Data Science Mission

Competencies : similarity indicators, skills and knowledge mapping of different jobs

Presented By:
BANTHITA BOONPUN, NATHAN DESTREZ,
KRITI SHARMA, BOHAN WANG

Mentors:
JULIEN CAPITAINE
MARIE JOUBERT

Agenda

25.03.2023

01 Business Problem

02 Project Goal

03 Code Snippet

04 Solutions

05 Future Applications

Business Problem

- Inadequate streamlined process and organizational structure of data
- Takes too long to establish relationships between skills/knowledge and occupations
- Pressing need for a similarity that can identify and analyze the intersection of skills/knowledge and relevant occupations

Project Goal

- Collect data from the ESCO website
- Store cleaned and transformed data into a more flexible solution to be able to create a hierarchy in AI for future use and future analysis

Code Snippet

Entrée [17]:
```
response = requests.get(df_DataJob['_links_href'][0])
response = response.json()
```

Entrée [18]:
```
response.keys()
```

Out[18]:
```
dict_keys(['className', 'classId', 'uri', 'title', 'referenceLanguage', 'preferredLabel', 'alternativeLabel', 'preferredTerm', 'alternativeTerms', 'description', 'status', 'code', 'codes', '_links', '_embedded'])
```

Entrée [19]:
```
response['_links']
```

Out[19]:
```
{'self': {'href': 'https://ec.europa.eu/esco/api/resource/occupation?uri=http://data.europa.eu/esco/occupation/68d973df-bf10-4bf7-9a1b-fbd9f604b9db&language=en',
  'uri': 'http://data.europa.eu/esco/occupation/68d973df-bf10-4bf7-9a1b-fbd9f604b9db',
  'title': 'bioinformatics scientist'},
 'isTopConceptInScheme': [{'href': 'https://ec.europa.eu/esco/api/resource/taxonomy?uri=http://data.europa.eu/esco/concept-scheme/member-occupations&language=en',
   'uri': 'http://data.europa.eu/esco/concept-scheme/member-occupations',
   'title': 'ESCO member occupations'}],
 'isInScheme': [{'href': 'https://ec.europa.eu/esco/api/resource/taxonomy?uri=http://data.europa.eu/esco/concept-scheme/member-occupations&language=en',
   'uri': 'http://data.europa.eu/esco/concept-scheme/member-occupations',
   'title': 'ESCO member occupations'},
  {'href': 'https://ec.europa.eu/esco/api/resource/taxonomy?uri=http://data.europa.eu/esco/concept-scheme/occupations&language=en',
   'uri': 'http://data.europa.eu/esco/concept-scheme/occupations',
   'title': 'ESCO Occupations'}],
 'regulatedProfessionNote': {'href': 'https://ec.europa.eu/esco/api/resource/concept?uri=http://data.europa.eu/esco/regulated-professions/unregulated&language=en',
  'uri': 'http://data.europa.eu/esco/regulated-professions/unregulated',
```

Entrée [20]:
```
links = response['_links']
```

Entrée [71]:
```
links.keys()
```

Out[71]:
```
dict_keys(['self', 'isTopConceptInScheme', 'isInScheme', 'regulatedProfessionNote', 'broaderIscoGroup', 'hasEssentialSkill', 'hasOptionalSkill'])
```

Entrée [21]:
```
ess_skills = links['hasEssentialSkill']
opt_skills = links['hasOptionalSkill']
```

Entrée [74]:
```
ess_skills
```

...

Entrée [22]:
```
ess_skills[0]['title'] , ess_skills[0]['skillType']
```

Out[22]:
```
('manage database', 'http://data.europa.eu/esco/skill-type/skill')
```

Code Snippet

Occupation DF

for main node

Entrée [25]:
```
response.keys()
```

Out[25]:
```
dict_keys(['className', 'classId', 'uri', 'title', 'referenceLanguage', 'preferredLabel', 'alternativeLabel', 'preferredTerm', 'alternativeTerms', 'description', 'scopeNote', 'status', 'code', 'codes', '_links', '_embedded'])
```

Entrée [26]:
```
response['title'], response['preferredLabel']['fr'], response['alternativeLabel']['en'], response['description']['en']

```

Out[26]:
```
('ICT business analyst',
 'analyste d’étude en TIC',
 ['ICT business analysts',
  'business process specialist',
  'IT business analyst'],
 {'literal': "ICT business analysts are in charge of analysing and designing an organisation's processes and systems, assessing the business model and its integration with technology. They also identify change needs, assess the impact of the change, capture and document requirements and then ensure that these requirements are delivered whilst supporting the business through the implementation process.",
  'mimetype': 'plain/text'})
```

Entrée [27]:
```
response['alternativeLabel']['en']
```

Out[27]:
```
['ICT business analysts', 'business process specialist', 'IT business analyst']
```

Entrée [28]:
```
# Create an empty list to store the data
data = []

# Iterate over the _links_href column and extract the required data
for href in df_DataJob['_links_href']:
    # Send a request to the API endpoint and get the JSON response
    response = requests.get(href).json()
    # Extract the required data from the response
    title = response['title']
    pref_label = response['preferredLabel']['fr']
    alt_label = ", ".join([label['value'] if isinstance(label, dict) else label for label in response['alternativeLabel']
    desc = response['description']['en']
    # Append the data to the "data" list
    data.append({'Title': title, 'PreferredlabelinFR': pref_label, 'AltLabel': alt_label, 'Description': desc})

# Create a new DataFrame from the data list
df_new = pd.DataFrame(data, columns=['Title', 'PreferredlabelinFR', 'AltLabel', 'Description'])

```

Code Snippet

Entrée [34]:
```
1 df_occupation
```

Out[34]:

	Title	PreferredlabelinFR	AltLabel	Description
0	bioinformatics scientist	bioinformaticien/bioinformaticienne	environmental scientist, bioinformatics analys...	Bioinformatics scientists analyse biological p...
1	business analyst	analyste de gestion	standardisation expert, business strategy anal...	\nBusiness analysts research and understand th...
2	chief data officer	directeur des données/directrice des données	chief data officers, chief analytics officer, CDO	Chief data officers manage companies enterpris...
3	computer scientist	informaticien/informaticienne	ICT scientist, computer scientist, IT research...	Computer scientists conduct research in comput...
4	data analyst	analyste de données	data warehousing analyst, data analysts, data ...	Data analysts import, inspect, clean, transfor...
5	data centre operator	opérateur de centre de données/opératrice de c...	data center administrator, data centre operato...	Data centre operators maintain computer operat...
6	data quality specialist	analyste qualité des données	data quality expert, data integrity officer, d...	Data quality specialists review organisations ...
7	data scientist	scientifique des données	data scientists, data engineer, research data ...	Data scientists find and interpret rich data s...
8	data warehouse designer	concepteur d'entrepôt de données/conceptrice d...	data warehouse architect, data warehouse devel...	Data warehouse designers are responsible for p...
9	database administrator	administrateur de base de données/administratr...	database manager, DBA, database administrators...	Database administrators test, implement and ad...
10	database designer	concepteur de bases de données /conceptrice de...	data architect, database designers, data base ...	Database designers specify the databases logic...
11	database developer	développeur de base de données/développeuse de...	database developers, data base developer, data...	Database developers program, implement and coo...
12	database integrator	intégrateur de base de données/intégratrice de...	data base integrators, data base integrator, d...	Database integrators perform integration among...
13	ICT business analyst	analyste d'étude en TIC	ICT business analysts, business process specia...	ICT business analysts are in charge of analysi...

