

**Máster universitario de Ciencia de Datos**

**Práctica 1 – PRA1**

**Uso de base de datos NoSQL - Cassandra, MongoDB y Neo4j.**

Autor:

Mario Ubierna San Mamés

|  |
| --- |
| Índice de Contenido |

[Índice de Contenido 3](#_Toc89849666)

[Índice de tablas 4](#_Toc89849667)

[Índice de ilustraciones 5](#_Toc89849668)

[1. Ejercicio 1: Cassandra 6](#_Toc89849669)

[2. Ejercicio 2: MongoDB 7](#_Toc89849670)

[3. Ejercicio 3: Neo4j 8](#_Toc89849671)

[4. Ejercicio 4: Neo4j 9](#_Toc89849672)

[5. Bibliografía 10](#_Toc89849673)

|  |
| --- |
| Índice de tablas |

**No se encuentran elementos de tabla de ilustraciones.**

|  |
| --- |
| Índice de ilustraciones |

**No se encuentran elementos de tabla de ilustraciones.**

|  |
| --- |
| Ejercicio 1: Cassandra |

A

|  |
| --- |
| Ejercicio 2: MongoDB |

A

|  |
| --- |
| Ejercicio 3: Neo4j |

*Para la realización de este ejercicio se seguirán las instrucciones del caso de estudio ubicado en la siguiente URL: https://neo4j.com/developer/guide-importing-data-and-etl/ . Este caso se aborda también en el documento de ejercicios titulado “Transformación de una base de datos relacional a un modelo en grafo” pero a un nivel más superficial. Se recomienda leer dicho documento antes de realizar este ejercicio.*

*Se recomienda leer con atención la página web, ya que se introducen conceptos y buenas prácticas a seguir en ‘production’.*

*En la máquina virtual ya hay una base de datos precargada (twitter) que no es necesaria para realizar los ejercicios de esta práctica. Los nodos y relaciones que hay creados no interfieren en el ejercicio, los podéis ignorar tranquilamente.*

*Encontraréis las instrucciones para arrancar el servicio de Neo4j en la página 11 del documento con nombre “Uso de máquina virtual\_ Bases de datos no convencionales.pdf”. Recordad que el usuario / contraseña para acceder a Neo4j son: neo4j / uoc.*

## Carga de la base de datos (no puntúa)

Mirar el enunciado de la práctica.

## Ejercicios básicos (30%)

*Se propone responder las siguientes cuestiones:*

*1. Una vez realizada la carga de datos, utilizando cypher y su interfaz web, se pide:*

*a. obtener una visualización del esquema de los nodos/relaciones que se han creado.*

*b. contar cuántos nodos de tipo Supplier se han creado (adjuntar una captura de pantalla de las query y resultado).*

*2. Explicar brevemente (máx 3 líneas) la funcionalidad implementada de la línea:*

*ON CREATE SET c.categoryName = row.CategoryName, c.description = row.Description;*

*3. En la página web, se crea un índice para cada tipo de nodo y una restricción de unicidad para los nodos de tipo Order. Explicar brevemente (máx 3 líneas): ¿Por qué se recomienda crear un índice para cada tipo de nodo?*

### Comprobaciones de la carga de datos

#### Obtener una visualización del esquema

Para obtener una visualización del esquema de los nodos/relaciones que se han creado usamos “*call db.schema,visualization()*”.



Ilustración – función para obtener el esquema.

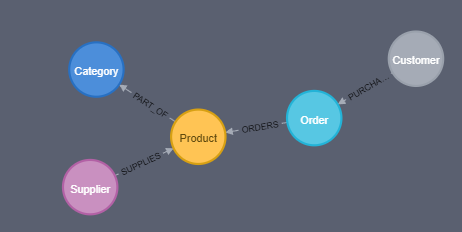


Ilustración - Visualización del esquema.

#### Cuántos nodos tipo *Supplier* se han creado

Para obtener el número de nodos hay que ejecutar la siguiente consulta y obtendremos dicho resultado:



Ilustración - Consulta para obtener el número de nodos de tipo Supplier.

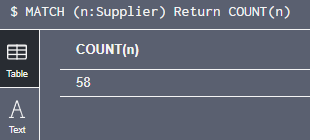
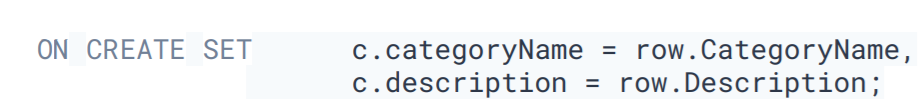


Ilustración - Resultado del número de nodos de tipo Supplier.

### Explicar la funcionalidad implementada



Lo que se está haciendo es definir los valores de las dos propiedades que tiene cada nodo del tipo *Category*, es decir, por cada línea del fichero CSV leído, coge los valores de las columnas (*CategoryName* y *Description*) y los añade a las propiedades correspondientes.

### ¿Por qué se recomienda crear un índice para cada tipo de nodo?

Básicamente se recomienda crear un índice para cada tipo de nodo para garantizar la búsqueda de nodos de forma óptima, dando así un valor único para cada nodo del mismo tipo.

|  |
| --- |
| Ejercicio 4: Neo4j |

A

|  |
| --- |
| Bibliografía |

A