Comparativa de rendimiento entre Docker y máquina virtual utilizando consultas Cypher en Neo4j

1. Introducción

Este trabajo tiene como objetivo comparar el rendimiento de una base de datos Neo4j ejecutando consultas Cypher en dos entornos distintos: una máquina virtual (VM) con Ubuntu 24.04 y un contenedor Docker ejecutado sobre un host con Ubuntu 22.04. La intención es analizar cuál de los dos entornos ofrece un mejor rendimiento, estabilidad y eficiencia al ejecutar operaciones sobre grafos.

2. Objetivos

- Comparar el rendimiento de Neo4j en una máquina virtual y en un contenedor Docker.
- Medir el tiempo de ejecución de diferentes tipos de consultas Cypher.
- Evaluar la estabilidad de cada entorno a través de la desviación estándar.
- Documentar los resultados obtenidos y extraer conclusiones.

3. ¿Qué es Neo4j y Cypher?

Neo4j es una base de datos orientada a grafos que permite modelar relaciones entre entidades de forma nativa, representando los datos como nodos y relaciones. Cypher es el lenguaje de consulta de Neo4j, similar a SQL pero diseñado para grafos.

4. Entornos de prueba

Máquina virtual

- Sistema operativo: Ubuntu 24.04
- Neo4j instalado manualmente (Community Edition)
- Consultas ejecutadas con `cypher-shell`

Docker

- Host: Ubuntu 22.04
- Contenedor: Imagen oficial `neo4j:latest`
- Neo4j ejecutado con `NEO4J_AUTH=none`

5. Base de datos utilizada

Se simuló una pequeña red social con los siguientes elementos:

Nodos:

- `User`: 50 usuarios con nombre y edad.
- `Post`: 100 publicaciones con contenido y fecha.
- `Tag`: 7 etiquetas temáticas.

Relaciones:

- `(:User)-[:CREATED]->(:Post)`
- `(:Post)-[:HAS_TAG]->(:Tag)`
- `(:User)-[:FOLLOWS]->(:User)`

Los datos se generaron usando consultas Cypher automáticas con `UNWIND` y `rand()` para aleatoriedad controlada.

6. Consultas y metodología de evaluación

Se definieron tres consultas de diferentes niveles de complejidad:

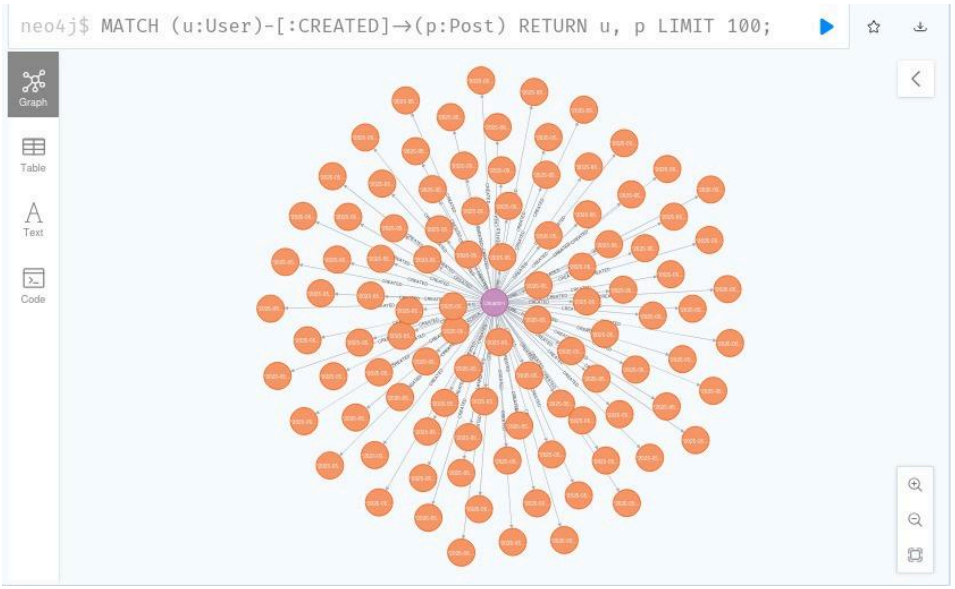
- **Consulta ligera:**
  MATCH (u:User) RETURN count(u);

  Cuenta los usuarios. Sirve como referencia para una carga mínima.

