Aplicaciones de Datos en Redes Complejas

Trabajo Parcial

Caparachin Villaverde, Nanto Gustavo (E202110812)
García Godos Villavicencio, Jorge Daniel (E202110981)
Huamani Franco, Ismael (E202110991)
Poma Ludeña, Rodrigo Renato (E202111018)

Agenda

- Instrucciones
- Primera Propuesta de Base de Datos
  - Descripción de la Información
    - Descripción de los nodos
    - Descripción de las relaciones
  - Volumen de la Información
  - Construcción de la Base de Datos de Grafos
  - Ingesta de Datos
  - Consultas de Interés de la Base de Datos de Grafo
- Segunda Propuesta de Base de Datos
  - Descripción de la Información
  - Análisis y Limpieza de la Información
  - Construcción de la Base de Datos de Grafos
  - Metadatos de la Base de Datos de Grafos
  - Consultas de Interés de la Base de Datos de Grafo

Instrucciones

Desarrollar una base de datos en Neo4j con las siguientes características:

- Contener como mínimo 50 nodos; no hay un número máximo de nodos.
- Contener como mínimo 40 relaciones entre nodos (en total)
- Cada nodo debe tener como mínimo una propiedad.
- Debe haber como mínimo 3 etiquetas.
- Debe haber como mínimo 3 tipos de relaciones entre nodos.
- Los temas recomendados son los siguientes:
    1. Ventas(teniendo la lista de productos, y la cantidad de ventas de cada producto)
    2. Visitas de página web (teniendo como métrica el horario de visita y la página visitada)
    3. Comentarios en una red social (teniendo una clasificación como positivos o negativos.
- Opcionalmente se puede escoger otro tema, teniendo en mente los datos dependientes e independientes a fin de desarrollar un modelo predictivo en la sesión 12.

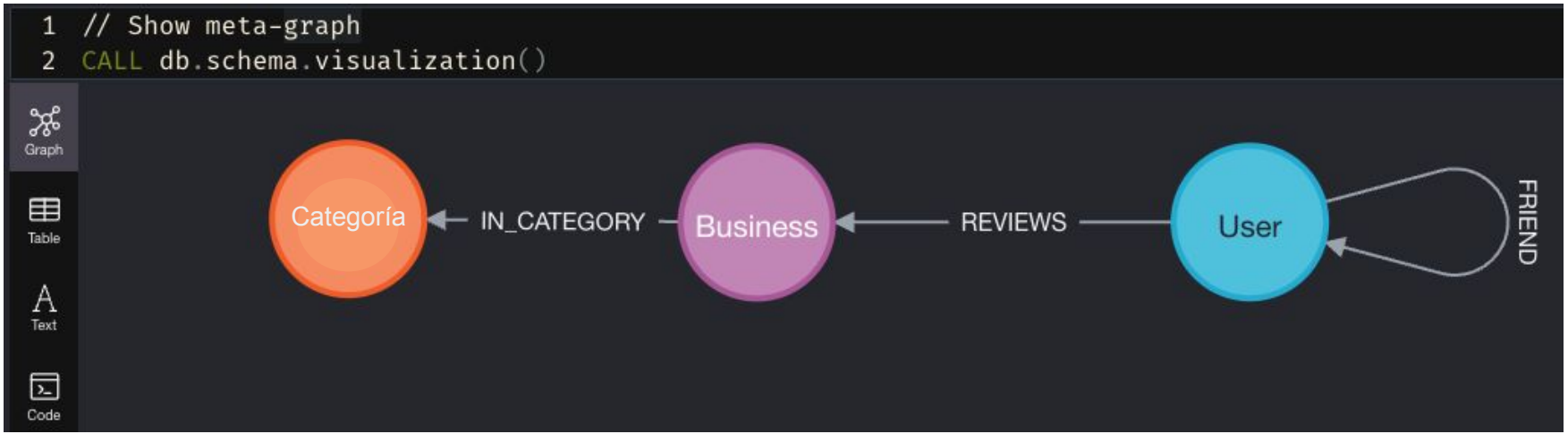
Realizar consultas de interés acerca del grafo que expliquen la relación entre los nodos de distintas etiquetas.

