

**UNIVERSIDAD DA VINCI DE GUATEMALA**

**FACULTAD DE INGENIERIA INDUSTRIA Y TECNOLOGIA**

**CARRERA: LICENCIATURA EN INGENIERIA EN SISTEMAS**



**SEGUNDA EVALUACION PARCIAL**

Byron Rodolfo Maldonado Palacios 202300076

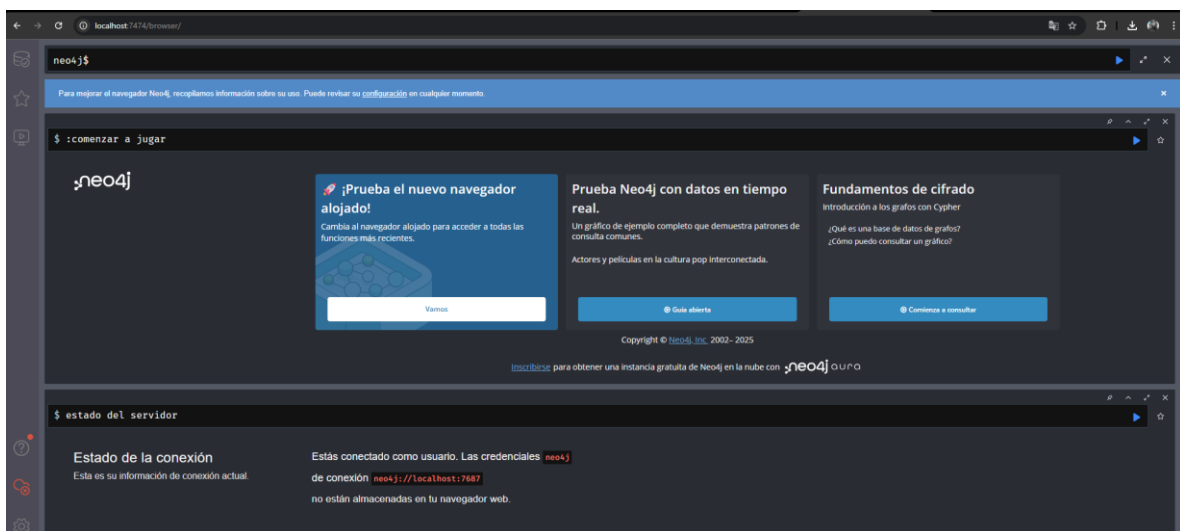
Guatemala, 02 de noviembre de 2025

## Vamos a Descargar la imagen oficial de Neo4j

```
PS C:\Users\Usuario> docker pull neo4j:latest
latest: Pulling from library/neo4j
4f4fb700ef54: Pull complete
d207cc66da44: Pull complete
afbcb3f6dc81: Pull complete
4ce987248665: Pull complete
d35c4f909d8e: Pull complete
3ca9ec796b46: Pull complete
Digest: sha256:d1081996365240ff8646671699085d1fc5214ad226c4febd30a54de6052c7541
Status: Downloaded newer image for neo4j:latest
docker.io/library/neo4j:latest
PS C:\Users\Usuario> |
```

## Crear y ejecutar el contenedor de Neo4j

```
PS C:\Users\Usuario> docker run -d `
>> --name neo4j_container `
>> -p 7474:7474 -p 7687:7687 `
>> -e NEO4J_AUTH=neo4j/test123 `
>> neo4j:latest
7aa19fcee1238c60b9168131b7579ff4f8e012c51a96c9321969b7b1edb3f5d1
PS C:\Users\Usuario> |
```



## Grafo (pseudonodos y relaciones)

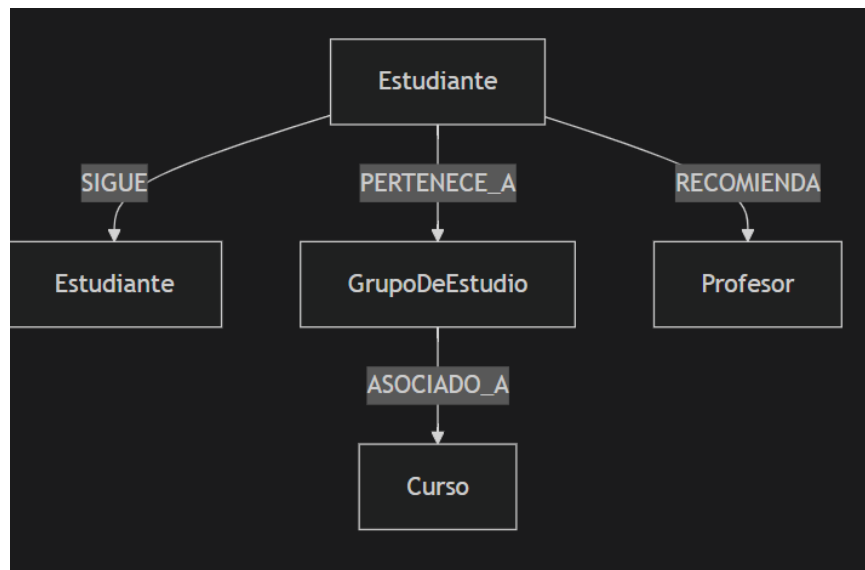
Esquema (tipos de nodos y relaciones)

