NoSQL - DIA2

MANNAI Hasna (≈ 33%)
CHENIK Yassine (≈ 33%)
BOUCHIBA Emine (≈ 33%)

RAPPORT
TP Cassandra

TABLE DES MATIERES

Importation du json dans Cassandra

Dézipper le fichier JSON :

Utilisation de WinRar pour dézipper le fichier et le mettre dans notre répertoire de travail.

Lancement de Docker :

Une fois Docker lancé et le container Cassandra lancé :

- ➔ **On lance le terminal Windows dans lequel on tape la ligne de commande suivante :**
  - ✓ **Docker cp "C:\Users\emine\Documents\Ecole\A4\S8\Advanced_topics_in_NoSql_databases\TP_Cassandra\companies2.json" Cassandra:/**
- ➔ **On entre dans le terminal Cassandra :**
  - ✓ **cqlsh (dans le terminal Docker)**
- ➔ **On lance la création d'un KeySpace :**
  - ✓ **CREATE KEYSPACE IF NOT EXISTS Companies2_Cassandra WITH REPLICATION = { 'class' : 'SimpleStrategy', 'replication_factor': 3 };**
- ➔ **On se met dans ce KeySpace :**
  - ✓ **USE Companies2_Cassandra;**

Création des tables

Script de création des Types que l'on va utiliser :

✓ relationshipsType :

- CREATE TYPE IF NOT EXISTS relationshipsType (is_past BOOLEAN,title TEXT,person map<TEXT,TEXT>);

✓ investmentsType :

- CREATE TYPE IF NOT EXISTS investmentsType(map<TEXT,TEXT>,financial_org map<TEXT,TEXT>, person map<TEXT,TEXT>);

✓ roundType :

- CREATE TYPE IF NOT EXISTS roundType(id INT , round_code TEXT, source_url TEXT, source_description TEXT, raised_amount INT, raised_currency_code TEXT, funder_year INT, funder_month INT, funder_day INT, investments list<frozen<investmentsType>>);

Script de création de la table Company :

```
CREATE TABLE  companies(_id map<TEXT,TEXT>, name TEXT, permalink TEXT, crunchbase_url TEXT, homepage_url TEXT, blog_url TEXT, blog_feed_url TEXT , twitter_username TEXT, category_code TEXT, number_of_employees INT, founded_year INT, founded_month INT, founded_day INT, deadpooled_year INT, deadpooled_month INT, deadpooled_day INT, deadpooled_url TEXT, tag_list TEXT, alias_list TEXT, email_address TEXT, phone_number TEXT, description TEXT, created_at TEXT, updated_at TEXT, overview TEXT, total_money_raised TEXT, PRIMARY KEY (_id, permalink)
);
ALTER TABLE companies WITH GC_GRACE_SECONDS = 0;
CREATE INDEX IF NOT EXISTS  company_id ON companies(_id) ;
CREATE INDEX IF NOT EXISTS  permalink_id ON companies(permalink);
```

Script de création de la table Product :

```
CREATE TABLE Product (_id map<TEXT,TEXT>,products list<map<TEXT,TEXT>>,
PRIMARY KEY (_id)
);
ALTER TABLE Product WITH GC_GRACE_SECONDS = 0;
CREATE INDEX IF NOT EXISTS  product_id ON Product(_id) ;
```

Création des tables

Script de création de la table Office_Company :

```
CREATE TABLE Office_Company(_id map<TEXT,TEXT>,offices list<frozen<map<TEXT,TEXT>>>,
coord frozen<map<TEXT,frozen<map<TEXT,list<double>>>>>,
PRIMARY KEY (_id)
);
ALTER TABLE Office_Company WITH GC_GRACE_SECONDS = 0;
CREATE INDEX IF NOT EXISTS  office_company_id ON Office_Company (_id);
```

Script de création de la table Relationships :

```
CREATE TABLE Relationships(_id map<TEXT,TEXT>,relationships list<frozen<relationshipsType>>,
PRIMARY KEY (_id);
ALTER TABLE Relationships WITH GC_GRACE_SECONDS = 0;
CREATE INDEX IF NOT EXISTS  relationships_id ON Relationships(_id);
```

Script de création de la table RoundTable :

```
CREATE TABLE  RoundTable(_id map<TEXT,TEXT>, funding_rounds list<frozen< roundType>>,
PRIMARY KEY(_id));
ALTER TABLE RoundTable WITH GC_GRACE_SECONDS = 0;
CREATE INDEX IF NOT EXISTS  roundtable _id ON RoundTable (_id);
```

