

Projet Graph Analytics

Réalisé par:

LABIAD Soukaina

PLAN

1. Introduction
2. Préparation de la base de données
3. Le voyageur du Monde
4. Les interconnectés

Introduction

- Analyser le trafic aérien.
- Résoudre des problèmes.
- Proposer des solutions.

Le jeu de données est extrait du site : <https://openflights.org/data.html>



Préparation de la base de données

Le jeu de données

5 fichiers .csv:



- les aéroports
- les compagnies aériennes
- les routes aériennes
- les informations sur les pays
- les informations sur les avions

Préparation de la base de données

Nettoyage et Indexation des données

L'indexation rend les recherches de données plus rapides et efficaces.

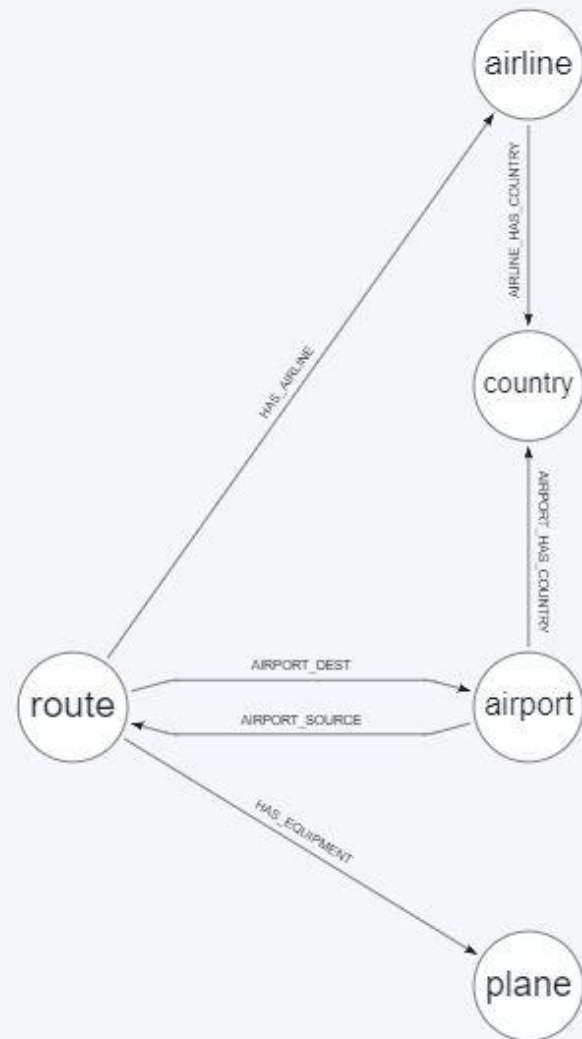
\$: schema

Index Name	Type	Uniqueness	EntityType	LabelsOrTypes	Properties	State
constraint_2619603d	BTREE	UNIQUE	NODE	["route"]	["Route ID"]	ONLINE
constraint_359b776	BTREE	UNIQUE	NODE	["airline"]	["Airline ID"]	ONLINE
constraint_c218edec	BTREE	UNIQUE	NODE	["country"]	["Name"]	ONLINE
constraint_c5e0a97	BTREE	UNIQUE	NODE	["airport"]	["AirportID"]	ONLINE
constraint_f28b0e99	BTREE	UNIQUE	NODE	["plane"]	["Name"]	ONLINE
index_343aff4e	LOOKUP	NONUNIQUE	NODE	[]	[]	ONLINE
index_f7700477	LOOKUP	NONUNIQUE	RELATIONSHIP	[]	[]	ONLINE

Préparation de la base de données

Modélisation de la base de données graphe

1. Créer une instance de base de données avec **neo4j auroraDB**.
2. Importer les données dans la base.
3. Modéliser la base de données graphe.



