Práctica MongoDB

IT BOARDING

BOOTCAMP

Instalar MongoDB

Vamos a utilizar el gestor de paquete `brew` , vamos a añadir la referencia a las herramientas para poder luego instalar mongo:

1. `brew tap mongodb/brew`
2. `brew update`
3. `brew install mongodb-community@6.0`

Una vez instalado mongo, ahora debemos ejecutar el servicio para empezar a utilizarlo

```
brew services start mongodb-community@6.0
```

Finalmente vamos a descargar e instalar Compass, un cliente con interfaz gráfica:

Download and Install Compass

Práctica

Para resolver los siguientes ejercicios vamos a usar la colección “restaurantes”.

- Se recomienda leer las preguntas guías de la siguiente PPT para ir entrando en tema.
- Crear una base de datos denominada como sample_restaurantes mediante MongoDB Compass.
- Descargar la colección e importarla de manera local en su base de datos recien creada.
- Una vez adentro, desplegar Mongo Shell y ejecutar

> use sample_restaurantes

También pueden resultar útiles estas referencias para los 1ros pasos: MongoDB Cheat Sheet.

Para empezar

Estas preguntas pueden responderse utilizando la interfaz gráfica de Compass.

1. ¿Cuántas colecciones tiene la base de datos?
2. ¿Cuántos documentos hay en cada colección? ¿Cuánto pesa cada colección?
3. ¿Cuántos índices en cada colección? ¿Cuánto espacio ocupan los índices de cada colección?
4. Traer un documento de ejemplo de cada colección. db.collection.find(...).pretty() nos da un formato más legible.
5. Para cada colección, listar los campos a nivel raíz (ignorar campos dentro de documentos anidados) y sus tipos de datos.

😀 Todo listo para empezar.

Ejercicio 1: SQL

Usando Mongo Shell. Colección restaurantes se requiere:

1. Devolver restaurante_id, nombre, barrio y tipo_cocina pero excluyendo _id para un documento (el primero).

2. Devolver restaurante_id, nombre, barrio y tipo_cocina para los primeros 3 restaurantes que contengan 'Bake' en alguna parte de su nombre.

3. Contar los restaurantes de comida (tipo_cocina) china (*Chinese*) o tailandesa (*Thai*) del barrio (barrio) Bronx. Consultar or versus in.

Ejercicio 2: NoSQL

1. Traer 3 restaurantes que hayan recibido al menos una calificación de grado 'A' con puntaje mayor a 20. Una misma calificación debe cumplir con ambas condiciones simultáneamente; investigar el operador elemMatch.

2. ¿A cuántos documentos les faltan las coordenadas geográficas? En otras palabras, revisar si el tamaño de direccion.coord es 0 y contar.

3. Devolver nombre, barrio, tipo_cocina y grados para los primeros 3 restaurantes; de cada documento **solo la última calificación**. Ver el operador slice.

Ejercicio 3: Popurri

1. ¿Cuál es top 3 de tipos de cocina (cuisine) que podemos encontrar entre los datos? *Googlear* "mongodb group by field, count it and sort it". Ver etapa limit del *pipeline* de agregación.

2. ¿Cuáles son los barrios más desarrollados gastronómicamente? Calcular el promedio ($avg) de puntaje (grades.score) por barrio; considerando restaurantes que tengan más de tres reseñas; ordenar barrios con mejor puntaje arriba. **Ayuda**:

    a. match es una etapa que filtra documentos según una condición, similar a db.orders.find(<condición>).

    b. Parece necesario deconstruir las listas grades para producir un documento por cada puntaje utilizando la etapa unwind.

3. Una persona con ganas de comer está en longitud -73.93414657 y latitud 40.82302903, ¿qué opciones tiene en 500 metros a la redonda? Consultar geospatial tutorial.

Soluciones

SQL

1.

```
db.restaurants.findOne(
  { },
  { restaurant_id: 1, name: 1, borough: 1, cuisine: 1, _id: 0 }
)
```

2.

```
db.restaurants.find(
  { name: /Bake/ },
  { restaurant_id: 1, name: 1, borough: 1, cuisine: 1 }
).limit(3)
```

3.

```
db.restaurants.count(
  { cuisine: { $in: [ "Chinese", "Thai" ] }, borough: "Bronx" }
)
```

Query

1.

```
db.restaurantes.findOne
    { },
    { restaurante_id: 1, nombre: 1, barrio: 1, tipo_cocina: 1,  _id: 0
)
```

2.

```
db.restaurantes.find
    { nombre: /Bake/ },
    { restaurante_id: 1, nombre: 1, barrio: 1, tipo_cocina: 1
).limit(3)
```

3.

```
db.restaurantes.count
    { tipo_cocina: { $in: [ "Chinese", "Thai" ] }, barrio: "Bronx"
)
```

NoSQL

1.

```
db.restaurantes.find(
  { grades: { $elemMatch: { grade:"A", score: { $gt: 20 } } }}
).limit(3)
```

2.

```
db.restaurantes.count(
  { "address.coord": { $size: 0 } }
)
```

3.

```
db.restaurantes.find(
  { },
  { name: 1, borough: 1, cuisine: 1, grades: { $slice: -1 } }
).limit(3)
```

Query

1.

```
db.restaurantes.find
    { grados: { $elemMatch: { grado:"A", puntaje: { $gt: 20 } } }
).limit(3)
```

2.

```
db.restaurantes.count
    { "direccion.coord": { $size: 0 }
)
```

3.

```
db.restaurantes.find
    { },
    { nombre: 1, barrio: 1, tipo_cocina: 1, grados: { $slice: -1 } }
).limit(3)
```