



Ejecución de Spark en la máquina local

4 minutos de lectura · 9 de noviembre de 2019



Islam Shariful

Seguir



Escuchar



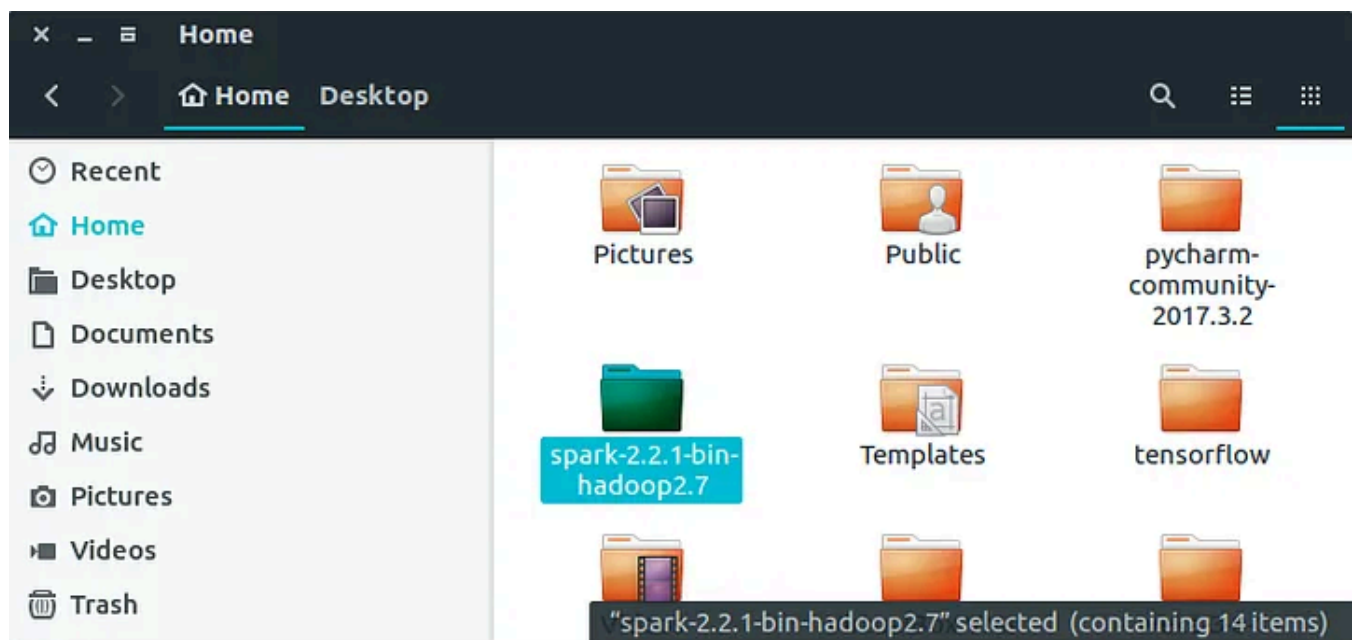
Compartir

Apache Spark es un sistema de computación en clúster rápido y de propósito general. Para aprovechar al máximo su potencial, Spark debe ejecutarse en un sistema distribuido. Sin embargo, es posible que no se tenga acceso a un sistema distribuido constantemente. Especialmente para fines de aprendizaje, es posible que se desee ejecutar Spark en el propio ordenador. Esto es realmente muy sencillo. Hay varias maneras de hacerlo. Les mostraré lo que hice para ejecutar Spark en mi portátil.



El primer paso es descargar Spark desde este [enlace](#) (en mi caso, lo guardé en el directorio de inicio). Luego, descomprime la carpeta usando la línea de comandos o

haciendo clic derecho en el archivo *.tar . La siguiente figura muestra la carpeta descomprimida, desde donde ejecutaría Spark.



Spark descargado (descomprimido)

Ejecutar Spark desde la línea de comandos

Ahora podemos ejecutar Spark fácilmente desde la línea de comandos. Necesitamos obtener la ubicación de la carpeta. En mi caso, la ubicación (que se puede copiar desde la barra de direcciones con Ctrl + L) es [nombre del /home/user_name/spark-2.2.1-bin-hadoop2.7 archivo]. Ahora necesito configurar algunas variables

\$SPARK_HOME y \$PYSPARK_PYTHON [nombre del archivo]. Después de esto, puedo ejecutar Spark fácilmente escribiendo [nombre del archivo] \${SPARK_HOME}/bin/pyspark . Las líneas que usé en mi línea de comandos se muestran a continuación:

```
1 user_name:~$ export SPARK_HOME=/home/shanto/spark-2.2.1-bin-hadoop2.7
2 user_name:~$ export PYSPARK_PYTHON=python3
3 user_name:~$ ${SPARK_HOME}/bin/pyspark
```

pyspark_env.txt hosted with ♥ by GitHub

[view raw](#)