- **Consulta intermedia:**

  MATCH (u:User)-[:CREATED]->(p:Post) RETURN count(p);

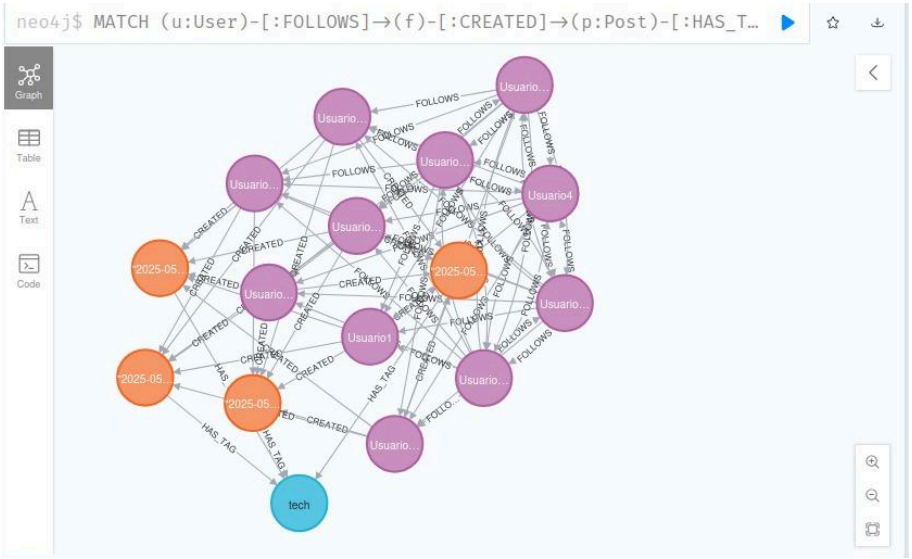
  Cuenta las publicaciones creadas por usuarios. Recorre una relación.



- **Consulta compleja:**

  MATCH (u:User)-[:FOLLOWS]->(f)-[:CREATED]->(p:Post)-[:HAS_TAG]->(t:Tag) RETURN count(p);

  Busca publicaciones hechas por usuarios que son seguidos por otros usuarios, y que además están etiquetadas. Luego cuenta esas publicaciones.



- Cada consulta se ejecutó 5 veces seguidas en ambos entornos usando:

  time cypher-shell "<consulta>"

  Se extrajo el tiempo `real` para cada ejecución. Luego se calculó la media y desviación estándar.

7. Resultados

(Las fotos se encuentran al final del documento)