Code Snippet

Entrée [33]:

```
1 df_skills
```

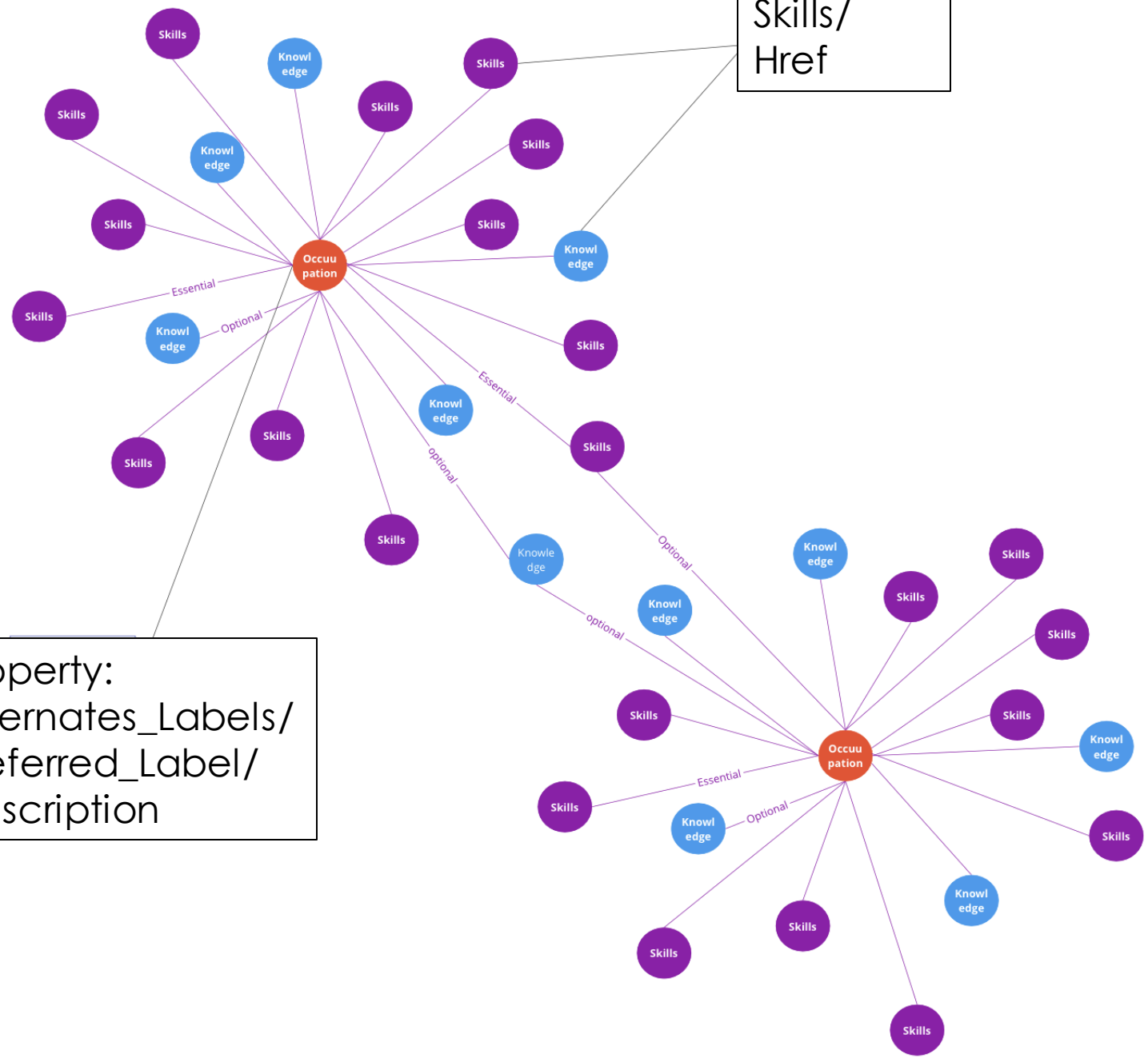
Out[33]:

	Title	Href	Relation	Skill_title	Skill_type
0	bioinformatics scientist	https://ec.europa.eu/esco/api/resource/occupat...	essential	apply scientific methods	skill
1	bioinformatics scientist	https://ec.europa.eu/esco/api/resource/occupat...	essential	present reports	skill
2	bioinformatics scientist	https://ec.europa.eu/esco/api/resource/occupat...	essential	use databases	skill
3	bioinformatics scientist	https://ec.europa.eu/esco/api/resource/occupat...	essential	contact scientists	skill
4	bioinformatics scientist	https://ec.europa.eu/esco/api/resource/occupat...	essential	gather data	skill
...	...	...	...	...	...
771	ICT business analyst	https://ec.europa.eu/esco/api/resource/occupat...	optional	visual presentation techniques	knowledge
772	ICT business analyst	https://ec.europa.eu/esco/api/resource/occupat...	optional	provide user documentation	skill
773	ICT business analyst	https://ec.europa.eu/esco/api/resource/occupat...	optional	information architecture	knowledge
774	ICT business analyst	https://ec.europa.eu/esco/api/resource/occupat...	optional	cloud technologies	knowledge
775	ICT business analyst	https://ec.europa.eu/esco/api/resource/occupat...	optional	business ICT systems	knowledge

776 rows × 5 columns

GRAPH DB for 2 occupations nodes

Property:
Skills/
Href



Property:
Alternates_Labels/
Preferred_Label/
Description

Occupation

Skill

Knowledge

Essential

Optional

Solutions

- NoSQL databases offer a flexible and scalable solution for managing unstructured and semi-structured data
- Provide suggestions instantly by searching the data relationships.
- Link the two occupations to see the similarities by using the score

Why Neo4J

Neo4J: our main tool to visualize and store data for this project

- RDBMS are poor at handling relationships between data points
- We need visual technique to portray the information
- Two potential graph database solutions: Gephi and Neo4j
- They are both used for maintaining and displaying graph data, although they each have unique advantages and applications
- Gephi is best suited for exploring and studying small to medium-sized networks because it is largely a visualization tool

Why Neo4J (cont'd)

Neo4J: our main tool to visualize and store data for this project

- Neo4J is optimized for storing and querying relationships between data and user-friendly using Cypher (similar query language as SQL)
- Research recommends to use Neo4J as it's the popular graph database system and has a large and active community to support

