Mocoa, Fecha (16 – 10 – 2025)
Programa: Ingeniería de sistemas
Syllabus: Bases de datos no relacionales
Semestre: 7
Docente: Fredy Jacanamijoy
3204414573 – fredy.jacanamijoy@itp.edu.co

Integración de Flask, MongoDB y Redis

Este tutorial explica cómo integrar Flask (framework web en Python) con MongoDB (base de datos NoSQL) y Redis (base de datos en memoria) para crear una API eficiente que use caché para mejorar el rendimiento.

1. Instalación de componentes

Asegúrate de tener Python, MongoDB y Redis instalados.
Para instalar las dependencias de Python, usa:

```
1. pip install flask pymongo redis
```

2. Conexión entre Flask, MongoDB y Redis

El siguiente código muestra cómo conectar Flask con MongoDB y Redis, seleccionando una base de datos específica de Redis (por ejemplo, db=2).

```python
from flask import Flask, jsonify
from pymongo import MongoClient
import redis

app = Flask(__name__)

# Conexión a MongoDB
mongo_client = MongoClient("mongodb://localhost:27017/")
db_mongo = mongo_client["mi_bd"]
usuarios_col = db_mongo["usuarios"]

# Conexión a Redis (usando base de datos 2)
cache = redis.Redis(
    host="localhost",
    port=6379,
    db=2,  # Base de datos seleccionada
    decode_responses=True
)

@app.route("/usuario/<nombre>")
def obtener_usuario(nombre):
    if (usuario_cache := cache.get(nombre)):
        return jsonify({"origen": "redis", "datos": eval(usuario_cache)})

    usuario = usuarios_col.find_one({"nombre": nombre}, {"_id": 0})
    if usuario:
        cache.set(nombre, str(usuario), ex=60)  # Guarda en cache 60s
        return jsonify({"origen": "mongo", "datos": usuario})
    return jsonify({"error": "Usuario no encontrado"}), 404

if __name__ == "__main__":
    app.run(debug=True)
```

3. Flujo de funcionamiento

1. Flask recibe una solicitud de usuario (por ejemplo, /usuario/Ana).
2. Flask busca el dato en Redis (db=2).
3. Si no está en Redis, consulta MongoDB.
4. Luego guarda el resultado en Redis para futuras consultas.
5. En la siguiente solicitud, Redis responde directamente (más rápido).

4. Verificación desde la consola de Redis

Puedes verificar que los datos se hayan guardado en Redis usando su consola interactiva:

```
2. redis-cli
3. SELECT 2
4. KEYS *
5. GET Ana
6. TTL Ana
```

- `KEYS *` muestra todas las claves almacenadas.
- `GET <clave>` devuelve el valor asociado.
- `TTL <clave>` indica cuántos segundos quedan antes de que expire.

5. Ejemplo de respuesta JSON

Si los datos provienen de Redis:

```
{"origen": "redis", "datos": {"nombre": "Ana", "edad": 30}}
```

Si los datos provienen de MongoDB:

```
{"origen": "mongo", "datos": {"nombre": "Ana", "edad": 30}}
```

6. Visualización del cache en Redis

También puedes ver en tiempo real las claves almacenadas en Redis con herramientas gráficas como:
- RedisInsight (oficial de Redis)
- Mediante la CLI con `MONITOR` para observar comandos en tiempo real.

EJEMPLOS PARA INGRESO DE DATOS A MONGO

```
db.usuarios.insertOne({
          nombre: "Ana",
          edad: 30,
          ciudad: "Bogotá"
        })
```

```
db.usuarios.insertMany([
          { nombre: "Carlos", edad: 28, ciudad: "Cali" },
          { nombre: "Laura", edad: 35, ciudad: "Medellín" },
          { nombre: "Diego", edad: 40, ciudad: "Pasto" }
        ])
```