Primera Propuesta de Base de Datos

Descripción de la Información

La información que decidimos utilizar es un fragmento de los datos de las diferentes revisiones de negocios en la plataforma YELP.

Los datos tienen la siguiente estructura:



Descripción de los NODOS

User


```
{
  "yelping_since": "2007-01-25 16:47:26",
  "cool": 5994,
  "name": "Walker",
  "id": "qVc8ODYU5SZjKXVBgXdI7w",
  "useful": 7217,
  "funny": 1259,
  "fans": 267
}
```


Business

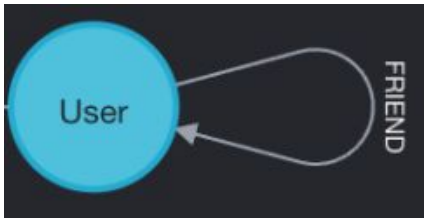

```
{
  "address": "1616 Chapala St, Ste 2",
  "city": "Santa Barbara",
  "is_open": false,
  "latitude": 34.4266787,
  "name": "Abby Rappoport, LAC, CMQ",
  "stars": 5.0,
  "id": "Pns2l4eNsfO8kk83dixA6A",
  "state": "CA",
  "longitude": -119.7111968
}
```


Category


```
{
  "id": "Doctors"
}
```

Descripción de los RELACIONES

FRIENDS

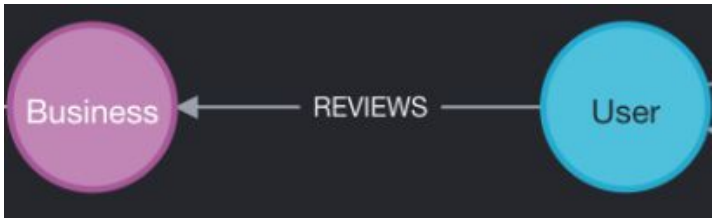


Relationship properties

FRIEND

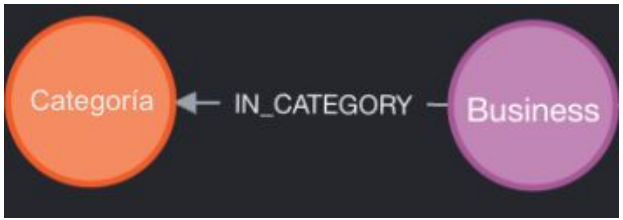
<id> 673086

REVIEWS



<id>	22072106
date	2016-09-27 21:42:25
stars	5.0
text	I've used this UPS location on numerous occasions and have always had friendly, reliable service, whether it be shipping an item, faxing or any other ... Show all