Ahora, ejecutar código Spark también es fácil. Solo necesito escribir la siguiente línea en la línea de comandos (sparkcode.py es el archivo donde escribí algunas líneas de código Spark).

```
1 user_name:~$ ${SPARK_HOME}/bin/spark-submit sparkcode.py
```

spark_submit.txt hosted with ♥ by GitHub

[view raw](#)

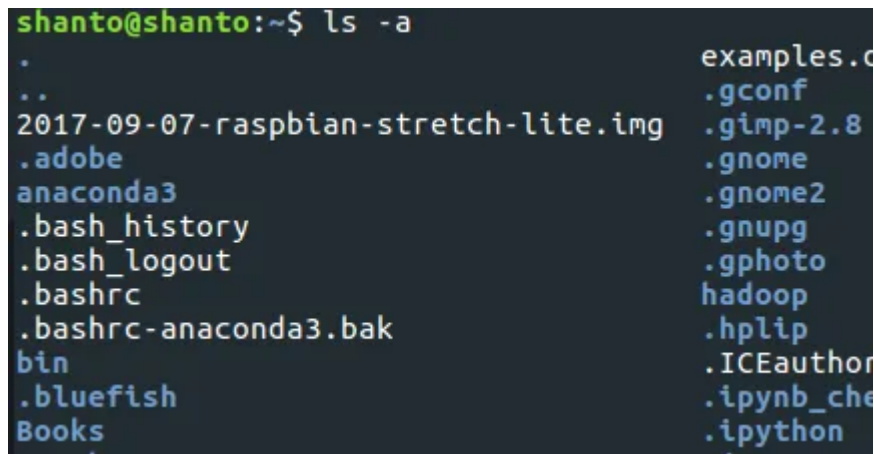
Bien. Todo funciona correctamente hasta ahora. Sin embargo, si cerramos la terminal y escribimos `${SPARK_HOME}/bin/pyspark` en una nueva, no funcionará porque las variables de entorno que configuré ya no existen. ¿Necesito configurar las variables cada vez que abro una nueva terminal? Para que sea permanente, debemos editar el `.bashrc` archivo en tu directorio personal. Comprobemos si el archivo está ahí.

```
1 user_name:/$ cd ~
2 user_name:~$ ls -a
```

[add_env_to_bashrc.txt](#) hosted with ♥ by [GitHub](#)

[view raw](#)

El archivo es un archivo oculto en su directorio de inicio, como se muestra en la siguiente figura.



```
shanto@shanto:~$ ls -a
.          examples.d
..         .gconf
2017-09-07-raspbian-stretch-lite.img .gimp-2.8
.adobe     .gnome
anaconda3 .gnome2
.bash_history .gnupg
.bash_logout .gphoto
.bashrc      hadoop
.bashrc-anaconda3.bak .hplip
bin         .ICEauthor
.bluefish   .ipython
Books       .ipython
```

Ahora necesitamos abrirlo `.bashrc` en cualquier editor (lo estoy abriendo con nano) y agregar las líneas como se muestra en el siguiente fragmento de código.

```
1 user_name:~$ sudo nano .bashrc
2 password for user_name:
```

[change_bashrc.txt](#) hosted with ♥ by [GitHub](#)