Préparation de la base de données

Balayage des données

Pour bien comprendre le jeu de données et les relations entre les noeuds, nous avons essayé à répondre à quelques questions:



Préparation de la base de données

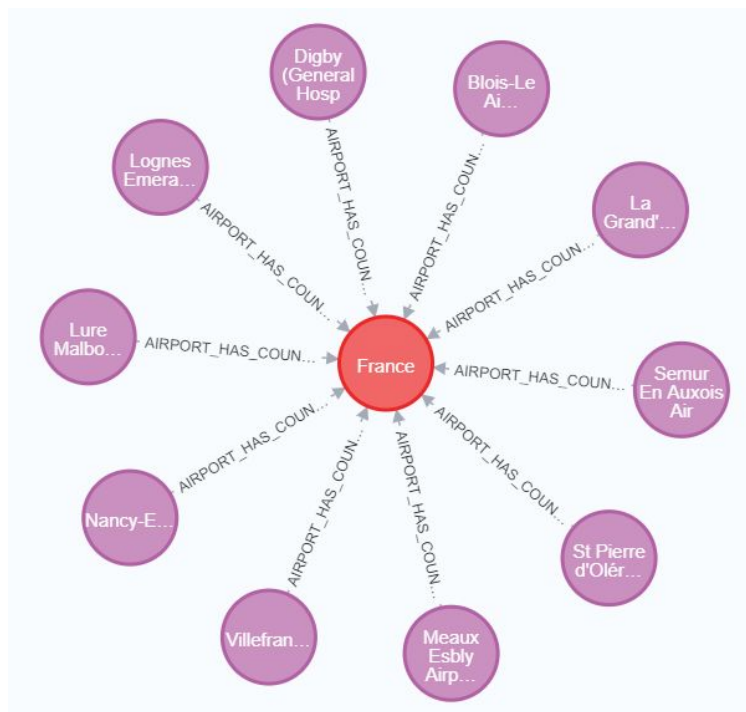
Balayage des données

Quels sont les aéroports français ?

Requête:

```
neo4j$ MATCH (n1:airport)-  
  [r:AIRPORT_HAS_COUNTRY]→  
  (n2:country{Name:'France'})  
RETURN r, n1, n2 LIMIT 10
```

Résultat:



Préparation de la base de données

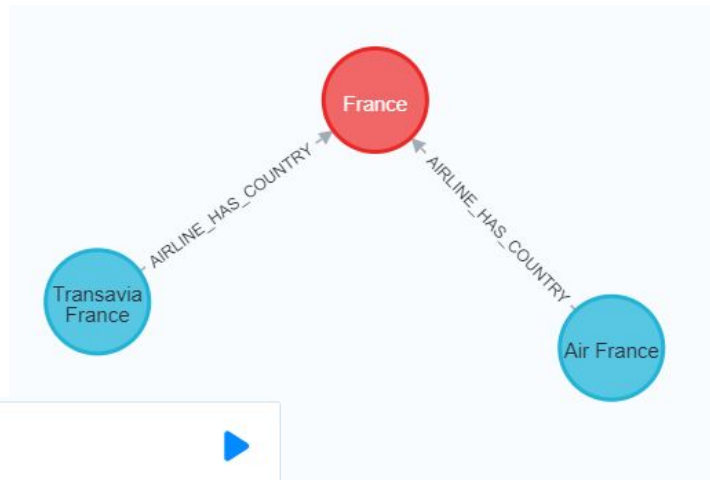
Balayage des données

Quelles sont les compagnies aériennes
françaises actives ?

Résultat:

Requête:

```
neo4j$ MATCH (n1:airline{Active:'Y'})-  
[r:AIRLINE_HAS_COUNTRY]→  
(n2:country{Name:'France'}) RETURN r, n1,  
n2 LIMIT 25
```



Préparation de la base de données

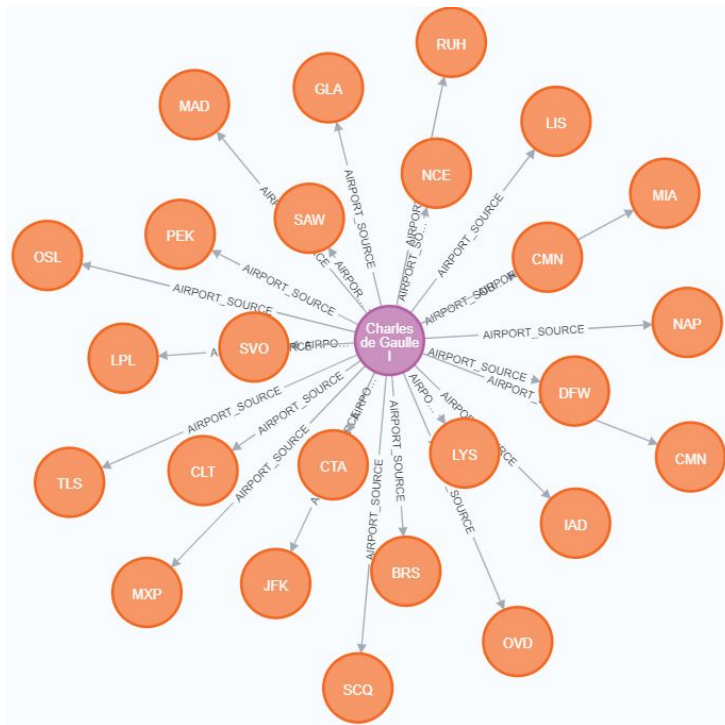
Balayage des données

Quelles sont les lignes au départ de l'aéroport Charles de Gaulle (CDC) ?