IN_CATEGORY

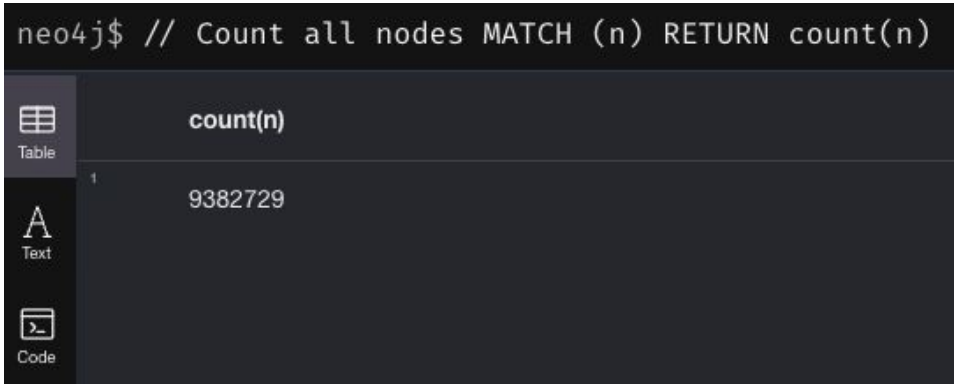


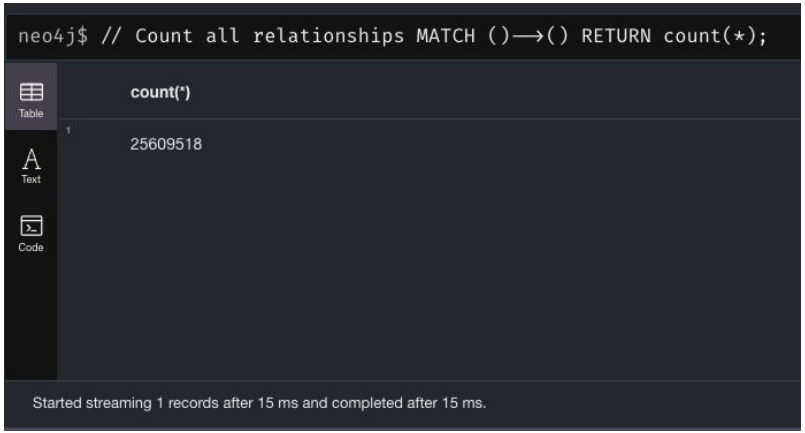
Relationship properties

IN_CATEGORY

<id> 660449

Volumen de la información





Tamaño en archivos : 8GB
Tamaño de base de datos en Neo4j : 42GB

Construcción de la Base de Datos de Grafos (Transformación)

User

```
data = json.loads(file//:user.json)
user['user_id'] = data['user_id']
user['name'] = data['name']
user['yelping_since'] = data['yelping_since']
user['useful'] = data['useful']
user['funny'] = data['funny']
user['cool'] = data['cool']
user['friends'] = str(data['friends']).split(', ')
user['fans'] = data['fans']
```

Review

```
data = json.loads(line)
review['review_id'] = data['review_id']
review['user_id'] = data['user_id']
review['business_id'] = data['business_id']
review['stars'] = data['stars']
review['useful'] = data['useful']
review['funny'] = data['funny']
review['cool'] = data['cool']
review['text'] = data['text']
review['date'] = data['date']
```

Business

```
data = json.loads(line)
business_line['business_id'] = data['business_id']
business_line['name'] = data['name']
business_line['address'] = data['address']
business_line['city'] = data['city'].title()
business_line['state'] = str(data['state']).upper()
business_line['postal_code'] = data['postal_code']
business_line['latitude'] = data['latitude']
business_line['longitude'] = data['longitude']
business_line['stars'] = data['stars']
business_line['categories'] = str(data['categories']).split(', ')
```

Ingesta de datos

```
LOAD CSV WITH HEADERS FROM "file://busines.csv" AS row
MERGE (b: Business { id: row.business_id })
SET b.name = row.name
SET b.address = row.address
SET b.city = row.city
SET b.state = row.state
SET b.postal_code = row.postal_code
SET b.latitude = row.latitude
SET b.longitude = row.longitude
SET b.longitude = row.longitude
SET b.stars = row.stars
SET b.categories = row.categories;

LOAD CSV WITH HEADERS FROM "file://user.csv" AS row
MERGE (u: User { id: row.user_id })
SET u.name = row.name
SET u.yelping_since = row.yelping_since
SET u.useful = row.useful
SET u.funny = row.funny
SET u.cool = row.cool
SET u.friends = row.friends
SET u.fans = row.fans
```