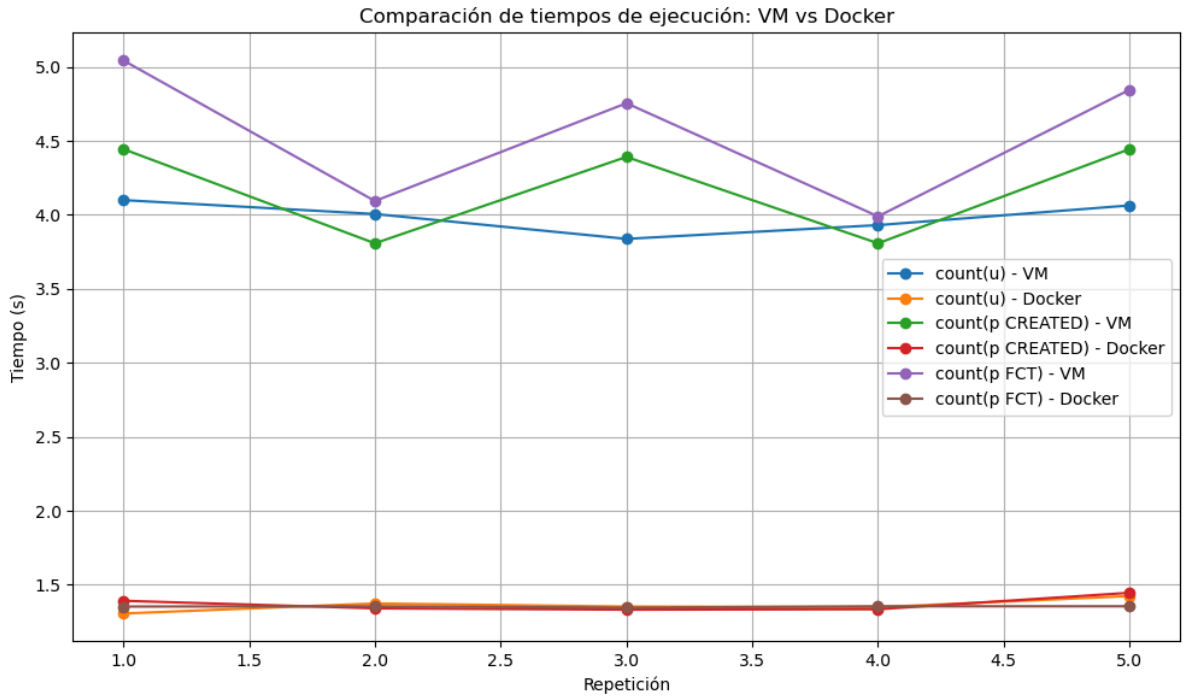
Tabla comparativa:

Consulta	Media VM (s)	Desv. VM	Media Docker (s)	Desv. Docker
count(u)	3.99	0.09	1.35	0.04
count(p) CREATED	4.17	0.21	1.38	0.04
FOLLOWS->CREATED->HAS_TAG	4.33	0.29	1.38	0.06

8. Análisis comparativo

En las tres consultas, Docker demostró ser significativamente más rápido que la máquina virtual, con mejoras de aproximadamente un 65%. Además, los tiempos fueron más estables en Docker (menor desviación estándar), lo que indica mayor predictibilidad.

Esto puede deberse a que Docker comparte el kernel del sistema operativo y evita la sobrecarga de virtualizar hardware como ocurre en una VM.



9. Conclusión

Docker ofrece una forma más eficiente y rápida de ejecutar Neo4j para este tipo de cargas de trabajo. Es ideal para entornos de desarrollo, pruebas o despliegue continuo. Sin embargo, las máquinas virtuales siguen siendo útiles cuando se requiere aislamiento total del sistema operativo.

10. Repositorio GitHub

El proyecto está disponible en:
https://github.com/beatriizorozco/Docker_vs_VM

Incluye los scripts de configuración, datos generados, consultas ejecutadas y capturas de pantalla de los entornos.

