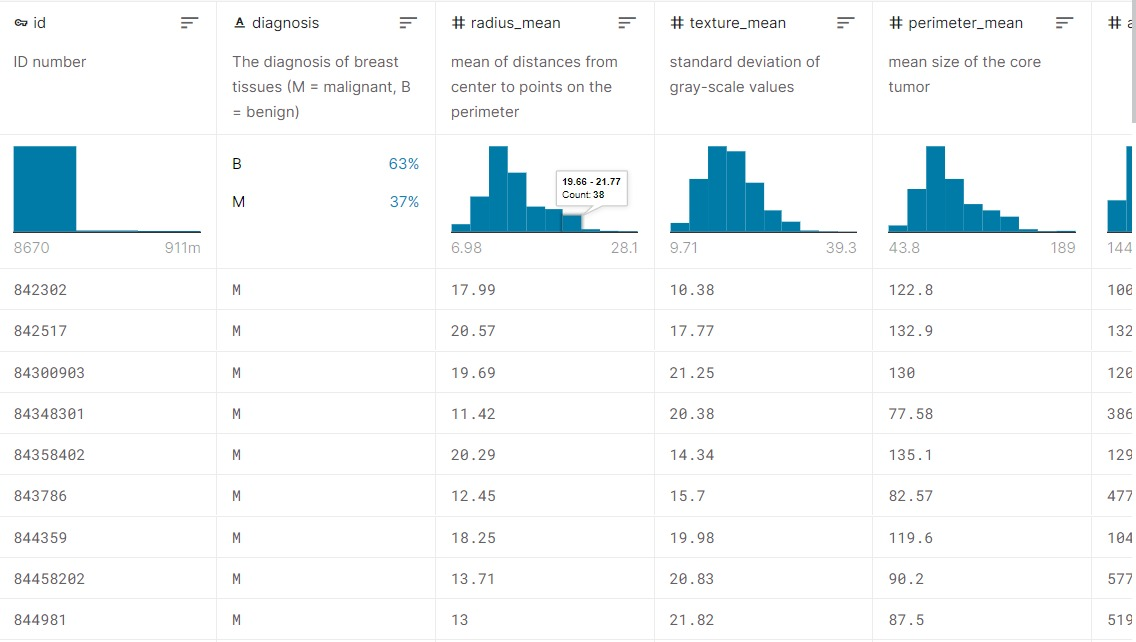
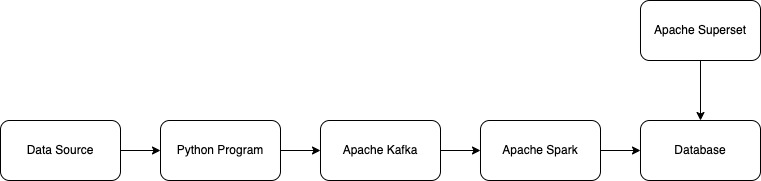
**Projet Data Mining**

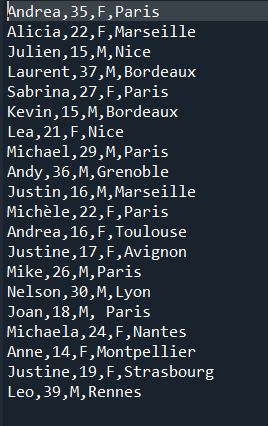
Le projet que nous devons réaliser se base sur un fichier csv récupéré sur Kaggle contenant des données de patients atteint de cancer bénin et malin. Notre but est de faire en sorte que le médecin puisse avoir de manière rapide accès aux informations qui l’intéresse sans avoir à parcourir tout le fichier.



Le projet que nous devons réaliser consiste a mettre en œuvre la pipeline de traitement de data suivante :



Ainsi, pour tester le fonctionnement de notre pipeline, nous avons choisi comme source de données un fichier csv test contenant des données utilisateurs comme le nom, l’âge, la date de naissance ou encore la ville d’un utilisateur.



Nous ouvrirons ce fichier à travers un programme Python « Producer » qui nous permettra de les envoyer dans un bus Kafka.

