Proyecto Final: Sistema RAG NoSQL con MongoDB

Asignatura: Bases de Datos No Relacionales

1. Información General del Proyecto

Descripción

Los estudiantes desarrollarán un sistema de Recuperación y Generación Aumentada (RAG) utilizando MongoDB como base de datos principal. El sistema aprovechará las capacidades NoSQL de MongoDB para almacenar documentos flexibles, implementar búsqueda vectorial mediante Atlas Vector Search y construir un pipeline RAG completo que integre un LLM accesible mediante API gratuita.

El proyecto combina el modelado flexible de documentos NoSQL con técnicas modernas de procesamiento de lenguaje natural y búsqueda semántica, culminando en un sistema inteligente capaz de responder preguntas complejas sobre una base de conocimiento diversa.

Requisito mínimo: El sistema debe procesar y vectorizar al menos texto e imágenes, soportar consultas híbridas (filtros tradicionales + similaridad vectorial) y generar respuestas contextualizadas mediante RAG.

2. Objetivos de Aprendizaje

Objetivos Principales

- Diseñar esquemas NoSQL flexibles utilizando estrategias de embedding vs. referencing según el caso de uso
- Implementar modelado de documentos que optimice consultas frecuentes y minimice operaciones costosas
- Construir pipelines de agregación complejos usando el Aggregation Framework de MongoDB
- Configurar y utilizar índices especializados (texto, compuestos, vectoriales)
- Procesar y vectorizar contenido multimodal (texto, imágenes, metadatos)
- Implementar búsqueda híbrida combinando filtros tradicionales con similaridad vectorial

- Integrar un pipeline RAG completo con LLM gratuito y técnicas de context retrieval

Infraestructura

Opción A: MongoDB Atlas

- Cluster gratuito M0 (512MB)
- Acceso nativo a Atlas Vector Search
- Atlas Search para búsqueda de texto completo
- Monitoreo integrado

Opción B: MongoDB Local

- Docker Compose con MongoDB 7.0+
- Configuración manual para búsqueda vectorial
- Mayor control pero menor funcionalidad integrada

3. Alcance Técnico Mínimo

- **Diseño de Datos NoSQL:** Definir las colecciones y su estructura.
  - **Embedding vs. Referencing**
  - **Embedded**: Metadatos pequeños, historial de consultas del usuario
  - **Referenced**: Imágenes grandes, documentos compartidos entre usuarios
  - **Híbrido**: Documento principal con referencias a imágenes pero embeddings de metadatos

- **Ingesta y Limpieza de Datos**
  - Scripts en Python o Node.js para cargar datos desde JSON/CSV.
  - Validación básica de esquema usando MongoDB Schema Validation.

- **Consultas y Agregación**
  - Uso de Aggregation Pipeline con $match, $project, $group, $lookup.

    - Si se usa Atlas Search: $search con operadores text, vectorSearch, compound.

- **Índices**
    - Índice compuesto en { "fecha": 1, "idioma": 1 }.
    - Índice de texto en "contenido_texto".
    - Si se usa Vector Search: índice knnVector en el campo de embeddings.

- **Funciones/Triggers (Opcional)**
    - Usar Atlas Triggers para mantener campos derivados o auditoría.

- **API Mínima**
    - Desarrollar una API con Node.js/Express o Python/FastAPI con endpoints:
        - POST /search → búsqueda híbrida o vectorial.
        - POST /rag → genera respuesta usando contexto de MongoDB + LLM.

- **Pipeline RAG**
    - **Embeddings:** Usar modelo como all-MiniLM-L6-v2 para texto.
    - **Almacenamiento:** Guardar embeddings en una colección con knnVector.
    - **Recuperación:** Usar $vectorSearch o $search con filtros por metadatos.
    - **LLM:** Integrar con API gratuita (Ej.: Groq + Llama 3.1).
    - **Prompt Engineering:** Incluir contexto recuperado en el prompt.

Entregas del Proyecto

Entrega 1: Diseño y Configuración

Entregables

1. **Documento de Análisis**
    - Universo del discurso y análisis de requerimientos

- Justificación de decisiones de modelado NoSQL
- Comparación embedding vs. referencing

2. **Diseño de Esquema NoSQL**
    - Definición de colecciones con ejemplos de documentos
    - Estrategias de indexing planificadas
    - Schema validation rules
3. **Configuración de Entorno**
    - Cluster MongoDB configurado (Atlas o local)
    - Scripts de inicialización
    - Conexión verificada desde aplicación
4. **Dataset Preparado**
    - Mínimo 100 documentos de texto
    - Mínimo 50 imágenes asociadas
    - Formato JSON válido para carga

Entrega 2: Implementación RAG Completa (Semana 12)

Entregables

1. **Sistema RAG Funcional**
    - Pipeline completo de ingesta con embeddings
    - API REST con endpoints documentados
    - Integración con LLM gratuito configurada
2. **Demostración de Consultas**
    - 5 consultas de ejemplo con evidencias
    - Métricas de rendimiento (tiempo de respuesta, precisión)
    - Casos de uso texto-texto, imagen-imagen, multimodal
3. **Código Fuente Completo**

    - Repositorio Git con estructura clara
    - README con instrucciones de instalación
    - Scripts de carga y configuración
4. **Informe Final**
    - Arquitectura técnica implementada
    - Resultados y evaluación del sistema
    - Lecciones aprendidas y recomendaciones
    - Comparación con enfoque relacional

Casos de Prueba Obligatorios

1. **Búsqueda Semántica**: "¿Qué documentos hablan sobre sostenibilidad ambiental?"
2. **Filtros Híbridos**: "Artículos en inglés sobre tecnología publicados en 2024"
3. **Búsqueda Multimodal**: "Imágenes similares a esta foto de arquitectura"
4. **RAG Complejo**: "Explica las principales tendencias en energías renovables según los documentos"

Tecnologías Recomendadas

Base de Datos

- MongoDB Atlas (Vector Search incluido) o MongoDB 7.0+ local
- Compass para exploración visual

ML y Embeddings

- **Texto**: sentence-transformers (all-MiniLM-L6-v2)
- **Imágenes**: OpenCLIP o transformers (clip-vit-base-patch32)
- **Multimodal**: CLIP para búsquedas texto↔imagen

APIs de LLM Gratuitas

- **Groq API**: Llama 3.1, Mixtral (rápido, cuota generosa)

- **Hugging Face Inference API**: Modelos open-source
- **OpenAI Free Tier**: GPT-3.5-turbo (limitado)
- **Ollama**: Local, sin límites de API