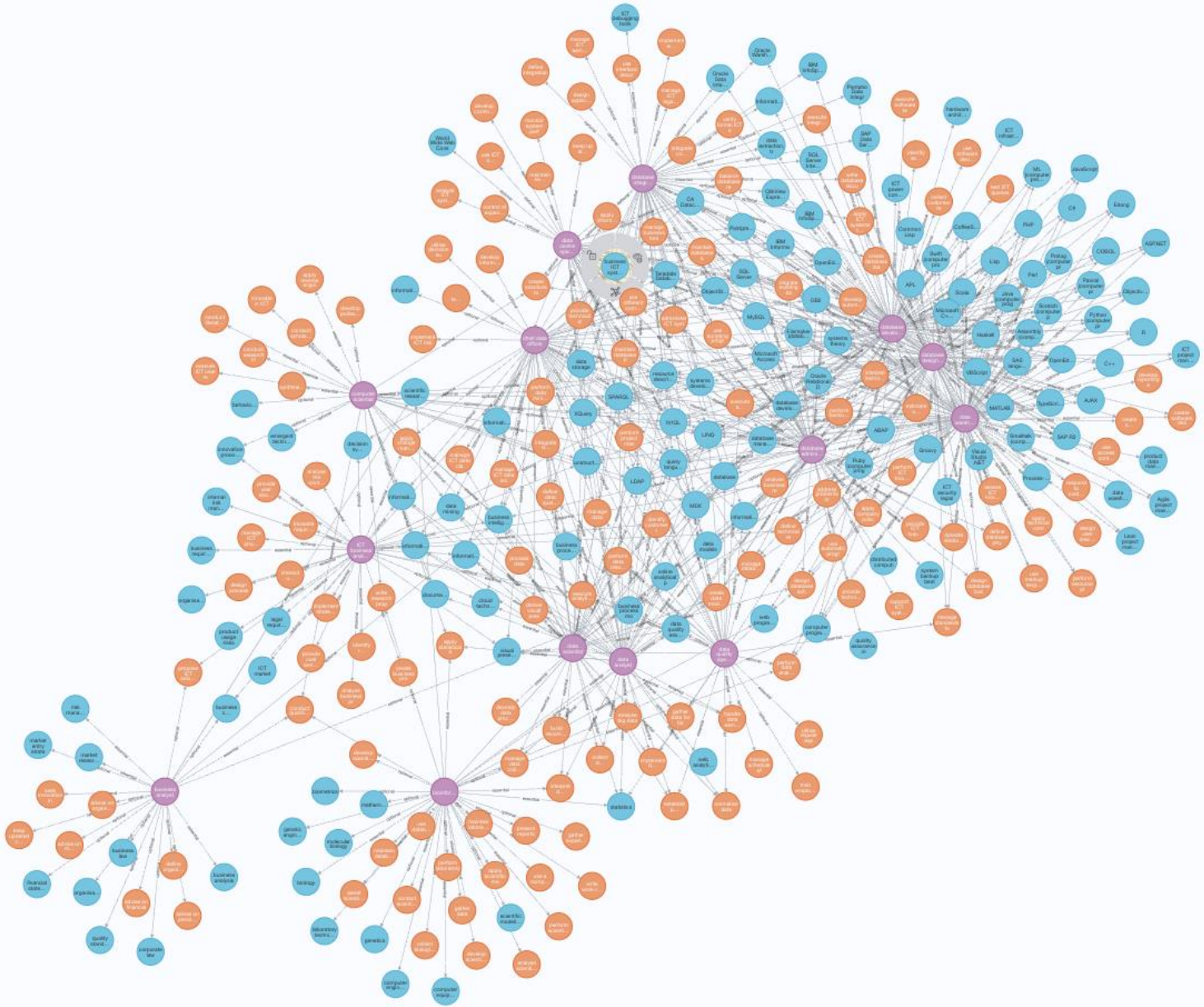
```
1 MATCH (n:Title) MATCH (s:Skills)
2 MATCH (k:Knowledge)
3 RETURN n,s,k
```

