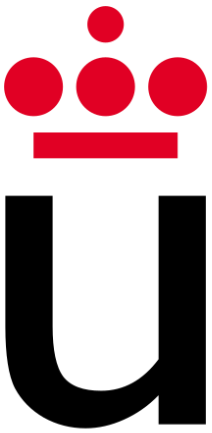

Universidad
Rey Juan Carlos

Bases de Datos NO Relacionales
PRÁCTICA TEMA 5 y 6

GRADO EN CIENCIA E INGENIERÍA DE DATOS
CURSO:2025/2026

Cristian Gómez Macías

Material docente en abierto de la Universidad Rey Juan Carlos

TABLA DE CONTENIDO

Práctica Temas 5 (MongoDB) y 6 (Neo4j)

Contexto

La empresa MetroCampus Data Services (MDS) desarrolla soluciones de análisis de datos para sistemas de transporte urbano y planificación académica. En el curso 2025/2026 se plantea un proyecto centrado en la ciudad de Madrid cuyo objetivo es estudiar cómo se relacionan:

- La red de Metro de Madrid (líneas y estaciones). Link: https://www.metromadrid.es/es/viaja-en-metro/plano-de-metro-de-madrid
- Las correspondencias con Renfe existentes en determinadas estaciones de Metro.
- Los campus universitarios públicos (UCM, UPM, UC3M, URJC) y los estudios de Grado y Máster que se imparten en ellos.

Se pretende disponer de un sistema de información que permita responder preguntas como:

- ¿Qué líneas y estaciones de Metro dan mejor servicio a los distintos campus?
- ¿Qué estaciones pueden considerarse “hubs universitarios”?
- ¿Qué trayectos en Metro resultan más adecuados para cursar un determinado Grado o Máster, según la estación de origen de la persona interesada?

Para cubrir tanto la parte de modelo de datos operativo como la parte de análisis de rutas y relaciones, se ha decidido combinar:

- Una base de datos documental basada en MongoDB. Se recomienda usar MongoDB Atlas.
- Una base de datos de grafos basada en Neo4j.

La práctica se enmarca en este proyecto y se centra en el diseño del modelado, carga y explotación de los datos mencionados, utilizando ambas tecnologías de manera complementaria.

Objetivo

Con esta práctica se pretende cubrir los siguientes aspectos vistos en los temas 5 y 6 de la asignatura:

- Diseño de un modelo documental en MongoDB para representar líneas, estaciones, campus y estudios.
- Uso de operaciones CRUD y agregaciones para consultas típicas del dominio.
- Diseño de un modelo de grafo en Neo4j a partir del mismo dominio (líneas, estaciones, campus y estudios).
- Definición y ejecución de consultas en Cypher sobre estructuras de grafos.
- Implementación de consultas de rutas y recomendación de campus según trayectos en Metro.
- Comparación razonada del papel de MongoDB y Neo4j en un sistema real basado en datos no relacionales.

Requisitos funcionales

Las partes de cada práctica deben contener al menos (deben cumplir):

Parte A — MongoDB

1. **Modelo de datos**

Definir un esquema documental que incluya, al menos, las colecciones:

- lineas (líneas de Metro).
- estaciones (estaciones de Metro).
- campus (campus universitarios públicos).
- estudios (si no se embeben en campus).

Justificar las decisiones de:

- Embebido vs referencias (lineas–estaciones, campus–estudios).
- Representación de estaciones cercanas a campus.
- Representación de la correspondencia con Renfe.

2. **Datos y carga**

Población de datos realista con, como mínimo:

- 4 líneas de Metro.
- 40–60 estaciones con:
    - Algunas estaciones que pertenezcan a varias líneas.
    - Varias estaciones con correspondencia Renfe (como información adicional).

Campus de UCM, UPM, UC3M y URJC.

Para cada campus:

- Estaciones de Metro cercanas (principal y alternativas).
- Estudios de GRADO y de MÁSTER (distribuidos entre campus y universidades).

Entrega de scripts o ficheros de inserción (JSON, scripts .js, etc.).

3. **Operaciones CRUD**

Inserciones, actualizaciones y borrados significativos sobre:

- Líneas de Metro.
- Estaciones.
- Campus.
- Estudios (Grado/Máster).

