

Prérequis pour les utilisateurs de Windows uniquement

Si vous utilisez une distribution Linux ou Mac OS passez directement à la section LAB

Afin de pouvoir utiliser correctement Spark avec le système de fichier Windows il est nécessaire de disposer du binaire winutils.exe. Ce binaire permet d'écrire des données depuis Spark le système de fichier Windows.

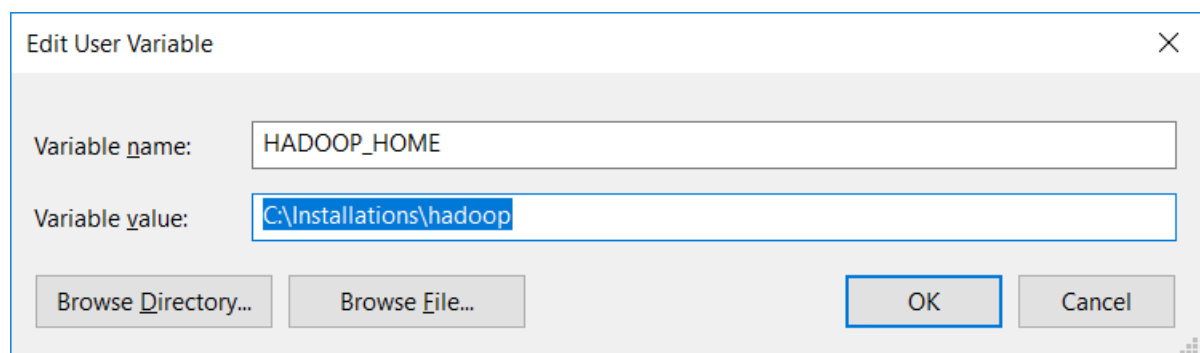
Spark attend winutils.exe dans l'installation Hadoop "<Répertoire d'installation Hadoop>/bin/winutils.exe" (notez le dossier "bin")

Création de HADOOP_HOME

- Créer un répertoire pour les binaire hadoop: e.g: C:\Program Files\hadoop\bin\
- Télécharger le fichier winutils.exe depuis le github :
<https://github.com/yacineab/ESGI-spark-core/blob/master/hadoop/bin/winutils.exe>
- Placer le fichier winutils.exe dans le répertoire ..\hadoop\bin\

Définir la variable d'environnement HADOOP_HOME

- Sous Windows 10 et Windows 8 accédez à Panneau de configuration> Système> Paramètres système avancés.
- Windows 7, cliquez avec le bouton droit sur Poste de travail et sélectionnez Propriétés> Avancé.
- Cliquez sur le bouton Variables d'environnement.
- Sous Variables système, cliquez sur Nouveau.
- Dans le champ Nom de variable, entrez : HADOOP_HOME
- Dans le champ Valeur, entrez votre chemin du répertoire hadoop. **e.g:** C:\Program Files\hadoop
- Cliquez sur OK et appliquez les modifications



LAB

Les données utilisées dans ce TP se trouvent sur le github dans le répertoire [data](#)

Lab 1 : Flight-Data

Nous allons travailler sur les données des vols entre différents pays pour l'année 2015. Ce jeu de données comptabilise le nombre de vols effectué entre deux destinations au cours de l'année 2015. Le dataset se trouve dans ce lien : [2015-summary.csv](#)

1. Télécharger le fichier csv et le placer dans un répertoire data
2. Créer un nouveau projet sbt depuis votre IDE avec la **version scala 2.12.x**
3. Ajouter les dépendances spark-core et spark-sql dans le fichier build.sbt
4. Créer une nouvelle classe de type object nommée FlightData qui va contenir votre code Spark
5. Créer une variable spark et instancier le SparkSession

Le point d'entrée dans l'API Spark DataFrame est le SparkSession, qui est instancié ici dans la variable appelée spark. L'appel de la fonction pour créer le DataFrame se fait donc à partir de cette variable.

6. Créer le DataFrame flightData2015 à partir des données du fichier précédent.
7. Utiliser la fonction printSchema() pour afficher le schéma de données de votre DataFrame
8. Afficher les 20 premières lignes de votre DataFrame

Quelques transformations

1. Créer un nouveau DataFrame où le pays de destination est égal au pays d'origine.
2. Faire un show() sur ce DataFrame
3. Faire explain sur ce DataFrame. Que comprenez-vous de retour de la fonction explain ?

Sorting

1. Trier le dataframe par le nombre de vol par destination i.e. la colonne count. Afficher les 20 premières lignes de ce DataFrame
2. Appeler la fonction explain sur ce dernier DataFrame. Que constateriez-vous ?

DataFrame à partir d'un fichier JSON

1. Télécharger le fichier JSON et le placer dans un répertoire data: [2010-summary.json](#)
2. Créer un DataFrame flightData2010Json à partir des données du fichier JSON
3. Afficher les premières lignes de votre DataFrame

DataFrame à partir d'un fichier PARQUET

1. Télécharger le fichier parquet et le placer dans un répertoire data: [flights.parquet](#)
2. Créer un DataFrame flightsParquet à partir du fichier précédent
3. Afficher les premières lignes de votre DataFrame