Laura VERCHERE
Andrea MACHEDA
Liam HUISSOUD

Pour le 22 mars 2024

HANDS-ON ORIENTÉ GRAPHS
TP SNCF

L'objectif du projet est de pouvoir indiquer à l'utilisateur le trajet ferroviaire à emprunter pour rejoindre une gare de départ A à une gare de départ B, toutes deux préalablement renseignées par celui-ci. Un affinement par heure de départ ou d'heure d'arrivée sera également possible.

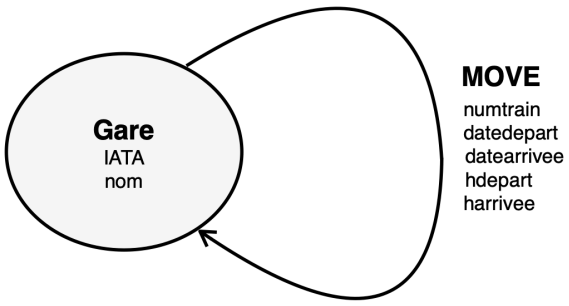
Avant d'arriver à indiquer le trajet à l'utilisateur, il est nécessaire de récupérer et stocker toutes les données du réseau ferroviaire récupérées sur le site de la SNCF. Ici, nous nous sommes concentrées sur les trains TGV INOUI et INTERCITÉS, ce qui représente plus de 350 000 données.

Etape indispensable avant toute exploitation du jeu de données : le Data Cleaning. Effectivement, il est nécessaire de supprimer toutes les données inutiles afin de réduire le nombre de données à exploiter et ainsi améliorer le temps d'exécution de notre programme. Ainsi, nous avons supprimer les lignes de données pour lesquelles :

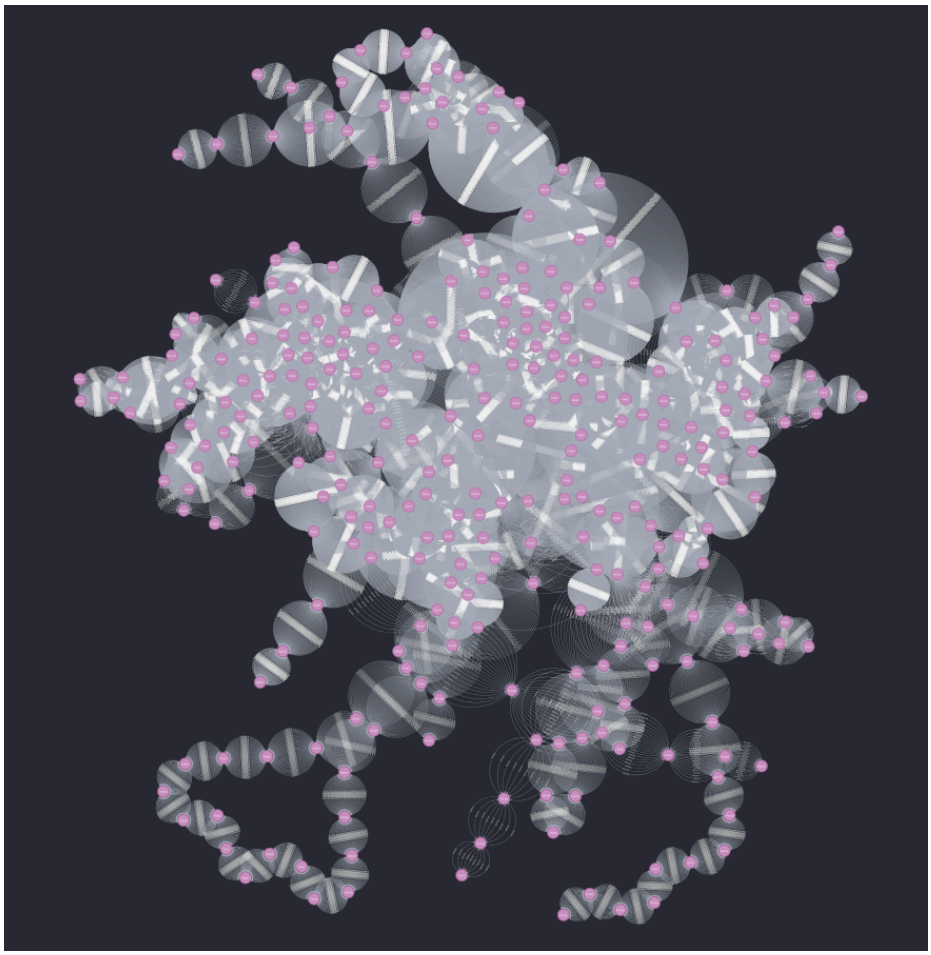
- Le code IATA était DEEAA puisque certaines informations étaient nulles
- Le nom de la gare de départ ou d'arrivée était TBD : nom de gare non recensée et il s'agissait d'autocars et non de trains
- Les doublons. Par exemple, dans la BDD initiale nous avions les trajets A —> B, B—> C et A —> C qui correspondait à un même train. Après traitement, nous n'avions plus que le trajet A—> B —> C

Nous avons dû également nous occuper des trains de nuits et gérer la création d'une date d'arrivée en la fixant au lendemain de la date de départ du train.

Le Data Cleaning nous a ainsi permis de passer de 350 000 données à un peu moins de 100 000. Ainsi, nous avons chercher à optimiser au maximum nos données, pour nous permettre, par la suite, de traiter efficacement ces dernières et ainsi améliorer considérablement le temps d'exécution de notre algorithme final. Après avoir analyser et « cleaner » la base de données, nous pouvons définir le schéma du modèle de celle-ci :



Cette clarification et simplification du modèle de la base de données, nous a permis de la modéliser sous forme de graphe en neo4 :



Ici, les noeuds représentent les gares tandis que les liaisons correspondent aux trains entre deux gares.

Enfin, la dernière étape est l'exploitation de cette nouvelle base de données. Il nous a été demandé d'élaborer un code de programmation permettant les 3 points suivants :

- Afficher tous les trains permettant d'effectuer le trajet entre une gare A et une gare B à une certaine date, toutes trois renseignées par l'utilisateur
- Afficher le trajet direct entre une gare A et une gare B à une date précise, toujours renseignées par l'utilisateur qui pourra aussi indiquer soit une heure de départ ou bien une heure d'arrivée
- L'itinéraire pour réaliser un tour de France pour lequel l'utilisateur devra choisir le nombre d'escales ainsi que les villes qu'il souhaite visiter

Afin de répondre aux fonctionnalités ci-dessous, il nous fallait concevoir un algorithme permettant de trouver le chemin (ici, le train) le plus court entre un nœud de départ (ici, gare de départ) et un nœud de destination (ici, gare d'arrivée). Nous avions deux choix d'algorithme pour cela : le A* ou le dijkstra.
Après avoir implémenter l'algorithme A* sur notre base de données, nous nous sommes rapidement rendu compte qu'il n'était pas l'algorithme le plus optimal dans notre situation, puisqu'il ne nous affichait pas le trajet le plus rapide, bien au contraire. Malgré une rapidité d'exécution nettement moins importante que l'algorithme A*, l'algorithme dijkstra nous permet d'afficher le plus court trajet. Notre choix s'est donc porté sur cet algorithme. S