Ensuite, dans un programme Spark, nous nous connecterons à Kafka afin de récupérer les données reçues dans le bus Kafka, ligne par ligne. Nous effectuerons ensuite plusieurs transformations sur les données et enfin, ces données modifiées finaux seront ensuite sauvegardées dans une table d’ une base de données MySql puis affichées à l’aide de l’outil de visualisation Superset.

Afin de réaliser ce projet, nous avons suivi les étapes suivantes :

-Nous avons commencer par installer et initialiser trois machines virtuelles à l’aide de vagrant en utilisant les fichiers d’installation donnés et en utilisant la commande « vagrant up » :

* 1ère machine : Spark, on attribue l’ adresse IP : 192.168.33.12
* 2ème machine : Kafka, on attribue l’adresse IP : 192.168.33.13. Pour démarrer Kafka, on lance les deux commandes suivantes dans deux sessions de terminaux différents :
  + $ bin/zookeeper-server-start.sh config/zookeeper.properties
  + $ bin/kafka-server-start.sh config/server.properties

On y crée notre sujet « projet\_datamining » avec la commande suivante :

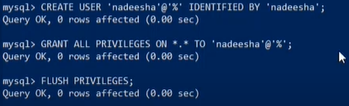
* + $ bin/kafka-topics.sh --create --topic projet\_datamining --bootstrap-server 192.168.33.13:9092

Pour tester le fonctionnement directement dans un terminal, nous pouvons écrire dans le topic en tant que producer en lançant cette commande :

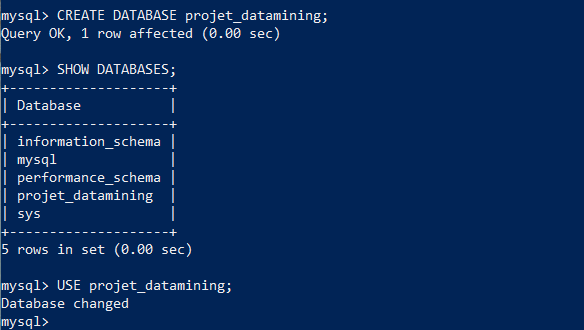
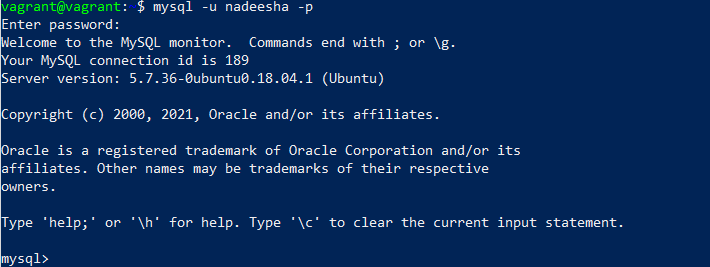
* + $ bin/kafka-console-producer.sh --topic projet\_datamining --bootstrap-server 192.168.33.13:9092

Et pour lire les données en tant que consumer, on lance cette commande :

* + $ bin/kafka-console-consumer.sh --topic projet\_datamining --from-beginning --bootstrap-server 192.168.33.13:9092
* 3ème machine : on y installe MySql en suivant les étapes de ce lien : https://www.digitalocean.com/community/tutorials/how-to-install-mysql-on-ubuntu-20-04-fr et on attribue l’adresse IP : 192.168.33.14. On configure les préférences de sécurité puis nous créons l’utilisateur « nadeesha » et on donne à l’utilisateur le droit de tout faire sur toutes les tables :



Puis on se connecte en tant que nadeesha en utilisant la commande : « mysql -u nadeesha -p » avec comme mot de passe :  « nadeesha » :

On crée maintenant la base de données « projet\_datamining » : 