Graph

Table

Text

Code



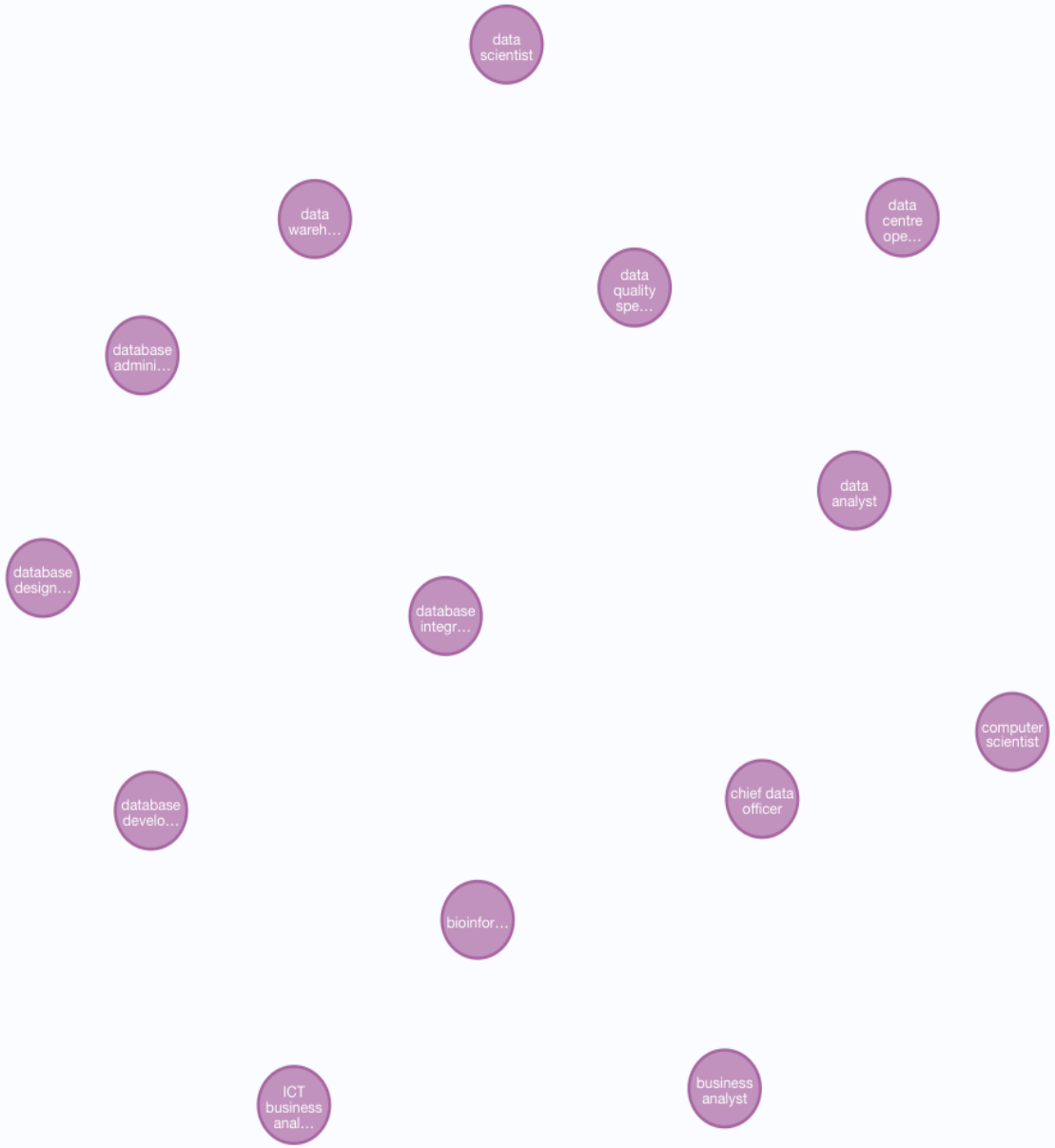
```
1 MATCH (n:Title)
2 RETURN n
```

Graph

Table

Text

Code



Overview

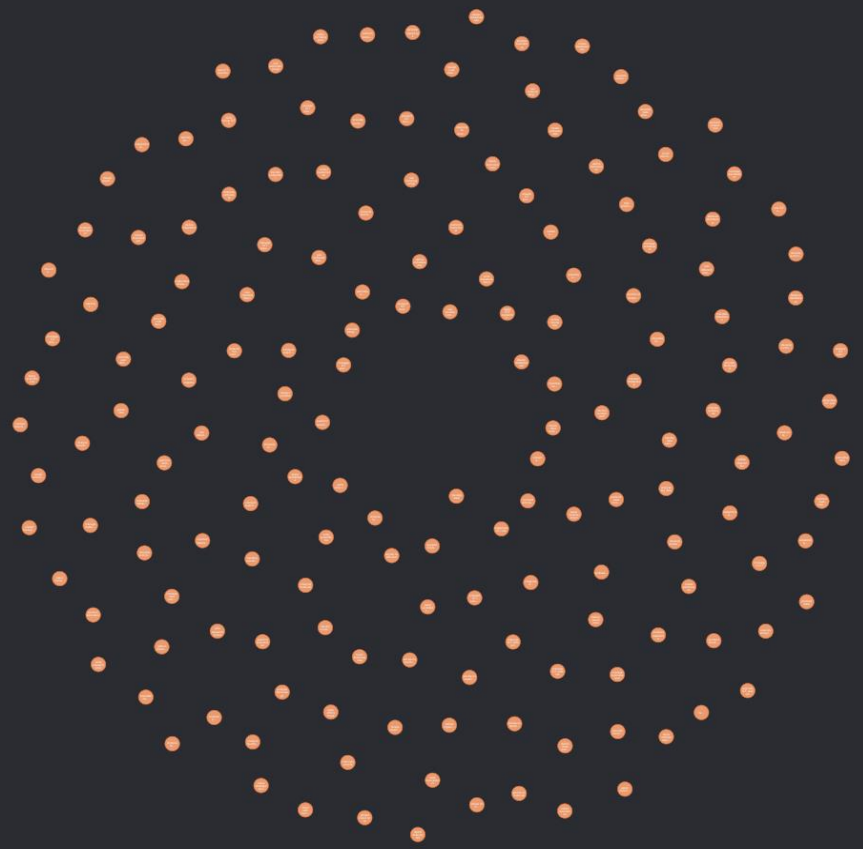
Node labels

* (14) Title (14)

Displaying 14 nodes, 0 relationships.

neo4j$ MATCH (s:Skills) RETURN s

Graph
Table
Text
Code



Overview

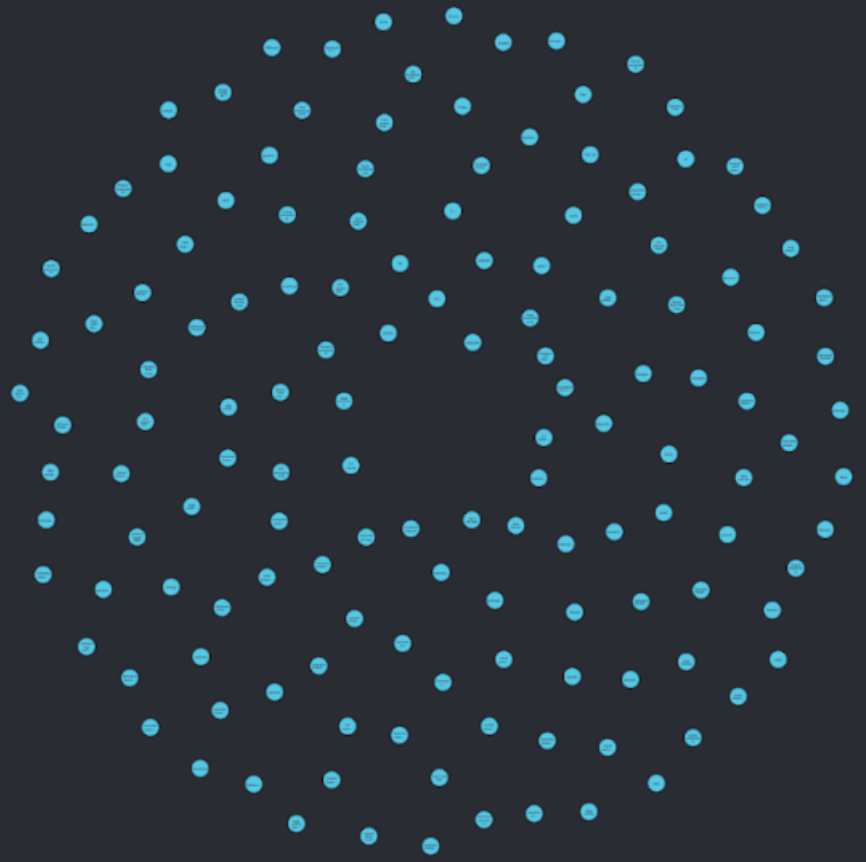
Node labels

* (167) Skills (167)

Displaying 167 nodes, 0 relationships.

```
1 MATCH (s:Knowledge)
2 RETURN s
```

Graph
Table
Text
Code



Overview

Node labels

* (141)
Knowledge (141)

Displaying 141 nodes, 0 relationships.

```
1 MATCH (n:Title {title:'data scientist'})
2 RETURN n
```

Graph

Table

Text

Code

Node properties

Title

<id>	327
alternativeLabel	data scientists, data engineer, research data scientist, data expert, data research scientist
description	Data scientists find and interpret rich data sources, manage large amounts of data, merge data sources, ensure consistency of data-sets, and create vi... Show all
preferredLabelFR	scientifique des données
title	data scientist

```
1 MATCH (s:Skills)
2
3 RETURN s Limit 1
```

Graph

Table

Text

Code

Node properties

Skills

<id>	2
Href	https://ec.europa.eu/esco/api/resource/occupation?uri=http://data.europa.eu/esco/occupation/60082a99-d8ef-4e84-9290-78902681b6ed&language=en
skills	keep updated on the political landscape

```
(n:Title{title: 'data scientist'})- [:essential]→(k:Knowledge)
 n,k
```

Overview

Node labels

* (10) Title (1) Knowledge (9)

Relationship types

* (9) essential (9)

Displaying 10 nodes, 0 relationships.