Requête:

```
MATCH (n1:airport{AirportID:'1382'})-  
[r:AIRPORT_SOURCE]→(n2:route) RETURN r,  
n1, n2 LIMIT 25
```

Résultat:



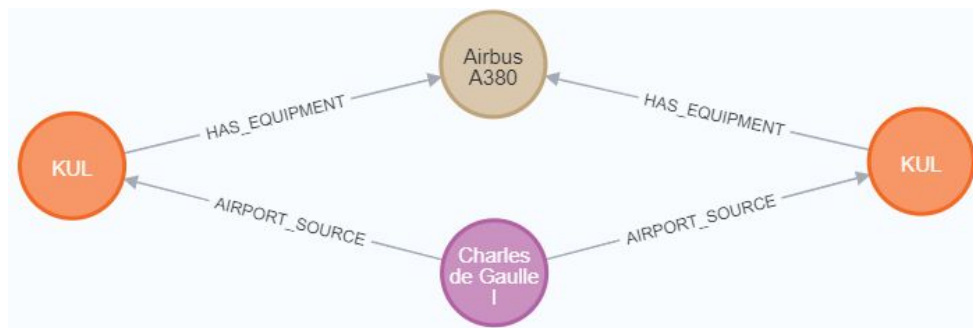
Préparation de la base de données

Balayage des données

Quelles sont les lignes au départ de l'aéroport CDC délivrées par un A380 ?

Requête:

Résultat:



```
neo4j$ MATCH (n1:airport{AirportID:'1382'})-  
[r:AIRPORT_SOURCE]→(n2:route)-  
[r2:HAS_EQUIPMENT]→(n3:plane{Name:"Airbus  
A380"}) RETURN r, n1, n2 LIMIT 25
```





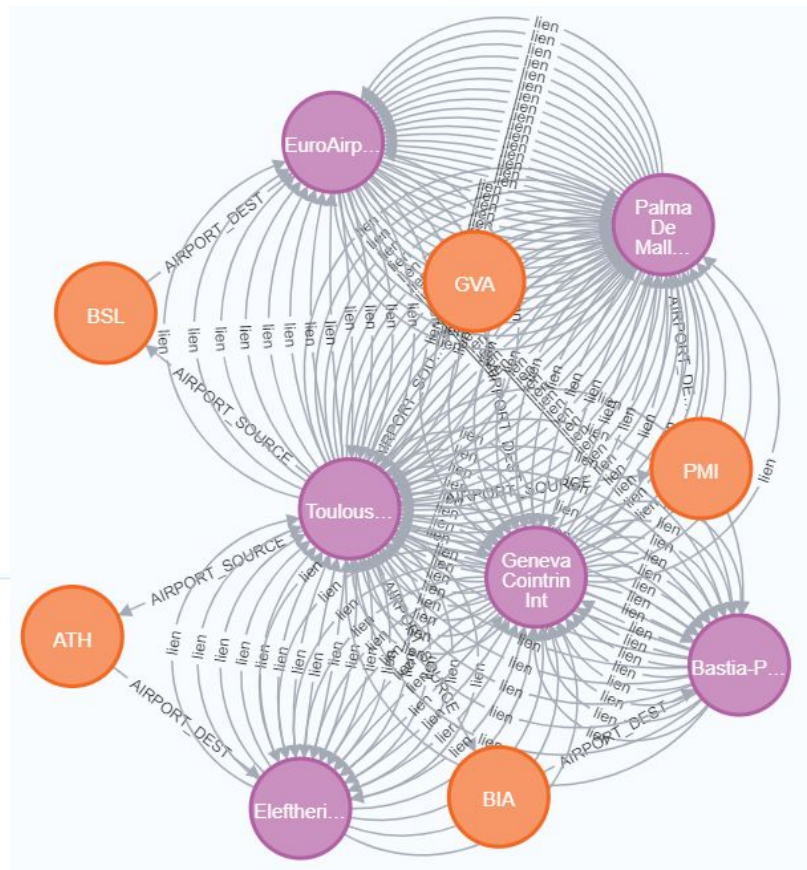
Le voyageur du Monde

Le voyageur du Monde

L'aéroport de départ est **Toulouse-Blagnac Airport** → le plus proche à TOURNEFEUILLE.

