ENSAI

Année scolaire 2016-2017

Projet de Technologies NoSQL

Filière SID

Quel est le meilleur endroit où habiter à New York ?

Laure Nicollet

Qu'est-ce que le "meilleur endroit" ?

Ca y est ! Nos billets d'avion pour New York sont réservés ! Mais nous n'avons toujours pas de logement... Nous ne pouvons pas chercher parmi tous les appartements à louer à New York, il y en aurait beaucoup trop ! Mais alors où chercher... ?

Il n'y a en réalité qu'une chose qui nous tienne vraiment à coeur : un peu de verdure ! Après 3 ans passés entre les étangs verdoyants du campus de Ker Lann, difficile de s'imaginer entourés d'immeubles grisâtres ! Notre premier critère de recherche seront donc de trouver un logement à proximité immédiate d'un parc. Et si possible, un parc suffisament grand pour oublier un peu la circulation qui ne manquera pas de l'entourer.

Mais cela ne suffit évidemment pas. A quoi bon trouver un appartement de rêve, avec vue sur un magnifique parc, si nous perdons chaque jour 2h à aller au travail et que n'avons plus le temps d'en profiter ? Le sport le plus physique que nous aimons exercer étant une petite promenade tranquille dans ce charmant parc qui bordera notre appartement, le vélo est exclu. Nous essayerons donc d'être proches d'un arrêt de métro. Et par proche, disons moins d'1km, toujours pour éviter trop d'exercice physique...

Évidemment, nous aimerions aussi être près d'un cinéma, de musées, de bons restaurants ou autres lieux de sorties tous plus plaisants les uns que les autres. Mais il nous semble impossible qu'un quartier réunisse toutes ces conditions. Par ailleurs, si nous sommes suffisamment proches d'un arrêt de métro, nous pourrons rapidement aller à d'autres endroits pour ces sorties plus occasionnelles. Nous nous cantonnerons donc à ces deux critères qui semblent déjà définir un lieu de vie somme toute très sympathique.

Comment trouver cet endroit idéal ?

Avant toute chose, le dossier projet_nicollet récupéré sur github doit être placé dans le répertoire nommé "Dossier personnel" fourni par Ubuntu.

0 - Installation de R

Comme nous allons télécharger nos données et les préparer avec le langage R, il faut dans un premier temps installer ce logicel en éxécutant les commandes suivantes dans un nouveau terminal :

- Se placer dans le répertoire du projet (`cd projet_nicollet`)
- Rendre le script d'installation exécutable avec `chmod a+x 0_installR.sh`
- Exécuter le script d'installation : `./0_installR.sh`.

1 - Téléchargement des bases de données

Nous allons à présent télécharger les bases de données qui seront utilisées. Pour cela, rester dans le terminal, toujours dans le répertoire contenant les scripts à éxécuter (`projet_nicollet`). Il faut ensuite éxécuter le script de téléchargement et de préparation des données, fait en langage R : `sudo Rscript 1_dataPreparation.R`.

Un sous-répertoire nommé **data** est créé. Il contient les 3 jeux de données qui seront utilisés, tous au format geojson :

- Le premier contient la liste des blocs de recensements, équivalents aux IRIS en France. Cela permet un maillage très fin de la ville de New York, sur lequel on se basera pour décider où habiter : nous choisirons de vivre dans le bloc qui correspondra le mieux à nos critères.
- Le deuxième contient la liste des parcs de New York.
- Le troisième contient la liste des stations de métros.

Chacun de ces fichiers contient les coordonnées des objets concernés, sous forme de point (latitude, longitude) pour les stations de métros, et sous forme de polygone (liste de points) pour les parcs et blocs de recensement. En plus des coordonnées, nous avons conservés quelques autres attributs susceptibles de nous servir dans le choix du meilleur endroit (mais avons supprimé la majorité des attributs fournis au départ car ils ne nous intéressent pas et il n'est donc pas nécessaire de les charger en base). Pour les stations de métros, nous conservons en plus un identifiant, le nom de la station de métro et les lignes qui y passent. Nous gardons l'identifiant, le nom, et la taille des parcs. Nous ne conservons rien d'autre que les coordonnées et l'identifiant pour les blocs de recensement.

