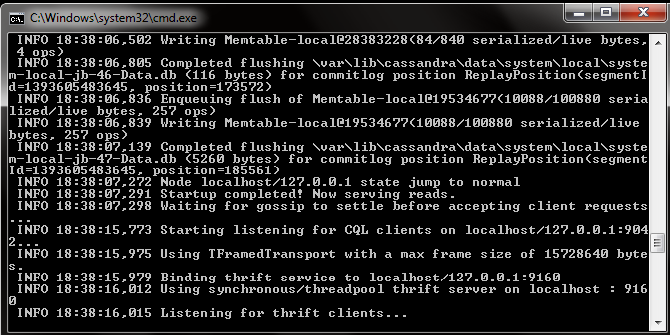
**Tutorial**

Au cours de ce tutorial, il faut signaler que l’installation se fait sur Windows. Cassandra est disponible sur Linux aussi. Et cela dépend de la version du système.

**Installation**:

1. Télécharger Apache- Cassandra du lien suivant : <http://cassandra.apache.org/download/>
2. Télécharger Java SE Development Kit 7 approprie à cette version de cassandradu lien suivant : <http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html>
3. Extraire les fichiers du ʺCassandra ʺ  téléchargé et les mettre dans le répertoire approprié.

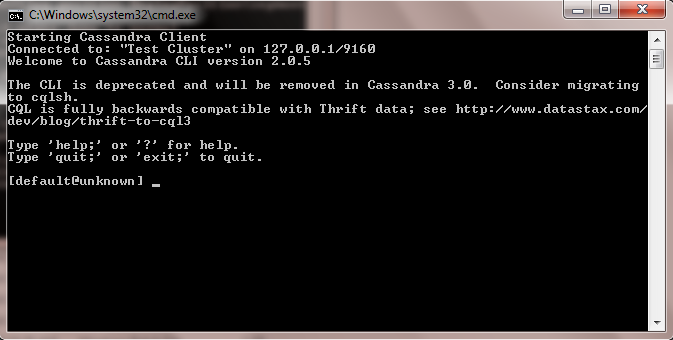
**Démarrage**:

Pour démarrer Cassandra il suffit d’exécuter cassandra.bat qui se trouve dans le fichier bin.   
Il faut l’exécuter en mode administratif.

Le message ʺListening for thrift clientsʺ indique que Cassandra est démarrée avec succès.

**Interface de connection**

Cassandra représente une interface pour se connecter et faire des commandes sur la base de données. Il suffit d’exécuter Cassandra-cli dans le fichier bin.



Pour créer un Keyspace il suffit de taper la commande suivante :   
CREATE KEYSPACE nomkeyspace  
with placement\_strategey = 'org.apache.cassandra.locator.SimpleStrategy' and strategy\_options= {replication\_factor:2};

Les datas se sauvegardent dans le fichier var/lib/Cassandra/data.

**Application Java pour tester les méthodes CRUD**

Dans cette partie on exécute une application java pour tester les méthodes Crud en connectant à la base de données Cassandra.

Ce qui nous nécessite :

1. Eclipse IDE
2. Apache Cassandra version 1.2.15
3. Jdbc Driver 1.2.5 installé du lien suivant : https://code.google.com/a/apache-extras.org/p/cassandra-jdbc/downloads/list
4. Jdk 1.7

Création de l’application :

1. Créer un nouveau projet de type java application du menu eclipse.
2. Ajouter les librairies qui se trouvent dans le fichier apache Cassandra/lib suivantes en appuyant sur config build path du projet :   
   apache-Cassandra-thrift-1.1.6.jar  
    apache-Cassandra-clientutil-1.1.6.jar  
   libthrift-0.7.0.jar   
   apache-Cassandra-1.1.6.jar  
   guava.jar  
    log4j and sl4j-api-1.6.1.jar   
   sl4j-log4j12-1.6.1.jar  
   En plus la librairie jdbc : Cassandra-jdbc-1.1.2.jar
3. Créer une classe qui contient les méthodes suivantes :

a. createColumnFamily (): creer un column family

b. dropColumnFamily (): effacer un column family

c. insertData (): Ajouter des informations au column family spécifié

d. deleteData (): Effacer les informations d’un column family spécifié

e. updateData (): Mettre à jour des informations sur un column family spécifié

f. listData (): Montrer les informations qui se trouve dans un column family spécifié

Le code suivant présente la classe entière :

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

public class CassandraCql {

private static java.sql.Connection con = null;

public static void main(String[] a) {

try {

Faire une connection à Cassandra ici

Class.forName("org.apache.cassandra.cql.jdbc.CassandraDriver");

con = DriverManager.getConnection("jdbc:cassandra://localhost:9160/testkeyspace");

/\* -- Functions to perform on Keyspace -- \*/

createColumnFamily();

Appel des méthodes ici

pouplateData();

deleteData();

updateData();

listData();

dropColumnFamily("news");

} catch (ClassNotFoundException e) {

e.printStackTrace();

} catch (SQLException e) {

e.printStackTrace();

}

}

public static void createColumnFamily() throws SQLException {

String data = "CREATE columnfamily news (key int primary key, category text , linkcounts int ,url text)";

Statement st = con.createStatement();

st.execute(data);

}

public static void dropColumnFamily(String name) throws SQLException {

String data = "drop columnfamily " + name + ";";

Statement st = con.createStatement();

st.execute(data);

}

public static void pouplateData() throws SQLException {

String data = "BEGIN BATCH \n" + "insert into news (key, category, linkcounts,url) values (1,'class',71,'news.com') \n"

+ "insert into news (key, category, linkcounts,url) values (2,'education',15,'tech.com') \n"

+ "insert into news (key, category, linkcounts,url) values (3,'technology',415,'ba.com') \n"

+ "insert into news (key, category, linkcounts,url) values (4,'travelling',45,'google.com/teravel') \n" + "APPLY BATCH;";

Statement st = con.createStatement();

st.executeUpdate(data);

}

public static void deleteData() throws SQLException {

String data = "BEGIN BATCH \n" + "delete from news where key='user5' \n" + "delete category from news where key='user2' \n"

+ "APPLY BATCH;";

Statement st = con.createStatement();

st.executeUpdate(data);

}

public static void updateData() throws SQLException {

String t = "update news set category='sports', linkcounts=1 where key='user5'";

Statement st = con.createStatement();

st.executeUpdate(t);

}

public static void listData() throws SQLException {

String t = "SELECT \* FROM news";

Statement st = con.createStatement();

ResultSet rs = st.executeQuery(t);

System.out.println("rows nb: " + rs.getRow());

while (rs.next()) {

System.out.println(rs.getString("category") + "\n");

for (int j = 1; j < rs.getMetaData().getColumnCount() + 1; j++) {

System.out.println(rs.getMetaData().getColumnName(j) + " : " + rs.getString(rs.getMetaData().getColumnName(j)));

}

}

}

}

**Problèmes et difficultés**

Cassandra présente une base de donnée NoSql nouvelle pour cela c’est une technique n’est pas mature suffisamment et elle n’est pas stable. Pour cela j’ai confronté quelques problèmes et difficultés.

Premièrement, la nouvelle version apache Cassandra 2.0.0 ne possède aucun driver jdbc pour connecter du java pour cette raison j’étais obliger d’utiliser une ancienne version.

Deuxièmement, j’ai fait une recherche pour essayer de trouver une interface qui facilite le traitement des données dans la base de donnée sans utiliser les commandes sur cassandra-cli mais ils sont faibles.