Simple Queries

1. Get all the companies that have a category_code=”nanotech” :

- SELECT * FROM companies WHERE category_code=”nanotech” ALLOW FILTERING;
- ✓ *Ici, on veut afficher les entreprises qui sont dans le domaine de la nanotechnologie Et comme « **category_code**» n'a pas été indexé, on doit ajouter le « **ALLOW FILTERING** ».*

2. Get all the Companies that founded in 2008 :

- SELECT * FROM companies WHERE founded_year =2008 ALLOW FILTERING;
- ✓ *Ici, on veut afficher les entreprises qui ont été fondés en 2008. Et comme « **founded_year** » n'a pas été indexé, on doit ajouter le « **ALLOW FILTERING** ».*

3. Get the number of Companies whith more than 100 employees :

- SELECT count(*) FROM companies WHERE number_of_employees >100 ALLOW FILTERING;
- ✓ *Ici, on veut afficher les entreprises qui ont plus de 100 employés. Puisque « **number_of_employees** » n'a pas été indexé, on doit ajouter l'option « **ALLOW FILTERING** ».*

Ou

- SELECT count(*) FROM Companies WHERE token(_id)>100;
- ✓ *Ici, on utilise le token(_id) afin de compter les entreprises qui ont plus de 100 salariés sans se préoccuper de l’attribut « **number_of_employees** » ce qui permet ainsi d'éviter l'utilisation du « **ALLOW FILTERING** »*

4. Get all companies founded in May 2019 :

- SELECT * FROM companies WHERE (founded_year =2019 AND founded_month=5) ALLOW FILTERING;

✓ *Ici nous cherchons toutes les entreprises qui ont été fondées en mai 2019. Et puisque, ni « **founded_year** », ni **« founded_month** » n'ont étaient indexées, on utilise le « **ALLOW FILTERING** ».*

5. Get all companies that starts with an "F" :

- SELECT * FROM companies WHERE name LIKE 'F%' ALLOW FILTERING;

✓ *Ici, on cherche à afficher toutes les entreprises qui commencent par la lettre F. Pour ce faire, on utilise la fonction **'LIKE'** qui nous permet de reconnaître un motif dans une chaîne de caractères et on lui donne comme argument **'F%'**. De plus, l'attribut « **name** » n'étant pas indexé, nous utilisant l'option « **ALLOW FILTERING** ».*

6. Get all comapnies that raised more than 10 million dollars :

- SELECT * FROM companies WHERE total_money_raised > 10000000 ALLOW FILTERING;

✓ *Ici, on cherche à afficher les entreprises qui ont levé plus de 10M de dollars. L'attribut « **total_money_raised** » n'étant pas indexé, on utilise l'option « **ALLOW FILTERING** ».*

1. Get the number of employees of each Company that have as category_code =”software” :

- SELECT _id,number_of_employees FROM Companies WHERE category_code=”software” GROUP BY_id;

✓ *Ici, on cherche à avoir le **nombre d'employés** travaillant au sein d'entreprises dans le domaine du **Software.***

2. Get the product names that have a permalink =”ripcode-v4” :

- CREATE OR REPLACE FUNCTION project(key text, tab map) RETURNS NULL ON NULL INPUT RETURNS INT LANGUAGE Java AS 'return tab.get(key);';
- SELECT project(‘name', products) FROM Product WHERE(select project(‘permalink ', products) =”ripcode-v4”) ALLOW FILTERING;

✓ *Ici, on veut afficher le nom des produits avec un « **permalink** » égal à "**ripcode-v4**". Pour ce faire, on commence par créer une fonction **project** qui nous permet d'afficher la **value de tab[‘key’]**. Et puis on exécute la requête en ajoutant l'option « **ALLOW FILTERING** ».*

Hard Queries

1. Give the product names that belong to the campanie “5a5c533c942d09e481c15829” :

- CREATE OR REPLACE FUNCTION project(key text, tab map) RETURNS NULL ON NULL INPUT RETURNS INT LANGUAGE Java AS 'return tab.get(key);';
- SELECT project(‘name', products) FROM Product WHERE(select project(‘$oid’, _id) =”5a5c533c942d09e481c15829”) ALLOW FILTERING;

✓ *Ici, on cherche à afficher le nom de tous les produits qui apparaissent au sein de l'entreprise ayant un « **_id** » égal à « **5a5c533c942d09e481c15829** ». Pour ce faire, on commence par créer une fonction « **project** » qui prend en argument une « **key** » et un « **tab** » et qui renvoie la « **value** » de tab à l’emplacement key (**tab["key"]**). Puis, exécute la requête en ajoutant l’option « **ALLOW FILTERIGN** ».*