Université Hassan 1er

Ecole Nationale des Sciences Appliquées de Berrechid

Département de mathématique et informatique

Filière : Ingénierie des Systèmes d'Information et BIG DATA

Module : Bases de données NoSQL et BIG DATA

Semestre : S8

Compte rendu TP : Flink

Réalisé par : EL ANSSARI Yassine et EZ-ZAHAR Zakaria

Année universitaire : 2021/2022

Application 1 : Word Count

Plate-forme

- Système d'exploitation : Ubuntu (ou n'importe quelle version de Linux)
- Java 7.x ou supérieure
- Éclipse – Dernière version

Installation Java :

- Update the source list

  $ sudo apt-get update

- Install Java

  $ sudo apt-get install oracle-java7-installer

- Verify Java Installation

  $ java -version

Installation Flink :

- Vous pouvez télécharger Flink à partir du site Web officiel d'Apache: Cliquez ici
- Décompressez le fichier d'installation:

  $ tar -xzvf flink- 1 . 1.3 -bin-hadoop26-scala_2. 11 . tgz

- Renommer le répertoire d'installation:

  $ mv flink- 1 . 1.3/ flink

- Changez le répertoire de travail en Flink Home:

  $ cd flink

- Démarrer le flink:

  ./start-cluster.sh

- Vérifier l'état des services en cours d'exécution**:**

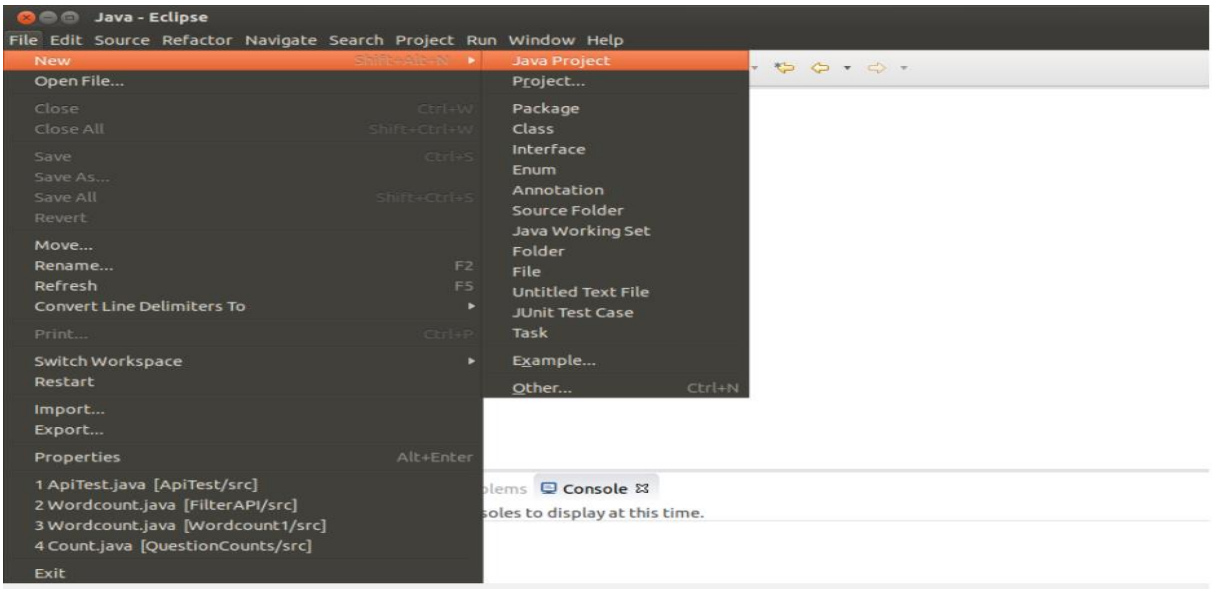
$ jps

- Interface utilisateur Web Apache Flink:

