Rapport Mongo DB (réseau de transport)

Alexis Araujo
But info3 fi

DEMANDE 1

Proposer une organisation des données du TP1 sous la forme d'une base de données Mongo DB.

Collection Arrets_lignes

Je propose de faire une collection "arrets_lignes" :
Cette collection stockera des informations sur les arrêts de transport, y compris leur emplacement géographique et des détails supplémentaires, ainsi que les informations de la ligne associée.

Exemple de document pour cette collection :


```
[
  {
    "route_id": "IDFM:C01727",
    "stop_id": "IDFM:monomodalStopPlace:47905",
    "stop_name": "Bouray",
    "stop_lon": 2.2900778137432436,
    "stop_lat": 48.53290340283496,
    "OperatorName": "SNCF",
    "Nom_commune": "Lardy",
    "Code_insee": "91330"
  },
  {
    "route_id": "IDFM:C01727",
    "stop_id": "IDFM:monomodalStopPlace:43081",
    "stop_name": "Étréchy",
    "stop_lon": 2.1947692381982216,
    "stop_lat": 48.49387742150091,
    "OperatorName": "SNCF",
    "Nom_commune": "Étréchy",
    "Code_insee": "91226"
  }
  // Ajoutez d'autres documents ici
]
```


Collection Routes

Je propose de faire une collection "routes" :
Cette collection contiendra des informations sur les itinéraires de transport, y compris les noms courts et longs des itinéraires.

Exemple de document pour cette collection :

```
[
  {
    "route_id": "IDFM:C01729",
    "route_short_name": "E",
    "route_long_name": "E"
  },
  {
    "route_id": "IDFM:C01728",
    "route_short_name": "D",
    "route_long_name": "D"
  }
  // Ajoutez d'autres documents ici
]
```

Collection Stop_times

Je propose de faire une collection “stop_times” :
Cette collection stockera des informations sur les heures d'arrivée et de départ aux arrêts pour chaque voyage.

Exemple de document pour cette collection :

```
[
  {
    "trip_id": "IDFM:TN:SNCF:28bf25f5-5248-48be-96b0-c62b689c4543",
    "stop_id": "IDFM:monomodalStopPlace:44411",
    "stop_sequence": 1,
    "arrival_time": "12:58:50",
    "departure_time": "12:58:50"
  },
  {
    "trip_id": "IDFM:TN:SNCF:28bf25f5-5248-48be-96b0-c62b689c4543",
    "stop_id": "IDFM:monomodalStopPlace:47599",
    "stop_sequence": 2,
    "arrival_time": "13:00:20",
    "departure_time": "13:01:00"
  }
  // Ajoutez d'autres documents ici
]
```

Collection Stops

Je propose de faire une collection “stops” :
Cette collection contiendra des informations sur les arrêts de transport, y compris leur nom et la station parente associée.

Exemple de document pour cette collection :

```
[
  {
    "stop_id": "IDFM:monomodalStopPlace:47052",
    "stop_name": "Courcelle-sur-Yvette",
    "parent_station": "IDFM:62951"
  },
  {
    "stop_id": "IDFM:monomodalStopPlace:43246",
    "stop_name": "La Ferté-sous-Jouarre",
    "parent_station": "IDFM:68918"
  }
  // Ajoutez d'autres documents ici
]
```

Collection Trips

Je propose de faire une collection “trips” :
Cette collection stockera des informations sur les voyages, y compris l'itinéraire associé et la destination.

Exemple de document pour cette collection :

```
[
  {
    "route_id": "IDFM:C01727",
    "trip_id": "IDFM:TN:SNCF:28bf25f5-5248-48be-96b0-c62b689c4543",
    "trip_headsign": "MONA"
  },
  {
    "route_id": "IDFM:C01727",
    "trip_id": "IDFM:TN:SNCF:fb1d9600-d2bc-4284-bf22-0f7e5e0e9f79",
    "trip_headsign": "CIME"
  }
  // Ajoutez d'autres documents ici
]
```

DEMANDE 2

Créer des script pour restructurer les données du TP2 conformément à la structure de document (objet JSON) choisie. Les scripts peuvent être définis dans un langage choisi librement.

Voir le script python : main_convert_tsv_to_json et également voir le dossier json avec les données.

DEMANDE 3

Exprimer les requêtes données dans le premier TP sous la forme des requêtes MongoDB.

