

1- Introduction

Dans ce chapitre nous voulons utiliser les processus de CRUD (Créates, Read, Update, Delete) à la fois sur le mongo dB, Cassandra SQL server 2008, savoir qui est le meilleur d'entre eux et la première chose que nous commençons par la définition des instruments utilisés dans ce test.

Cassandra est une base de données NO SQL orientée colonnes, destinée pour de grands volumes de données, hétérogènes, de structure et taille évolutives et hautement disponibles, sans compromettre la performance. Un très large panel d'entreprises, dont Twitter, Netflix et eBay, utilisent Cassandra, CQL(**Cassandra Query Language**) est l'interface par défaut dans le SGBD Cassandra. C'est un langage SQL-Like, pour la manipulation des données dans la base, télécharger la dernière version de Cassandra sur: <http://cassandra.apache.org/download/>.

Table plus : C'est un programme favoriser pour Cassandra, et nous l'utilisons dans ce chapitre pour afficher le temps d'exécution de chaque opération, télécharger la version sur : <https://tableplus.com/blog/2017/02/changelogs.html>

Mongo DB est un Système de Gestion de Base de Données tout comme MySQL, Oracle et bien d'autres, seulement, Mongo DB est un SGBD orienté documents et non relationnel. En conséquence, les données ne sont pas stockées dans des tables sous forme de ligne/tuples, mais elles sont stockées dans des collections. Ces collections vont contenir des documents JSON, Ce système est classé No SQL, ce qui veut dire que Mongo DB n'utilise pas de langage Structured Query Language (SQL).

Mongo DB Compass Community : C'est un programme favoriser pour : Mongo dB orientée document Et nous l'utilisons dans ce chapitre pour naviguer dans les tables.

Microsoft SQL Server 2008 est un système de gestion de base de données (SGBD) en langage SQL incorporant entre autres un SGBDR (SGBD relationnel) développé et commercialisé par la société Microsoft. Il fonctionne sous les OS Windows et Linux (depuis mars 2016).

2- Source De Données :

Après la création des tables de la base de données, nous allons utiliser un générateur de données pour remplir toutes les tables avec les données correspondantes.

Mockaroo <https://mockaroo.com/> offre un service gratuit pour générer des données qui peut être personnalisé pour émuler nos besoins de conception et les données des tables. Les données peuvent être exportées vers un certain nombre de formats mais dans notre cas, on va s'intéresser au format SQL.JSON.

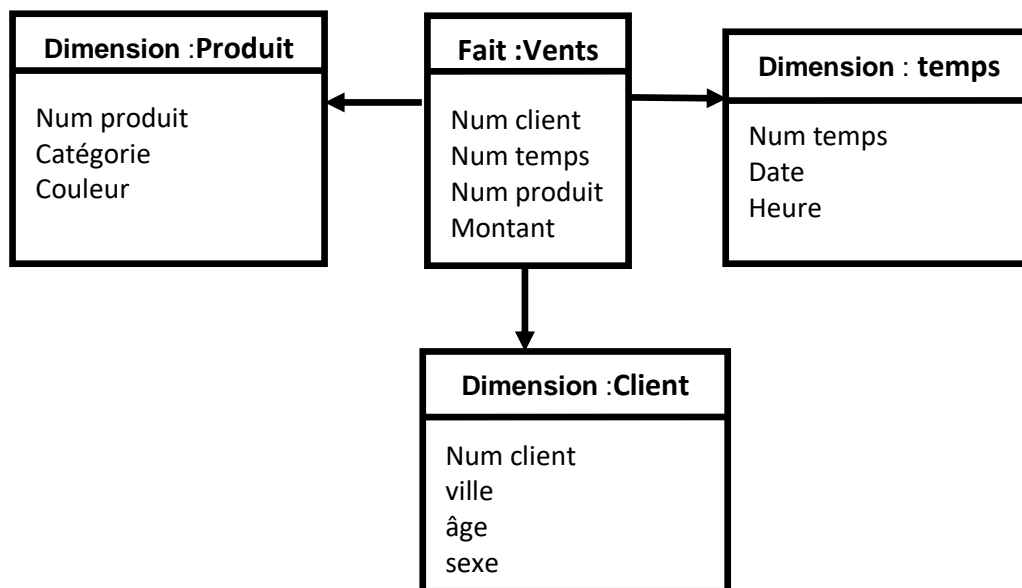


Figure 3.3 schéma en étoile

3- Protocol experimental

Nous utilisons Cassandra v3.11.4 et table plus v 2.4 pour tester le modèle orienté colonnes et Mongo DB Shell v4.0.6 et Mongo dB Compass Community v1.17.0 pour celui orienté documents. Et SQL server 2008 pour tester les modèles relationnelles.

Matériel et systèmes de gestion de données. Nous utilisons ce test PC d'un type (i3-acer, 4 Go de RAM, 500 G de disque. La base de données sur laquelle nous mènerons les expériences se compose de quatre tableaux (schéma en étoile comme Figure 3.3) un fait (vente) et trois table de dimensions (clients-produit-temps) chaque table il ya 1000 enregistrement .le format des tables csv pour mongo dB et Cassandra, Le processus complet est présenté en figure 4.1 Les données sont chargées à partir de fichiers dans Cassandra et Mongo DB et sql server en utilisant leurs instructions propres.