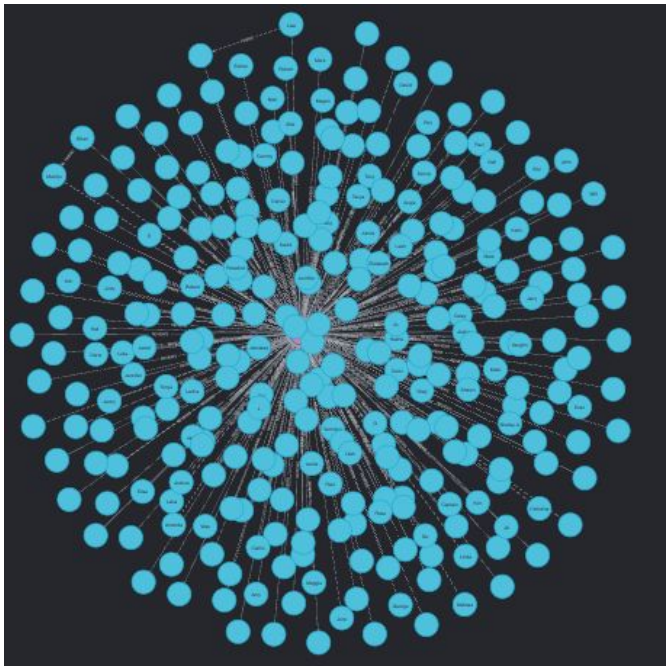
```
MATCH (source: business { id: business.id })
MATCH (target: categories { id: business.categories })
MERGE (business)-[:IN_CATEGORY]->(categories);

MATCH (source: user { id: user.id })
MATCH (target: user { id: source.friends })
MERGE (source)-[:FRIEND]-(target);

LOAD CSV WITH HEADERS FROM "file://review.csv" AS row
MATCH (source: user { id: user.id })
MATCH (source: business { id: business.id })
MERGE (r: Review { id: row.review_id })
SET r.name = row.name
SET r.stars = row.stars
SET r.useful = row.useful
SET r.funny = row.funny
SET r.cool = row.cool
SET r.text = row.text
SET r.date = row.date
MERGE (source)-[r:REVIEWS]->(target);
```

Consultas de Interés de la Base de Datos de Grafo

OBTENER EL NEGOCIO CON MÁS REVIEWS.

```
MATCH (u)-[:REVIEWS]->(b)
RETURN b, COLLECT(u) as user
ORDER BY SIZE(user) DESC LIMIT 10
```


```
{
  "identity": 32225,
  "labels": [
    "Business"
  ],
  "properties": {
    "address": "441 Royal St",
    "city": "New Orleans",
    "is_open": true,
    "latitude": 29.95647323,
    "name": "Royal House",
    "stars": 4.0,
    "state": "LA",
    "id": "VQcCL9PiNL_wkGf-uF3fjg",
    "longitude": -90.066386051
```

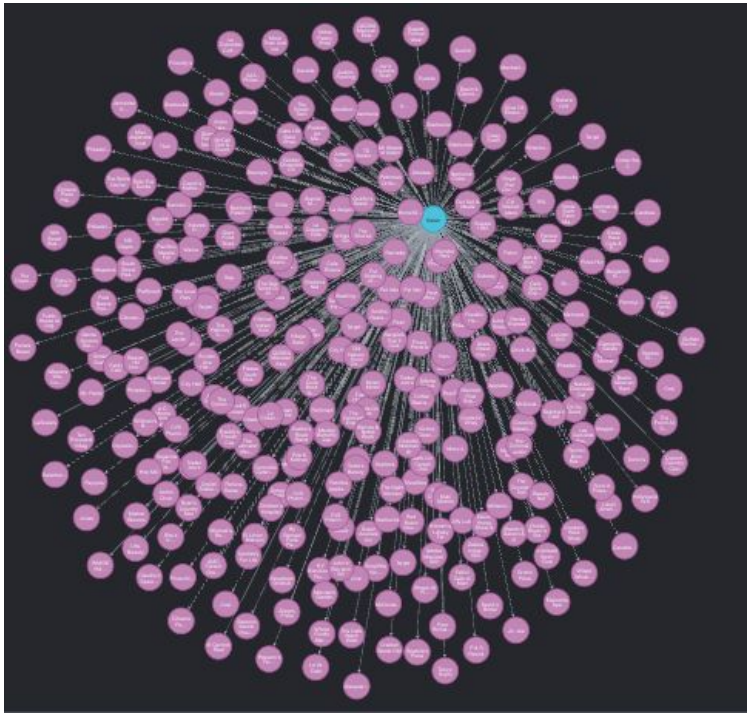



Consultas de Interés de la Base de Datos de Grafo

OBTENER EL USUARIO CON MÁS REVIEWS.

```
MATCH (u)-[:REVIEWS]->(b)
RETURN u, COLLECT(b) as bussines
ORDER BY SIZE(bussines) DESC LIMIT 1
```


```
{
  "identity": 173608,
  "labels": [
    "User"
  ],
  "properties": {
    "yelping_since": "2008-05-29 12:29:54",
    "cool": 9759,
    "name": "Karen",
    "id": "_BcWyKQL16ndpBdggh2kNA",
    "useful": 16950,
    "funny": 5203,
    "fans": 558
  },
  "elementId": "173608"
}
```