Nodos (labels y prop. mínima)

- (Estudiante {nombre})
- (GrupoDeEstudio {nombre})
- (Curso {nombre})
- (Profesor {nombre})

Relaciones (dirigidas)

- (e1:Estudiante)-[:SIGUE]->(e2:Estudiante)
- (e:Estudiante)-[:PERTENECE\_A]->(g:GrupoDeEstudio)
- (g:GrupoDeEstudio)-[:ASOCIADO\_A]->(c:Curso)
- (e:Estudiante)-[:RECOMIENDA]->(p:Profesor)



# RELACIONES Y CONSULTAS

## Relacionar alumnos con la sesión de estudio

```
neo4j$ MATCH (pedro:Alumno {nombre:'Pedro Ruiz'}), (luisa:Alumno {nombre:'Luisa Martin'}), (jose:Alumno {nombre:'Jose Torres'}), (grupo2:SesionDeEstudio {nombre:'Desarrollo Web'}) CREATE (pedro)-[:ASISTE_
```

Created 3 relationships, completed after 16 ms.

## Relacionar la sesión con la materia

```
neo4j$ MATCH (grupo2:SesionDeEstudio {nombre:'Desarrollo Web'}), (materia:Asignatura {nombre:'Desarrollo Web Avanzado'}) CREATE (grupo2)-[:TRATA SOBRE]->(materia);
```

Created 1 relationship, completed after 12 ms.

## Relacionar docente con la materia y la sesión.

```
neo4j$ MATCH (maestra:Docente {nombre:'Sofia Jimenez'}), (materia:Asignatura {nombre:'Desarrollo Web Avanzado'}), (grupo2:SesionDeEstudio {nombre:'Desarrollo Web'}) CREATE (maestra)-[:IMPARTE]->(materia);
```

Created 2 relationships, completed after 14 ms.

## Alumnos que asisten a la sesión "Desarrollo Web"

```
neo4j$ MATCH (a:Alumno)-[:ASISTE_A]-(s:SesionDeEstudio {nombre:'Desarrollo Web'}) RETURN a.nombre AS alumno, s.nombre AS sesion;
```

	alumno	sesion
1	"Pedro Ruiz"	"Desarrollo Web"
2	"Luisa Martin"	"Desarrollo Web"
3	"Jose Torres"	"Desarrollo Web"

Started streaming 3 records after 8 ms and completed after 9 ms.

## Materia que imparte la docente "Sofia Jimenez"

```
neo4j$ MATCH (d:Docente {nombre:'Sofia Jimenez'})-[:IMPARTE]->(m:Asignatura) RETURN d.nombre AS docente, m.nombre AS asignatura;
```

	docente	asignatura
1	"Sofia Jimenez"	"Desarrollo Web Avanzado"

Started streaming 1 records after 8 ms and completed after 9 ms.

## Asignatura que se trata en la sesión "Desarrollo Web"

```
neo4j$ MATCH (s:SesionDeEstudio {nombre:'Desarrollo Web'})-[:TRATA SOBRE]->(m:Asignatura) RETURN s.nombre AS sesion, m.nombre AS asignatura;
```

	sesion	asignatura
1	"Desarrollo Web"	"Desarrollo Web Avanzado"

Started streaming 1 records after 7 ms and completed after 7 ms.

## Mostrar qué docente guía la sesión y a qué alumnos

```
neo4j$ MATCH (a:Alumno)-[:ASISTE_A]-(s:SesionDeEstudio)-[:GUIADO POR]->(d:Docente) RETURN a.nombre AS alumno, s.nombre AS sesion, d.nombre AS docente;
```

	alumno	sesion	docente
1	"Jose Torres"	"Desarrollo Web"	"Sofia Jimenez"
2	"Luisa Martin"	"Desarrollo Web"	"Sofia Jimenez"
3	"Pedro Ruiz"	"Desarrollo Web"	"Sofia Jimenez"

Started streaming 3 records after 8 ms and completed after 8 ms.