La requête Cypher utilisée:

```
1 MATCH (a1:airport {Name:
   'Toulouse-Blagnac Airport'})-
   [rel1:AIRPORT_SOURCE]→(r:route)-
   [rel2:AIRPORT_DEST]→(a2:airport)
2 RETURN a1,rel1,r,rel2,a2 LIMIT 5
3
```



Le voyageur du Monde

On avait besoin d'ajouter la propriété **distance** comme coût de la relation entre les aéroports de départ et d'arrivée.

La requête Cypher utilisée:

```
1 MATCH (source:airport) -  
  [linkSource:AIRPORT_SOURCE] →  
  (r:route) - [linkDest:AIRPORT_DEST]  
  → (dest:airport)  
2 CREATE (source) - [l:lien {distance:  
  point.distance(Point({latitude:  
  source.Latitude, longitude:  
  source.longitude}), Point({latitude:  
  dest.Latitude, longitude:  
  dest.longitude})))/1000}] → (dest)
```

3



Table



Code

Created 38131 relationships, completed after 215 ms.

Le voyageur du Monde

Puis on a stocké notre graphe sous le nom **voyageurMonde**, qui contient tous les aéroports de l'Europe en noeuds et la distance entre eux sur les arêtes.

La requête Cypher utilisée:

```
1 CALL gds.graph.project(  
2   'voyageurMonde',  
3   'airport',  
4   {lien: {orientation: 'NAT',  
5     'distance'}}  
6 )
```

relationshipProjection

```
{  
  "lien": {  
    "orientation": "NATURAL",  
    "aggregation": "DEFAULT",  
    "type": "lien",  
    "properties": {  
      "distance": {  
        "defaultValue": null,  
        "property": "distance",  
        "aggregation": "DEFAULT"  
      }  
    }  
  }  
}
```

graphName

nodeCount

relationshipCount

"voyageurMonde"

7698

305048

Le voyageur du Monde

Choix d'algorithme: All Pairs Shortest Path

- Trouve le chemin le plus court entre toutes les paires de nœuds.
- Se base sur un graph pondéré.

Résultat:



- Matrice des distances minimales entre chaque nœud et tous les autres nœuds du graphs.

```
1 CALL gds.alpha.allShortestPaths.stream('voyageurMonde', {
2   | relationshipWeightProperty: 'distance'
3 })
4 YIELD sourceNodeId, targetNodeId, distance
5 WITH sourceNodeId, targetNodeId, distance
6 WHERE gds.util.isFinite(distance) = true
7
8 MATCH (source:airport) WHERE id(source) = sourceNodeId
9 MATCH (target:airport) WHERE id(target) = targetNodeId
10 WITH source, target, distance WHERE source <> target
11 RETURN source.Name AS source, target.Name AS target, distance
12 ORDER BY distance ASC LIMIT 10
```


Le voyageur du Monde

Résultat:

Fichier .csv qui contient les distances entre les différents aéroports de l'Europe dans un ordre ascendant.

 voyageur_monde_FV 

```
1 source,target,distance
2 Lanzarote Airport,Fuerteventura Airport,60.39055141525801
3 Saarbrücken Airport,Luxembourg-Findel International Airport,79.785658362836
4 Luxembourg-Findel International Airport,Saarbrücken Airport,79.785658362836
5 Aarhus Airport,Aalborg Airport,100.00687717907952
6 Aalborg Airport,Aarhus Airport,100.00687717907952
7 Helsinki Vantaa Airport,Lennart Meri Tallinn Airport,100.88566230018043
8 Lennart Meri Tallinn Airport,Helsinki Vantaa Airport,100.88566230018043
9 Lycksele Airport,Arvidsjaur Airport,118.99224664597621
```

Le voyageur du Monde

Exploitation du résultat pour répondre au besoin:

→ Algorithm Dijkstra pour trouver le plus court chemin en connaissant l'aéroport source et l'aéroport target.

```
def start_process(start_airport, target_airport):  
    nodes, init_graph = generate_graph_from_csv()  
    graph = Graph(nodes, init_graph)
```

Voyageur du Monde Start !