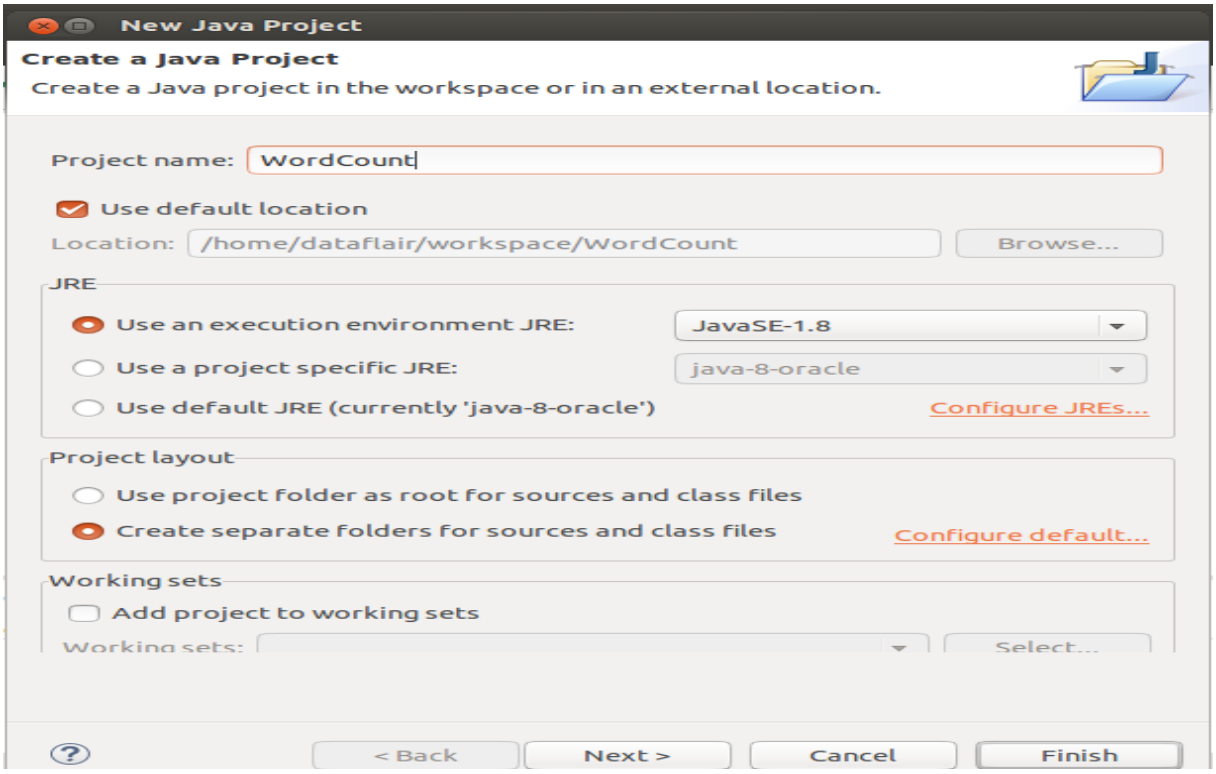
localhost:8081

Les étapes pour projet Word count :

