

PROYECTO I

BASE DE DATOS NoSQL



SEBASTIÁN TORO FRANCO
CRISTIAN STEVEN OSORIO PAZ
WILLIAM AGUIRRE ZAPATA

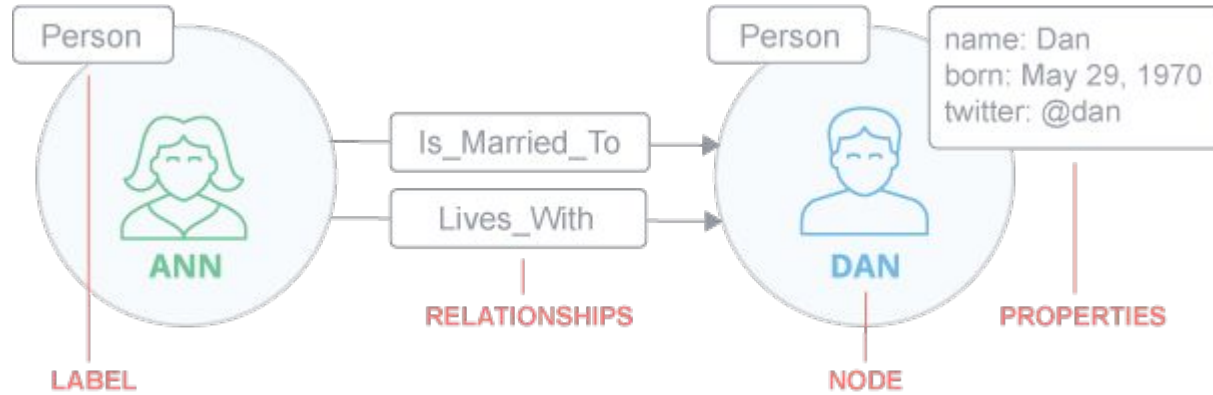
¿Qué es Neo4?

Neo4j es un software libre de Base de datos NoSQL orientada a grafos, implementado en Java. Neo4j fue desarrollado por Neo Technology, una startup sueca. Un motor de persistencia embebido, basado en disco.



Conceptos básicos de Neo4j

El modelo grafo de propiedades etiquetadas.

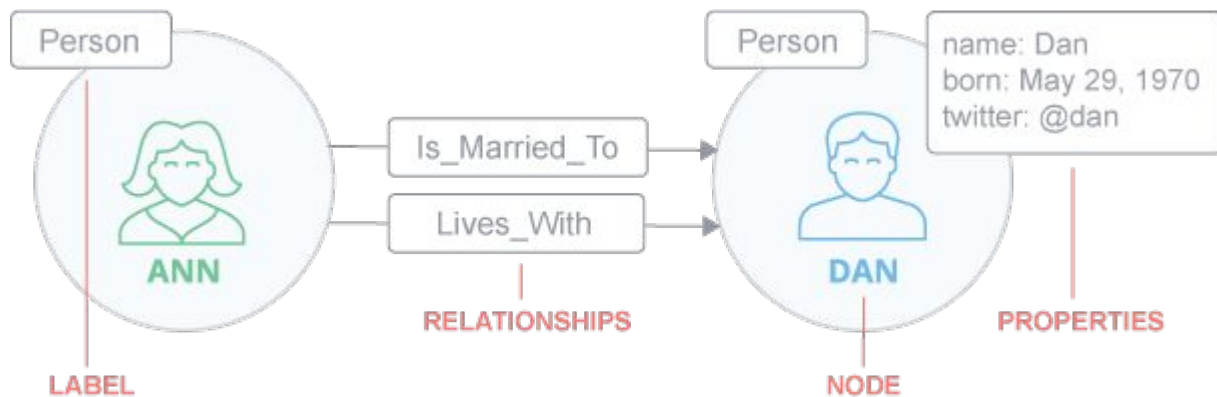


Nodos

- Los nodos son los principales elementos de datos
- Los nodos están conectados a otros nodos a través de las **relaciones**
- Los nodos pueden tener una o más **propiedades** (es decir, atributos almacenados como pares clave / valor)
- Los nodos tienen una o más **etiquetas** que describen su rol en el grafo

Conceptos básicos de Neo4j

El modelo grafo de propiedades etiquetadas.