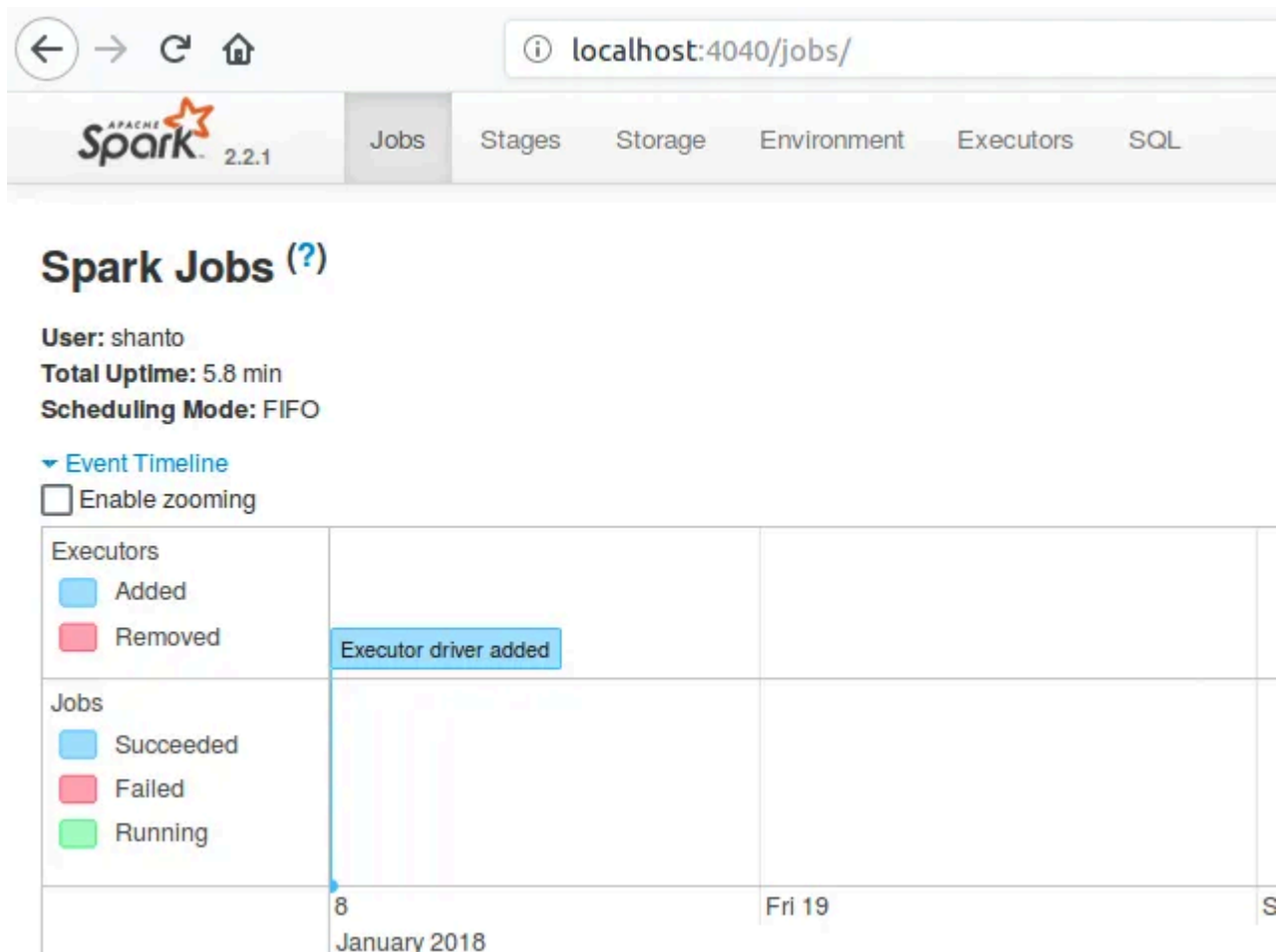
[view raw](#)

Agregue estas líneas al final del archivo, guarde el archivo (CTRL + X) y salga:

```
1 export SPARK_HOME=/home/shanto/spark-2.2.1-bin-hadoop2.7
2 export PYSARK_PYTHON=python3
3 export PATH=$SPARK_HOME/bin:$PATH
```

[add_to_bashrc.txt](#) hosted with ♥ by [GitHub](#)

[view raw](#)



Ejecución de Spark en un notebook de Jupyter

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text [1]. This is also very popular among data scientists. I use Jupyter a lot, specially for small size project and where I need to add explanation and graphs for visualization. As I also use Spark for different projects, I need to run Spark from my Jupyter notebook. There are a few ways of doing that. The simplest way is to install the package *findspark*.

I already have Jupyter installed in my laptop. Now I just need to install *findspark* using the following command in command line. **Without any arguments, the `$SPARK_HOME` environmental variable will be used by *findspark*, so the previous step where we set the value of `$SPARK_HOME` is a prerequisite.**

jupyter sparktest Last Checkpoint: a minute ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help

Code

```
In [1]: import findspark
findspark.init()

In [2]: import pyspark

In [3]: from pyspark import SparkConf, SparkContext
import sys
from pyspark.sql import SparkSession, functions, types, SQLContext

In [4]: spark = SparkSession.builder.appName('temperature range sql').getOrCreate()
sc = spark.sparkContext
sqlContext = SQLContext(sc)

In [6]: # from: https://stackoverflow.com/questions/43751509/how-to-create-new-dataframe-with-dict
cMap = {"k1": "v1", "k2": "v1", "k3": "v2", "k4": "v2"}
a_cMap = [(k,)+(v,) for k,v in cMap.items()]
data = spark.createDataFrame(a_cMap, ['key','val'])

from pyspark.sql.functions import count
data = data.groupBy('key').pivot('val').agg(count('val'))
data.show()

+---+---+---+
|key| v1| v2|
+---+---+---+
| k2| 1|null|
| k4|null| 1|
| k1| 1|null|
| k3|null| 1|
+---+---+---+
```

In []:

So, we are all set. Let's have some fun with Spark.

Spark

Pyspark

Jupyter Notebook

Big Data



Seguir

Written by Shariful Islam

19 followers · 54 following

Data Scientist, Machine Learning and AI Enthusiast