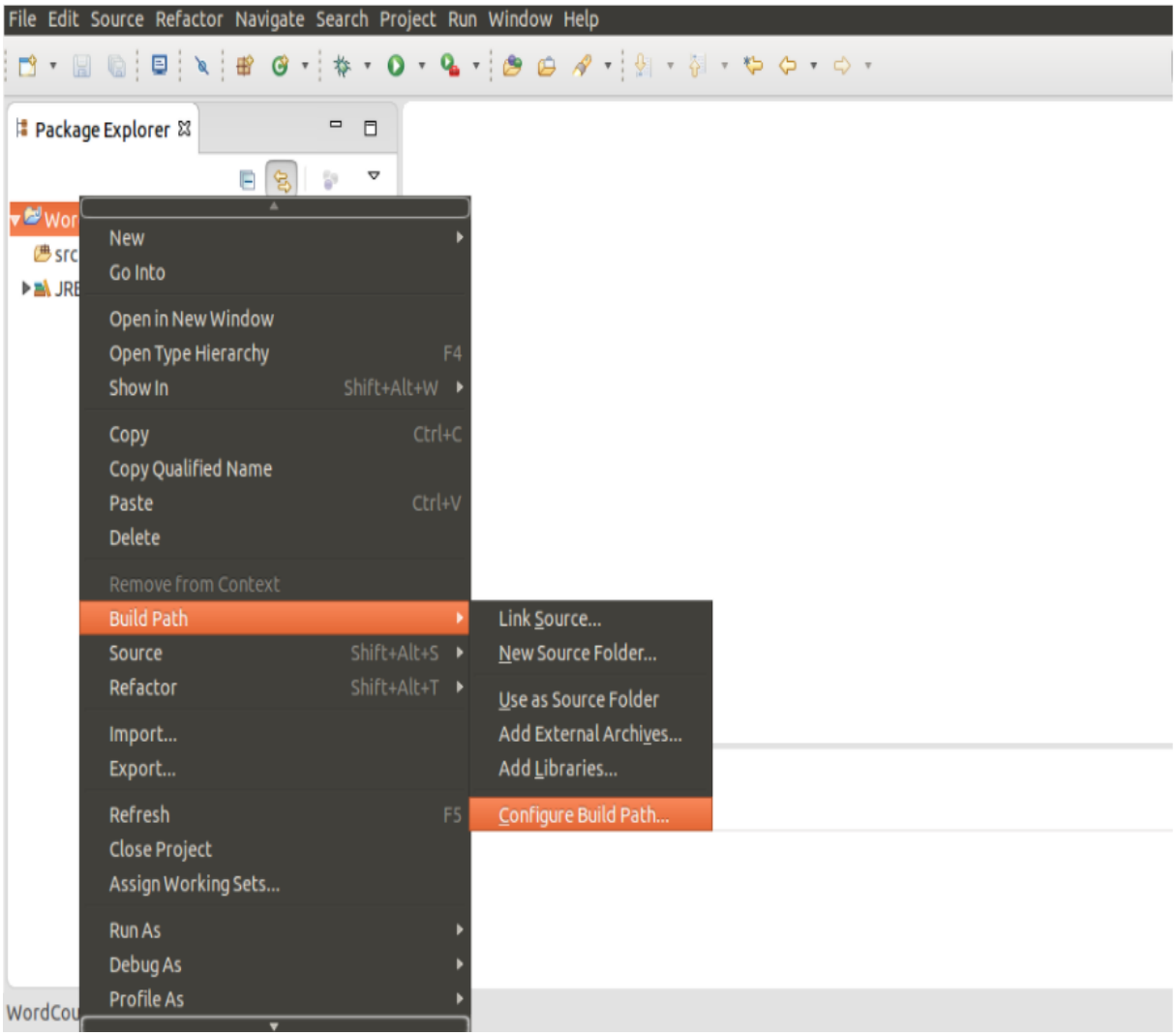
Etape 1 : Créer un nouveau projet Java



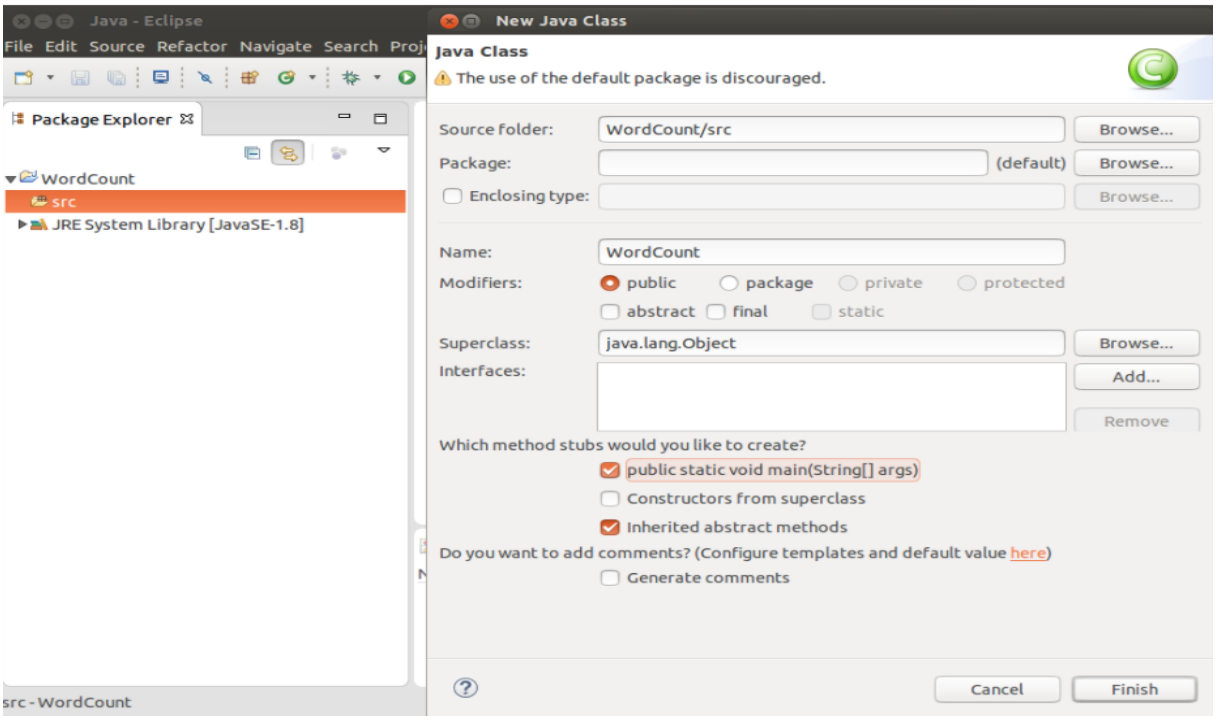
- Vous pouvez nommer le projet comme WordCount ou avec un nom de votre choix.



- Ajoutez les fichiers JAR dans Build Path. Vous pouvez trouver les fichiers jar dans le répertoire lib de la page Flink home.
- Pour faire ça, cliquez avec le bouton droit sur le projet et sélectionnez l'option Configure build path à partir du build path