2 - Choix et installation de la base de donées

Nous avons choisi d'utiliser la base MongoDB. En effet, l'aspect graphe de Neo4j ne semble pas être le plus pratique pour les traitements que nous souhaitons faire. Par ailleurs, après une recherche rapide pour comparer Cassandra, Redis et MongoDB, les outils pour effectuer des requêtes spatiales les plus appropriés à nos jeux de données sont ceux de MongoDB, notamment car ils permettent de traiter simplement les polygones donnés au format geojson.

Voici les étapes nécessaires à l'installation de MongoDB, toujours dans un terminal :

— Vérifier que vous êtes toujours dans le même répertoire que précédemment (sinon exécuter `cd projet_nicollet`)

— Rendre le script d'installation exécutable avec `chmod a+x 2_mongoInstall.sh`

— Exécuter le script d'installation : `./2_mongoInstall.sh`. Ce dernier installe mongoDB, mais aussi robomongo, qui permet d'avoir une interface plus user-friendly. Il lance ensuite mongo et robomongo (ce qui peut nécessiter que votre mot de passe soit demandé).

Dans le fenêtre de robomongo qui vient de s'ouvrir, cliquer sur create pour créer une nouvelle connection. Entrez les paramètres NY dans le champ Name, et localhost :27017 dans le champ Address, puis cliquez sur Save. Sélectionnez alors la connection NY que nous venons de créer, puis cliquez sur connect.

3 - Insertion en base des jeux de données

Nous allons maintenant importer nos jeux de données geojson dans notre base mongoDB. Pour cela, il faut éxécuter les étapes suivantes :

— Ouvrir un nouveau terminal et se placer dans le bon répertoire avec `cd projet_nicollet`

— Rendre le script de chargement des données en base exécutable avec `chmod a+x 3_import.sh`

— Exécuter ce script : `./3_import.sh`. Ce dernier crée 3 collections dans la base NY, que nous avons créée précédemment dans mongoDB. Les trois collections correspondent aux différents types de données que nous utilisons : les métros (**subways**), les parcs (**parks**), et les blocs de rencensement (**censusBlocks**).

4 - Requêtes sur la base de données pour trouver l'endroit idéal

L'ensemble des requêtes exécutées pour trouver le meilleur endroit figure dans le document requetes.txt. Le code complet est à copier puis exécuter dans un shell robomongo. L'algorithme complet exécuté pour obtenir le lieu où l'on voudrait habiter est le suivant :

— Des index sont créés sur les trois collections afin de rendre possible ou plus rapides (selon l'opérateur) les requêtes géospatiales. Pour les collections **parks** et **censusBlocks**, la création d'un index de façon traditionnelle échoue car certains objets semblent mal formés. Pour pallier à cela, une astuce est utilisée : on crée d'abord l'index sur la collection vide puis on ajoute les objets un par un. Ceux qui sont mal formés ne sont pas ajoutés. On perd ainsi 8 parcs sur les 2008 de départ, ce qui importe peu. On passe aussi de 38794 blocs de recensement à un peu plus de 24000, ce qui est plus gênant, mais devrait néanmoins suffire à trouver un endroit qui correspondrait à nos critères de recherche (cf encadré à la fin de cette section si le nombre de blocs conservé est très différent *).

— On choisit alors de commencer par ne garder que les blocs situés à moins d'un kilomètre d'un arrêt de métro, grâce aux opérateurs $near et $maxDistance. Tous les blocs qui conviennent sont placés dans une nouvelle collection appellée **censusBlocksNearSubway**.