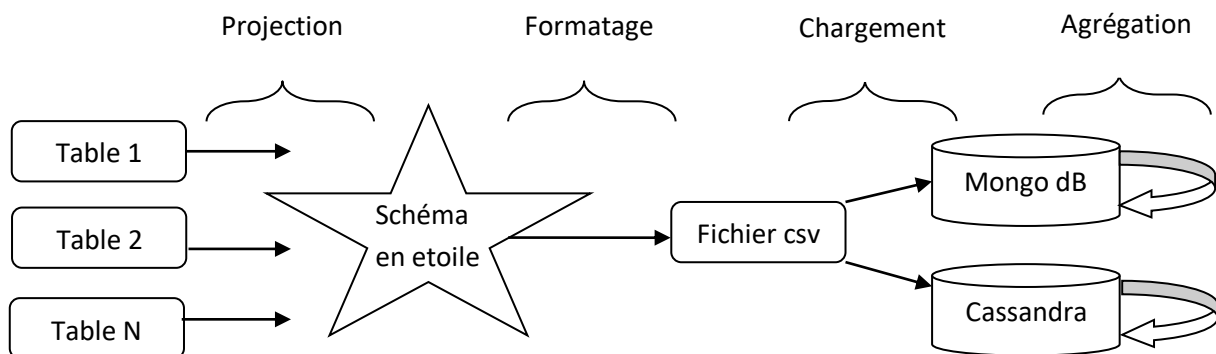


Figure 4.1 Processus du protocole expérimental

3.1 Les fonctions utilisées pour l'importation (Mongo dB):

-Mongo import -d Star -c vente --type csv --file C:\Users\info\Desktop\base\vente.csv –
Header line

Mongo import -d Star -c produit --type csv --file C:\Users\info\Desktop\base\produit.csv –
Header line

Mongo import -d Star -c clients --type csv --file C:\Users\info\Desktop\base\ clients.csv –
Header line

Mongo import -d Star -c temps --type csv --file C:\Users\info\Desktop\base\temps.csv –
Header line

3.2 Les fonctions utilisées pour l'importation (Cassandra)

COPY vente FROM '/Users/info/Desktop/Cassandra/vente.csv';

COPY temps FROM '/Users/info/Desktop/Cassandra/temps.csv';

COPY produit FROM '/Users/info/Desktop/Cassandra/produit.csv';

COPY clients FROM '/Users/info/Desktop/Cassandra/clients.csv';

4- Résultats avec les temps d'exécution des différentes requêtes :

Requête	temps d'exécution	Opération	OBS
insert into produit(idproduit, categorie, couleur) VALUES(1010, 'symbol','khaled');	272 ms	Créâtes	Cassandra
Select * from produit where idproduit = 53 ;	8 ms	Read	
UPDATE produit SET categorie='Delhi',couleur='50000' WHERE idproduit=23;	6 ms	Update	
DELETE FROM star.produit WHERE idproduit = 1010 ;	4 ms	Delete	
DBCcollection temps = db.getCollection("vente"); BasicDBObject query = new BasicDBObject(); query.put("montant", 120.85);	1 seconde	Read	mongodb
DBCcollection vente = db.getCollection("vente"); BasicDBObject doc1 = new BasicDBObject(); doc1.put("num produit",1002); doc1.put("num temps",784); doc1.put("num clients",782); doc1.put("montant", 120.85); vente.insert(doc1);	7 secondes	Create	
DBCcollection vente = db.getCollection("vente"); BasicDBObject findTestItemQuery = new BasicDBObject(); findTestItemQuery.put("num produit", 1002); DBCursor testItemsCursor = vente.find(findTestItemQuery); if(testItemsCursor.hasNext()) { DBObject testCodeItem = testItemsCursor.next(); testCodeItem.put("num produit",1003);	1 seconde	Update	

Expérimentation

vente.save(testCodeItem);			TDS
<pre> DBCcollection vente = db.getCollection("vente"); BasicDBObject deleteQuery = new BasicDBObject(); deleteQuery.put("num produit", 1002); DBCursor cursor = vente.find(deleteQuery); while (cursor.hasNext()) { DBObject item = cursor.next(); vente.remove(item); </pre>	1 seconde	Delete	
INSERT INTO produit ([num produit], catégorie, couleur)VALUES (1001, 'khaled', 'amani') ;	30 m/s	Create	
UPDATE produit SET [num produit] = 1002, catégorie = 'nawal', couleur='youcef'WHERE [num produit]=1001;	8 m/s	Update	
<pre> SELECT [num produit] ,[num temps] ,[num client] ,[montant] FROM [etoile].[dbo].[vente\$] WHERE [num produit]=126 </pre>	7 m/s	Read	
DELETE FROM vente WHERE [num produit]=126', [num temps]=84, [num clients]=826 ,mantant=458.42;	16 m/s	Delete	

Tab.4.1 les temps d'exécution des différentes requêtes

Nom opération	SQL serveur	Mongo dB	Cassandra
Créates time	30 m/s	7 second	272 m/s
Read time	7 m/s	1 second	8 m/s
update time	8 m/s	1 second	6 m/s
Delete time	16 m/s	1 second	4 m/s

Tab.4.2 Tableau comparatif des opérations CRUD sur les différents SGBD

5- Conclusion

nous avons conduit un ensemble d'expérimentations pour étudier le processus de chargement (chargement du schéma en étoile et calcul CRUD opération) et d'interrogation. Nous avons utilisé Cassandra comme solution No SQL orientée colonnes et Mongo DB comme solution orientée documents. les données (tables mille ligne) , Les premiers résultats montrent des performances meilleures pour SQL server Les bases de données sont petites et c'est ce qui convient SQL server et par contre les bases de données sont grandes, c'est ce qui convient l'approche verticale orientée colonne et 'approche orientée documents.