Etape 2 : Faire une classe WordCount

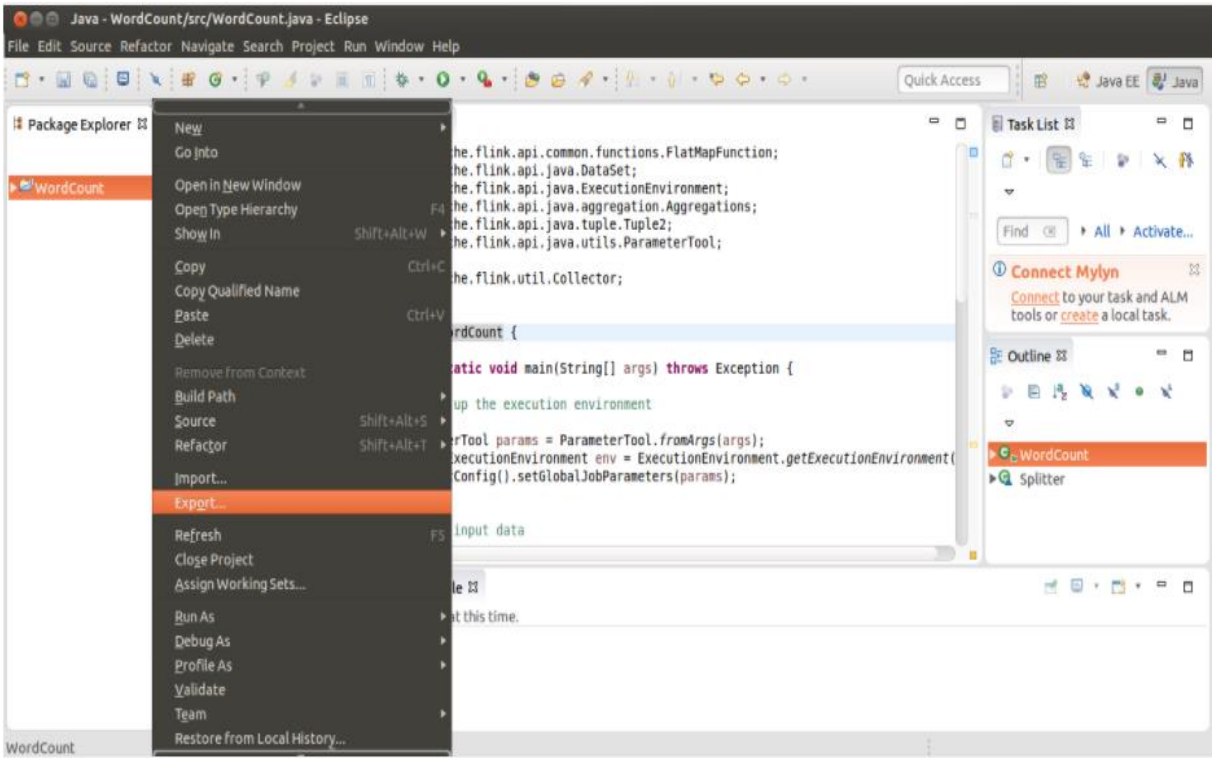


Copiez ci-dessous le code Apache Flink Wordcount dans l'éditeur

```
import org.apache.flink.api.common.functions.FlatMapFunction;
import org.apache.flink.api.java.DataSet;
import org.apache.flink.api.java.ExecutionEnvironment;
import org.apache.flink.api.java.aggregation.Aggregations;
import org.apache.flink.api.java.tuple.Tuple2;
import org.apache.flink.api.java.utils.ParameterTool;
import org.apache.flink.util.Collector;
public class WordCount {
public static void main(String[] args) throws Exception {
// set up the execution environment
final ParameterTool params = ParameterTool.fromArgs(args);
final ExecutionEnvironment env =
ExecutionEnvironment.getExecutionEnvironment();
env.getConfig().setGlobalJobParameters(params);
// get input data
DataSet<String> text = env.readTextFile(params.get("input"));
DataSet<Tuple2<String, Integer>> counts =
// split up the lines in pairs (2-tuples) containing: (word,1)
