("executionStats").find({punto:{$lt :2}},{desc:1,_id:0})
```

Utiliza el índice (Sí/no): sí

¿Cuál? ipunto

Claves: 1000

Documentos: 999

7) Ahora queremos ver los datos de la consulta

db.plano.find({desc:{$gt:"(5"}},{desc:1,_id:0})

¿Cuántos documentos son accedidos? ¿por qué?

Solución:

Núm. Documentos: 0

¿Por qué? (una frase): el índice tiene toda la información que hace falta para mostrar la solución.

8) MongoDB permite preguntar por aquellos documentos que están dentro de la zona determinada por un polígono. Por ejemplo, podemos preguntar por los puntos que están dentro del cuadrado especificado por las coordenadas `[ 0 , 0 ], [ 3 , 0 ], [ 3 , 3 ], [0,3]`

Escribe en la solución la respuesta.

Solución:

```
db.plano.find( {punto: {  $geoWithin:

              { $polygon: [ [ 0 , 0 ], [ 3 , 0 ], [ 3 , 3 ], [0,3] ] } }   } )
```

o

```
db.plano.find( {punto: {  $geoWithin: { $box: [ [ 0 , 0 ],  [ 3 , 3 ] ] } }   } )
```

9) La consulta anterior no utiliza ningún índice. Escribe en la solución qué indice crearías para mejorar la eficiencia. ¿Se puede crear el índice? ¿Por qué?

Solución

Índice:

Instrucción para el índice: db.plano.createIndex({ punto: "2d" })

Se puede: no

¿Por qué? Porque hay puntos que no tienen coordenadas lomgitud, latitud válidas.

10) Supón que con los índices existentes se quiere ejecutar la consulta:

find({punto:{$lt :2}, desc:{$gt:"(5"}},{desc:1,_id:0})

¿Podría crearse un índice que mejore el rendimiento de esta consulta?

Si la respuesta es sí ¿cuál y cómo sabes que mejora?

Si la respuesta es no ¿por qué?

Solución.

Sí: Se puede crear un índice que mejore. Por ejemplo

find({punto:{$lt :2}, desc:{$gt:"(5"}},{desc:1,_id:0})

El número de documentos examinados es 110 y el número de claves es 113, cuando antes de crear el índice era 999 y 1000, respectivamente. Además el plan original aparece como rechazado tras crear este índice.