Deusto
Facultad de Ingeniería
Ingeniaritzako Fakultatea

Práctica BD NoSQL

Arquitectura Tecnológica para Big Data

Grupo 4

Oscar Basaguren

En Bilbao, a 16 de Diciembre del 2025Ç

1. Elección de MongoDB

Para el modelado del caso de uso de registro y análisis de terremotos, la base de datos NoSQL seleccionada es MongoDB. La elección se fundamenta en que este tipo de información presenta alta variabilidad estructural, gran volumen de inserciones y consultas frecuentes por filtros geográficos y temporales, lo que encaja perfectamente con un modelo documental.

A continuación se presentan las ventajas y desventajas principales que justifican esta elección.

Flexibilidad del modelo documental (schema-less con validación opcional)
Los registros sísmicos no siempre contienen los mismos campos: algunos incluyen parámetros de falla o información de riesgo de tsunami, y otros no.
MongoDB permite almacenar documentos con estructuras distintas sin necesidad de un esquema rígido, eliminando columnas nulas,dependencias entre tablas, y la necesidad de realizar JOINs.Cada evento se representa de forma natural como un documento independiente.

Alto rendimiento en inserciones y lecturas
MongoDB está optimizado para:ingesta rápida de datos, actualizaciones frecuentes de valores (magnitud ,intensidad,riesgo) y consultas en grandes volúmenes.

Desnormalización eficiente
Factores como la ubicación, magnitud, parámetros de falla y riesgo de tsunami pueden almacenarse dentro del mismo documento. Así, acelerar las consultas, evitar múltiples accesos a distintas tablas y mejorar la coherencia interna de cada evento.

Soporte de consultas geoespaciales e índices compuestos
MongoDB incluye:

- Índices 2dsphere para búsquedas por coordenadas (cercanía, intersección con áreas, etc.).
- Índices compuestos para mejorar consultas por país, año, magnitud o riesgo.
- Operadores específicos para análisis geográfico.

No obstante también presenta ciertas desventajas:

Ausencia de claves foráneas y consistencia referencial estricta
Relaciones como país → región o región → zona sísmica no se garantizan automáticamente; si se requieren, deben implementarse mediante validaciones personalizadas o control desde la aplicación, lo que puede añadir complejidad cuando existen dependencias entre entidades.

Mayor coste en actualizaciones masivas debido a la desnormalización
Si un dato común cambia (por ejemplo, renombrar una zona sísmica), debe modificarse en varios documentos. Aunque MongoDB permite actualizaciones en bloque, sigue siendo menos eficiente que una actualización única en un modelo relacional normalizado.

Agregaciones complejas menos instuitivas
Aunque el aggregation pipeline de MongoDB es potente, ciertos análisis avanzados pueden resultar más simples en motores especializados, como bases de datos columnares o de series temporales (por ejemplo, InfluxDB).

MongoDB resulta ser adecuado para este caso porque ofrece una estructura flexible, alto rendimiento en ingesta y lectura, soporte geoespacial avanzado y simplicidad en la consulta de documentos autosuficientes, de modo que su modelo documental encaja de forma natural con la representación completa de un evento sísmico.

2. Definición del Esquema

El modelo sigue la estructura lógica del dataset, agrupando la información en cinco bloques principales:

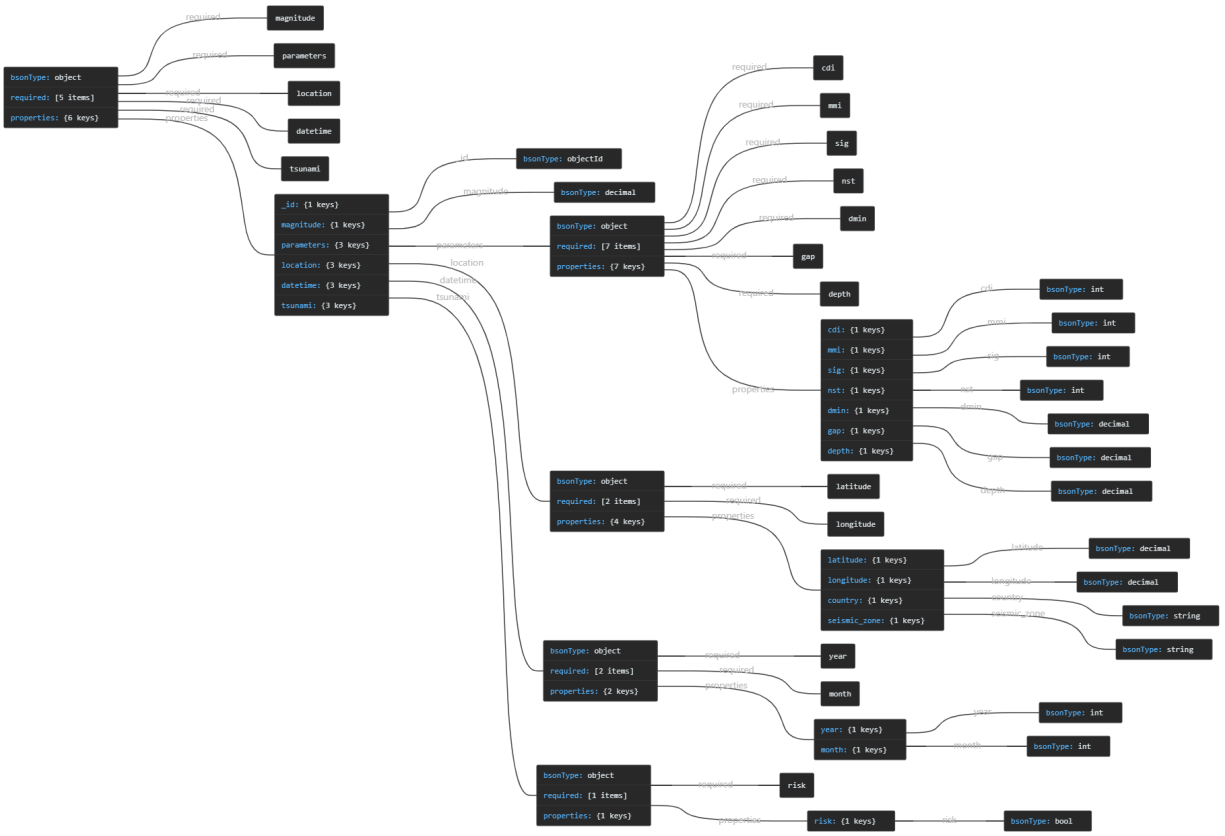
- **magnitude**: magnitud del terremoto (Mw u otras escalas unificadas).
- **parameters**: parámetros técnicos del evento (intensidad, número de estaciones, profundidad…).
- **location**: coordenadas geográficas y metadatos de ubicación.
- **datetime**: información temporal del evento.
- **tsunami**: indicador de riesgo de tsunami asociado.

Este diseño evita valores nulos, favorece la lectura de un evento completo en una única consulta y permite ampliaciones futuras sin modificar un esquema rígido.

A continuación se muestra el validador completo utilizado para la colección earthquakes:

JSON

```
{
  "bsonType": "object",
  "required": ["magnitude", "parameters", "location", "datetime", "tsunami"],
  "properties": {
    "_id": { "bsonType": "objectId" },
    "magnitude": { "bsonType": "decimal" },
    "parameters": {
      "bsonType": "object",
      "required": ["cdi", "mmi", "sig", "nst", "dmin", "gap", "depth"],
      "properties": {
        "cdi": { "bsonType": "int" },
        "mmi": { "bsonType": "int" },
        "sig": { "bsonType": "int" },
        "nst": { "bsonType": "int" },
        "dmin": { "bsonType": "decimal" },
        "gap": { "bsonType": "decimal" },
        "depth": { "bsonType": "decimal" }
      }
    },
    "location": {
      "bsonType": "object",
      "required": ["latitude", "longitude"],
      "properties": {
        "latitude": { "bsonType": "decimal" },
        "longitude": { "bsonType": "decimal" },
        "country": { "bsonType": "string" },
        "seismic_zone": { "bsonType": "string" }
      }
    },
    "datetime": {
      "bsonType": "object",
      "required": ["year", "month"],
      "properties": {
        "year": { "bsonType": "int" },
        "month": { "bsonType": "int" }
      }
    },
    "tsunami": {
      "bsonType": "object",
      "required": ["risk"],
      "properties": {
        "risk": { "bsonType": "bool" }
      }
    }
  }
}
```



El diseño propuesto ofrece un documento compacto en el que toda la información esencial del terremoto se integra en una sola estructura, optimizando las lecturas, y aplica una validación ligera que impone reglas básicas para evitar inconsistencias sin perder la flexibilidad del modelo documental. Además, organiza el contenido en bloques claramente separados :magnitud, ubicación, parámetros técnicos y riesgo se agrupan de forma coherente según su función. Finalmente, es un modelo extensible que permite añadir nuevos datos,sin afectar al resto de los documentos.