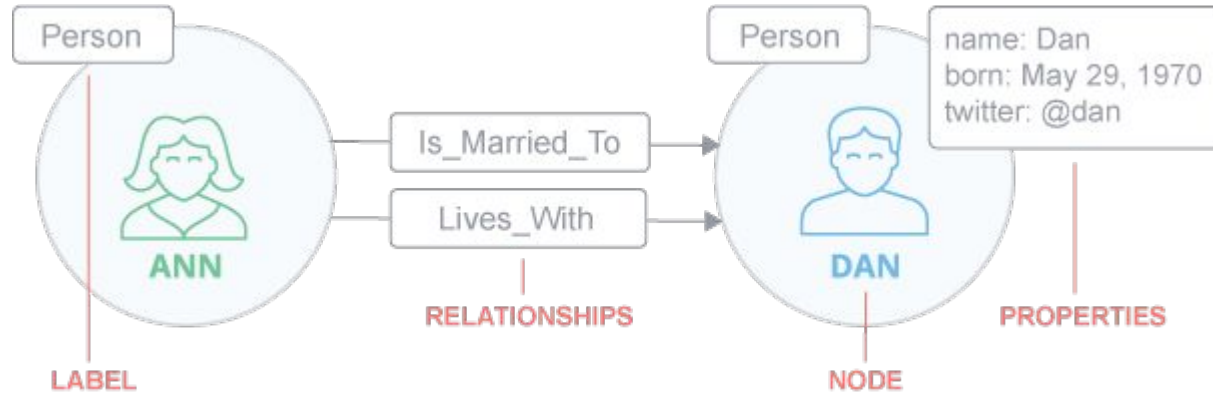


Relaciones

- Las relaciones conectan dos nodos
- Las relaciones son direccionales
- Los **nodos** pueden tener relaciones múltiples, incluso recursivas
- Las relaciones pueden tener una o más **propiedades** (es decir, atributos almacenados como pares clave / valor)

Conceptos básicos de Neo4j

El modelo Grafo de propiedades etiquetadas.

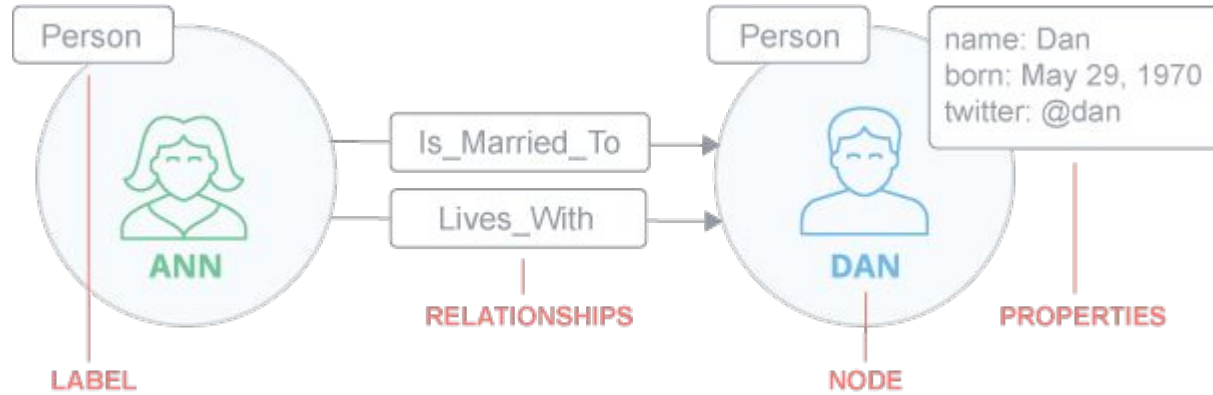


Propiedades

- Las propiedades se denominan valores donde el nombre (o clave) es una cadena
- Las propiedades se pueden indexar y restringir
- Los índices compuestos se pueden crear a partir de propiedades múltiples