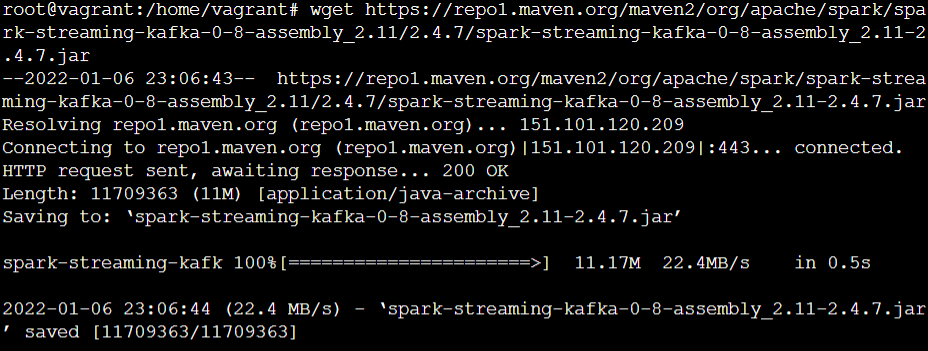
Nous installerons également dans cette même machine virtuelle Apache Superset.

* Lien entre les programmes
* Nous avons ensuite lier Spark et Kafka. Pour cela, on suit les commandes suivantes dans la machine virtuelle Spark:

On désactive la variable d’environnement PYSPARK\_DRIVER\_PYTHON :

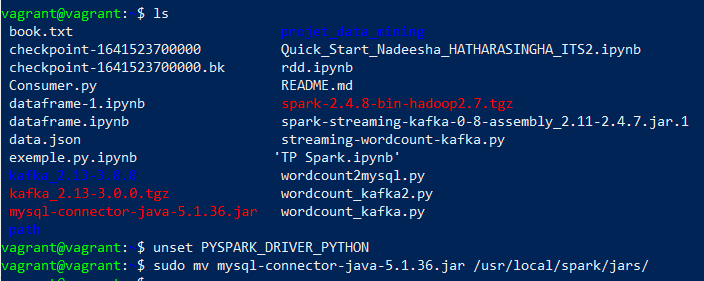


On télécharge la librairie correspondante contenue dans le jar suivant :



Puis on le déplace dans le dossier local où se trouve les dépendances de spark :

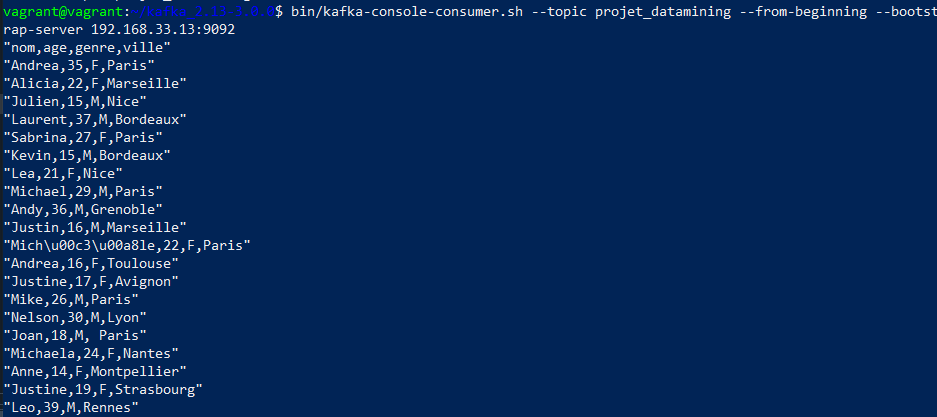


* On suit les mêmes étapes pour la librairie mysql-connector-java-5.1.36.jar permettant de lier mysql et spark.
* 
* Création du programme ‘Producer’ Python :

Nous avons ensuite créer un programme Python « Producer » qui lit un fichier csv ligne par ligne et l’envoie dans le bus Kafka toutes les 10 secondes. Pour cela, nous nous sommes connecter à Kafka en indiquant l’adresse IP de la VM correspondante ainsi que le port et nous avons utiliser le code suivant :



On vérifie avec la commande Consumer pour lire les données citée ci-dessus :



* Création du programme ‘Consumer’ Spark :

Nous avons ensuite créer le programme Consumer Spark.

Ce programme consiste à se connecter à Kafka pour récupérer les données du fichier source csv qui ont été envoyés dans le bus Kafka puis à appliquer des transformations afin de trier et organiser les données reçues et finalement les transmettre dans notre base de données MySqlafin de minimiser les taches du médecin.