Toulouse-Blagnac Airport -> Hamburg Airport -> Halmstad Airport

2580.9032110587505

Halmstad Airport -> Växjö Kronoberg Airport -> Il Caravaggio International Airport -> Košice Airport

3062.8490264636825

Košice Airport -> Il Caravaggio International Airport -> Toulouse-Blagnac Airport

1988.0184455594317

```
start_process("Halmstad Airport", "Košice Airport")  
start_process("Košice Airport", "Toulouse-Blagnac Airport")
```

Le voyageur du Monde

Visiter 6 villes

Distance totale = **7 630 Km**

Voyageur du Monde Start !

Toulouse-Blagnac Airport -> Hamburg Airport -> Halmstad Airport

2580.9032110587505

Halmstad Airport -> Växjö Kronoberg Airport -> Il Caravaggio International Airport -> Košice Airport

3062.8490264636825

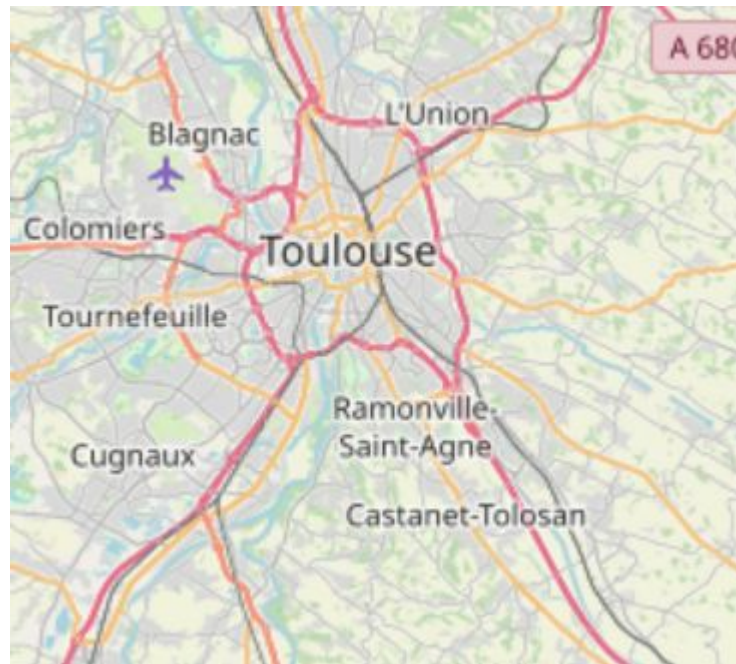
Košice Airport -> Il Caravaggio International Airport -> Toulouse-Blagnac Airport

1988.0184455594317

Le voyageur du Monde

Fonction de visualisation de l'aéroport sur la carte en utilisant Folium

```
▼ def show_me_my_airport(my_airport):  
    loc = return_location(my_airport)  
    map = folium.Map(location=loc)  
    map.save(my_airport+".html")  
  
▼ def return_location(airport_name):  
    location = []  
    airport = airports_df[airports_df['Name'] == airport_name]  
    lat = airport.iloc[0]['Latitude']  
    long = airport.iloc[0]['Longitude']  
    location.append(lat)  
    location.append(long)  
    return location  
  
show_me_my_airport("Toulouse-Blagnac Airport")
```



Demo



Les interconnectés

Les interconnectés

La distance minimale entre deux aéroports est de 150 **km**.

à partir d'un fichier ordonné par distance entre aéroports ascendants, on déduit les aéroports à supprimer.

NoteBook:

http://localhost:8888/notebooks/Documents/GraphAnalytics/5_les%20interconnectes/5.Les%20interconnect%C3%A9s.ipynb

23 aéroports

Résultat:

- Lanzarote Airport
- Saarbrücken Airport
- Luxembourg-Findel International Airport
- Aarhus Airport
- Aalborg Airport
- Helsinki Vantaa Airport
- Lennart Meri Tallinn Airport
- Arvidsjaur Airport
- Lycksele Airport
- Halmstad Airport
- Poitiers-Biard Airport
- La Rochelle-Île de Ré Airport
- Palma De Mallorca Airport
- Menorca Airport
- Munich Airport
- Nuremberg Airport
- Ibiza Airport
- Tampere-Pirkkala Airport
- Ivalo Airport
- Copenhagen Kastrup Airport
- Fuerteventura Airport
- Växjö Kronoberg Airport
- Kittilä Airport

Demo





Merci pour votre attention