In [34]:

```
                                    Universidad Politénica Salesiana
Estudiante: Rayner Palta

   Diseñe y desarrolle un sistema recopilador que permita obtener las noticias
Webscraping es la técnica de extraer datos contenidos en un formato no estru
Es por ello, que se desea crear nuevos métodos que permitan la recopilación
En base a ello, vamos a obtener los datos de lo que esta hablando las notici

```

In [ ]:

```
from neo4j import GraphDatabase
class Neo4jService(object):
        def __init__(self, uri, user, password):
        self._driver = GraphDatabase.driver(uri, auth=(user, password))
        #self._driver = GraphDatabase.driver("neo4j://localhost:7687",
        auth=("ne 6    def close(self):
      self._driver.close()
  #nodos principales
   def nodo_cabeceraTitulo(self, tx, nombre):
   tx.run("CREATE (:Titulo {nombre: $nombre})", nombre=nombre) 11    def
   nodo_candidatoAsambleista(self, tx, nombre):
      tx.run("CREATE (:Candidato {nombre: $nombre})", nombre=nombre)
  def nodo_noticia(self, tx, nombre):
      tx.run("CREATE (:Noticias {nombre: $nombre})", nombre=nombre) 15
def nodo_fechaNoticia(self, tx, nombre):
      tx.run("CREATE (:Fecha {nombre: $nombre})", nombre=nombre)

  #nodos secundarios
    def nodo_contenidoNoticia(self, tx, nombre):
    tx.run("CREATE (:Contenido {nombre: $nombre})", nombre=nombre)   21
    def nodo_titulosNoticias(self, tx, nombre):
      tx.run("CREATE (:Titulos {nombre: $nombre})", nombre=nombre) 23
def nodo_fechasNoticias(self, tx, nombre):
      tx.run("CREATE (:Fechas {nombre: $nombre})", nombre=nombre)

    #relacionaion
    def relacion_noticia(self, tx, nombre_noticias, nombre_noticia):
      tx.run("MATCH (a:Noticias {nombre: $nombre_noticias}) " 29
"MATCH (b:Contenido {nombre: $nombre_noticia}) " 30
"MERGE (a)-[:Noticias]->(b)",
31              nombre_noticias=nombre_noticias, nombre_noticia=nombre_noticia) 32
def relacion_titulosCabeceras(self, tx, nombre_titulos, nombre_titulo):
33      tx.run("MATCH (a:Titulo {nombre: $nombre_titulos}) " 34
"MATCH (b:Titulos {nombre: $nombre_titulo}) " 35
"MERGE (a)-[:TituloNoticia]->(b)",
36              nombre_titulos=nombre_titulos, nombre_titulo=nombre_titulo) 37
def relacion_fechaNoticias(self, tx, tiempo_fechas, nombre_fecha):
38      tx.run("MATCH (a:Fecha {nombre: $tiempo_fechas}) " 39
"MATCH (b:Fechas {nombre: $nombre_fecha}) " 40
"MERGE (a)-[:Fecha]->(b)",
41              tiempo_fechas=tiempo_fechas, nombre_fecha=nombre_fecha)
42
43    #relacion de candidato   titulo fecha contenido Noticia
44    def relacion_candidato_tituloNoticia(self, tx, candidato, titulo_noticia):
45      tx.run("MATCH (a:Candidato {nombre: $candidato}) " 46
"MATCH (b:Titulo {nombre: $titulo_noticia}) " 47              "MERGE (a)-
[:TituloNoticia_Candidato]->(b)",
48              candidato=candidato, titulo_noticia=titulo_noticia)
49
50   def relacion_candidato_fechaNoticia(self, tx, candidato, fecha_noticia):
51      tx.run("MATCH (a:Candidato {nombre: $candidato}) " 52
"MATCH (b:Fecha {nombre: $fecha_noticia}) " 53
"MERGE (a)-[:FechaNoticia_Candidato]->(b)",
54              candidato=candidato, fecha_noticia=fecha_noticia)
55
56   def relacion_candidato_contenidoNoticia(self, tx, candidato, noticia_conteni
```

```
57         tx.run("MATCH (a:Candidato {nombre: $candidato}) " 58 
"MATCH (b:Noticias {nombre: $noticia_contenido}) " 59                "MERGE 
(a)-[:Noticias_Candidato]->(b)",
60                 candidato=candidato, noticia_contenido=noticia_contenido)
```