Consultas de Interés de la Base de Datos de Grafo

OBTENER LOS USUARIOS MÁS RELEVANTE (PAGE RANK)

```
1 CALL gds.pageRank.stream('yelp')
2 YIELD nodeId, score
3 RETURN gds.util.asNode(nodeId).name AS name, score
4 ORDER BY score DESC, name ASC LIMIT 10
```

Table | Text | Code

	name	score
1	*null*	7524.944999991683
2	"Michelle"	15.21007410384705
3	*null*	14.990037071481838
4	"Morgan"	13.791469817727835
5	"Morris"	10.104882211500028
6	*null*	9.547438445902989
7		

Started streaming 10 records after 1 ms and completed after 12468 ms.

Los nodos con name null son bussines o categorías

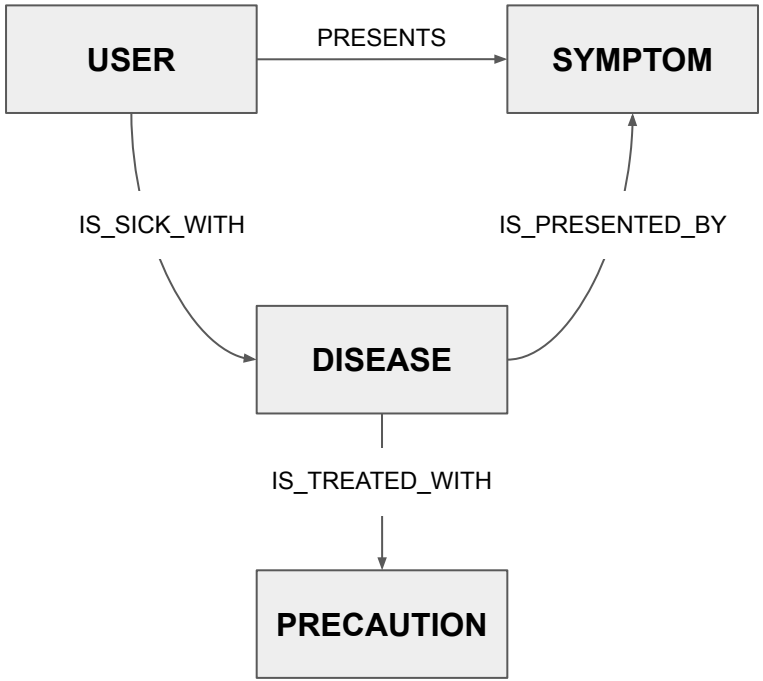
Segunda Propuesta de Base de Datos

Diagnóstico de Enfermedades

Descripción de la Información

Conjunto de datos relacionado a la atención médica mediante plataformas digitales. Tiene información sobre los usuarios que han hecho uso de la aplicación, sus síntomas, las enfermedades diagnosticadas y las precauciones a tomar. De este conjunto de datos, se puede construir un modelo predictivo, que utilice los síntomas descritos por el usuario y pueda indicar que enfermedad presenta. Adicional a ello, se puede ofrecer un conjunto de precauciones a considerar según la enfermedad predicha.