SCRIPT JS PARA LA CREACIÓN DE LA BD:

Tras definir el esquema de la BD para la colección earthquakes, mediante este scripts JavaScript ejecutado en el shell de MongoDB se crea la BD.

Este script crea la base de datos, elimina la colección si existe previamente y aplica el validador correspondiente.

JavaScript

```
const DB_NAME = "bigdata_practica";
const COLL = "earthquakes";
const dbRef = db.getSiblingDB(DB_NAME);

if (dbRef.getCollectionNames().includes(COLL)) {
  dbRef[COLL].drop();
}

// Crear colección con validador JSON Schema
dbRef.createCollection(COLL, {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["magnitude", "parameters", "location", "datetime", "tsunami"],
      properties: {
        _id: { bsonType: "objectId" },
        magnitude: { bsonType: "decimal" },
        parameters: {
          bsonType: "object",
          required: ["cdi", "mmi", "sig", "nst", "dmin", "gap", "depth"],
          properties: {
            cdi: { bsonType: "int" },
            mmi: { bsonType: "int" },
            sig: { bsonType: "int" },
            nst: { bsonType: "int" },
            dmin: { bsonType: "decimal" },
            gap: { bsonType: "decimal" },
            depth: { bsonType: "decimal" }
          }
        },
        location: {
          bsonType: "object",
          required: ["latitude", "longitude"],
          properties: {
            latitude: { bsonType: "decimal" },
            longitude: { bsonType: "decimal" },
            country: { bsonType: "string" },
```

```
            seismic_zone: { bsonType: "string" }
          }
        },
        datetime: {
          bsonType: "object",
          required: ["year", "month"],
          properties: {
            year: { bsonType: "int" },
            month: { bsonType: "int" }
          }
        },
        tsunami: {
          bsonType: "object",
          required: ["risk"],
          properties: {
            risk: { bsonType: "bool" }
          }
        }
      }
    }
  },
  validationAction: "error"
});


print(`Base de datos '${DB_NAME}' creada correctamente.`);
print(`Colección '${COLL}' creada con el esquema especificado.`);
```

El script crea la base de datos bigdata_practica y genera en ella la colección earthquakes aplicando un validador JSON Schema.
Antes de crearla, comprueba si la colección ya existe y, en ese caso, la elimina para asegurar una creación limpia.
Posteriormente define la estructura esperada para cada documento, incluyendo los bloques de magnitud, parámetros técnicos, ubicación, fecha y riesgo de tsunami, y establece la validación estricta para impedir inserciones que no cumplan el esquema.

3. Inserción de 100 registros

INSERCIÓN DE 100 REGISTROS Y GEOLOCALIZACIÓN

Para poblar la colección earthquakes se han insertado los primeros 100 registros del dataset original.
Debido a que los datos del CSV no incluyen información explícita sobre país ni zona sísmica, ha sido necesario crear un script en Python que:

1. Lee los datos del fichero CSV.
2. Enriquece cada registro determinando el país y la zona sísmica mediante una API de geocodificación inversa.
3. Convierte los valores numéricos a tipos válidos en MongoDB (por ejemplo, Decimal128).
4. Construye documentos válidos conforme al JSON Schema definido.
5. Inserta los registros en la colección mediante PyMongo.

Dado que el CSV proporciona únicamente las coordenadas,para la obtención de los campos location.country y location.seismic_zone, se ha hecho uso de la API de geolocalización Nominatim, incluida en la librería geopy.