— Comme nous aimerions vivre à proximité immédiate d'un parc, nous éxécutons alors une seconde requête. Parmi ces blocs suffisamment proches d'un arrêt de métro, nous ne conservons que ceux dont l'intersection avec un parc n'est pas nulle. Les blocs correspondant sont conservés dans une dernière collection nommée **censusBlocksNearSubwayAndPark**. Ainsi, ces blocs sont suffisamment proches d'un arrêt de métro, et une partie du bloc au moins fait partie d'un parc ou en contient un complet. Dans cette table, nous ajoutons à chaque objet 2 attributs : le nom du parc qui est en intersection avec le bloc, et la taille de ce parc.

— Enfin, pour choisir parmi la dizaine de blocs retenus, nous trions les résultats en fonction de la taille du parc associé à chaque bloc. Le bloc retenu est celui associé au plus grand parc.

L'avant dernière commande du documents requetes.txt affiche dans mongoDB le bloc retenu. En habitant entre les coordonnées du polygone données dans l'attribut geometry, nous pourrons ainsi profiter des 7,85 acres (3,18 hactares) du parc Sara D. Roosevelt pour notre plus grand plaisir, tout en étant à moins d'un kilomètre d'une station de métro qui nous permettra par la suite de nous rendre rapidement en tout point de notre nouvelle ville.

Par ailleurs, la dernière requête nous permet de vérifier que nous sommes proches de multiples arrêts et lignes de métros, puisqu'il y à déja 13 stations à moins d'un kilomètre du premier point du bloc de recensement chois, et donc potentiellement plus si l'on regarde aussi lesquelles sont à moins d'un kilomètre de l'autre bout du bloc (malheureusement, l'opérateur $near ne peut pas s'appliquer à un polygone mais seulement à un point). Par ailleurs, même en réduisant à 500 mètres si nous voulons faire preuve d'une grande phobie de tout exercice physique, nous avons toujours 3 stations de métros proches, avec 6 lignes différentes (F, 4, 6, 6 express, J et Z), ce qui devrait déjà nous premettre de nous rendre en de multiple endroits suffisamment rapidement et facilement.

*** IMPORTANT** : Nous avons remarqué que d'une éxécution à l'autre, le résultat varie. En effet, lors de la création d'un index sur les blocs de recensement ayant un format valide (lignes 30 à 37 du document requetes.txt), le nombre de documents considérés comme valides et conservés change d'une éxécution à l'autre. N'ayant pas réussi à trouver d'explication, nous avons conservé les résultats obtenus lors d'un éxécution qui conservait un maximum de blocs de recensement (environ 25 000 alors qu'il n'en gardait parfois à peine plus de 10 000). Le résultat de toutes les autres commandes n'a pas semblé changer entre une éxécution et l'autre, hormis les changements impliqués par le fait que certains blocs de recensement n'étaient pas conservés.

5 - Déterminer l'adresse de notre appartement idéal

Puisque toute adresse dans le bloc nous convient, nous pouvons chercher l'adresse correspondant à n'importe quel point du bloc. En rentrant les coordonées du premier point (longitude : -73.9902736164793, latitude : 40.7234260182932) sur le site http ://www.coordonnees-gps.fr/conversion-coordonnees-gps, on trouve l'adresse suivante : **200 Forsyth St, New York, NY 10002, États-Unis**.

Pour avoir plus de choix, on peut aussi alors retrouver tout le bloc de recensement correspondant (avec les coordonnées et l'id 10036013000) sur la carte de l'open data de la ville de new york où l'on avait téléchargé les données (https ://data.cityofnewyork.us/City-Government/2010-Census-Blocks/v2h8-6mxf). Il correspond au nord du parc Sara D. Roosevelt. Une rapide recherche sur google maps nous permet de trouver que la zone est bordée par les numéros **172 à 200, Forsyth St** à l'est, et les numéros **179 à 229, Chrystie St**, à l'ouest. C'est donc parmi ces numéros que nous chercherons notre nouvel appartement (les rues longeant le parc au nord et au sud ne semblent pas disposer d'habitations).

Nous pouvons donc à présent quitter robomongo, et exécuter la commande `service mongod stop` dans un terminal avant de nous lancer dans la recherche active d'un appartement situé à l'endroit souhaité.