- Variables independientes: síntomas
- Variable dependiente: enfermedad



Construcción de la Base de Datos de Grafos

Nodes:

```
// CREATE DISEASE NODE
LOAD CSV WITH HEADERS FROM
"https://raw.githubusercontent.com/RodrigoRenatoPomaLudena/disease_complex_networks_neo4j/master/dataset_cleaned/disease_node.csv" AS row
MERGE (disease:Disease {disease:row.disease})
SET disease.description=row.description


// CREATE PRECAUTION NODE
LOAD CSV WITH HEADERS FROM
"https://raw.githubusercontent.com/RodrigoRenatoPomaLudena/disease_complex_networks_neo4j/master/dataset_cleaned/precaution_node.csv" AS row
MERGE (precaution:Precaution {precaution:row.precaution})
SET precaution.weight=row.weight


// CREATE SYMPTOM NODE
LOAD CSV WITH HEADERS FROM
"https://raw.githubusercontent.com/RodrigoRenatoPomaLudena/disease_complex_networks_neo4j/master/dataset_cleaned/symptom_node.csv" AS row
MERGE (symptom:Symptom {symptom:row.symptom})
SET symptom.weight=row.weight


// CREATE USER  NODE
LOAD CSV WITH HEADERS FROM
"https://raw.githubusercontent.com/RodrigoRenatoPomaLudena/disease_complex_networks_neo4j/master/dataset_cleaned/user_node.csv" AS row
MERGE (user:User {user:row.user})
SET user.email=row.email, user.postalZip=row.postalZip,
user.region=row.region, user.country=row.country
```

Relationships:

```
// CREATE USER AND DISEASE RELATIONSHIP
LOAD CSV WITH HEADERS FROM
"https://raw.githubusercontent.com/RodrigoRenatoPomaLudena/disease_complex_networks_neo4j/master/dataset_cleaned/disease_and_user_node.csv" AS row
MATCH (user:User) WHERE user.user = row.user
MATCH (disease:Disease) WHERE disease.disease = row.disease
MERGE (user)-[:IS_SICK_WITH]->(disease)


// CREATE USER AND SYMPTOM RELATIONSHIP
LOAD CSV WITH HEADERS FROM
"https://raw.githubusercontent.com/RodrigoRenatoPomaLudena/disease_complex_networks_neo4j/master/dataset_cleaned/user_and_symptom_node.csv" AS row
MATCH (user:User) WHERE user.user = row.user
MATCH (symptom:Symptom) WHERE symptom.symptom = row.symptom
MERGE (user)-[:PRESENTS]->(symptom)

//CREATE DISEASE AND PRECAUTION RELATIONSHIP
LOAD CSV WITH HEADERS FROM
"https://raw.githubusercontent.com/RodrigoRenatoPomaLudena/disease_complex_networks_neo4j/master/dataset_cleaned/precaution_and_symptom_node.csv" AS
row
MATCH (disease:Disease) WHERE disease.disease = row.disease
MATCH (precaution:Precaution) WHERE precaution.precaution = row.precaution
MERGE (disease)-[:IS_TREATED_WITH]->(precaution)

//CREATE DISEASE AND SYMPTOM RELATIONSHIP
LOAD CSV WITH HEADERS FROM
"https://raw.githubusercontent.com/RodrigoRenatoPomaLudena/disease_complex_networks_neo4j/master/dataset_cleaned/disease_and_symptom_node.csv" AS row
MATCH (disease:Disease) WHERE disease.disease = row.disease
MATCH (symptom:Symptom) WHERE symptom.symptom = row.symptom
MERGE (disease)-[:IS_PRESENTED_BY]->(symptom)
```