text.flatMap(new Splitter())
// group by the tuple field "0" and sum up tuple field "1"
.groupBy(0)
```

```
.aggregate(Aggregations.SUM, 1);
// emit result
counts.writeAsText(params.get("output"));
// execute program
env.execute("WordCount Example");
}
}
//The operations are defined by specialized classes, here the Splitter class.
class Splitter implements FlatMapFunction<String, Tuple2<String, Integer>> {
@Override
public void flatMap(String value, Collector<Tuple2<String, Integer>> out) {
// normalize and split the line into words
String[] tokens = value.split("\\W+");
// emit the pairs
for (String token : tokens) {
if (token.length() > 0) {
out.collect(new Tuple2<String, Integer>(token, 1));
}
}
}
}
```

- Avant d'exécuter création Apache Flink l'application de word count, nous devons créer un fichier jar. Clicque droit sur projet >> export



1.1.1.1

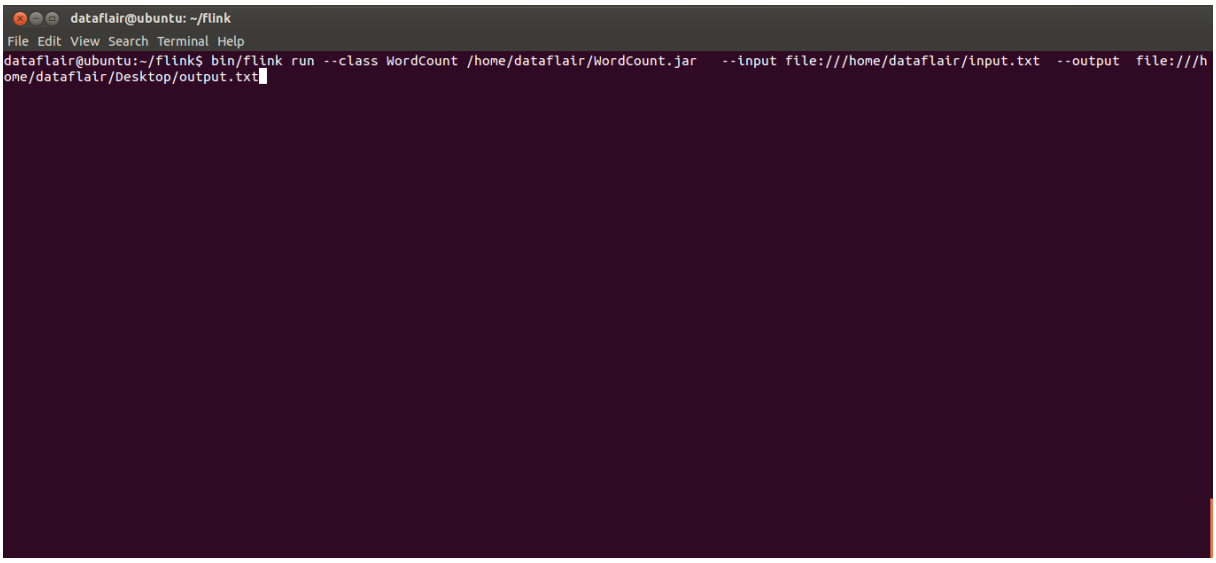
1.1.1.2 Accédez au répertoire d'accueil d'Apache Flink