In [35]:

```
import requests
from bs4 import BeautifulSoup
neo4j = Neo4jService('bolt://localhost:7687', 'neo4j', 'neo4jj') 4 with
neo4j._driver.session() as session:
    session.write_transaction(neo4j.nodo_candidatoAsambleista , "Candidato")
    session.write_transaction(neo4j.nodo_cabeceraTitulo , "Titulos")
    session.write_transaction(neo4j.nodo_fechaNoticia , "Fecha")
    session.write_transaction(neo4j.nodo_noticia , "Noticias") 9
    print('Nodos creados')
10    with open("/Users/rayner/Downloads/links.txt","r") as archivo:
11    for linea in archivo:
12       #print(linea)
13        pageGoo= requests.get(linea)
14        soupGoo = BeautifulSoup(pageGoo.content,'html.parser')
15        resultaFecha = soupGoo.find_all("span",{"class":"r0bn4c rQMQod"})
16        resultaTitulo = soupGoo.find_all("div",{"class":"BNeawe vvjwJb AP7Wnd"})
17        resultaTexto = soupGoo.find_all("div",{"class":"kCrYT"})
18        print("-----INICIO SCRAPING GOOGLE------") 19
          contenidoTextoGoogle=list() 20        for resu in resultaTexto:
21          contenidoTextoGoogle.append(resu.text)
22      auxGo=list(set(contenidoTextoGoogle)) 23
for ll in auxGo:
24            session.write_transaction(neo4j.nodo_contenidoNoticia,ll)
25            session.write_transaction(neo4j.relacion_noticia,"Noticias",ll) 26
              tituloGoo=list()
27      for i in resultaTitulo: 28
tituloGoo.append(i.text) 29        for
titulo in tituloGoo:
30            session.write_transaction(neo4j.nodo_titulosNoticias,titulo)
31            session.write_transaction(neo4j.relacion_titulosCabeceras,"Titulos",
32            #print(titulo) 33        enlaceFecha=list() 34        for e2 in
              resultaFecha: 35            enlaceFecha.append(e2.text) 36
              for fecha in enlaceFecha:
37            session.write_transaction(neo4j.nodo_fechasNoticias,fecha)
38            session.write_transaction(neo4j.relacion_fechaNoticias,"Fecha",fecha
              39
40 session.write_transaction(neo4j.relacion_candidato_tituloNoticia,"Candidato","Ti
41 session.write_transaction(neo4j.relacion_candidato_contenidoNoticia,"Candidato",
42 session.write_transaction(neo4j.relacion_candidato_fechaNoticia,"Candidato", "Fe
43 print("fin del proceso")
```

```
Nodos creados
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
```

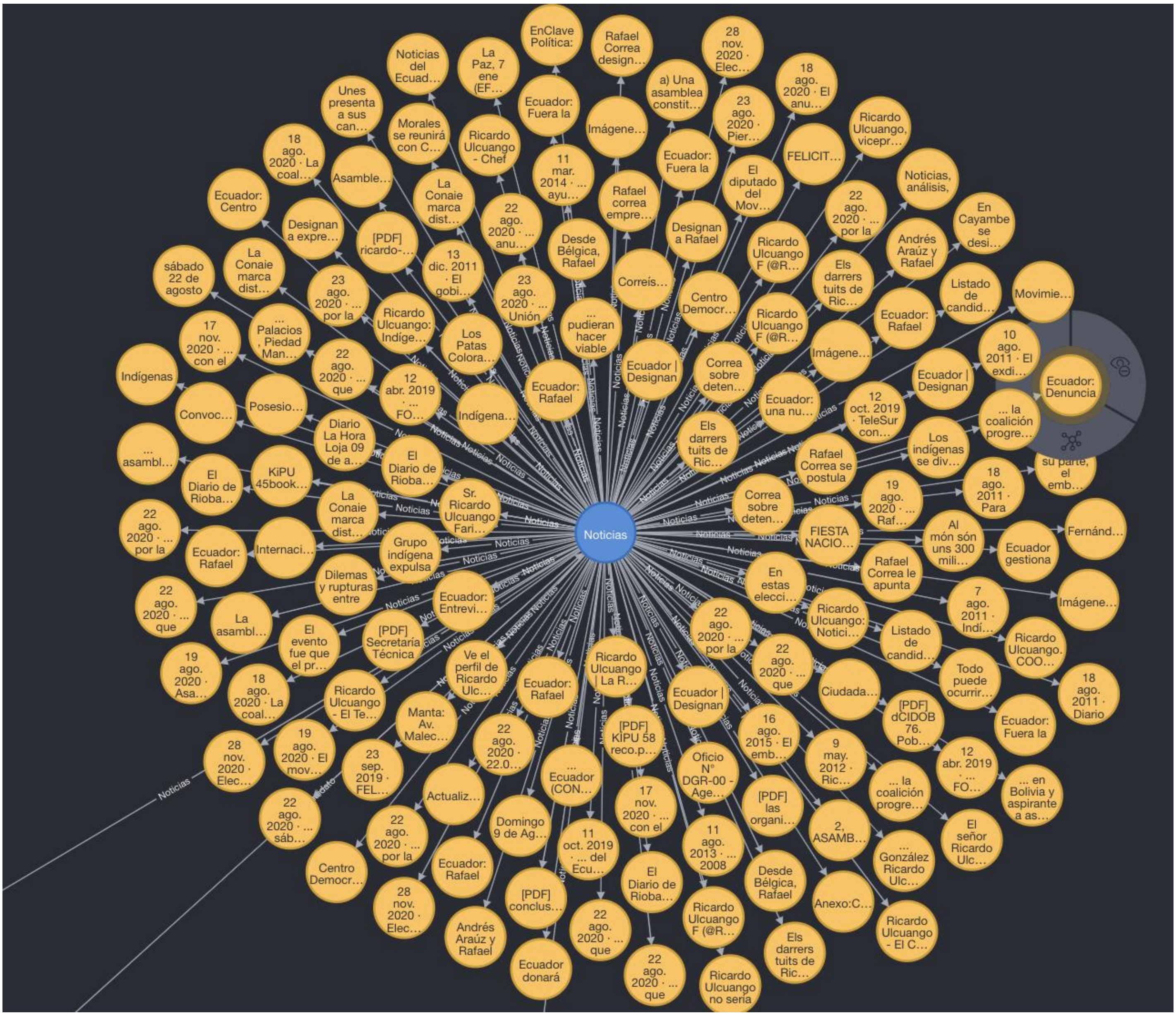
```
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------
-----INICIO SCRAPING GOOGLE------ ---
--INICIO SCRAPING GOOGLE------ fin
del proceso
```