Metadatos de la Base de Datos de Grafos

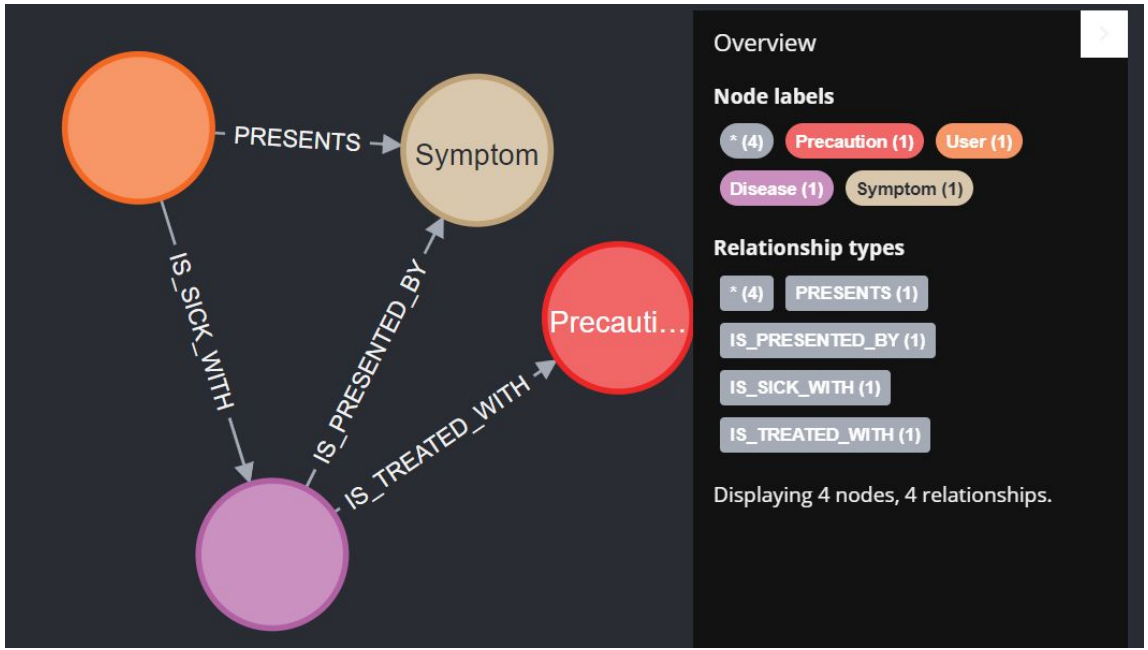
El grafo presenta 4 nodos interconectados, donde: el usuario presenta síntomas y haciendo uso de estos, puede determinarse la enfermedad que padece, para posteriormente poder darle precauciones; las cuales permitan salvaguardar su bienestar.

La cantidad de nodos por etiqueta son:

- Disease: 41
- Precaution: 96
- Symptom: 132
- User: 4920

La cantidad de relaciones entre nodos son:

- PRESENTS: 36276
- IS_SICK_WITH: 4553
- IS_TREATED_WITH: 150
- IS_PRESENTED_BY: 298



Consultas de Interés de la Base de Datos de Grafo

La enfermedad con mayor variedad de síntomas (sintomatología variada)

```
1 MATCH (d:Disease)-[r]-(s:Symptom)
2 WITH d.disease AS disease, COUNT(s) AS quantity
3 RETURN *
4 ORDER BY quantity DESC
5 LIMIT 1
```

Table

	disease	quantity
1	"common cold"	17

```
1 MATCH (d:Disease)-[r]-(s:Symptom)
2 WITH d, s
3 WHERE d.disease = 'common cold'
4 RETURN s.symptom, s.weight
5 ORDER BY s.weight DESC
```