Mediante geocodificación inversa, Nominatim permite obtener el país a partir de coordenadas GPS y para la zona sísmica se aplica una heurística sencilla basada en latitud:si la latitud se encuentra entre ±20° y ±40°, se clasifica como Zona de subducción,en caso contrario, como Zona continental.

```python
import pandas as pd
from pymongo import MongoClient
from bson.decimal128 import Decimal128
from geopy.geocoders import Nominatim
import time


MONGO_URI = "mongodb://root:12345@localhost:27017/admin"
DB_NAME = "bigdata_practica"
COLLECTION = "earthquakes"
USE_API = True


csv_file = "./data/earthquakes.csv"
df = pd.read_csv(csv_file)
df = df.head(100)

print(f" CSV cargado correctamente con {len(df)} registros.")
print(f" Columnas detectadas: {list(df.columns)}")
```

```python
# Usa un user_agent con tu nombre, correo o proyecto (Nominatim lo exige)
geolocator = Nominatim(
    user_agent="BD_EARTHQUAKES",  # cámbialo si quieres
    timeout=10
)

# Cache local para evitar repetir consultas de la misma zona
cache_paises = {}

def obtener_pais_y_zona(lat, lon, i):
    """Obtiene el país y la zona sísmica de unas coordenadas"""
    coord_key = (round(lat, 2), round(lon, 2))
    if coord_key in cache_paises:
        return cache_paises[coord_key]

    if not USE_API:
        return "Desconocido", "Zona simulada"

    try:
        print(f" [{i}] Consultando país para ({lat}, {lon}) ...")
        location = geolocator.reverse((lat, lon), language="en", zoom=5,
addressdetails=True)

        if location and "country" in location.raw.get("address", {}):
            country = location.raw["address"]["country"]
        else:
            country = "Desconocido"

        # Guarda en cache para reutilizar
        cache_paises[coord_key] = (country, "Zona de subducción" if 20 < abs(lat)
< 40 else "Zona continental")
        return cache_paises[coord_key]

    except Exception as e:
        print(f"[{i}] Error geolocalizando: {e}")
        return "Desconocido", "Zona continental"


def preparar_registro(row, i):
    lat = float(row["latitude"])
    lon = float(row["longitude"])
    country, seismic_zone = obtener_pais_y_zona(lat, lon, i)

    return {
        "magnitude": Decimal128(str(row["magnitude"])),
        "parameters": {
```

```
            "cdi": int(row["cdi"]),
            "mmi": int(row["mmi"]),
            "sig": int(row["sig"]),
            "nst": int(row["nst"]),
            "dmin": Decimal128(str(row["dmin"])),
            "gap": Decimal128(str(row["gap"])),
            "depth": Decimal128(str(row["depth"]))
        },
        "location": {
            "latitude": Decimal128(str(lat)),
            "longitude": Decimal128(str(lon)),
            "country": country,
            "seismic_zone": seismic_zone
        },
        "datetime": {
            "year": int(row["Year"]),
            "month": int(row["Month"])
        },
        "tsunami": {
            "risk": bool(row["tsunami"])
        }
    }


client = MongoClient(MONGO_URI)
db = client[DB_NAME]
collection = db[COLLECTION]

documentos = []

print("\n Procesando e insertando registros...\n")
for i, fila in df.iterrows():
    print(f" [{i+1}/{len(df)}] Procesando registro...")
    try:
        doc = preparar_registro(fila, i+1)
        documentos.append(doc)
        print(f"  [{i+1}] País={doc['location']['country']},
Zona={doc['location']['seismic_zone']}")
    except Exception as e:
        print(f" [{i+1}] Error preparando registro: {e}")
    if USE_API:
        time.sleep(1.2)

print("\n Insertando en MongoDB...")
if documentos:
    result = collection.insert_many(documentos)
    print(f" Se insertaron {len(result.inserted_ids)} registros en
'{DB_NAME}.{COLLECTION}' correctamente.")
```

```
else:
    print(" No se generaron documentos para insertar.")

client.close()
```

Para la ejecución de este simplemente con el Docker levantando el contenedor de MongoDB, basta con instalar las dependencias del proyecto y ejecutar el script.

