eBook Gratuit

APPRENEZ pyspark

eBook gratuit non affilié créé à partir des **contributeurs de Stack Overflow.**

#pyspark

Table des matières

À propos

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: pyspark

It is an unofficial and free pyspark ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official pyspark.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapitre 1: Démarrer avec pyspark

Remarques

Cette section fournit une vue d'ensemble de ce qu'est pyspark et pourquoi un développeur peut vouloir l'utiliser.

Il devrait également mentionner tous les grands sujets dans pyspark, et établir un lien avec les sujets connexes. La documentation de pyspark étant nouvelle, vous devrez peut-être créer des versions initiales de ces rubriques connexes.

Examples

Installation ou configuration

Instructions détaillées sur l'installation ou la configuration de pyspark.

Exemple de compte de mots dans Pyspark

L'exemple sous-jacent est juste celui donné dans la documentation officielle de pyspark. Veuillez cliquer ici pour atteindre cet exemple.

```
# the first step involves reading the source text file from HDFS
text_file = sc.textFile("hdfs://...")

# this step involves the actual computation for reading the number of words in the file
# flatmap, map and reduceByKey are all spark RDD functions
counts = text_file.flatMap(lambda line: line.split(" ")) \
             .map(lambda word: (word, 1)) \
             .reduceByKey(lambda a, b: a + b)

# the final step is just saving the result.
counts.saveAsTextFile("hdfs://...")
```

Consommer des données à partir de S3 en utilisant PySpark

Deux méthodes vous permettent de consommer des données à partir du compartiment AWS S3.

1. Utilisation de l'API sc.textFile (ou sc.wholeTextFiles): Cette API peut également être utilisée pour HDFS et le système de fichiers local.

```
aws_config = {}  # set your aws credential here
sc._jsc.hadoopConfiguration().set("fs.s3n.awsSecretAccessKey",
aws_config['aws.secret.access.key'])
sc._jsc.hadoopConfiguration().set("fs.s3n.awsSecretAccessKey",
aws_config['aws.secret.access.key'])
s3_keys = ['s3n/{bucket}/{key1}', 's3n/{bucket}/{key2}']
data_rdd = sc.wholeTextFiles(s3_keys)
```

2. En le lisant en utilisant une API personnalisée (Say a boto downloader):

```
def download_data_from_custom_api(key):
    # implement this function as per your understanding (if you're new, use [boto][1] api)
    # don't worry about multi-threading as each worker will have single thread executing your
job
    return ''

s3_keys = ['s3n/{bucket}/{key1}', 's3n/{bucket}/{key2}']
# numSlices is the number of partitions. You'll have to set it according to your cluster
configuration and performance requirement
key_rdd = sc.parallelize(s3_keys, numSlices=16)

data_rdd = key_rdd.map(lambda key: (key, download_data_from_custom_api(key))
```

Je recommande d'utiliser l'approche 2 car, tout en travaillant avec l'approche 1, le pilote télécharge toutes les données et les travailleurs le traitent. Cela présente les inconvénients suivants:

1. Vous allez manquer de mémoire à mesure que la taille des données augmente.
2. Vos travailleurs resteront inactifs jusqu'à ce que les données aient été téléchargées

Lire Démarrer avec pyspark en ligne: https://riptutorial.com/fr/pyspark/topic/5126/demarrer-avec-pyspark

Crédits

S. No	Chapitres	Contributeurs
1	Démarrer avec pyspark	Ashutosh, Community, Rahul Lakhanpal