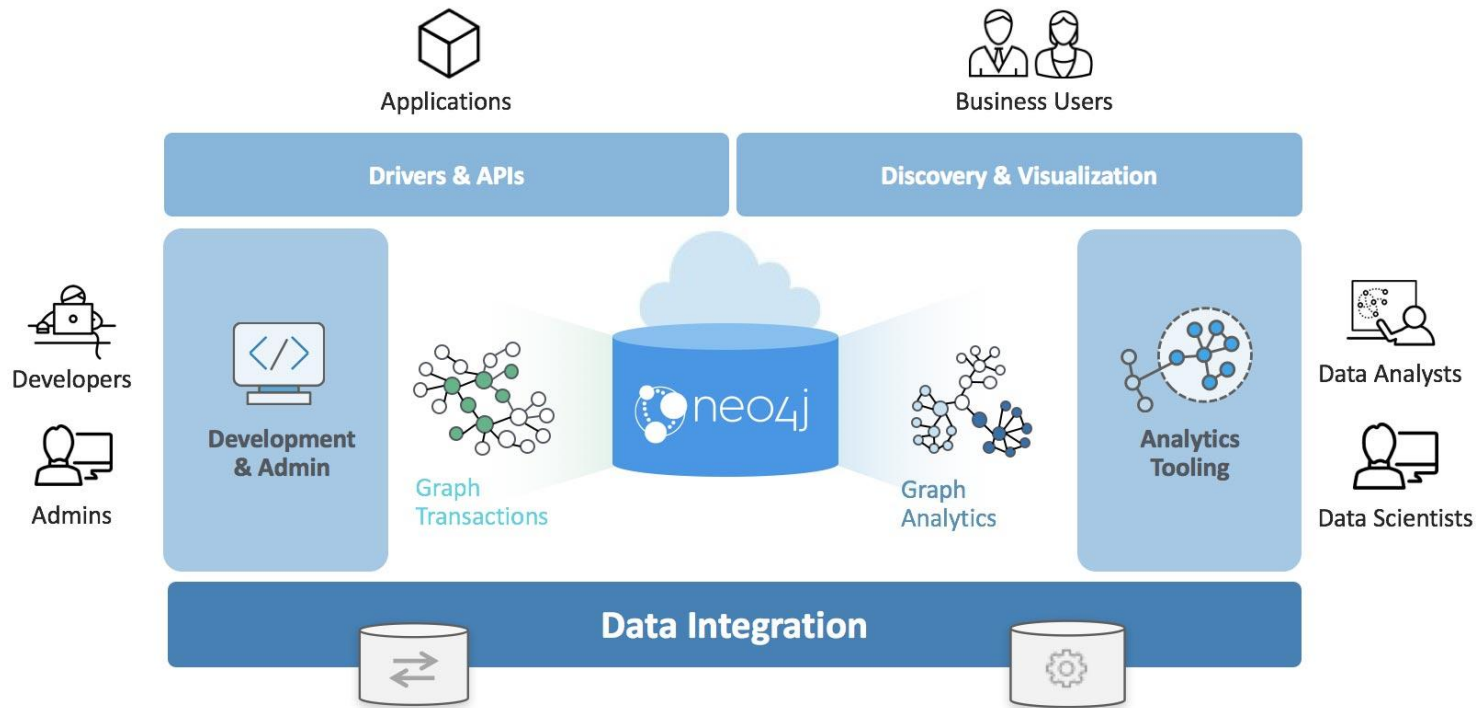
Conceptos básicos de Neo4j

El modelo grafo de propiedades etiquetadas.



Etiquetas

- Las etiquetas se utilizan para agrupar nodos en conjuntos
- Un **nodo** puede tener múltiples etiquetas
- Las etiquetas se indexan para acelerar la búsqueda de nodos en el grafo
- Los índices de etiquetas nativas están optimizados para la velocidad




El análisis de grafos ayuda a los científicos de datos a obtener nuevas perspectivas sobre los datos a través de grafos