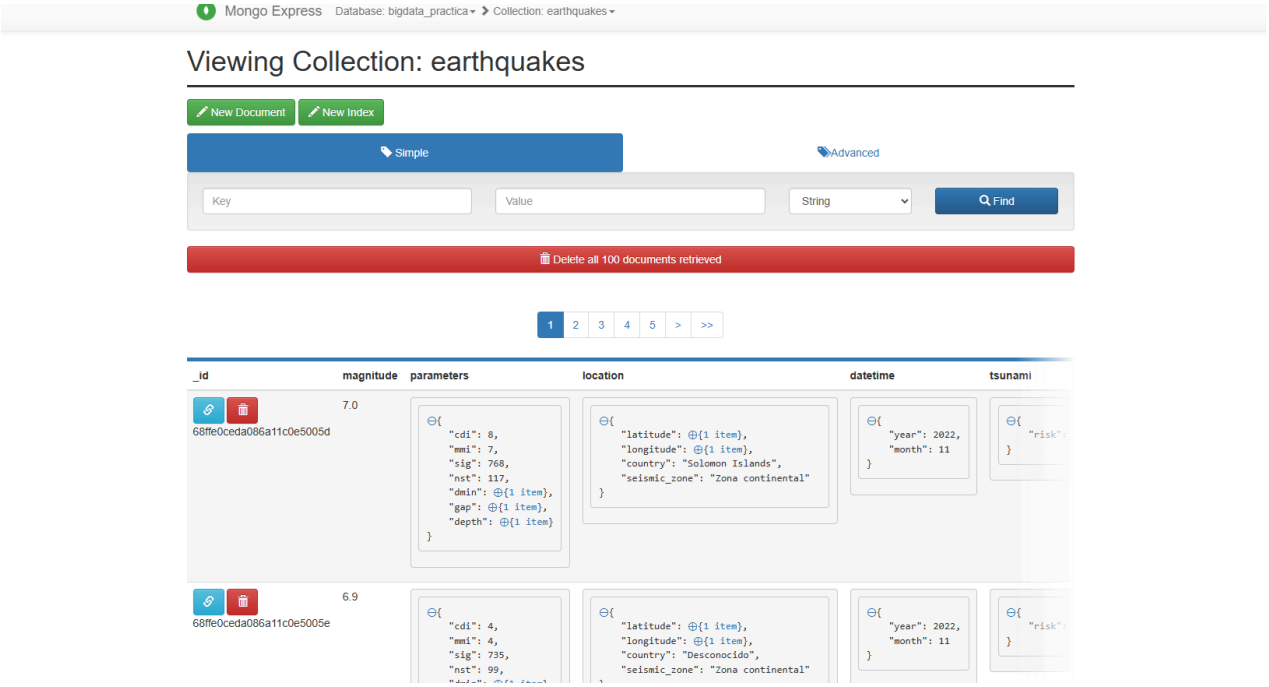
Primeramente,se comprueba cuántos documentos hay antes de ejecutar el script:

```
Shell
bigdata_practica> db.earthquakes.countDocuments()
0
```

Y tras ejecutar el script dentro del contenedor

```
Shell
bigdata_practica> db.earthquakes.countDocuments()
100
```

Mongo Express:



Collection Stats

Documents	100
Total doc size	34.46 KB
Average doc size	352 Bytes
Pre-allocated size	24 KB
Indexes	1
Total index size	20 KB
Padding factor	
Extents	

Desde la UI de Mongo Express se muestran 100 documentos en la colección earthquakes.

4. Sentencias e modificación de documentos

Para este apartado se han seleccionado dos registros de la colección earthquakes y se ha modificado el campo location.seismic_zone convirtiendo su valor a mayúsculas, tal como solicita la práctica.

Antes de realizar las actualizaciones se ejecutó una consulta de verificación para identificar los documentos objetivo.

1º Consultamos antes de la conversión:

La siguiente consulta devuelve los documentos cuyo país es *Solomon Islands* o *Viti*, mostrando únicamente el país y la zona sísmica:

```
Shell
bigdata_practica> db.earthquakes.find(
  { "location.country": { $in: ["Solomon Islands", "Viti"] } },
  { _id: 0, "location.country": 1, "location.seismic_zone": 1 }
)
[
  {
    location: { country: 'Solomon Islands', seismic_zone: 'Zona continental' }
  },
  { location: { country: 'Viti', seismic_zone: 'Zona de subducción' } }
]
```

2º Realizamos la conversión:

Solomon Islands → convertir 'Zona continental' a 'ZONA CONTINENTAL'

```
Shell
bigdata_practica> db.earthquakes.updateOne(
   {
    "location.country": "Solomon Islands",
    "location.seismic_zone": "Zona continental"
  },
  {
    $set: { "location.seismic_zone": "ZONA CONTINENTAL" }
  }
)
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
```

```
  modifiedCount: 1,
  upsertedCount: 0
}
```

