HAI914I / HMIN340 : Gestion des données NoSQL

- Session : 1
- Date : 14 janvier 2021
- Durée : 2H
- Documents autorisés : tous
- Matériel utilisé : aucun

1. Questions de cours (6 points)

Vous expliquerez les mérites comparés des systèmes NOSQL face aux SGBD relationnels, en exploitant au besoin les notions suivantes :

1. scalabilité (verticale et horizontale)
2. réplication
3. partitionnement
4. persistance polyglotte
5. modèle semi-structuré
6. postulat CAP et système distribué

Vous pouvez vous aider des trois systèmes NOSQL étudiés et de votre connaissance du relationnel pour illustrer vos propos.

Réponse

1. Les systèmes NoSQL sont facilement scalable (verticalment ET horizontalement) au détriment de la cohérence (C) ou de la disponibilité (A) des données énoncés par le postulat CAP. On peut mettre cette avantage en comparaison avec les systèmes relationnels qui ont une limite de scalabilité verticale au détriment du partionnement (P) des données énoncés par le postulat CAP. Certains systèmes NoSQL, sont utilisés parce qu'ils permettent d'aggréger un grand volume de données (la charge est répartie sur plusieurs noeuds).
2. Un seconde avantage des systèmes NoSQL, réside dans la flexibilité des données, qui est du au modèle semi-structuré. Alors qu'un système relationnels ne supporte que des modèles structurés, inflexible et nécéssitant de la normalisation. Par ailleurs, cette normalisation des systèmes relationnels vient enmpiétés sur la performance des requêtes de jointure relationnels, qui est moindre dans un système semi-structuré.

2. Le réseau de transport de la TAM et NoSQL

2.1 Enoncé

Les parties portant sur les systèmes de persistance (vus en cours) exploitent tous le contexte des transports urbains de l'agglomération de Montpellier (TAM)

2.2 Partie Neo4J (6 points)

1. Des ordres de création de noeuds et d'arêtes vous sont donnés dans la syntaxe Cypher. Vous construirez un premier graphe à partir des deux ordres de création donnés ci-dessous.

```
CREATE
    (s1:Station {nom:'Occitanie', latitude:43.63435811, longitude:3.84863696}) -[:STOP_DE]-> (la1:Liaison {distance:500}),
    (la1) -[:APPARTIENT_A]-> (li1:Ligne {nom:'Mosson-Odysseum', num:1, type:'tram',longueur:15700, tempsTotal:50}),
    (la2:Liaison {distance:1000}) -[:APPARTIENT_A]-> (li1),
    (s2:Station {nom:'Hopital Lapeyronie',latitude:43.63166867, longitude:3.85260055}) -[:STOP_DE]-> (la1),
    (s1) -[:STOP_DE]-> (la2),
    (s3:Station {nom:'Chateau d O',latitude:43.63131727, longitude:3.84299936}) -[:STOP_DE]-> (la2),
    (s4:Station {nom:'Univ. Sciences et Lettres',latitude:43.63131727, longitude:3.84299936}) -[:STOP_DE]-> (la3:Liaison {distance:1
    (s2) -[:STOP_DE]-> (la3),
    (la3)-[:APPARTIENT_A]-> (li1)
RETURN s1, s2, s3, s4, li1, la1, la2, la3
```

graph2.1.png

```
MATCH (o:Station {nom:'Occitanie'})
CREATE
    (sp:Station {nom:'Saint-Priest', latitude:43.836699, longitude:4.360054}) -[:STOP_DE]-> (l:Liaison {distance:400}),
    (l) -[:APPARTIENT_A]-> (li:Ligne {nom:'Euromedecine-Pas du Loup', num:6, type:'bus'}),
    (o) -[:STOP_DE]-> (l)
```

graph2.2.png

Vous choisirez le nom des stations, le numéro (num) des lignes, et la distance des liaisons pour donner une étiquette aux noeuds visualisés et vous mentionnerez les types de chacun des noeuds. Les autres propriétés des noeuds ne seront pas représentées.