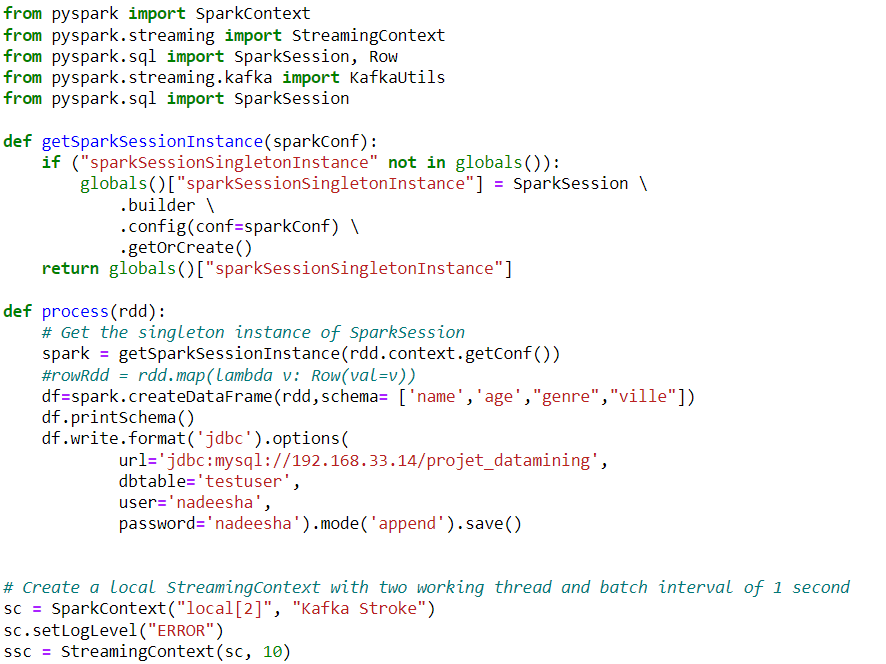
Voici quelques exemples des transformations que nous allons effectuer :

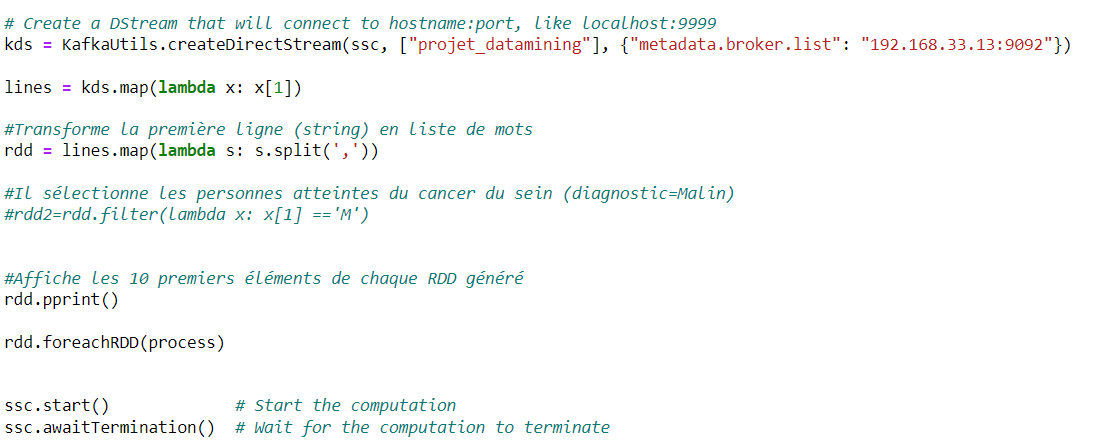
* ***Ligne.split(‘,’)*** : Pour transformer les chaines de caractères en liste
* ***Filter*** : Pour afficher seulement les personnes dont le diagnostic est Malin

Nous afficherons finalement la moyenne de chaque donnée du diagnostic pour toutes les personnes atteintes d’un cancer du sein. Le médecin pourra ensuite se servir de cette moyenne pour déterminer le diagnostic de ses nouveaux patients.