Viti → convertir 'Zona de subducción' a 'ZONA DE SUBDUCCIÓN'

```Shell
bigdata_practica> db.earthquakes.updateOne(
   {
    "location.country": "Viti",
    "location.seismic_zone": "Zona de subducción"
  },
  {
    $set: { "location.seismic_zone": "ZONA DE SUBDUCCIÓN" }
  }
)
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Ambas operaciones han coincidido con un documento (matchedCount: 1) y lo han modificado correctamente (modifiedCount: 1).

Tras realizar las actualizaciones, se vuelve a ejecutar la consulta inicial para verificar los cambios:

```Shell
db.earthquakes.find(
  { "location.country": { $in: ["Solomon Islands", "Viti"] } },
  { _id: 0, "location.country": 1, "location.seismic_zone": 1 }
)

[
  {
    location: { country: 'Solomon Islands', seismic_zone: 'ZONA CONTINENTAL' }
  },
```

```
  { location: { country: 'Viti', seismic_zone: 'ZONA DE SUBDUCCIÓN' } }
]
```

5. Definir las siguientes consultas

Para la ejecución de las consultas solicitadas se ha realizado un script *consultas_earthquakes.py* en Python que automatiza el proceso de filtrado, mide el tiempo de respuesta de cada consulta y genera un informe completo en el archivo *resultados_earthquakes.txt.*

Estas consultas permiten validar tanto la estructura del modelo como el rendimiento básico de lecturas sobre la colección earthquakes.

El script desarrollado (*consultas_earthquakes.py*) ejecuta todas las consultas, mide su tiempo de ejecución en milisegundos y genera un archivo de salida con: el resumen de tiempos, el número de documentos devueltos, y el detalle de cada documento encontrado.

Python

```python
from pymongo import MongoClient
from tabulate import tabulate
import time
import pprint

MONGO_URI = "mongodb://root:12345@localhost:27017/admin"
DB_NAME = "bigdata_practica"
COLLECTION_NAME = "earthquakes"

# Fichero donde se guardará toda la salida
OUTPUT_FILE = "resultados_earthquakes.txt"

client = MongoClient(MONGO_URI)
db = client[DB_NAME]
collection = db[COLLECTION_NAME]

# País concreto para las consultas B-i y B-ii
PAIS_CONCRETO = "Japan"  # cámbialo si quieres

consultas = {
    # A-i) Año del evento mayor de 2015
    "A-i) Terremotos con año > 2015": {
        "datetime.year": {"$gt": 2015},
    },
```

```
    # A-ii) País que empiece por “Japa”
    "A-ii) Terremotos en países que empiezan por 'Japa'": {
        "location.country": {"$regex": "^Japa", "$options": "i"},
    },

    # B-i) País concreto, magnitud > 7.0
    "B-i) País concreto con magnitud > 7.0": {
        "location.country": PAIS_CONCRETO,
        "magnitude": {"$gt": 7.0},
    },

    # B-ii) País concreto, riesgo de tsunami
    "B-ii) País concreto con riesgo de tsunami": {
        "location.country": PAIS_CONCRETO,
        "tsunami.risk": True,
    },
}

resultados = []
detalles_resultados = {}

with open(OUTPUT_FILE, "w", encoding="utf-8") as f:

    def log(msg=""):
        """Imprime en terminal y guarda también en el fichero de salida."""
        print(msg)
        f.write(str(msg) + "\n")

    log("\nINICIO DE EJECUCIÓN DE CONSULTAS SOBRE
'bigdata_practica.earthquakes'\n")
    log("=" * 90)

    for nombre, filtro in consultas.items():
        log(f"\nEjecutando consulta: {nombre}")
        inicio = time.time()
        resultados_cursor = list(collection.find(filtro, {"_id": 0}))
        fin = time.time()
        tiempo_ms = (fin - inicio) * 1000  # en milisegundos

        resultados.append([nombre, round(tiempo_ms, 2), len(resultados_cursor)])
        detalles_resultados[nombre] = resultados_cursor

        log(f"   Documentos devueltos: {len(resultados_cursor)} | Tiempo:
{round(tiempo_ms, 2)} ms")

    log("\n" + "=" * 90)

    log("\nTABLA RESUMEN DE TIEMPOS DE EJECUCIÓN:\n")
```

```
    tabla = tabulate(resultados, headers=["Consulta", "Tiempo (ms)", "Documentos
devueltos"], tablefmt="fancy_grid")
    log(tabla)

    log("\nRESULTADOS DETALLADOS DE LAS CONSULTAS:\n")
    for nombre, docs in detalles_resultados.items():
        log("\n" + "-" * 90)
        log(nombre)
        log("-" * 90)
        if docs:
            for doc in docs:
                texto_doc = pprint.pformat(doc, sort_dicts=False)
                log(texto_doc)
                log("-" * 60)
        else:
            log("No se encontraron resultados para esta consulta.")
        log("-" * 90)