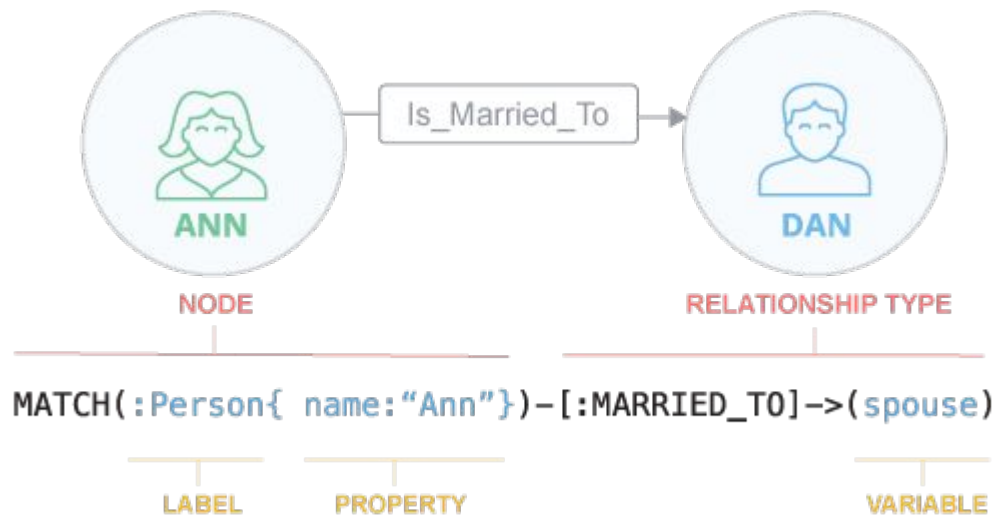
¿Qué es Cypher?

Cypher es un lenguaje de consulta propio de Neo4j. Es la forma declarativa de realizar consultas.

Cypher es a Neo4j lo que T-SQL es a Sql Server.

(Neo4j)
- [:] ->
(Cypher)

Los fundamentos de Cypher



Cypher describe nodos, relaciones y propiedades como arte ASCII directamente en el idioma, haciendo que las consultas sean fáciles de leer y reconocer como parte de los datos de su grafo.

PASOS DE INSTALACIÓN Y USO DE LA HERRAMIENTA

Descargue de <https://neo4j.com/download-center/#releases>

https://neo4j.com/download-center/#releases

Current Releases

Enterprise Server	Community Server	Neo4j Desktop
Neo4j Desktop 1.1.10		
OS	Download	
Mac	Neo4j Desktop (dmg)	
Linux	Neo4j Desktop (ApplImage)	
Windows	Neo4j Desktop (exe)	

PARA LINUX DESKTOP

Seleccione la pestaña Neo4j Desktop, después clic en Neo4j Desktop (ApplImage). Enseguida de descargado el archivo use *chmod x+ <filename>* De esta manera tendrá una instalación portable y local de Neo4j en Linux.

PARA LINUX SERVER

```
wget -O - https://debian.neo4j.org/neotechnology.gpg.key | sudo apt-key add -
```

```
echo 'deb https://debian.neo4j.org/repo stable/' | sudo tee  
/etc/apt/sources.list.d/neo4j.list
```

```
sudo apt-get update
```

```
sudo apt-get install neo4j
```



INSTALACIÓN DEL MÓDULO DE NEO4J PARA PYTHON EN LINUX DEBIAN

Requisitos tener instalado Python 3.6 y Pip3

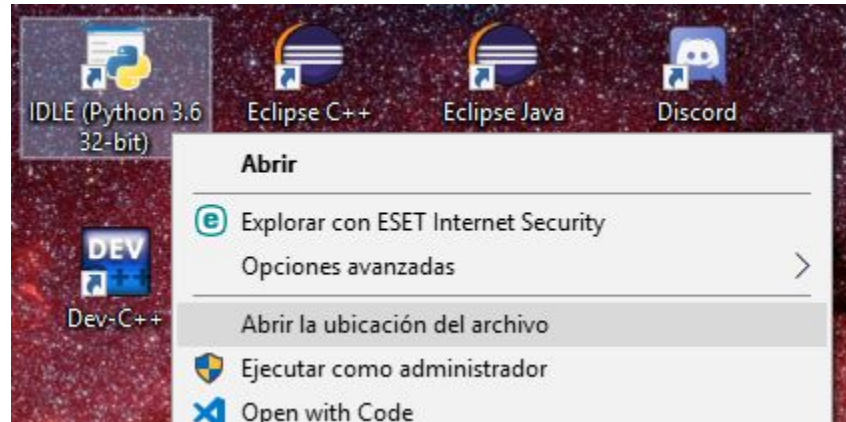
1	<code>sudo apt-get install python3.6</code>
2	<code>sudo apt-get install python3-pip</code>

Py2neo es biblioteca cliente y un completo juego de herramientas para trabajar con Neo4j desde las aplicaciones de Python y desde la línea de comandos.