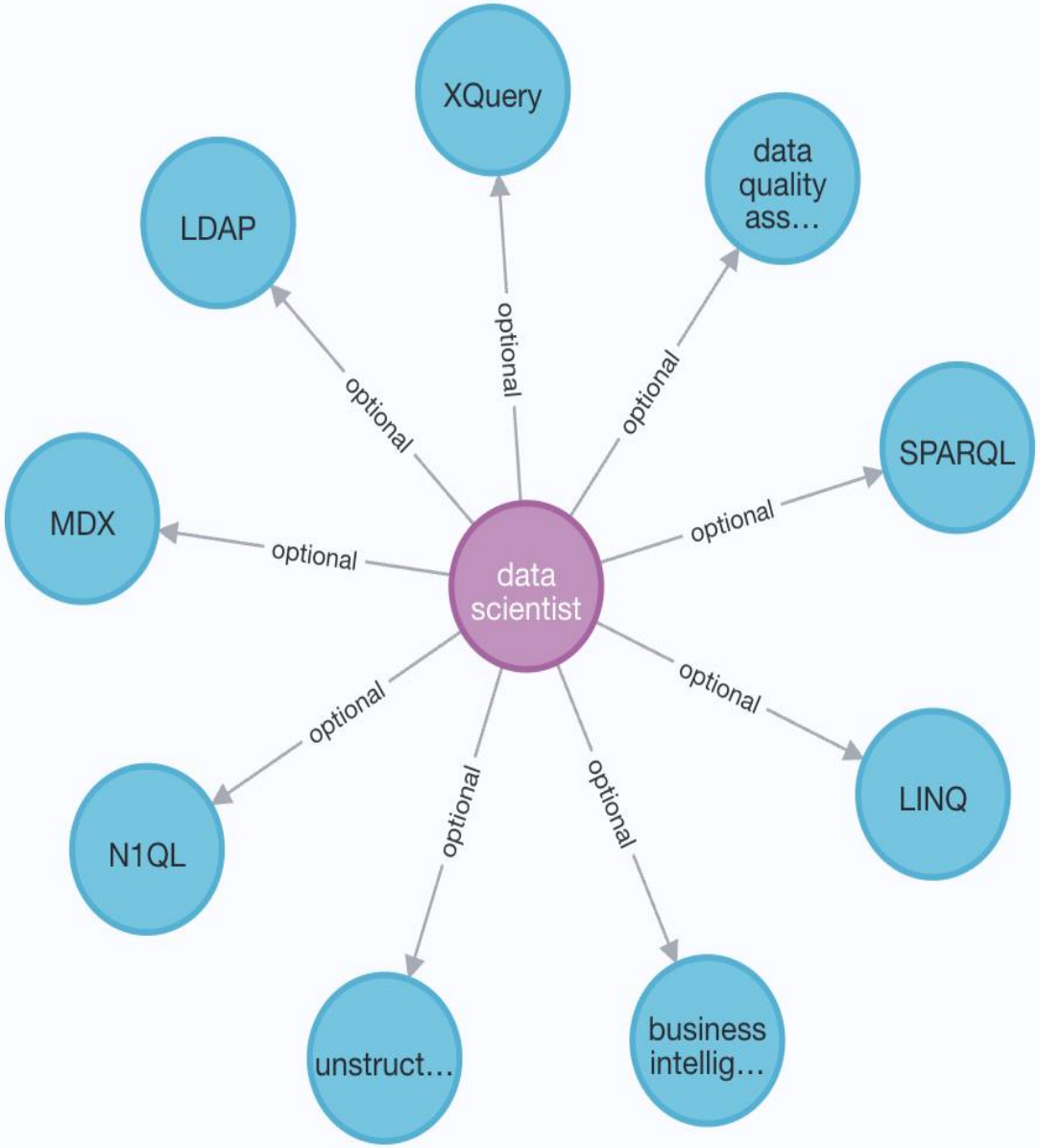
```
1 MATCH (n:Title {title: 'data scientist'})-[:optional]→(k:Knowledge)
2 RETURN n, k
```

Graph

Table

Text

Code



Overview

Node labels

* (10) Title (1) Knowledge (9)

Relationship types

* (9) optional (9)

Displaying 10 nodes, 0 relationships.

```
MATCH (n:Title {title: 'data scientist'})-[rel1:optional]→(k:Knowledge)
OPTIONAL MATCH(n) -[rel2:essential]→ (k2:Knowledge)
OPTIONAL MATCH(n) -[rel3:optional] →(s:Skills)
OPTIONAL MATCH(n) -[rel4:essential]→ (s2:Skills)
RETURN n, k,s,k2,s2,rel1,rel2,rel3,rel4
```

Graph

Table

Text

Code

Overview

Node labels

* (40) | Title (1) | Knowledge (18) | Skills (21)

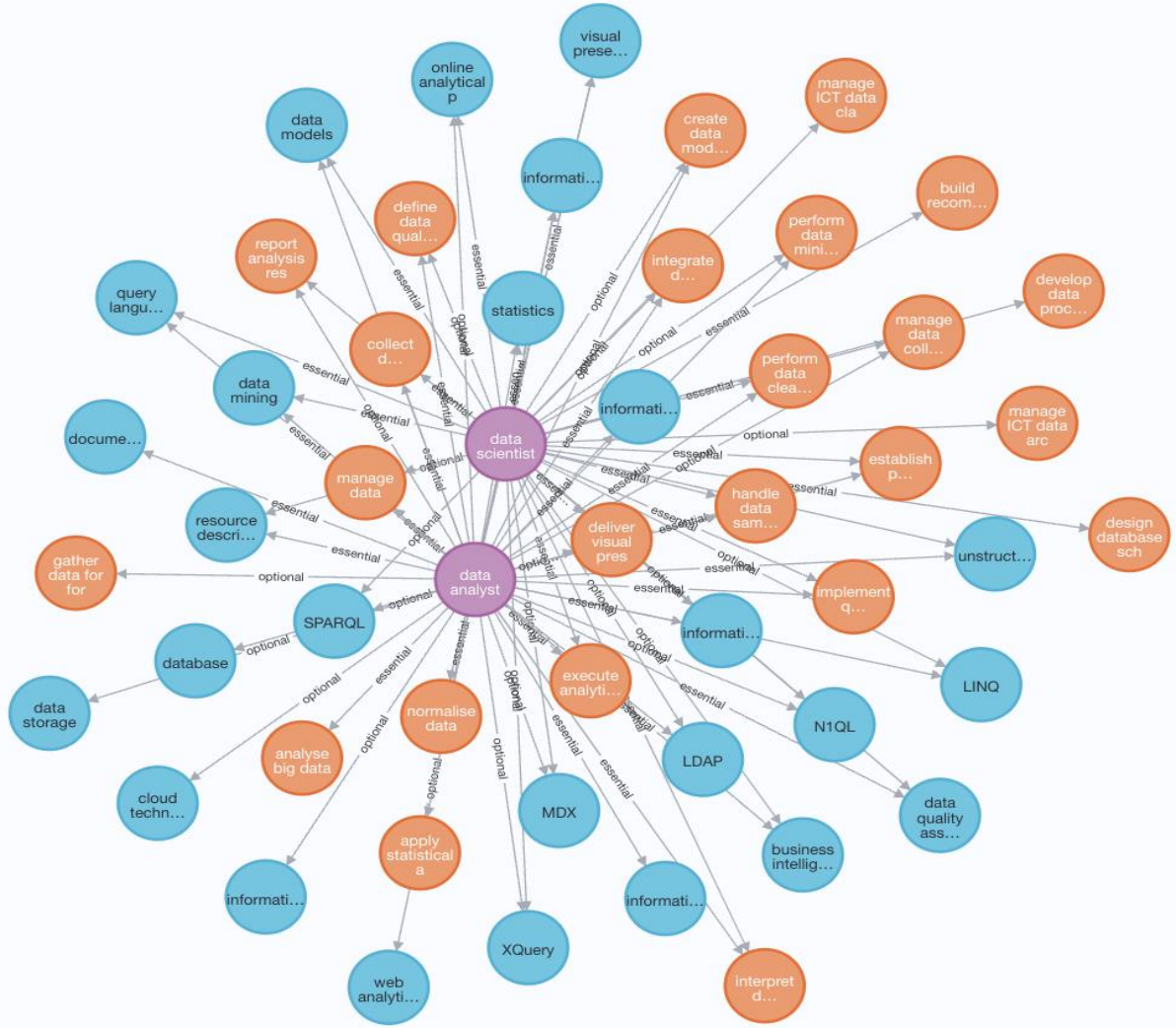
Relationship types

* (39) | optional (16) | essential (23)

Displaying 40 nodes, 31,752 relationships.

```
MATCH (n:Title)
WHERE n.title IN ['data scientist','data analyst']
OPTIONAL MATCH(n) -[rel1:optional]→(k:Knowledge)
OPTIONAL MATCH(n) -[rel2:essential]→ (k2:Knowledge)
OPTIONAL MATCH(n) -[rel3:optional] →(s:Skills)
OPTIONAL MATCH(n) -[rel4:essential]→ (s2:Skills)
RETURN n, k,s,k2,s2,rel1,rel2,rel3,rel4
```