    #
    log(f"\nSalida completa guardada en: {OUTPUT_FILE}")

print(f"\n(Info) También se ha guardado en el archivo: {OUTPUT_FILE}")
```

RESULTADO

El resultado además de devolverlo por consola, tambien se almacena en un fichero .txt. En el bloque que se presenta a continuación no se muestra el resultado completo, sino un resumen,eliminando parte del resultado para no mostrarlo por completo.El resultado completo de las consultas se encuentra en *resultados_earthquakes.txt* .

Shell

```
INICIO DE EJECUCIÓN DE CONSULTAS SOBRE 'bigdata_practica.earthquakes'

================================================================================
=========

Ejecutando consulta: A-i) Terremotos con año > 2015
   Documentos devueltos: 100 | Tiempo: 51.04 ms

Ejecutando consulta: A-ii) Terremotos en países que empiezan por 'Japa'
   Documentos devueltos: 2 | Tiempo: 3.36 ms

Ejecutando consulta: B-i) País concreto con magnitud > 7.0
   Documentos devueltos: 0 | Tiempo: 2.77 ms

Ejecutando consulta: B-ii) País concreto con riesgo de tsunami
   Documentos devueltos: 2 | Tiempo: 3.13 ms

================================================================================
=========

TABLA RESUMEN DE TIEMPOS DE EJECUCIÓN:

╒═══════════════════════════════════════════════════════╤═══════════════╤═══════════
═══════════╕
| Consulta                                              |   Tiempo (ms) |
Documentos devueltos |
╞═══════════════════════════════════════════════════════╪═══════════════╪═══════════
═══════════╡
| A-i) Terremotos con año > 2015                        |         51.04 |
100 |
├───────────────────────────────────────────────────────┼───────────────┼───────────
──────────┤
| A-ii) Terremotos en países que empiezan por 'Japa' |          3.36 |
2 |
├───────────────────────────────────────────────────────┼───────────────┼───────────
──────────┤
| B-i) País concreto con magnitud > 7.0              |          2.77 |
0 |
```

```
├──────────────────────────────────────────────────────┼──────────────┼──────────
──────────┤
| B-ii) País concreto con riesgo de tsunami          |         3.13 |
2 |
└──────────────────────────────────────────────────────┴──────────────┴──────────
══════════┘

RESULTADOS DETALLADOS DE LAS CONSULTAS:


--------------------------------------------------------------------------------
---------
A-i) Terremotos con año > 2015
--------------------------------------------------------------------------------
---------
{'magnitude': Decimal128('7.0'),
 'parameters': {'cdi': 8,
                'mmi': 7,
                'sig': 768,
                'nst': 117,
                'dmin': Decimal128('0.509'),
                'gap': Decimal128('17.0'),
                'depth': Decimal128('14.0')},
 'location': {'latitude': Decimal128('-9.7963'),
              'longitude': Decimal128('159.596'),
              'country': 'Solomon Islands',
              'seismic_zone': 'Zona continental'},
 'datetime': {'year': 2022, 'month': 11},
 'tsunami': {'risk': True}}
------------------------------------------------------------
{'magnitude': Decimal128('6.9'),
 'parameters': {'cdi': 4,
                'mmi': 4,
                'sig': 735,
                'nst': 99,
                'dmin': Decimal128('2.229'),
                'gap': Decimal128('34.0'),
                'depth': Decimal128('25.0')},
 'location': {'latitude': Decimal128('-4.9559'),
              'longitude': Decimal128('100.738'),
              'country': 'Desconocido',
              'seismic_zone': 'Zona continental'},
 'datetime': {'year': 2022, 'month': 11},
 'tsunami': {'risk': False}}
------------------------------------------------------------
...
--------------------------------------------------------------------------------
---------
```