In [ ]:

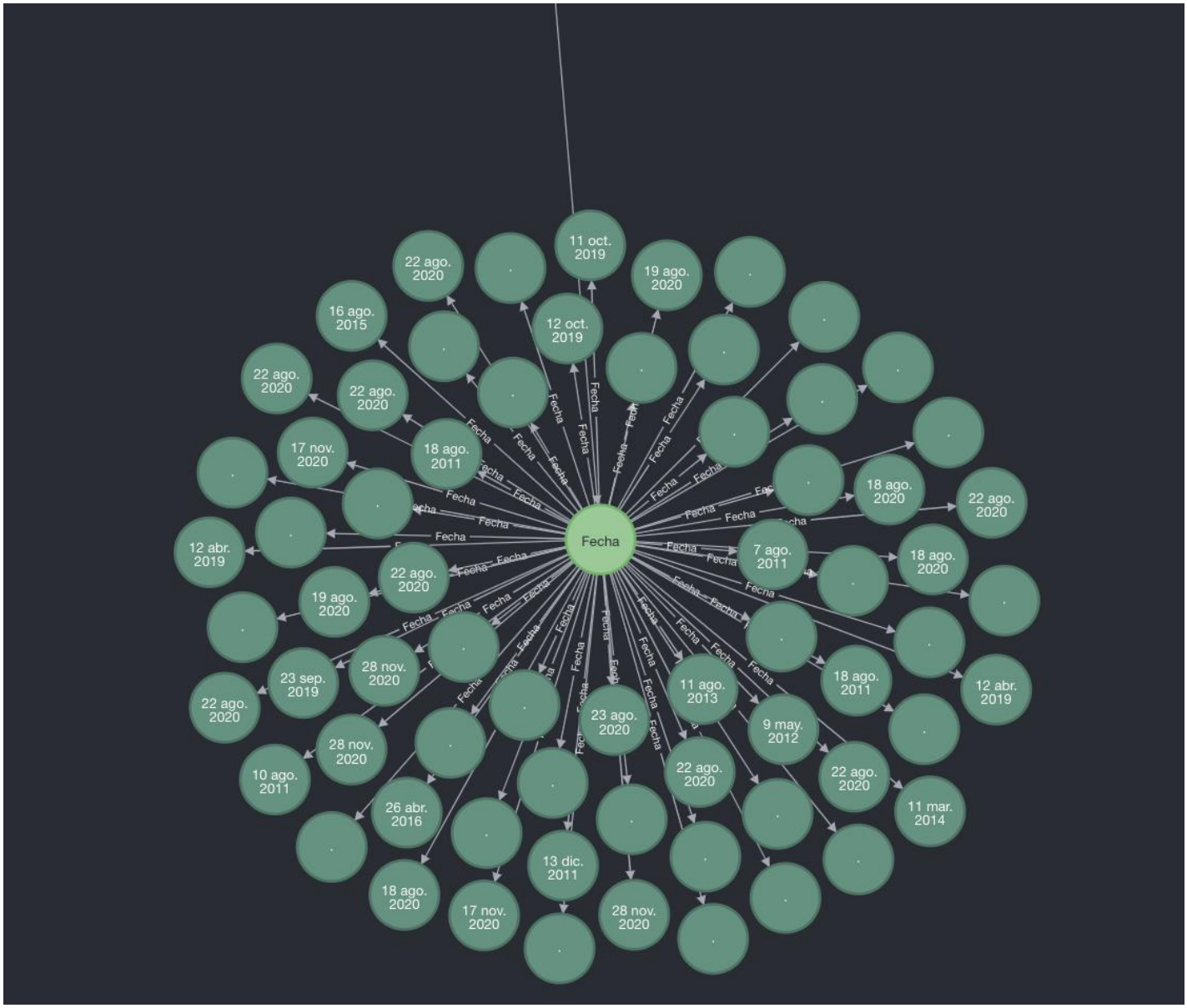
```

```

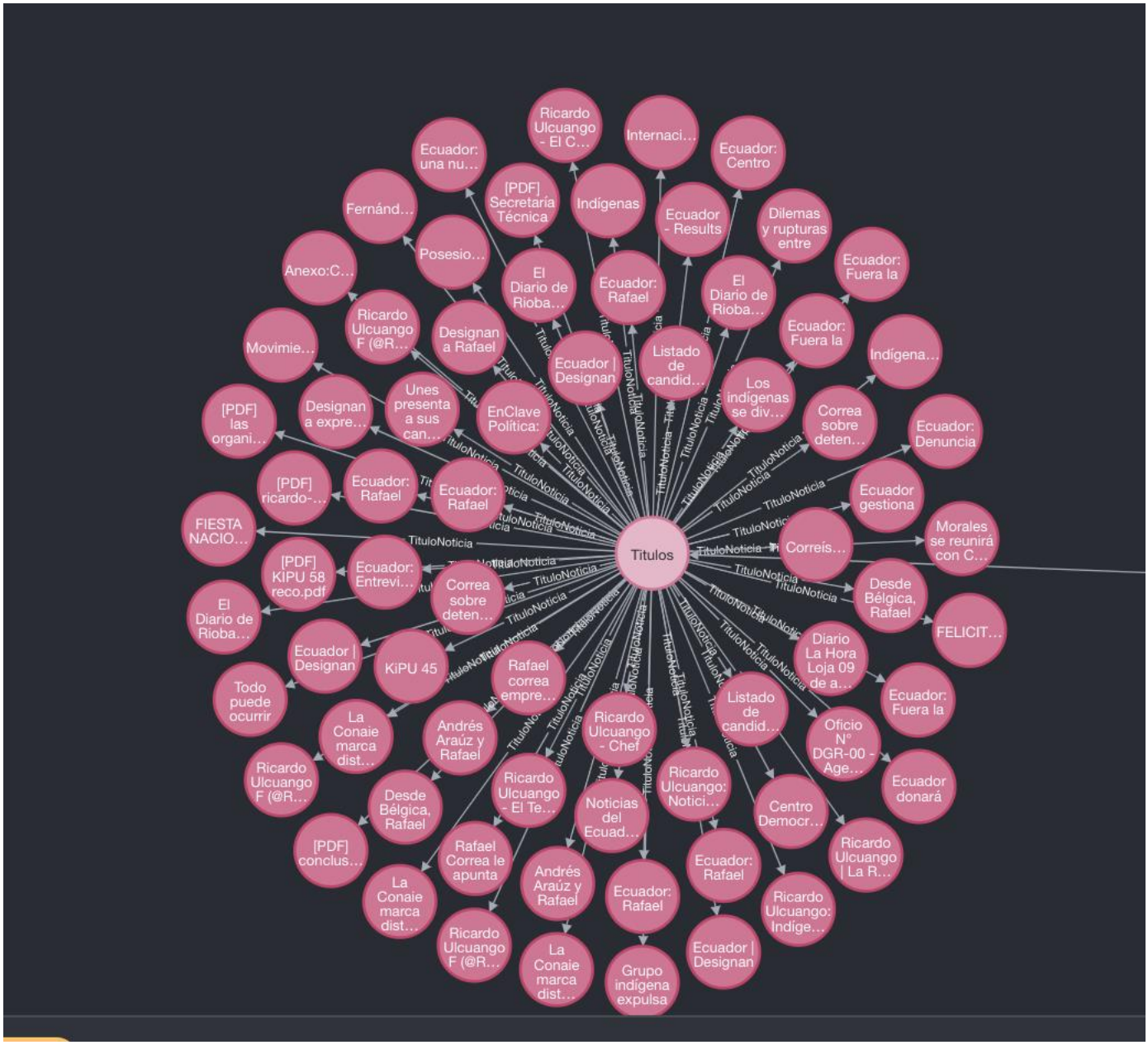
ANEXOS