3	<code>pip3 install neo4j-driver</code>
4	<code>pip3 installa py2neo</code>

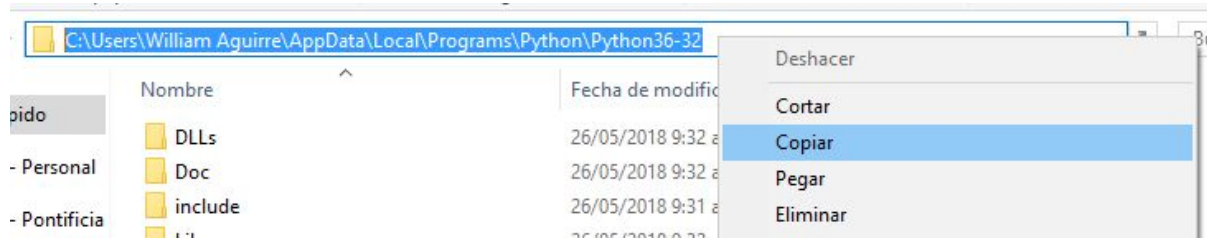
INSTALACIÓN DEL MÓDULO DE NEO4J PARA PYTHON EN WINDOWS

En Python 3.6 tenemos que ubicarnos en la carpeta donde se instaló Python, dando clic derecho en el acceso directo y en “Abrir la ubicación del archivo”



INSTALACIÓN DEL MÓDULO DE NEO4J PARA PYTHON EN WINDOWS

Copiamos la dirección de la ruta de la carpeta donde se encuentra la instalación de Python 3.6.



Abrimos un intérprete de comandos de Windows como CMD o PowerShell y escribimos “cd ” seguido de Ctrl+v para pegar la ruta anterior y damos ENTER.

INSTALACIÓN DEL MÓDULO DE NEO4J PARA PYTHON EN WINDOWS

```
C:\> Símbolo del sistema  
Microsoft Windows [Versión 10.0.17134.286]  
(c) 2018 Microsoft Corporation. Todos los derechos reservados.  
  
C:\Users\William Aguirre>cd C:\Users\William Aguirre\AppData\Local\Programs\Python\Python36-32_
```

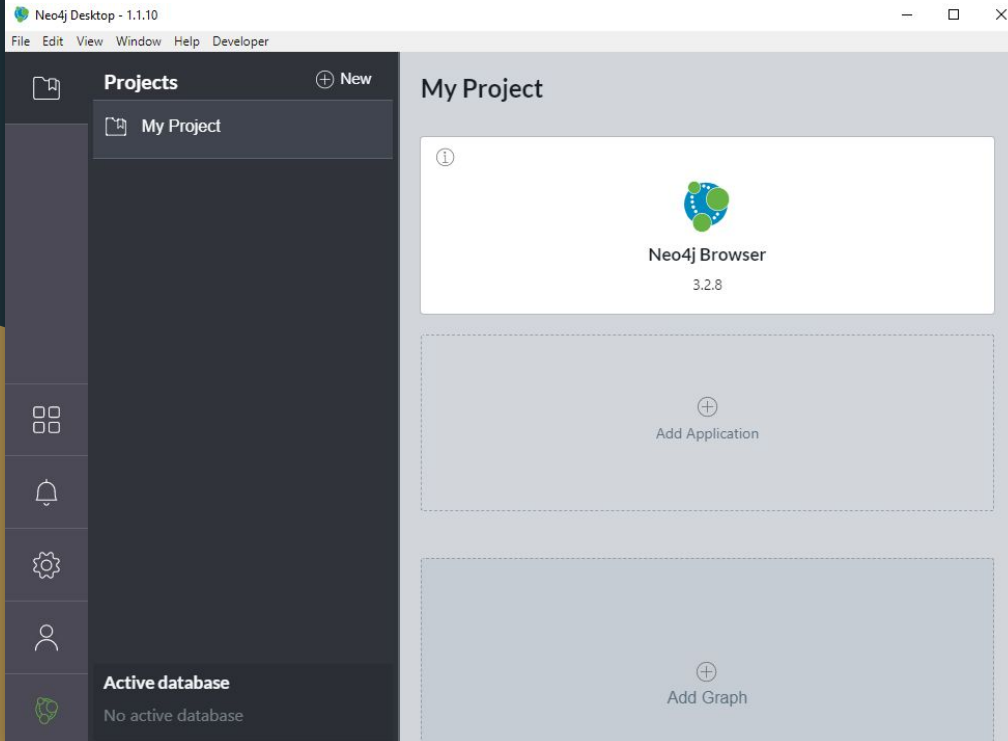
Una vez dentro de la ubicación de la carpeta en el CMD, escribimos “pip install neo4j-driver” y damos ENTER.

```
C:\Users\William Aguirre\AppData\Local\Programs\Python\Python36-32>pip install neo4j-driver
```