```
A-ii) Terremotos en países que empiezan por 'Japa'
--------------------------------------------------------------------------------
---------
{'magnitude': Decimal128('6.9'),
 'parameters': {'cdi': 7,
                'mmi': 6,
                'sig': 919,
                'nst': 0,
                'dmin': Decimal128('2.619'),
                'gap': Decimal128('35.0'),
                'depth': Decimal128('43.0')},
 'location': {'latitude': Decimal128('38.2296'),
              'longitude': Decimal128('141.665'),
              'country': 'Japan',
              'seismic_zone': 'Zona de subducción'},
 'datetime': {'year': 2021, 'month': 5},
 'tsunami': {'risk': True}}
------------------------------------------------------------
{'magnitude': Decimal128('7.0'),
 'parameters': {'cdi': 7,
                'mmi': 7,
                'sig': 1052,
                'nst': 0,
                'dmin': Decimal128('2.378'),
                'gap': Decimal128('35.0'),
                'depth': Decimal128('43.0')},
 'location': {'latitude': Decimal128('38.4752'),
              'longitude': Decimal128('141.607'),
              'country': 'Japan',
              'seismic_zone': 'Zona de subducción'},
 'datetime': {'year': 2021, 'month': 3},
 'tsunami': {'risk': True}}
------------------------------------------------------------
--------------------------------------------------------------------------------
---------

--------------------------------------------------------------------------------
---------
B-i) País concreto con magnitud > 7.0
--------------------------------------------------------------------------------
---------
No se encontraron resultados para esta consulta.
--------------------------------------------------------------------------------
---------

--------------------------------------------------------------------------------
---------
B-ii) País concreto con riesgo de tsunami
```

```
--------------------------------------------------------------------------------
---------
{'magnitude': Decimal128('6.9'),
 'parameters': {'cdi': 7,
                'mmi': 6,
                'sig': 919,
                'nst': 0,
                'dmin': Decimal128('2.619'),
                'gap': Decimal128('35.0'),
                'depth': Decimal128('43.0')},
 'location': {'latitude': Decimal128('38.2296'),
              'longitude': Decimal128('141.665'),
              'country': 'Japan',
              'seismic_zone': 'Zona de subducción'},
 'datetime': {'year': 2021, 'month': 5},
 'tsunami': {'risk': True}}
------------------------------------------------------------
{'magnitude': Decimal128('7.0'),
 'parameters': {'cdi': 7,
                'mmi': 7,
                'sig': 1052,
                'nst': 0,
                'dmin': Decimal128('2.378'),
                'gap': Decimal128('35.0'),
                'depth': Decimal128('43.0')},
 'location': {'latitude': Decimal128('38.4752'),
              'longitude': Decimal128('141.607'),
              'country': 'Japan',
              'seismic_zone': 'Zona de subducción'},
 'datetime': {'year': 2021, 'month': 3},
 'tsunami': {'risk': True}}
------------------------------------------------------------
--------------------------------------------------------------------------------
---------

Salida completa guardada en: resultados_earthquakes.txt
```

TIEMPOS DE EJECUCIÓN:

	CONSULTA	Tiempo (ms)	Documentos devueltos
1	Terremotos con año > 2015	51.04	100
2	Terremotos en países que empiezan por 'Japa'	3.36	2
3	País 'Japan' y magnitud > 7.0	2.77	0
4	País 'Japan', con riesgo de tsunami	3.13	2

Análisis de los resultados

Rendimiento:

- La consulta más pesada es la A-i (año > 2015) porque devuelve los 100 documentos completos y realiza una exploración total de la colección.
- El resto de consultas son extremadamente rápidas (2–3 ms), ya que aplican filtros más restrictivos y retornan muy pocos documentos.

Hallazgos:

- Solo dos documentos coinciden con países cuyo nombre empieza por “Japa”, ambos correspondientes a Japón.
- Para el país Japan, no existe ningún terremoto con magnitud mayor a 7.0 en los primeros 100 registros procesados.
- Sin embargo, sí existen dos terremotos en Japón con riesgo de tsunami, correspondientes a los años 2021.

Posible mejora del rendimiento
Si la colección creciera (miles o millones de registros), sería recomendable crear índices:

```javascript
db.earthquakes.createIndex({ "datetime.year": 1 })
db.earthquakes.createIndex({ "location.country": 1 })
db.earthquakes.createIndex({ "magnitude": -1 })
db.earthquakes.createIndex({ "tsunami.risk": 1 })
```