|  |
| --- |
| rectangle transparent colorérectangle coloréUne image contenant texte  Description générée automatiquementrectangle transparent coloré  RAPPORT  TP Cassandra |

|  |
| --- |
| NoSQL - DIA2  MANNAI Hasna ()  CHENIK Yassine ()  BOUCHIBA Emine () |

Table des matières

Importation du ficher json dans le container Cassandra 2

Création des tables 3

Simple Queries 5

Complex Queries 7

Hard Queries 8

|  |
| --- |
| 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 ».*

1. 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 ».*

1. 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 »*

|  |
| --- |
| Simple Queries |

1. 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 ».*

1. 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 ».*

1. 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 ».*

|  |
| --- |
| Complex Queries |

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.*

1. 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 ».*