- **VM:**

```
bea@bea-VirtualBox:~$ time cypher-shell -u neo4j -p test "MATCH (u:User) RETURN count(u);"
+----------+
| count(u) |
+----------+
| 50       |
+----------+

1 row
ready to start consuming query after 139 ms, results consumed after another 2 ms

real    0m4,100s
user    0m2,650s
sys     0m3,237s
```

```
bea@bea-VirtualBox:~$ time cypher-shell -u neo4j -p test "MATCH (u:User) RETURN count(u);"
+----------+
| count(u) |
+----------+
| 50       |
+----------+

1 row
ready to start consuming query after 139 ms, results consumed after another 2 ms

real    0m4,100s
user    0m2,650s
sys     0m3,237s
bea@bea-VirtualBox:~$ time cypher-shell -u neo4j -p test "MATCH (u:User)-[:CREATED]->(p:Post) RETURN count(p);"
+----------+
| count(p) |
+----------+
| 600      |
+----------+

1 row
ready to start consuming query after 659 ms, results consumed after another 53 ms

real    0m4,445s
user    0m2,279s
sys     0m3,400s
```

```
bea@bea-VirtualBox:~$ time cypher-shell -u neo4j -p test "MATCH (u:User)-[:FOLLOWS]->(f)-[:CREATED]->(p:Post)-[:HAS_TAG]
->(t:Tag) RETURN count(p);"
+----------+
| count(p) |
+----------+
| 6510     |
+----------+

1 row
ready to start consuming query after 450 ms, results consumed after another 126 ms

real    0m5,044s
user    0m2,705s
sys     0m3,562s
bea@bea-VirtualBox:~$ time cypher-shell -u neo4j -p test "MATCH (u:User) RETURN count(u);"
+----------+
| count(u) |
+----------+
| 50       |
+----------+

1 row
ready to start consuming query after 9 ms, results consumed after another 1 ms

real    0m4,006s
user    0m2,464s
sys     0m3,243s
```

```
bea@bea-VirtualBox:~$ time cypher-shell -u neo4j -p test "MATCH (u:User)-[:CREATED]->(p:Post) RETURN count(p);"
+----------+
| count(p) |
+----------+
| 600      |
+----------+

1 row
ready to start consuming query after 10 ms, results consumed after another 4 ms

real    0m4,222s
user    0m2,861s
sys     0m3,422s
bea@bea-VirtualBox:~$ time cypher-shell -u neo4j -p test "MATCH (u:User)-[:FOLLOWS]->(f)-[:CREATED]->(p:Post)-[:HAS_TAG]
->(t:Tag) RETURN count(p);"
+----------+
| count(p) |
+----------+
| 6510     |
+----------+

1 row
ready to start consuming query after 2 ms, results consumed after another 26 ms

real    0m4,846s
user    0m2,814s
sys     0m3,296s
```

```
bea@bea-VirtualBox:~$ time cypher-shell -u neo4j -p test "MATCH (u:User) RETURN count(u);"
+----------+
| count(u) |
+----------+
| 50       |
+----------+

1 row
ready to start consuming query after 6 ms, results consumed after another 2 ms

real    0m4,064s
user    0m2,674s
sys     0m3,487s
bea@bea-VirtualBox:~$ time cypher-shell -u neo4j -p test "MATCH (u:User)-[:CREATED]->(p:Post) RETURN count(p);"
+----------+
| count(p) |
+----------+
| 600      |
+----------+

1 row
ready to start consuming query after 4 ms, results consumed after another 7 ms

real    0m3,808s
user    0m2,518s
sys     0m2,925s
```

```
bea@bea-VirtualBox:~$ time cypher-shell -u neo4j -p test "MATCH (u:User)-[:FOLLOWS]->(f)-[:CREATED]->(p:Post)-[:HAS_TAG]
->(t:Tag) RETURN count(p);"
+----------+
| count(p) |
+----------+
| 6510     |
+----------+

1 row
ready to start consuming query after 5 ms, results consumed after another 11 ms

real    0m3,909s
user    0m2,605s
sys     0m3,327s
bea@bea-VirtualBox:~$ time cypher-shell -u neo4j -p test "MATCH (u:User) RETURN count(u);"
+----------+
| count(u) |
+----------+
| 50       |
+----------+

1 row
ready to start consuming query after 6 ms, results consumed after another 3 ms

real    0m3,931s
user    0m2,683s
sys     0m3,126s
```

```
bea@bea-VirtualBox:~$ time cypher-shell -u neo4j -p test "MATCH (u:User)-[:CREATED]->(p:Post) RETURN count(p);"
+----------+
| count(p) |
+----------+
| 600      |
+----------+

1 row
ready to start consuming query after 12 ms, results consumed after another 5 ms

real    0m3,990s
user    0m2,592s
sys     0m3,302s
bea@bea-VirtualBox:~$ time cypher-shell -u neo4j -p test "MATCH (u:User)-[:FOLLOWS]->(f)-[:CREATED]->(p:Post)-[:HAS_TAG]
->(t:Tag) RETURN count(p);"
+----------+
| count(p) |
+----------+
| 6510     |
+----------+

1 row
ready to start consuming query after 3 ms, results consumed after another 10 ms

real    0m4,755s
user    0m2,872s
sys     0m3,605s
```

```
bea@bea-VirtualBox:~$ time cypher-shell -u neo4j -p test "MATCH (u:User) RETURN count(u);"
+----------+
| count(u) |
+----------+
| 50       |
+----------+

1 row
ready to start consuming query after 4 ms, results consumed after another 1 ms

real    0m3,838s
user    0m2,702s
sys     0m3,022s
bea@bea-VirtualBox:~$ time cypher-shell -u neo4j -p test "MATCH (u:User)-[:CREATED]->(p:Post) RETURN count(p);"
+----------+
| count(p) |
+----------+
| 600      |
+----------+

1 row
ready to start consuming query after 6 ms, results consumed after another 3 ms

real    0m4,392s
user    0m2,760s
sys     0m3,576s
```