Graph
Table
Text
Code



Overview

Node labels

* (52) | Title (2) | Knowledge (26) | Skills (24)

Relationship types

* (84) | optional (33) | essential (51)

Displaying 52 nodes, 78,792 relationships.

```
1 MATCH (t:Title {title: "computer scientist"})-[:optional|essential]→(k:Knowledge
  {skills:"query languages"})←[:optional|essential]-(n:Title)
2 RETURN t, k, n
3
```

Graph
Table
Text
Code

Overview

Node labels

* (12) | Title (11) | Knowledge (1)

Relationship types

* (11) | essential (9) | optional (2)

Displaying 12 nodes, 0 relationships.

```
MATCH (t1:Title)- [r1: essential|optional]-(s:Skills)-[r2:essential|optional]-(t2:Title)
WHERE t1<> t2 AND r1 < r2
WITH t1, t2, COUNT(s) AS shared_skills
WHERE shared_skills > 10
RETURN t1. title AS title_1, t2.title AS title_2, shared_skills
```

Table

Text

Code

	title_1	title_2	shared_skills
1	"data analyst"	"data scientist"	16
2	"database designer"	"data warehouse designer"	22

Started streaming 2 records after 1 ms and completed after 4 ms.

```
MATCH(t1:Title)-[:essential|optional]-(s:Skills)-[:essential|optional]-(t2:Title)
WHERE t1 < t2
MATCH (t1)-[:essential|optional]-(k:Knowledge)-[:essential|optional]-(t2)
WITH t1.title AS title1, t2.title AS title2, count(DISTINCT s) AS shared_skills,
     count(DISTINCT k) AS shared_knowledge,
     0.5 * count(DISTINCT s) + 0.5 * count(DISTINCT k) AS similarity_score
RETURN title1, title2, shared_skills, shared_knowledge, similarity_score
ORDER BY similarity_score DESC
```

Table | Text | Code

	title1	title2	shared_skills	shared_knowledge	similarity_score
1	"data warehouse designer"	"database designer"	22	72	47.0
2	"database designer"	"database developer"	1	60	30.5
3	"data warehouse designer"	"database developer"	1	60	30.5
4	"database developer"	"database integrator"	7	31	19.0
5	"data analyst"	"data scientist"	16	18	17.0
6	"data centre operator"	"database administrator"	7	22	14.5
7					

Started streaming 67 records in less than 1 ms and completed after 16 ms.

neo4j$

```
MATCH (t1)-[:essential]→(s:Skills)←[:essential]-(t2)
WITH t1, t2, shared_essential_skills, COUNT(DISTINCT s) AS shared_essential_knowledge
MATCH (t1)-[:optional]→(k:Knowledge)←[:optional]-(t2)
WITH t1, t2, shared_essential_skills, shared_essential_knowledge, COUNT(DISTINCT k) AS
shared_optional_skills
MATCH (t1)-[:optional]→(k:Knowledge)←[:optional]-(t2)
WITH t1, t2, shared_essential_skills, shared_essential_knowledge, shared_optional_skills,
COUNT(DISTINCT k) AS shared_optional_knowledge
RETURN t1.title AS title1, t2.title AS title2,
       0.7 * (shared_essential_skills + shared_essential_knowledge) + 0.3 *
(shared_optional_skills + shared_optional_knowledge) AS similarity_score
ORDER BY similarity_score DESC
```

Table | Text | Code

	title1	title2	similarity_score
1	"data warehouse designer"	"database designer"	58.199999999999996
2	"database designer"	"database developer"	35.0
3	"data warehouse designer"	"database developer"	35.0
4	"database developer"	"database integrator"	17.599999999999998
5	"data centre operator"	"database administrator"	16.6
6	"database administrator"	"database developer"	15.799999999999999

Future Applications

- Create a model for the data and the links between the data to comprehend how suggestions may be generated
- Provide suggestions instantly by searching the data relationships
- Constantly add new information and connections to the model to make it richer

Thank you for participating!

datacraft*

em lyon business school

Annexes

Import

Entrée [1]:

```
#Scrap

import requests #Requests allows to send HTTP/1.1 requests

#Utilitaires
import json
import pandas as pd
import numpy as np
from pandas import json_normalize

from concurrent.futures import ThreadPoolExecutor

#Neo4j

from neo4j import GraphDatabase
from py2neo import Graph
from neomodel import StructuredNode, StringProperty, RelationshipTo, RelationshipFrom, config
```

Scrap Data

Entrée [2]:

```
s://ec.europa.eu/esco/api/search?language=en&isInScheme=http://data.europa.eu/esco/concept-scheme/occupations&limit=3561"
```

Entrée [3]:

```
response = requests.get(occ_url) #Store the request into a new variable
```

Entrée [4]:

```
Occupation_source = response.json()
```

Entrée [5]:

```
Occupation = Occupation_source['_embedded']
```

Entrée [6]:

```
df_Occupation = pd.DataFrame(Occupation["results"]) #Convert to pandas
```

Define List of necessary jobs

Entrée [7]:

```
DataJobsIndices = df_Occupation['title'].str.contains('op|stewart|learning|artificial|dev|math|intelligence|physics|compu
df_DataJob = df_Occupation[DataJobsIndices]
```

This code snippet is a filter to include only rows where the 'title' column contains certain keywords related to data and technology.

Entrée [8]:

```
print(len(df_DataJob))
```

```
620
```

Entrée [9]:

```
df_DataJob['title']
```

Out[9]:

```
3                            abrasive blasting operator
4                          absorbent pad machine operator
10                                   accounting analyst
36                                advertising copywriter
61                                agricultural scientist
                                  ...
3534    Wood processing and papermaking plant operators
3535                      Wood processing plant operators
3538                                wood router operator
3545                    wooden furniture machine operator
3547     Woodworking-machine tool setters and operators
Name: title, Length: 620, dtype: object
```

Entrée [10]:

```
#Occupation list from Julien
Nec_Occ = ['data analyst',
'data scientist',
'computer scientist',
'chief data officer',
'ICT research consultant',
'data centre operator',
'ICT business analyst',
'data quality specialist',
'data warehouse designer',
'business analyst','data protection officer','database administrator','database designer','database integrator',
'database developer','ICT security manager','knowledge engineer','test engineer','mathematician research engineer',
'bioinformatics scientist']
```

Entrée [11]:

```
df_DataJob = df_DataJob[df_DataJob['title'].isin(Nec_Occ)]
df_DataJob = df_DataJob.reset_index()
```

Data Exploration

Entrée [13]:
```
1 df_DataJob['_links'][0]
```

Out[13]:
```
{'self': {'href': 'https://ec.europa.eu/esco/api/resource/occupation?uri=http://data.europa.eu/esco/occupation/68d973df-bf10
-4bf7-9a1b-fbd9f604b9db&language=en',
  'uri': 'http://data.europa.eu/esco/occupation/68d973df-bf10-4bf7-9a1b-fbd9f604b9db',
  'title': 'bioinformatics scientist'}}
```

In the given JSON object, 'href' and 'uri' are used to represent the same resource, which is an occupation with the title of "bioinformatics scientist". The 'uri' provides a unique identifier for the occupation, while the 'href' provides the URL where the occupation's information can be accessed.