1. Quelles sont les stations d'une ligne X donnée (par exemple, "RER B") ?

```
db.arrets_lignes.find({ "route_id": "RER B" }, { "stop_name": 1, "_id": 0 })
```

2. Quelles sont toutes les stations au sud d'une station X donnée ?

```
db.arrets_lignes.find({ "stop_lat": { $lt: latitude_de_la_station_X } }, { "stop_name": 1, "_id": 0 })
```

3.

4. Y a-t-il une connexion directe de X à Y (un train qui va de X à Y) ?

```
// Remplacez "X" et "Y" par les IDs des stations X et Y
var stationX = "ID_DE_LA_STATION_X";
var stationY = "ID_DE_LA_STATION_Y";

// Vérifiez s'il existe un train qui dessert les stations X et Y dans l'ordre
var directConnection = db.stop_times.find(
    { "stop_id": stationX },
    { "trip_id": 1, "_id": 0 }
).forEach(function (doc) {
    var tripID = doc.trip_id;
    var hasStationY = db.stop_times.findOne({ "trip_id": tripID, "stop_id": stationY });
    if (hasStationY) {
        print("Il y a une connexion directe de X à Y.");
    }
});
```

5. Combien de stations séparent les stations X et Y de la même ligne ?

```
// Remplacez "X" et "Y" par les IDs des stations X et Y, et "Ligne_Z" par l'ID de la ligne
var stationX = "ID_DE_LA_STATION_X";
var stationY = "ID_DE_LA_STATION_Y";
var ligneZ = "ID_DE_LA_LIGNE_Z";

// Recherchez les positions des stations X et Y dans la liste des stations de la ligne Z
var positionX = db.arrets_lignes.findOne({ "stop_id": stationX, "route_id": ligneZ
}).stop_sequence;
var positionY = db.arrets_lignes.findOne({ "stop_id": stationY, "route_id": ligneZ
}).stop_sequence;

// Calculez le nombre de stations entre X et Y
var stationsSeparation = Math.abs(positionY - positionX) - 1;
print("Le nombre de stations entre X et Y est : " + stationsSeparation);
```

6. Quelle est la durée minimale d'un trajet de X à Y, étant donné qu'il existe une connexion directe ?

```
// Remplacez "X" et "Y" par les IDs des stations X et Y
var stationX = "ID_DE_LA_STATION_X";
var stationY = "ID_DE_LA_STATION_Y";

// Recherchez les horaires de départ de la station X et d'arrivée de la station Y
var departX = db.stop_times.find({ "stop_id": stationX }).sort({ "departure_time": 1
}).limit(1).next();
var arriveeY = db.stop_times.find({ "stop_id": stationY }).sort({ "arrival_time": -1
}).limit(1).next();

// Calculez la durée minimale
var dureeMinimale = (new Date(arriveeY.arrival_time) - new Date(departX.departure_time))
/ 1000; // en secondes
print("La durée minimale d'un trajet de X à Y est de " + dureeMinimale + " secondes.");
```

7. Y a-t-il une connexion avec un seul changement entre X et Y, étant donné qu'il n'y a pas de connexion directe ?

```
// Remplacez "X" et "Y" par les IDs des stations X et Y
var stationX = "ID_DE_LA_STATION_X";
var stationY = "ID_DE_LA_STATION_Y";

// Recherchez des itinéraires potentiels de X à Y avec un seul changement
var itineraires = db.itineraires.find({
    "stations": { $all: [stationX, stationY] },
    "changement": 1
});

if (itineraires.count() > 0) {
    print("Il y a une connexion avec un seul changement entre X et Y.");
} else {
    print("Aucune connexion avec un seul changement entre X et Y.");
}
```

8. Combien de changements de ligne pour aller de X à Y ?

```
// Remplacez "X" et "Y" par les IDs des stations X et Y
var stationX = "ID_DE_LA_STATION_X";
var stationY = "ID_DE_LA_STATION_Y";

// Recherchez des itinéraires potentiels de X à Y
var itineraires = db.itineraires.find({
    "stations": { $all: [stationX, stationY] }
});

// Pour chaque itinéraire, comptez le nombre de changements de ligne
var nombreChangementsMax = 0;

itineraires.forEach(function(itineraire) {
    var changements = itineraire.lignes.length - 1;
    if (changements > nombreChangementsMax) {
        nombreChangementsMax = changements;
```

```
    }
});

print("Le nombre maximal de changements de ligne pour aller de X à Y est : " +
nombreChangementsMax);
```