- **Docker:**

```
bea@UEA-C289:~$ time sudo docker exec -it neo4j-docker cypher-shell "MATCH (u:User) RETURN count(u);"
+----------+
| count(u) |
+----------+
| 50       |
+----------+

1 row
ready to start consuming query after 36 ms, results consumed after another 1 ms

real    0m1,424s
user    0m0,004s
sys     0m0,010s
bea@UEA-C289:~$ time sudo docker exec -it neo4j-docker cypher-shell "MATCH (u:User)-[:CREATED]->(p:Post) RETURN count(p);"
+----------+
| count(p) |
+----------+
| 300      |
+----------+

1 row
ready to start consuming query after 75 ms, results consumed after another 6 ms

real    0m1,446s
user    0m0,007s
sys     0m0,012s
```

```
bea@UEA-C289:~$ time sudo docker exec -it neo4j-docker cypher-shell "MATCH (u:User)-[:FOLLOWS]->(f)-[:CREATED]->(p:Post)-[:HAS_TAG]->(t:Tag) RETURN count(p);"
+----------+
| count(p) |
+----------+
| 1638     |
+----------+

1 row
ready to start consuming query after 88 ms, results consumed after another 8 ms

real    0m1,487s
user    0m0,007s
sys     0m0,012s
bea@UEA-C289:~$ time sudo docker exec -it neo4j-docker cypher-shell "MATCH (u:User) RETURN count(u);"
+----------+
| count(u) |
+----------+
| 50       |
+----------+

1 row
ready to start consuming query after 2 ms, results consumed after another 1 ms

real    0m1,349s
user    0m0,006s
sys     0m0,009s
```

```
bea@UEA-C289:~$ time sudo docker exec -it neo4j-docker cypher-shell "MATCH (u:User)-[:CREATED]->(p:Post) RETURN count(p);"
+----------+
| count(p) |
+----------+
| 300      |
+----------+

1 row
ready to start consuming query after 3 ms, results consumed after another 2 ms

real    0m1,345s
user    0m0,002s
sys     0m0,011s
bea@UEA-C289:~$ time sudo docker exec -it neo4j-docker cypher-shell "MATCH (u:User)-[:FOLLOWS]->(f)-[:CREATED]->(p:Post)-[:HAS_TAG]->(t:Tag) RETURN count(p);"
+----------+
| count(p) |
+----------+
| 1638     |
+----------+

1 row
ready to start consuming query after 2 ms, results consumed after another 5 ms

real    0m1,355s
user    0m0,004s
sys     0m0,012s
```

```
bea@UEA-C289:~$ time sudo docker exec -it neo4j-docker cypher-shell "MATCH (u:User) RETURN count(u);"
+----------+
| count(u) |
+----------+
| 50       |
+----------+

1 row
ready to start consuming query after 1 ms, results consumed after another 0 ms

real    0m1,332s
user    0m0,005s
sys     0m0,008s
bea@UEA-C289:~$ time sudo docker exec -it neo4j-docker cypher-shell "MATCH (u:User)-[:CREATED]->(p:Post) RETURN count(p);"
+----------+
| count(p) |
+----------+
| 300      |
+----------+