Terminada la instalación escribimos “pip install pyneo” y damos ENTER.

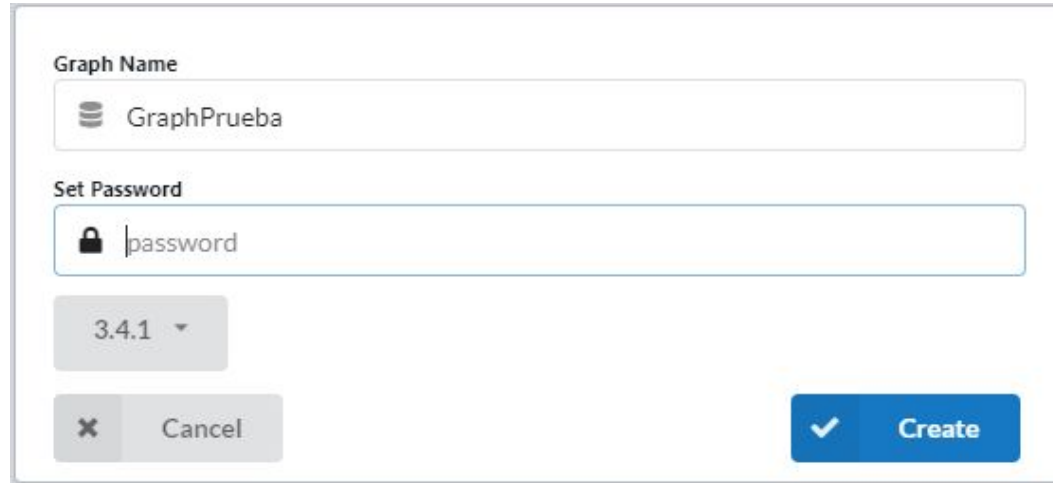
```
C:\Users\William Aguirre\AppData\Local\Programs\Python\Python36-32>pip install py2neo
```

CREACIÓN Y CONFIGURACIÓN DE UNA BASE DE DATOS EN NEO4J



Ejecutamos en Windows el programa “Neo4j Desktop” o en Linux el .ApplImage y esta es la primera pantalla del programa, damos clic en “Add Graph” para crear una base de datos nueva.

CREACIÓN Y CONFIGURACIÓN DE UNA BASE DE DATOS EN NEO4J

A screenshot of the Neo4j installation configuration window. It features a 'Graph Name' field with a database icon and the text 'GraphPrueba'. Below it is a 'Set Password' field with a lock icon and the text 'password'. A version selector shows '3.4.1' with a dropdown arrow. At the bottom are 'Cancel' and 'Create' buttons. The 'Create' button is highlighted with a yellow glow.