Documentación de las operaciones más importantes realizadas (comandos y breve explicación, pero no todos los comandos).

4. **Consultas de lectura**

Consultas para:

- Listar estaciones de una línea en orden de paso.
- Obtener estaciones con correspondencia Renfe.
- Obtener estaciones accesibles por zona tarifaria.

- Listar campus por universidad.
- Listar campus asociados a una estación principal.
- Listar estudios de GRADO por nombre, indicando campus y universidad.

5. **Agregaciones**

Pipelines aggregate() que permitan:

- Calcular el número de estaciones por línea.
- Calcular cuántas estaciones "universitarias" existen por zona tarifaria.
- Calcular cuántos estudios de GRADO y de MÁSTER ofrece cada universidad.
- Realizar una comparación simplificada de trayectos en una misma línea mediante un campo indiceEnLinea.

6. **Índices e impacto**

Creación de índices adecuados (por ejemplo, sobre estaciones.nombre, estaciones.tieneRenfe, campus.universidad, campus.estudios.nombre).

Parte B — Neo4j

1. Modelo de grafo

Definir nodos:

- `:Linea`
- `:Estacion`
- `:Campus`
- `:Estudio`

Definir relaciones:

`(:Linea)-[:TIENE_ESTACION {orden}]->(:Estacion)`

`(:Estacion)-[:SIGUIENTE {lineaId}]->(:Estacion)` para estaciones consecutivas en una línea.

`(:Campus)-[:CERCANA {minutos, rol}]->(:Estacion)` (principal / alternativa).

`(:Campus)-[:OFRECE]->(:Estudio)`

Opcional: `(:Estacion)-[:TRANSBORDO]->(:Estacion)` para correspondencias entre líneas.

Representar la información de correspondencias Renfe, como mínimo, mediante propiedades en nodos `:Estacion` (por ejemplo, `tieneRenfe`, `nombreEstacionRenfe`).

2. **Carga de datos**

Creación de nodos y relaciones a partir de los datos de MongoDB (directamente o vía ficheros intermedios).

Uso de UNWIND y MERGE en los scripts de carga para evitar duplicados.

Verificación del número de nodos por tipo y de relaciones totales.

3. **Consultas en lenguaje Cypher:**
   A. **Consultas estructurales**

   Consultas que obtengan:

   - Estaciones de una línea en orden de recorrido.
   - Estaciones que son hubs universitarios (relacionadas con más de un campus).
   - Estaciones que tienen correspondencia Renfe y dan servicio a algún campus.

   B. **Consultas sobre campus y estudios**

   Consultas para:

   - Localizar los campus que ofrecen un estudio de GRADO concreto.
   - Calcular, para cada universidad, el número de estudios de GRADO y de MÁSTER.
   - Localizar estudios de MÁSTER de una rama determinada y sus campus.

   C. **Consultas de rutas**

Consultas de caminos como el camino más corto, el más largo, etc. para:

- Dada una estación de origen y un campus concreto:
    - Encontrar el camino más corto en número de estaciones desde la estacion de origen hasta cualquier estación `:Estacion` conectada al campus por `:CERCANA` (u otra relación que se defina)
    - Utilizar **shortestPath** (https://neo4j.com/docs/cypher-manual/current/patterns/shortest-paths/) y las siguientes relaciones `:SIGUIENTE` (y `:TRANSBORDO` si se han definido
- Para un estudio de GRADO concreto:
    - Encontrar todos los campus que ofrecen ese estudio
    - Para cada campus, calcular el camino más corto desde la estación de origen dada por el usuario hasta sus estaciones cercanas.
    - Devolver, para cada campus:
        - Universidad.
        - Nombre del campus.
        - Estación de destino utilizada.
        - Longitud del camino (número de estaciones).

D. **Cambios de línea y comparativa avanzada**

Uso de la propiedad `lineaId` en `:SIGUIENTE` para:

- Calcular el número de cambios de línea en un trayecto.
- Comparar rutas por número de cambios de línea y número total de estaciones.