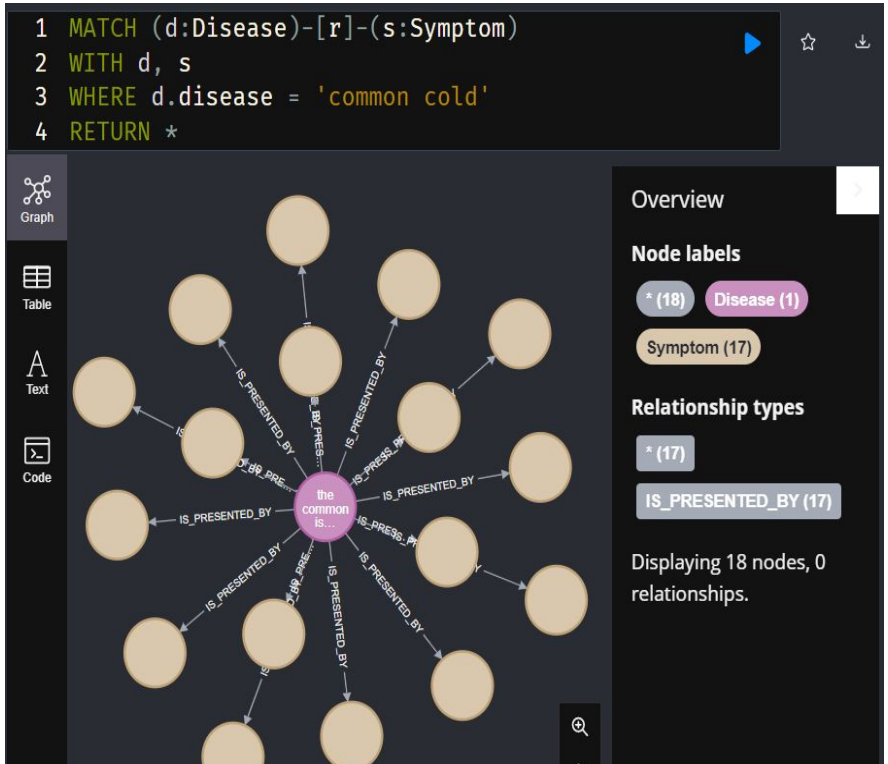
Table

	s.symptom	s.weight
1	"high fever"	"7"
2	"chest pain"	"7"

Text

Code

```
1 MATCH (d:Disease)-[r]-(s:Symptom)
2 WITH d, s
3 WHERE d.disease = 'common cold'
4 RETURN *
```



Consultas de Interés de la Base de Datos de Grafo

Las enfermedades y síntomas más comunes por país de origen del usuario

```
MATCH (u:User)-[r]-(d:Disease)
RETURN u.country, d.disease, COUNT(r) AS
ocurrency
ORDER BY ocurrency DESC
```

Table | Text | Code

	u.country	d.disease	ocurrency
1	"United States"	"acne"	11
2	"Austria"	"hepatitis b"	11
3	"Netherlands"	"allergy"	9
4	"United States"	"chronic cholestasis"	9
5	"Belgium"	"peptic ulcer diseae"	9
6	"Vietnam"	"common cold"	9

```
MATCH (u:User)-[r]-(s:Symptom)
RETURN u.country, s.symptom, COUNT(r) AS
ocurrency
ORDER BY ocurrency DESC
```

Table | Text | Code

	u.country	s.symptom	ocurrency
1	"Peru"	"vomiting"	73
2	"United States"	"vomiting"	72
3	"Vietnam"	"fatigue"	72
4	"Peru"	"fatigue"	71
5	"Indonesia"	"fatigue"	68
6	"Vietnam"	"vomiting"	67

Consultas de Interés de la Base de Datos de Grafo

Las enfermedades con mayor severidad de síntomas (sintomatología aguda) y sus precauciones

```
MATCH (d:Disease)-[r]-(s:Symptom)
WITH d, COUNT(s.symptom) AS n_symp,
SUM(toFloat(s.weight)) AS severity
RETURN d.disease, n_symp, severity,
ROUND(severity/n_symp,2) AS avg_severity
ORDER BY avg_severity DESC
```

Table | Text | Code

	d.disease	n_symp	severity	avg_severity
1	"urinary tract infection"	3	16.0	5.33
2	"aids"	4	21.0	5.25
3	"hepatitis e"	13	62.0	4.77
4	"heart attack"	4	19.0	4.75
5	"pneumonia"	11	52.0	4.73
6	"tuberculosis"	16	75.0	4.69

```
MATCH (d:Disease)-[r]-(p:Precaution)
WITH d, p
WHERE d.disease IN ['urinary tract infection',
'aids']
RETURN d.disease, p.precaution
ORDER BY d.disease ASC
```

Table | Text | Code

"d.disease"	"p.precaution"
"aids"	"consult doctor"
"aids"	"follow up"
"aids"	"avoid open cuts"
"aids"	"wear ppe if possible"
"urinary tract infection"	"drink plenty of water"
"urinary tract infection"	"take probiotics"
"urinary tract infection"	"increase vitamin c intake"
"urinary tract infection"	"drink cranberry juice"