Entrée [14]:
```
1 df_DataJob['_links_href'] = df_DataJob['_links'].apply(lambda x: x['self']['href'])
```

Entrée [15]:
```
1 df_DataJob['_links_href']
```

Out[15]:
```
0     https://ec.europa.eu/esco/api/resource/occupat...
1     https://ec.europa.eu/esco/api/resource/occupat...
2     https://ec.europa.eu/esco/api/resource/occupat...
3     https://ec.europa.eu/esco/api/resource/occupat...
4     https://ec.europa.eu/esco/api/resource/occupat...
5     https://ec.europa.eu/esco/api/resource/occupat...
6     https://ec.europa.eu/esco/api/resource/occupat...
7     https://ec.europa.eu/esco/api/resource/occupat...
8     https://ec.europa.eu/esco/api/resource/occupat...
9     https://ec.europa.eu/esco/api/resource/occupat...
10    https://ec.europa.eu/esco/api/resource/occupat...
11    https://ec.europa.eu/esco/api/resource/occupat...
12    https://ec.europa.eu/esco/api/resource/occupat...
13    https://ec.europa.eu/esco/api/resource/occupat...
Name: _links_href, dtype: object
```

Entrée [16]:
```
1 df_DataJob
```

Out[16]:

_links	isTopConceptInScheme	broaderIscoGroup	broaderConcept	_links_href
{'self': {'href': 'https://ec.europa.eu/esco/a...	[http://data.europa.eu/esco/concept-scheme/mem...	[http://data.europa.eu/esco/isco/C2131]	NaN	https://ec.europa.eu/esco/api/resource/occupat...
{'self': {'href': 'https://ec.europa.eu/esco/a...	[http://data.europa.eu/esco/concept-scheme/mem...	[http://data.europa.eu/esco/isco/C2421]	NaN	https://ec.europa.eu/esco/api/resource/occupat...
{'self': {'href': 'https://ec.europa.eu/esco/a...	[http://data.europa.eu/esco/concept-scheme/mem...	[http://data.europa.eu/esco/isco/C1330]	NaN	https://ec.europa.eu/esco/api/resource/occupat...
{'self': {'href': 'https://ec.europa.eu/esco/a...	[http://data.europa.eu/esco/concept-scheme/mem...	[http://data.europa.eu/esco/isco/C2511]	NaN	https://ec.europa.eu/esco/api/resource/occupat...

Entrée [17]:
```
response = requests.get(df_DataJob['_links_href'][0])
response = response.json()
```

Entrée [18]:
```
response.keys()
```

Out[18]:
```
dict_keys(['className', 'classId', 'uri', 'title', 'referenceLanguage', 'preferredLabel', 'alternativeLabel', 'preferredTerm', 'alternativeTerms', 'description', 'status', 'code', 'codes', '_links', '_embedded'])
```

Entrée [19]:
```
response['_links']
```

Out[19]:
```
{'self': {'href': 'https://ec.europa.eu/esco/api/resource/occupation?uri=http://data.europa.eu/esco/occupation/68d973df-bf10-4bf7-9a1b-fbd9f604b9db&language=en',
  'uri': 'http://data.europa.eu/esco/occupation/68d973df-bf10-4bf7-9a1b-fbd9f604b9db',
  'title': 'bioinformatics scientist'},
 'isTopConceptInScheme': [{'href': 'https://ec.europa.eu/esco/api/resource/taxonomy?uri=http://data.europa.eu/esco/concept-scheme/member-occupations&language=en',
   'uri': 'http://data.europa.eu/esco/concept-scheme/member-occupations',
   'title': 'ESCO member occupations'}],
 'isInScheme': [{'href': 'https://ec.europa.eu/esco/api/resource/taxonomy?uri=http://data.europa.eu/esco/concept-scheme/member-occupations&language=en',
   'uri': 'http://data.europa.eu/esco/concept-scheme/member-occupations',
   'title': 'ESCO member occupations'},
  {'href': 'https://ec.europa.eu/esco/api/resource/taxonomy?uri=http://data.europa.eu/esco/concept-scheme/occupations&language=en',
   'uri': 'http://data.europa.eu/esco/concept-scheme/occupations',
   'title': 'ESCO Occupations'}],
 'regulatedProfessionNote': {'href': 'https://ec.europa.eu/esco/api/resource/concept?uri=http://data.europa.eu/esco/regulated-professions/unregulated&language=en',
  'uri': 'http://data.europa.eu/esco/regulated-professions/unregulated',
```

Entrée [20]:
```
links = response['_links']
```

Entrée [71]:
```
links.keys()
```

Out[71]:
```
dict_keys(['self', 'isTopConceptInScheme', 'isInScheme', 'regulatedProfessionNote', 'broaderIscoGroup', 'hasEssentialSkill', 'hasOptionalSkill'])
```

Entrée [21]:
```
ess_skills = links['hasEssentialSkill']
opt_skills = links['hasOptionalSkill']
```

Entrée [74]:
```
ess_skills
```

...

Entrée [22]:
```
ess_skills[0]['title'] , ess_skills[0]['skillType']
```

Out[22]:
```
('manage database', 'http://data.europa.eu/esco/skill-type/skill')
```

Entrée [22]:
```
ess_skills[0]['title'] , ess_skills[0]['skillType']
```

Out[22]: ('manage database', 'http://data.europa.eu/esco/skill-type/skill')

Skills DF

Entrée [23]:
```
# Create a list to store the data
data = []

# Loop through the rows of the original DataFrame
for i, row in df_DataJob.iterrows():
    # Get the href value from the _links column
    href = row['_links']['self']['href']
    # Send a request to the API endpoint and get the JSON response
    response = requests.get(href).json()
    # Get the essential and optional skills from the JSON response
    ess_skills = response['_links']['hasEssentialSkill']
    opt_skills = response['_links']['hasOptionalSkill']
    # Get the title from the original DataFrame
    title = row['title']
   # Loop through the essential skills and append them to the "data" list
    for skill in ess_skills:
        skill_type = skill['skillType']
        data.append({'Title': title, 'Href': href, 'Relation': 'essential', 'Skill_title': skill['title'], 'Skill_type':
# Loop through the optional skills and append them to the "data" list
    for skill in opt_skills:
        skill_type = skill['skillType']
        data.append({'Title': title, 'Href': href, 'Relation': 'optional', 'Skill_title': skill['title'], 'Skill_type': s

# Create a new DataFrame from the data list
df_skills = pd.DataFrame(data)
```

Clean df

Entrée [24]:
```
# Get the last part of the string after the last "/"
df_skills['Skill_type'] = df_skills['Skill_type'].apply(lambda x: x.split("/")[-1])
df_skills
```

Out[24]:

	Title	Href	Relation	Skill_title	Skill_type
0	bioinformatics scientist	https://ec.europa.eu/esco/api/resource/occupat...	essential	apply scientific methods	skill
1	bioinformatics scientist	https://ec.europa.eu/esco/api/resource/occupat...	essential	present reports	skill
2	bioinformatics scientist	https://ec.europa.eu/esco/api/resource/occupat...	essential	use databases	skill

Occupation DF

for main node

Entrée [25]:
```
response.keys()
```

Out[25]:
```
dict_keys(['className', 'classId', 'uri', 'title', 'referenceLanguage', 'preferredLabel', 'alternativeLabel', 'preferredTer
m', 'alternativeTerms', 'description', 'scopeNote', 'status', 'code', 'codes', '_links', '_embedded'])
```

Entrée [26]:
```
response['title'], response['preferredLabel']['fr'], response['alternativeLabel']['en'], response['description']['en']