Graph Name

GraphPrueba

Set Password

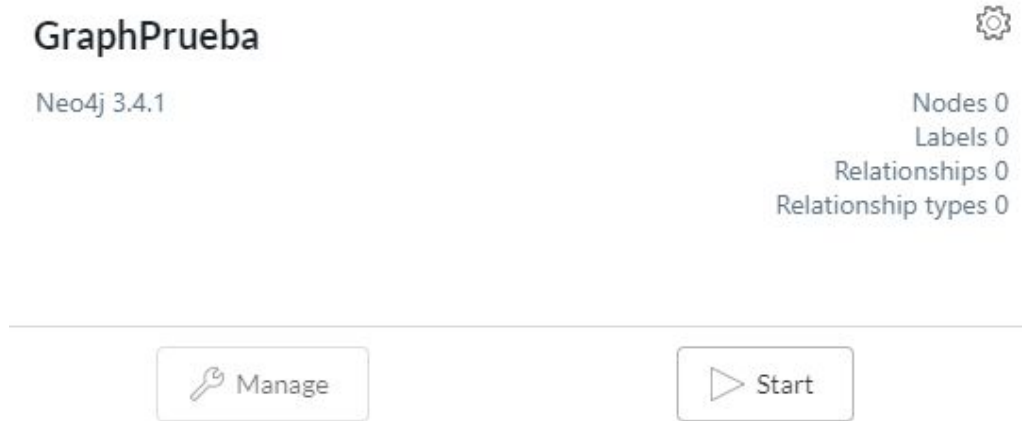
password

3.4.1

Cancel Create

Digitamos el nombre de la base de datos y su contraseña, también podemos seleccionar que versión de Neo4j queramos, pero este campo lo dejamos por defecto y damos clic en “Create”.

CREACIÓN Y CONFIGURACIÓN DE UNA BASE DE DATOS EN NEO4J



Damos clic en “Start” para empezar a desarrollar y trabajar en nuestra base de datos.

← → ↻ 🏠 ⓘ localhost:7474/browser/

📁 Aplicaciones 📁 Universidad 📁 Bancos 📁 Outlook 📁 Gmail 📁 Cursos Masivos Onlin 📁 Mis web 📁 Bases de datos cienti



\$

Database access not available. Please use `:server connect` to establish connection. There's a graph waiting for you.

\$:server connect

Connect to Neo4j

Database access requires an authenticated connection.

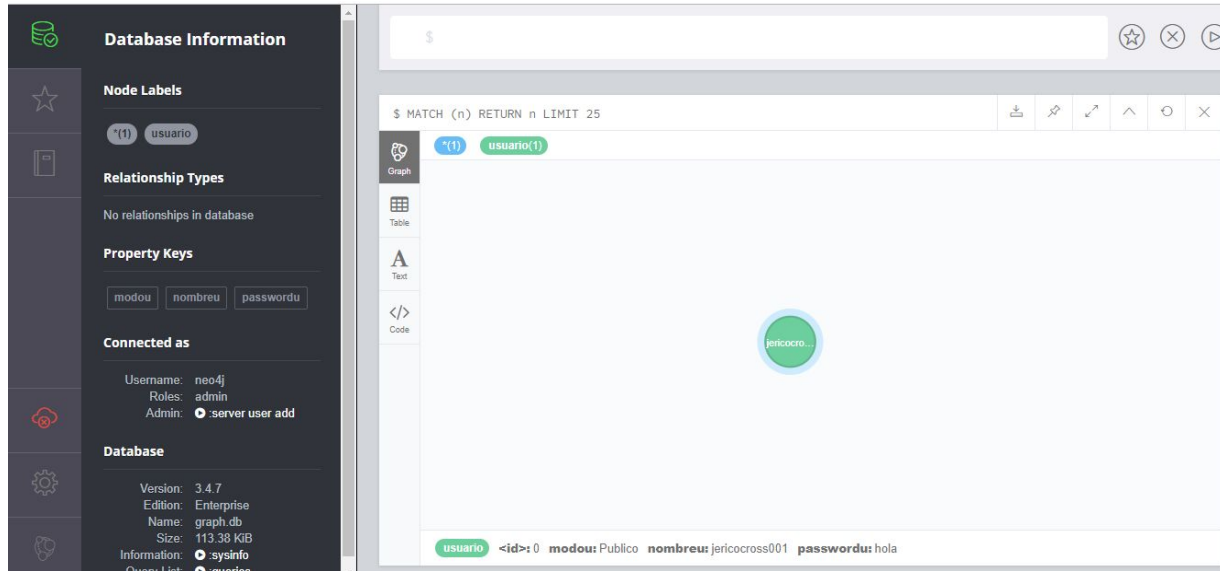
Host

Username

Password

Connect

AGREGA NODOS Y CONSULTAS



La barra superior donde aparece el signo pesos “\$” es el bash de neo4j que interpretará a Cypher

CONEXIÓN DE NEO4J CON PYTHON

En la construcción del proyecto en python “proyecto.py” se debe agregar `from py2neo import Graph, Node, Relationship` Librería de py2neo, aquella que permite definición de cada componente de esta y conexión con la base de datos

```
m=Graph("http://194.182.87.136:7474",host="194.182.87.136",password="bases12345")
```