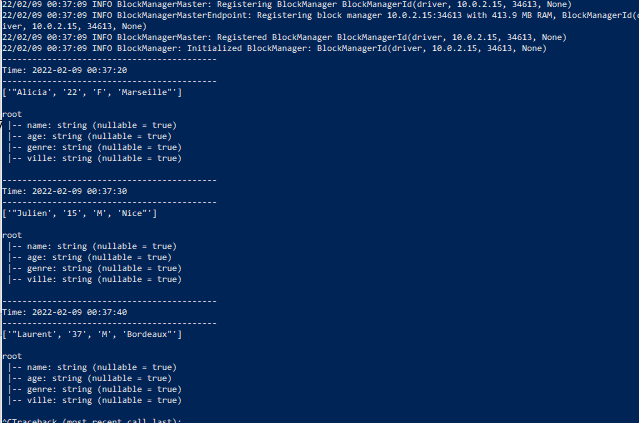
Pour cela, nous avons d’abord indiqué l’adresse IP de la machine virtuelle Kafka correspondante(192.168.33.13), ainsi que notre nom de topic « projet\_datamining ». Pour chaque ligne de mots envoyé qui sont donc séparés par des virgules car ils proviennent d’un fichier.csv, on transforme la chaine de caractères en liste de mots avec la commande : ligne.split(’,’) .

Enfin, on sauvegarde les données dans une nouvelle table « testuser » dans la base de données MySql « projet\_datamining » crée précédemment à l’aide du code suivant :

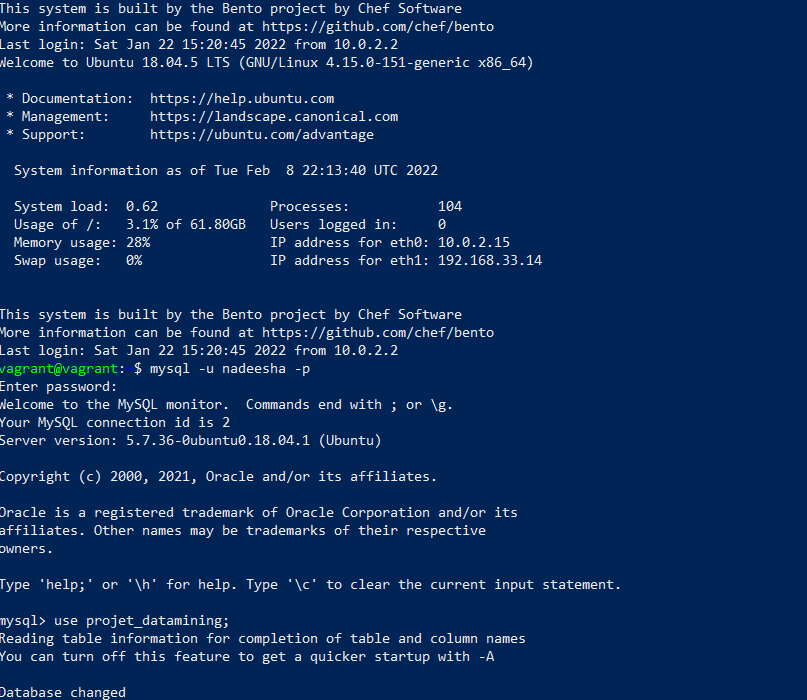


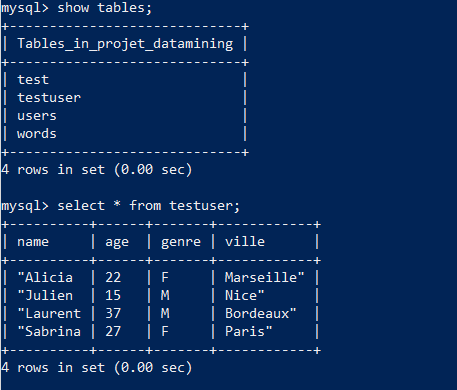


On teste ensuite dans Spark avec la commande suivante dans un terminal : /usr/local/spark/bin/spark-submit Consumer.py et on a le résultat suivant :



On vérifie maintenant que les entrées se sont bien ajoutés dans notre table « testuser » :





On voit ainsi que les données se sont bien ajoutés en temps réel dans la base de données « projet\_datamining » dans notre table « testuser ».

On teste maintenant l’affichage dans Apache Superset :