Démarrer les services Apache Flink en utilisant les commandes suivantes

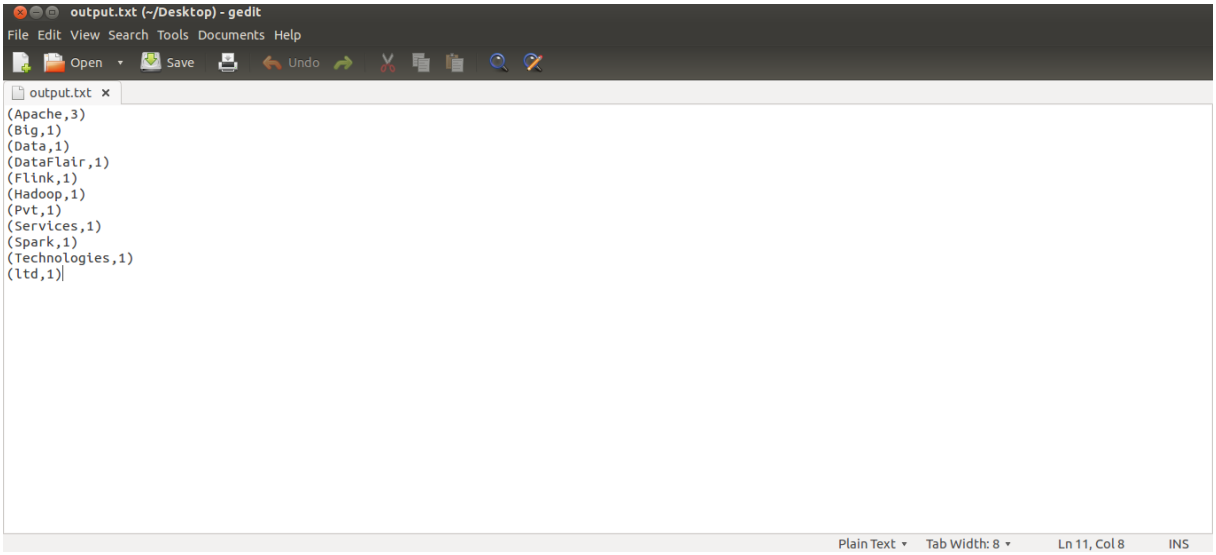
- Cd flink
- bin/start-cluster. sh

1.1.1.3 Soumettre l'application Apache Flink

Utilisez la commande suivante pour soumettre l'application Apache Flink

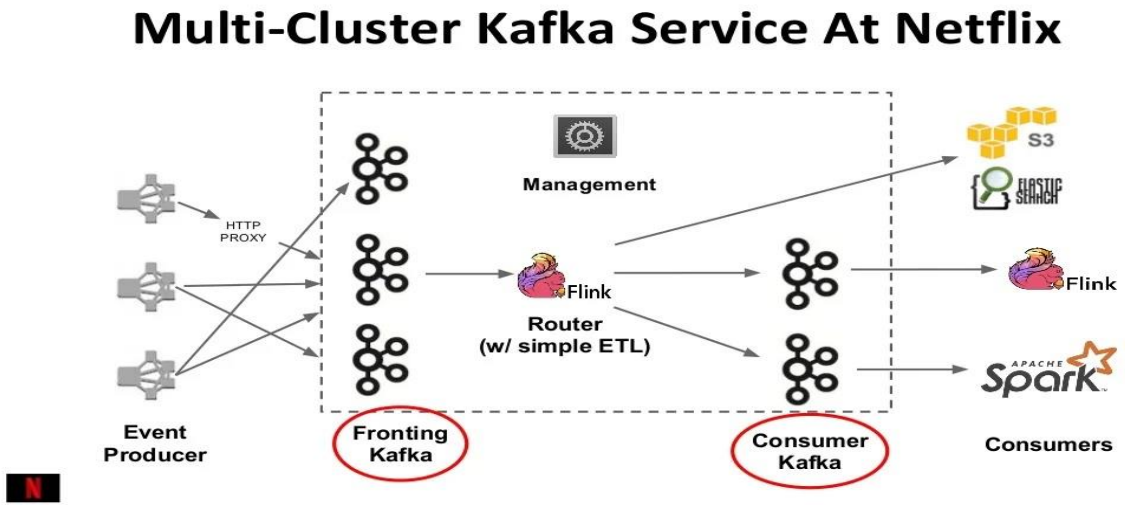

dataflair@ubuntu: ~/flink
File Edit View Search Terminal Help
dataflair@ubuntu:~/flink$ bin/flink run --class WordCount /home/dataflair/WordCount.jar --input file:///home/dataflair/input.txt --output file:///home/dataflair/Desktop/output.txt

La Sortie :


output.txt (~/Desktop) - gedit
File Edit View Search Tools Documents Help
Open
Save
Undo
output.txt
(Apache,3)
(Big,1)
(Data,1)
(DataFlair,1)
(Flink,1)
(Hadoop,1)
(Pvt,1)
(Services,1)
(Spark,1)
(Technologies,1)
(ltd,1)
Plain Text
Tab Width: 8
Ln 11, Col 8
INS

Application 2 : cas réel intégration d'apache flink avec kafka pour Netflix

Le but de ce Tp est de savoir comment netflix utilise la combinaison de cluster kafka avec apache flink pour le streaming et le traitement des données en temps réel à travers la réalisation de schéma en dessous.



- Netflix utilise plusieurs clusters Kafka pour générer les données en streaming qui vient de différentes sources, ces données vont être transformées à l'aide de moteur de traitement distribuée Apache Flink puis des consommateurs vont consommer ces données .

- En résumé on a 3 parties :

  Partie 1 : Génération des données de différentes sources par les clusters kafka

  Partie 2 : Flink va faire un faire un ETL : l'extraction des données qui sont

dans kafka ensuit la transformation de ces données et en fin le chargement dans kafka .

Partie 3 : la consommation et l'utilisation de ces données par d'autre moteurs de ces données.

- On va faire la réalisilation de ce schéma à travers 3 classes java :

  - ✓ Flink_Kafka_Receiver.java : dans ce class flink va jouer le rôle d'un consommateur des données qui vient de kafka .
  - ✓ Flink_Kafka _Sender.java : dans ce class flink va jouer le rôle d'un producteur des données vers kafka .
  - ✓ Flink_Kafka_Receiver_Sender.java : cette classe représente la réalisation de schéma générale uppercase comme transformation .

- J'ai utilisé 2 topics pour ce tp :

Le premier topic c'est pour les données qui viennent de différentes sources et le deuxième pour les données envoyées par flink après la transformation.

- Source de données : Producteur par défaut de kafka .

- Comme type de projet : Maven afin de travailler avec les Api flink et kafka et des jars pour faire la connexion entre flink et kafka .

Je vais mettre le code complet et les commandes dans le dépôt gitHub voilà le lien :

https://github.com/yassineelanssari/Integration-Apache-Flink-with-Kafka-Netflix-Case-