

Spark

一、核心概念

1. 每个 `spark` 应用都由一个驱动程序 (`driver program`) 来发起集群上的各种并行操作
 - `driver program` 包含了应用的 `main` 函数, 并且定义了集群上的分布式数据集, 还对这些分布式数据集应用了相关操作
 - `driver program` 通过一个 `SparkContext` 对象来访问 `spark`
 - `driver program` 一般要管理多个执行器 (`executor`) 节点
2. `SparkContext`: 该对象代表了对计算集群的一个连接
 - 在 `pyspark shell` 中, 当 `shell` 启动时, 已经自动创建了一个 `SparkContext` 对象, 它叫做 `sc`。
 - 通常可以用它来创建 `RDD`

二、安装和使用

1. 安装步骤:
 - 从 <http://spark.apache.org/downloads.html> 下载 `Pre-built Apache Hadoop xx and later` 的版本
 - 解压即可
2. 在 `pycharm` 中使用 `pyspark`:
 - `File->Settings->Project->Project Structure`, 选择右侧的 `Add Content Root`。
 - 添加 `spark` 目录下的 `python` 目录
 - 注意, 如果 `pycharm` 使用了 `python3`, 则需要在脚本中添加语句:

```
import os
os.environ["PYSPARK_PYTHON"]="python3"
```

三、pyspark shell

1. `spark` 带有交互式的 `shell`, 可以用于即时数据分析
 - `spark shell` 可以与分布式存储在许多机器的内存或者硬盘上的数据进行交互, 处理过程由 `spark` 自动控制
 - `pyspark shell` 是 `spark shell` 的 `python` 版本
2. 使用 `pyspark shell`: 进入 `spark` 的安装目录, 然后执行 `bin/pyspark`。
 - `ubuntu16.04` 中默认使用 `python2.7`
 - 如果需要使用 `python3`, 则使用 `export PYSPARK_PYTHON=python3` 来导出环境变量
 - 或者在代码中使用 `os.environ["PYSPARK_PYTHON"]="python3"`
 - 退出 `pyspark shell`: `CTRL+D`
3. 修改 `pyspark` 日志: 在 `conf` 目录下创建一个 `log4j.properties` 的文件。

- 可以直接使用模板 `log4j.properties.template` , 将 `log4j.rootCategory=INFO,console` 修改为 `log4j.rootCategory=WARN,console`

四、独立应用

1. 独立应用与 `pyspark shell` 的主要区别在于：你需要自行初始化 `SparkContext` , 除此之外二者使用的 API 完全相同。
2. 在 `python` 中，你需要把独立应用写成 `python` 脚本，然后使用 `Spark` 自带的 `bin/spark-submit` 脚本来运行：

```
bin/spark-submit my_script.py
```

`spark-submit` 会帮助我们引入 `python` 程序的 `spark` 依赖

3. 在独立应用中，通常使用下面方法初始化 `SparkContext` :

```
from pyspark import SparkConf, SparkContext
conf = SparkConf().setMaster('local').setAppName('My App')
sc = SparkContext(conf = conf)
```

首先创建一个 `SparkConf` 对象来配置应用，然后基于该 `SparkConf` 来创建一个 `SparkContext` 对象。

- `.setMaster()` 给出了集群的 URL , 告诉 `spark` 如何连接到集群上。这里 `'local'` 表示让 `spark` 运行在单机单线程上。
 - `.setAppName()` 给出了应用的名字。当连接到一个集群上时，这个值可以帮助你找到你的应用。
4. 关闭 `spark` 可以调用 `SparkContext` 的 `.stop()` 方法，或者直接退出应用（如调用 `System.exit(0)` 或者 `sys.exit()`）
 5. 如果需要使用 `python3` , 则使用 `export PYSARK_PYTHON=python3` 来导出环境变量。
 - 或者在代码中使用 `os.environ["PYSARK_PYTHON"]="python3"`