Parte C — Para ambas BD (MongoDB y Neo4j)

Funcionalidad de recomendación integrada de campus

Diseño de una funcionalidad que, para una estación de origen y un estudio de GRADO:

- Liste los campus que imparten ese estudio, con información de accesibilidad en Metro.
- Implementación en MongoDB de una **versión simplificada** (por ejemplo, restringida a una línea y usando indiceEnLinea).
- Implementación en Neo4j de una versión completa que considere rutas, cambios de línea y pasos por estaciones con Renfe.
- Comentario breve en la memoria sobre la diferencia de enfoque entre MongoDB y Neo4j para este problema.

Pautas rápidas

Se valorará especialmente que el modelo y las consultas se ajusten bien al dominio planteado. A modo orientativo, se recomienda:

Parte A — MongoDB

- Definir con claridad qué se embebe y qué se referencia (especialmente en lineas–estaciones y campus–estudios).
- Mantener un volumen de datos suficiente para que las agregaciones tengan sentido (no trabajar solo con 5 estaciones de ejemplo).
- Aprovechar las agregaciones para generar "mini-informes" (resúmenes por universidad, por zona, etc.).
- Diseñar índices alineados con las consultas de lectura más frecuentes.

Parte B — Neo4j

- Pensar el modelo de grafo desde las preguntas de rutas y relaciones que se quieren responder.
- Modelar explícitamente los caminos con :SIGUIENTE y, si se desea, :TRANSBORDO.
- Trabajar con consultas de rutas que integren bien estaciones, campus y estudios.
- Comparar de manera cualitativa cómo cambia la dificultad de ciertas consultas al pasar de MongoDB a Neo4j.

Rúbrica general de las prácticas

La parte de prácticas de MongoDB y Neo4j contribuye a la calificación final según la planificación general de la asignatura. A nivel interno, puede considerarse la siguiente orientación.

Parte A — MongoDB (sobre 10 puntos)

1. Diseño del modelo documental y justificación (3 puntos).
2. Población de datos y coherencia del dominio (2,5 puntos).
3. Operaciones CRUD (2 puntos).
4. Agregaciones, Índices y consultas (2,5 puntos).

Parte B — Neo4j (sobre 10 puntos)

1. Modelo de grafo bien definido y coherente con el dominio (2,5 puntos).
2. Inserción/Carga de datos correcta mediante Cypher (2 puntos).
3. Consultas estructurales y académicas (2,5 puntos).
4. Consultas de rutas y funcionalidad de recomendación (3 puntos).

ENTREGABLES

La práctica se entregará mediante el Aula virtual, concretamente en la sección de "**Prácticas**" de la asignatura de Bases de Datos No Relacionales. La entrega contendrá:

- Memoria de realización de la práctica, que incluya:
  - Portada.
  - Índice.
  - Explicación del modelo en MongoDB (colecciones, embebidos, referencias).
  - Explicación del modelo en Neo4j (nodos, relaciones, propiedades).
  - Descripción de la carga de datos en ambas bases de datos.

  - Resumen de las consultas más relevantes en MongoDB y Neo4j.
  - Descripción de la funcionalidad de recomendación (Parte C de requisitos funcionales).
  - Comentarios sobre pruebas y ejemplos de uso.
  - Imágenes o capturas ilustrativas del proceso (consola, clientes gráficos, etc.).
  - Estimación aproximada del esfuerzo y planificación de tareas del grupo.
- Scripts de MongoDB:
  - Creación/carga de datos (insertMany, mongoimport, etc.).
  - Consultas find y aggregate utilizadas.
  - Creación de índices.
- Scripts Cypher de Neo4j:
  - Creación de nodos y relaciones.
  - Consultas estructurales, de campus/estudios, de rutas y de recomendación.

FECHA DE ENTREGA

La práctica se entregará a través del Aula virtual, en la sección Prácticas de la asignatura, teniendo como fecha límite de entrega las 17:00h del día 8 de enero de 2026.

CONSIDERACIONES

La práctica requiere de la intervención de los 3 miembros del grupo.

Cualquier atisbo de plagio es calificado con un **0** para todos los miembros del grupo.

Cualquier entrega pasada la hora de la entrega es calificada con un **0** para todos los miembros del grupo.