TP : Spark

L'objectif de ce TP est de manipuler des données avec Spark

Exercice 1 : Lancer un job spark

1. Lancer un service jupyter-pyspark sur Onyxia et créer une SparkSession nommé `spark` en utilisant la configuration `local`. Récupérer le sparkContext ensuite dans une variable `sc`.

```
spark = (SparkSession
         .builder
         .appName("TP")
         .master("local[5]")
         .getOrCreate()
        )
sc = spark.sparkContext
```

> La valeur 5 dénote que Spark utilisera 5 threads pour la parallélisation

2. Lancer ce script qui calcule la valeur de Pi avec la méthode de Monte Carlo. Consulter le job spark via la *spark-ui* (le lien est disponible dans la note du service Onyxia)

```
import random

NUM_SAMPLES = 10000000

def inside(p):
    x, y = random.random(), random.random()
    return x*x + y*y < 1

count = sc.parallelize(range(0, NUM_SAMPLES)) \
             .filter(inside).count()
print ("Pi is roughly %f" % (4.0 * count / NUM_SAMPLES))
```

3. Eteindre la session spark avec `spark.stop()`

Exercice 2 : Explorer des données avec Spark

1. Créer une nouvelle SparkSession sans préciser le ressource manager (`kubernetes` est par défaut dans Onyxia). Lire ensuite fichier `s3a://projet-spark-lab/diffusion/formation/data/sirene.parquet`

2. Afficher le schéma du fichier avec la méthode `printSchema`.

3. En utilisant la syntaxe DataFrame :

- Calculer le nombre total d'entreprises dans le jeu de données (nombre de siret unique)
- Afficher les 5 premiers établissements dont les `activitePrincipaleEtablissement` commencent par `47`

Exercice 3 : Interconnecter Spark

1. Importer les données du fichier parquet sirene précédent avec la méthode `DataFrame.write.jdbc` dans une table postgresql.

> Un exemple est disponible ici

> Il faudra télécharger le driver pour accéder à la base de données et l'ajouter à la SparkSession : https://jdbc.postgresql.org/download/

2. Lire ensuite les données depuis la base PostgreSQL avec Spark et récupérer la première ligne

3. Compter le nombre de lignes dans la table en partionnant en 20 les données sur la colonne `siren` en spécifiant comme valeur minimal `100000000` et maximale `999999999`

Exercice 4 : Mise en place d'un streaming

1. Utiliser Spark Streaming pour afficher le topic Kafka `sirene` et le sauvegarder dans une variable `kafka_df`

> Il faudra créer le topic `sirene` et y ajouter des messages pour les afficher dans spark streaming

> Il faut ajouter la config Spark suivante pour télécharger les drivers pour accéder à Kafka

```
.config("spark.jars.packages", "org.apache.spark:spark-sql-kafka-0-
10_2.12:3.5.3")
```

> La lecture du stream dans le terminal se fait avec la fonction suivante :

```
query = kafka_df.writeStream \
    .queryName("kafka") \
    .format("console") \
    .start()
```

2. Transformer la valeur et la clé du message en chaine de caractère en ne gardant que ces deux colonnes, puis relancer le stream.