2. Une requête de consultation en langage Cypher, vous est donnée qui porte sur le graphe qui vient d'être créé. Vous donnerez la signification de cette requête, ainsi que le résultat renvoyé par cette requête.

```
MATCH (l:Ligne {type:'bus'}) <-[a:APPARTIENT_A] - () <-[st:STOP_DE]- (s:Station)
RETURN DISTINCT l.nom, s.nom
```

```
╒═══════════════════════════╤═══════════════╕
│"l.nom"                    │"s.nom"        │
╞═══════════════════════════╪═══════════════╡
│"Euromedecine-Pas du Loup"│"Occitanie"    │
├───────────────────────────┼───────────────┤
│"Euromedecine-Pas du Loup"│"Saint-Priest"│
└───────────────────────────┴───────────────┘
```

Grâce a cette requete, l'utilisateur souhaite récupéré tous les noms de station de bus et les noms des lignes de bus du réseau de la TAM.

3. Une nouvelle requête Cypher vous est donnée (toujours sur le graphe créé). Vous donnerez également la signification de cette requête et et vous proposerez le sous-graphe qui est le résultat de la requête.

```
MATCH (l1:Liaison) <-[:STOP_DE]- (s:Station) -[:STOP_DE]-> (l2:Liaison)
RETURN l1, s , l2
```

```
╒═══════════════╤═══════════════════════════════════════════════════════════════════════════════╤═══════════════╕
│"l1"           │"s"                                                                            │"l2"           │
╞═══════════════╪═══════════════════════════════════════════════════════════════════════════════╪═══════════════╡
│{"distance":1000}│{"nom":"Occitanie","latitude":43.63435811,"longitude":3.84863696}            │{"distance":400} │
├───────────────┼───────────────────────────────────────────────────────────────────────────────┼───────────────┤
│{"distance":500} │{"nom":"Occitanie","latitude":43.63435811,"longitude":3.84863696}            │{"distance":400} │
├───────────────┼───────────────────────────────────────────────────────────────────────────────┼───────────────┤
│{"distance":400} │{"nom":"Occitanie","latitude":43.63435811,"longitude":3.84863696}            │{"distance":1000}│
├───────────────┼───────────────────────────────────────────────────────────────────────────────┼───────────────┤
│{"distance":500} │{"nom":"Occitanie","latitude":43.63435811,"longitude":3.84863696}            │{"distance":1000}│
├───────────────┼───────────────────────────────────────────────────────────────────────────────┼───────────────┤
│{"distance":400} │{"nom":"Occitanie","latitude":43.63435811,"longitude":3.84863696}            │{"distance":500} │
├───────────────┼───────────────────────────────────────────────────────────────────────────────┼───────────────┤
│{"distance":1000}│{"nom":"Occitanie","latitude":43.63435811,"longitude":3.84863696}            │{"distance":500} │
├───────────────┼───────────────────────────────────────────────────────────────────────────────┼───────────────┤
│{"distance":500} │{"nom":"Hopital Lapeyronie","latitude":43.63166867,"longitude":3.85260│{"distance":1200}│
│               │055}                                                                           │               │
├───────────────┼───────────────────────────────────────────────────────────────────────────────┼───────────────┤
│{"distance":1200}│{"nom":"Hopital Lapeyronie","latitude":43.63166867,"longitude":3.85260│{"distance":500} │
│               │055}                                                                           │               │
└───────────────┴───────────────────────────────────────────────────────────────────────────────┴───────────────┘
```

~~Par cette requète, l'utilisateur désire connaitre les stations et les distances entre stations composants le sous-réseau obtenu (sans les ""terminus"" noeuds avec un seul arc sortant "stop_de").~~ Toutes les stations qui sont entre 2 stations. Le sous-graphe est composé des liaisons `[l, la2, la3, la1]` et des stations `[s2, s1]`

4. Vous écrirez en langage Cypher, la requête : "donner le nombre de stations qui sont situées sur la ligne portant le nom Mosson-Odysseum"

```
MATCH (s:Station)-[:STOP_DE]-(:Liaison)-[:APPARTIENT_A]-(l:Ligne {nom:"Mosson-Odysseum"})
RETURN distinct count(s) as nb_station_ligne_1
```

```
╒═══════════════════╕
│"nb_station_ligne_1"│
╞═══════════════════╡
│6                  │
└───────────────────┘
```

5. Le choix de représentation arrêté pour représenter les lignes, les stations et leurs connexions manque d'efficacité, proposez une simplification du modèle pour le rendre plus performant.
    1. Je simplifierai tout d'abords l'utilisation d'une type liaison, par un lien direct entre les stations et en injectant la distance de ce lien dans ce lien.
    2. Enfin, on dérive la relation Appartient_A, entre une station et une ligne.