```

Out[26]:
```
('ICT business analyst',
 'analyste d’étude en TIC',
 ['ICT business analysts',
  'business process specialist',
  'IT business analyst'],
 {'literal': "ICT business analysts are in charge of analysing and designing an organisation's processes and systems, assess
ing the business model and its integration with technology. They also identify change needs, assess the impact of the chang
e, capture and document requirements and then ensure that these requirements are delivered whilst supporting the business th
rough the implementation process.",
  'mimetype': 'plain/text'})
```

Entrée [27]:
```
response['alternativeLabel']['en']
```

Out[27]:
```
['ICT business analysts', 'business process specialist', 'IT business analyst']
```

Entrée [28]:
```
# Create an empty list to store the data
data = []

# Iterate over the _links_href column and extract the required data
for href in df_DataJob['_links_href']:
    # Send a request to the API endpoint and get the JSON response
    response = requests.get(href).json()
    # Extract the required data from the response
    title = response['title']
    pref_label = response['preferredLabel']['fr']
    alt_label = ", ".join([label['value'] if isinstance(label, dict) else label for label in response['alternativeLabel']
    desc = response['description']['en']
    # Append the data to the "data" list
    data.append({'Title': title, 'PreferredlabelinFR': pref_label, 'AltLabel': alt_label, 'Description': desc})

# Create a new DataFrame from the data list
df_new = pd.DataFrame(data, columns=['Title', 'PreferredlabelinFR', 'AltLabel', 'Description'])

```

Entrée [29]:
```
df_new['Description'][0]
```

Out[29]:
```
{'literal': 'Bioinformatics scientists analyse biological processes using computer programmes. They maintain or construct da
tabases containing biological information. Bioinformatics scientists gather and analyse biological data and may also assist
scientists in various fields, including in biotechnology and pharmaceutics. They perform scientific research and statistical
analyses, and report on their findings. Bioinformatics scientists may also collect DNA samples, discover data patterns and c
onduct genetic research.',
 'mimetype': 'plain/text'}
```

Entrée [30]:
```
# clean the 'Description' column
for index, row in df_new.iterrows():
    description = row['Description']['literal']
    cleaned_description = description.replace('\'', '').replace('{', '').replace('}', '').replace('literal:', '').replace
    df_new.at[index, 'Description'] = cleaned_description

df_new['Description'][0]
```

Out[30]:
```
'Bioinformatics scientists analyse biological processes using computer programmes. They maintain or construct databases cont
aining biological information. Bioinformatics scientists gather and analyse biological data and may also assist scientists i
n various fields, including in biotechnology and pharmaceutics. They perform scientific research and statistical analyses, a
nd report on their findings. Bioinformatics scientists may also collect DNA samples, discover data patterns and conduct gene
tic research.'
```

Entrée [31]:
```
df_new['Description'][2]
```

Out[31]:
```
'Chief data officers manage companies enterprise-wide data administration and data mining functions. They ensure data are us
ed as a strategic business asset at the executive level and implement and support a more collaborative and aligned informati
on management infrastructure for the benefit of the organisation at large.'
```

Entrée [32]:
```
df_occupation = df_new
```

Neo4j

Entrée [35]:

```python
class Neo4jConnection: #Init class
    '''
    Python class called Neo4jConnection that can be used to connect to a Neo4j database
    using the official Python driver GraphDatabase.driver.
    The __init__ method initializes the class with the database URI, username, and password.
    The close method closes the database driver if it is open.
    The query method executes a Cypher query with optional parameters and returns the query response as a list.

    '''

    def __init__(self, uri, user, pwd):
        self.__uri = uri
        self.__user = user
        self.__pwd = pwd
        self.__driver = None
        try:
            self.__driver = GraphDatabase.driver(self.__uri, auth=(self.__user, self.__pwd))
        except Exception as e:
            print("Failed to create the driver:", e)

    def close(self):
        if self.__driver is not None:
            self.__driver.close()

    def query(self, query, parameters=None, db=None):
        assert self.__driver is not None, "Driver not initialized!"
        session = None
        response = None
        try:
            session = self.__driver.session(database=db) if db is not None else self.__driver.session()
            response = list(session.run(query, parameters))
        except Exception as e:
            print("Query failed:", e)
        finally:
            if session is not None:
                session.close()
        return response

```

Entrée [36]:

```python
#Connect to DB
conn = Neo4jConnection(uri="bolt://18.213.115.214:7687",
                       user="neo4j",
                       pwd="fatigues-standing-credibility")
```

Clean db

Entrée [ ]:

```
#Don't run (only to reset db)
clean = conn.query("MATCH (n) DETACH DELETE n ")
clean
```

Create Label

Entrée [ ]:

```
conn.query('CREATE CONSTRAINT title IF NOT EXISTS FOR (t:Title)     REQUIRE t.title IS UNIQUE')
conn.query('CREATE CONSTRAINT skills IF NOT EXISTS FOR (s:Skills)     REQUIRE s.skills IS UNIQUE')
conn.query('CREATE CONSTRAINT knowledge IF NOT EXISTS FOR (k:Knowledge)     REQUIRE k.knowledge IS UNIQUE')
```

Title

Entrée [37]:

```
def add_title(title):
    """
    This is a Python function called add_title that takes in a pandas DataFrame
    and executes a Cypher query on a Neo4j database connected by a conn object

    The Cypher query in this function is designed to create a Title node in the database.
    With properties title, description, and alternativeLabel.

    """

    query = '''
        UNWIND $rows AS row
        MERGE (t:Title {title: row.Title})
        ON CREATE SET t.description = row.Description, t.alternativeLabel = row.AltLabel,
        t.preferredLabelFR = row.PreferredlabelinFR
        RETURN count(*) as total
        '''

    #The ON CREATE SET clause sets the description and alternativeLabel properties for the newly created Title node.
    return conn.query(query, parameters={'rows': title.to_dict('records')})
```

Entrée [65]:

```
add_title(df_occupation)
```

Skills/Knowledge

Entrée [69]:

```python
def create_skill_nodes(conn, df_skills):
    """This is a Python function called create_skill_nodes that takes in a pandas DataFrame
    and executes  Cypher queries on a Neo4j database connected by a conn object

    The Cypher queries in this function are designed to create a Skills and Knowledge nodes in the database.
    With properties href and "essential" or "optional" relation to the nodes t:Title. """

    for _, row in df_skills.iterrows():
        title_name = row['Title']
        skill_title = row['Skill_title']
        skill_type = row['Skill_type']
        relation = row['Relation']
        href = row['Href']

        # Determine the node label based on the skill type
        if skill_type == 'skill':
            node_label = 'Skills'
        elif skill_type == 'knowledge':
            node_label = 'Knowledge'
        else:
            raise ValueError(f"Invalid skill type: {skill_type}")

        # First, check if the skill already exists in the database
        existing_skills = conn.query(f"MATCH (s:{node_label} {{skills: $skill_title}}) RETURN s", {'skill_title': skill_t
        if existing_skills:
            skill_node = existing_skills[0]['s']
        else:
            # If the skill doesn't exist, create a new node
            skill_node = conn.query(f"CREATE (s:{node_label} {{skills: $skill_title, Href: $href}}) RETURN s", {'skill_ti

        # Get the Title node
        title_node = conn.query("MATCH (t:Title {title: $title_name}) RETURN t", {'title_name': title_name})[0]['t']

        # Create the relationship between the Title and the skill node
        conn.query("MATCH (t:Title {title: $title_name}), (s) WHERE ID(s) = $skill_id CREATE (t)-[r:" + relation + "]->(s
```

Entrée [68]:

```python
#Take time
create_skill_nodes(conn, df_skills)
```

GITHUB

https://github.com/NathanDestrez/DS_Mission