## MI PROPUESTA DE 3 claves y su tipo en Redis

### 1. user:{id}:profile — HASH

- Guarda campos del perfil del estudiante.
- Ejemplo de campos: name, email, career, joined\_at.
- Ventaja: lectura/escritura de campos individuales sin reescribir todo.

### 2. user:{id}:recent:courses — LIST

- Pila de “últimos cursos visitados”.
- Usas LPUSH para agregar al inicio y LTRIM para quedarte con los 5 más recientes.

### 3. groups:popularity — SORTED SET (ZSET)

- Ranking de grupos por popularidad (score = cantidad de miembros o visitas)
- Con ZINCRBY aumentas el score; con ZREVRANGE obtienes el top 3

## Arranque rápido con Docker

```
docker run -d --name redis -p 6379:6379 redis:latest
```

```
docker exec -it redis redis-cli
```

## Los últimos 5 cursos visitados por un estudiante

```
PS C:\Users\Usuario> docker run -d --name redis -p 6379:6379 redis:latest
127.0.0.1:6379> HSET user:101:profile name "Luisa Martin" email "Luisa@connectu.edu" career "Diseño Gráfico" joined_at "2025-02-10"
(integer) 4
127.0.0.1:6379> HGETALL user:101:profile
1) "name"
2) "Luisa Martin"
3) "email"
4) "Luisa@connectu.edu"
5) "career"
6) "Dise\x03\xblo Gr\x03\xalfico"
7) "joined_at"
8) "2025-02-10"
```

## Registrar visitas y limitar a 5

```
127.0.0.1:6379> LPUSH user:101:recent:courses DWA-2025
(integer) 1
127.0.0.1:6379> LPUSH user:101:recent:courses BDG-101
(integer) 2
127.0.0.1:6379> LPUSH user:101:recent:courses IAL-2025
(integer) 3
127.0.0.1:6379> LPUSH user:101:recent:courses EST-2025
(integer) 4
127.0.0.1:6379> LPUSH user:101:recent:courses UX-200
(integer) 5
127.0.0.1:6379> LTRIM user:101:recent:courses 0 4
OK
127.0.0.1:6379> LRANGE user:101:recent:courses 0 4
1) "UX-200"
2) "EST-2025"
3) "IAL-2025"
4) "BDG-101"
5) "DWA-2025"
```

## Los 3 grupos más populares del momento

```
127.0.0.1:6379> ZINCRBY groups:popularity 1 grupo:devweb
"1"
127.0.0.1:6379> ZINCRBY groups:popularity 1 grupo:data
"1"
127.0.0.1:6379> ZINCRBY groups:popularity 1 grupo:ux
"1"
127.0.0.1:6379> ZINCRBY groups:popularity 5 grupo:devweb
"6"
127.0.0.1:6379> ZINCRBY groups:popularity 3 grupo:data
"4"
127.0.0.1:6379> ZINCRBY groups:popularity 2 grupo:ux
"3"
```

## Top 3

```
127.0.0.1:6379> ZREVRANGE groups:popularity 0 2 WITHSCORES
1) "grupo:devweb"
2) "6"
3) "grupo:data"
4) "4"
5) "grupo:ux"
6) "3"
127.0.0.1:6379> |
```

### ¿Por qué Redis es útil junto con Neo4j en este caso?

- Caché de recomendaciones (read-through): las recomendaciones de grupos basadas en relaciones (intereses, amigos-de-amigos) se calculan en Neo4j, y se guardan en Redis como `user:{id}:reco:groups` (LIST/ZSET) con TTL (10–30 min). Las siguientes consultas responden en milisegundos sin recalculiar el grafo.
- Rankings y contadores en tiempo real: los ZSET de Redis mantienen el Top de grupos con `ZINCRBY` sin hacer agregaciones costosas en el grafo.
- Desacoplamiento y escalabilidad: Neo4j asegura la consistencia semántica del grafo; Redis atiende lecturas ultra rápidas, últimos visitados, sesiones y rate-limiting.
- Invalidación simple: si cambian relaciones en Neo4j (nuevo interés, unión a grupo), se invalida la caché con `DEL user:{id}:reco:groups` para regenerarla en la próxima lectura.

**Resultado:** menor latencia, menor carga sobre Neo4j y experiencia en tiempo real para el usuario.