2.3 Partie CouchDB (4 points)

```
{
  "docs": [
    {
      "_id": "Occitanie",
      "nom": "Occitanie",
      "latitude": 43.63435811,
      "longitude": 3.84863696,
      "type": "station",
      "correspondances": [
        {
          "libelle": "Mosson-Odysseum",
          "numLigne": 1,
          "distance": 471,
          "vitesseMax": 50,
          "typeLigne": "tram",
          "destination": "Hopital Lapeyronie"
        },
        {
          "libelle": "Mosson-Odysseum",
          "numLigne": 1,
          "distance": 799,
          "typeLigne": "tram",
          "destination": "Chateau d O"
        },
        {
          "libelle": "Euromedeine-Pas du Loup",
          "numLigne": 6,
          "typeLigne": "bus",
          "destination": "Saint-Priest"
        }
      ]
    },
    {
      "_id": "Hopital Lapeyronie",
      "nom": "Hopital Lapeyronie",
      "type": "station",
      "correspondances": [
        {
          "libelle": "Mosson-Odysseum",
          "numLigne": 1,
          "distance": 912,
          "vitesseMax": 40,
          "typeLigne": "tram",
          "destination": "Universites Sciences et Lettres"
        },
        {
          "libelle": "Mosson-Odysseum",
          "numLigne": 1,
          "distance": 471,
          "vitesseMax": 50,
          "typeLigne": "tram",
          "destination": "Occitanie"
        }
      ]
    }
  ]
}
```


Vous donnerez votre compréhension des vues Map et Map/Reduce suivantes (écriture en javascript du corps de la fonction map / des fonctions map/reduce). Quels sont les résultats renvoyés (illustrez les résultats obtenus avec les documents présentés au dessus).

1. Vue Map 1

```
function(doc) { if (doc.nom=='Occitanie')
    emit([doc._id, doc.latitude, doc.longitude], doc.correspondances);
}
```

- Compréhension : On souhaite récupéré l'id, la localisation gps et les correspondances d'un document ayant pour attribut nom "Occitanie"
- Résultat :


```
{ "total_rows":1, "offset":0, "rows" :[
    {"id":"Occitanie","key":{"_id":"Occitanie","latitude": 43.63435811,"longitude": 3.84863696,}, "value":["correspondances": [
        {
          "libelle": "Mosson-Odysseum",
          "numLigne": 1,
          "distance": 471,
          "vitesseMax": 50,
          "typeLigne": "tram",
          "destination": "Hopital Lapeyronie"
        },
        {
          "libelle": "Mosson-Odysseum",
          "numLigne": 1,
          "distance": 799,
          "typeLigne": "tram",
          "destination": "Chateau d O"
        },
        {
          "libelle": "Euromedeine-Pas du Loup",
          "numLigne": 6,
          "typeLigne": "bus",
          "destination": "Saint-Priest"
        }
      ]]
  }
]}
```


2. Vue Map 2

```
function(doc) { if (doc.type=='station')
    for(var idx in doc.correspondances) {
        correspondance = doc.correspondances[idx];
        emit(doc._id,
        correspondance.destination);
    }
}
```

- Compréhension : On souhaite acquérir le nom de toutes les destinations atteignables a partir des correspondances de toutes les stations. Ca reviendrai a acquérir le nom de toutes les stations composant le réseau de la TAM
- Résultat :


```
{ "total_rows":5, "offset":0, "rows" :[
    {"id":"Occitanie","key":"Occitanie", "value":"Hopital Lapeyronie"},
    {"id":"Occitanie","key":"Occitanie", "value":"Chateau d O"},
    {"id":"Occitanie","key":"Occitanie", "value":"Saint-Priest"},
    {"id":"Hopital Lapeyronie","key":"Hopital Lapeyronie", "value":"Universites Sciences et Lettres"},
    {"id":"Hopital Lapeyronie","key":"Hopital Lapeyronie", "value":"Occitanie"},
]}
```


3. Vue Map Reduce

```
function(doc) { if (doc.type=='station')
    for(var idx in doc.correspondances) {
        correspondance = doc.correspondances[idx];
        if (correspondance.numLigne==1){
            emit(doc._id,correspondance.distance);
        }
    }
}
```

```
function(keys, values) {
    return Math.max.apply(null, values);
}
```

- Compréhension : On veut calculer la station avec la distance maximale
- Résultat : "Hopital Lapeyronie" : "912"

```

```