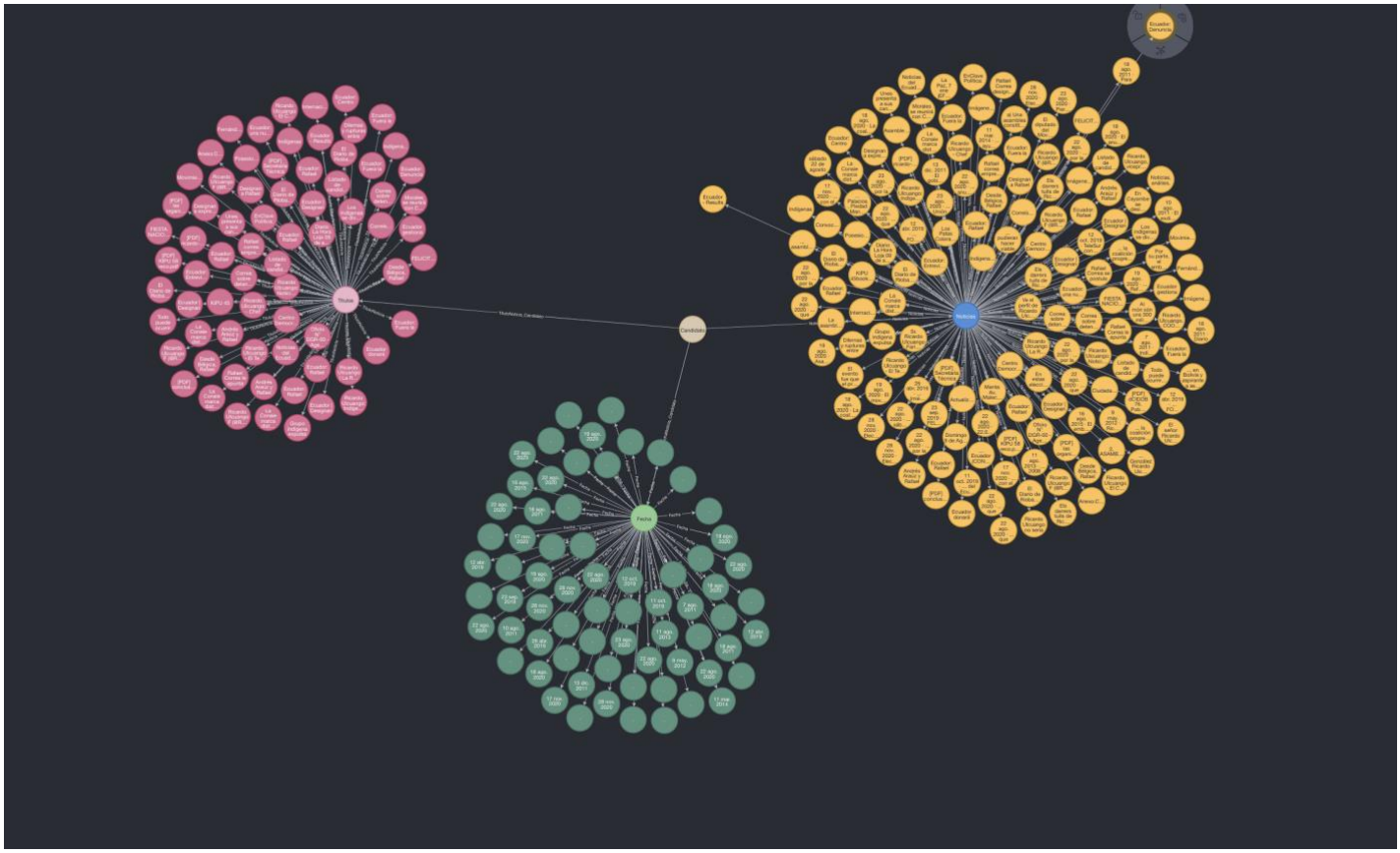
Creacion del nodo noticias con las respectivos nodos



Creacion del nodo fechas y sus relaciones



Creacion del nodo Titulo y sus respectivos documentos



Los tres nodos principales relacionados con el nodo Candidato