1 row
ready to start consuming query after 2 ms, results consumed after another 1 ms

real    0m1,335s
user    0m0,005s
sys     0m0,006s
```

```
bea@UEA-C289:~$ time sudo docker exec -it neo4j-docker cypher-shell "MATCH (u:User)-[:FOLLOWS]->(f)-[:CREATED]->(p:Post)-[:HAS_TAG]->(t:Tag) RETURN count(p);"
+----------+
| count(p) |
+----------+
| 1638     |
+----------+

1 row
ready to start consuming query after 3 ms, results consumed after another 4 ms

real    0m1,353s
user    0m0,010s
sys     0m0,007s
bea@UEA-C289:~$ time sudo docker exec -it neo4j-docker cypher-shell "MATCH (u:User) RETURN count(u);"
+----------+
| count(u) |
+----------+
| 50       |
+----------+

1 row
ready to start consuming query after 1 ms, results consumed after another 0 ms

real    0m1,351s
user    0m0,008s
sys     0m0,007s
```

```
bea@UEA-C289:~$ time sudo docker exec -it neo4j-docker cypher-shell "MATCH (u:User)-[:CREATED]->(p:Post) RETURN count(p);"
+----------+
| count(p) |
+----------+
| 300      |
+----------+

1 row
ready to start consuming query after 1 ms, results consumed after another 0 ms

real    0m1,374s
user    0m0,008s
sys     0m0,010s
bea@UEA-C289:~$ time sudo docker exec -it neo4j-docker cypher-shell "MATCH (u:User)-[:FOLLOWS]->(f)-[:CREATED]->(p:Post)-[:HAS_TAG]->(t:Tag) RETURN count(p);"
+----------+
| count(p) |
+----------+
| 1638     |
+----------+

1 row
ready to start consuming query after 1 ms, results consumed after another 3 ms

real    0m1,341s
user    0m0,003s
sys     0m0,010s
```

```
bea@UEA-C289:~$ time sudo docker exec -it neo4j-docker cypher-shell "MATCH (u:User) RETURN count(u);"
+----------+
| count(u) |
+----------+
| 50       |
+----------+

1 row
ready to start consuming query after 2 ms, results consumed after another 0 ms

real    0m1,306s
user    0m0,003s
sys     0m0,010s
bea@UEA-C289:~$ time sudo docker exec -it neo4j-docker cypher-shell "MATCH (u:User)-[:CREATED]->(p:Post) RETURN count(p);"
+----------+
| count(p) |
+----------+
| 300      |
+----------+

1 row
ready to start consuming query after 1 ms, results consumed after another 1 ms

real    0m1,392s
user    0m0,008s
sys     0m0,011s
```

```
bea@UEA-C289:~$ time sudo docker exec -it neo4j-docker cypher-shell "MATCH (u:User)-[:FOLLOWS]->(f)-[:CREATED]->(p:Post)-[:HAS_TAG]->(t:Tag) RETURN count(p);"
+----------+
| count(p) |
+----------+
| 1638     |
+----------+

1 row
ready to start consuming query after 1 ms, results consumed after another 2 ms

real    0m1,353s
user    0m0,003s
sys     0m0,010s
```