Se realiza la conexión a la base de datos local con atributos con contraseña, esto extrae un grafo al cual llamamos m

MODELO DE DATOS

El modelo propuesto para la creación de la base de datos se encuentra en dirección: <http://194.182.87.136:7474/>

Username: neo4j

Password: bases12345

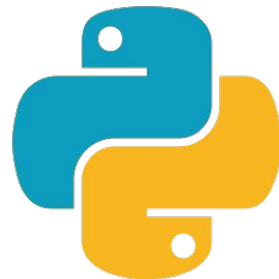
Utiliza la siguiente consulta para visualizar toda la base de datos “MATCH (n) RETURN n”



FUNCIONES EN PYTHON

En el proyecto se realizaron las siguiente funciones a manera especial:

- Ver mis seguidores
- Ver usuarios seguidos
- Personas que quizás conozcas
- Ver mis tweets
- Realizar tweet
- Ver tweets recibidos
- Explorar
- Buscar usuario
- Cambiar ajustes de privacidad
- Cambiar o establecer ciudad.
- Manejo de solicitudes de amistad



CONSULTAS DEL PROYECTO EN NEO4J A TRAVÉS DE PYTHON

- `MATCH (a:usuario) WHERE a.nombre="'+Nombre+'"`
`RETURN a`
- `MATCH (a:usuario)-[:viveen]->(b:ciudad) WHERE`
`a.nombre="'+user.getUsername()+'"` `RETURN`
`b.nombre AS Nombre`
- `MATCH (b:usuario)-[:follows {estado:['0']}]>(a:usuario)`
`where a.nombre="'+user.getUsername()+'"` `RETURN`
`b.nombre AS Nombre"`
- `MATCH (a:usuario)-[:follows {estado:['1']}]>(b:usuario)`
`where a.nombre="'+user.getUsername()+'"` `RETURN`
`b.nombre AS Nombre`
- `MATCH (b:usuario)-[:follows {estado:['1']}]>(a:usuario)`
`where a.nombre="'+user.getUsername()+'"` `RETURN`
`b.nombre AS Nombre`
- `MATCH (a:usuario)-[:follows`
`{estado:['1']}]>(b:usuario)-[:follows`
`{estado:['1']}]>(c:usuario)-[:follows`
`{estado:['1']}]>(d:usuario)-[:follows`
`{estado:['1']}]>(e:usuario)-[:follows`
`{estado:['1']}]>(f:usuario) WHERE`
`a.nombre="'+user.getUsername()+'"` `RETURN`
`f.nombre AS nombre`
- `MATCH (b:usuario)-[r:follows`
`{estado:['1']}]>(a:usuario) WHERE`
`a.nombre="'+nombre+'"` `and`
`b.nombre="'+node['nombre']+'"` `RETURN r`

CONSULTAS DEL PROYECTO EN NEO4J A TRAVÉS DE PYTHON

- `MATCH (a:usuario) WHERE a.nombreu="'+nombre+'" RETURN a`
- `MATCH (a:ciudad) WHERE a.nombre="'+str(ciudad)+'" RETURN a`
- `MATCH (a:usuario)-[r:viveen]->(b:ciudad) WHERE a.nombreu="'+user.getusername()+'" and b.nombreu="'+user.getciudad()+'" DELETE r`
- `MATCH (a:usuario) WHERE a.nombreu="'+user.getusername()+'" SET a.modou="'+user.getusermode()+'"'`
- `MATCH (a:usuario)-[:creatweet]->(b:tweet) where a.nombreu="'+nombre+'" RETURN b`
- `MATCH (a:usuario)-[:creatweet]->(t:tweet) WHERE a.nombreu = '"+x+"' RETURN t`
- `MATCH (a)-[:enviaa]->(b) WHERE b.nombreu = '"+user.getusername()+'" RETURN a`
- `MATCH (b:usuario)-[r:follows {estado:['0']}]>(a:usuario) WHERE a.nombreu="'+user.getusername()+'" and b.nombreu="'+user.getinvitations()[0-1]+' SET r.estado =['1']`



POR SU ATENCIÓN