|  |
| --- |
| Máster en Big Data |
| Tecnologías de Almacenamiento |
| 9. Hands-On: Spark |

PRESENTADO:

JOSE DAVID ANGULO GARCIA

ALBERT RIPOLL

Índice

[1. Introducción 3](#__RefHeading___Toc453_1462167123)

[2. Entorno 3](#__RefHeading___Toc455_1462167123)

[3. Inspección de los datos locales 3](#__RefHeading___Toc457_1462167123)

# Introducción

El objetivo de este Hands-On es el de familiarizarse con una de las librerías más populares del framework de Spark como es SparkSQL

# Entorno

Para la realización de los ejercicios se va a utilizar *spark-shell* en scala o python ya que nos proporciona un entorno muy dinámico para la introducción de funciones y nos permite recibir una respuesta inmediata.

Para ello, nos descargaremos Spark y levantaremos una *spark-shell* en local.

Sigue las siguientes instrucciones si es necesario:

https://sparkbyexamples.com/spark/install-apache-spark-on-mac/

https://sparkbyexamples.com/spark/apache-spark-installation-on-windows/

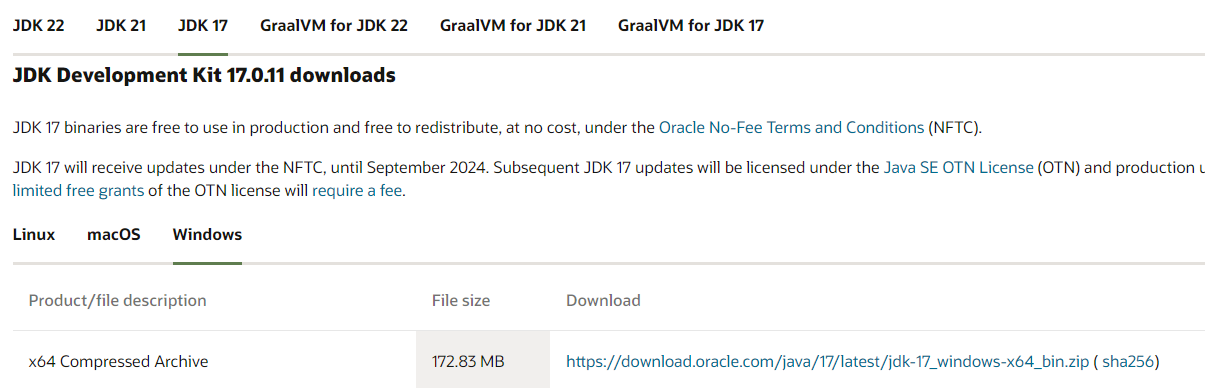
https://sparkbyexamples.com/spark/spark-installation-on-linux-ubuntu/

**Descarga e instalación del Spark y levantamiento de spark-shell en local.**

1. **Descargas.**

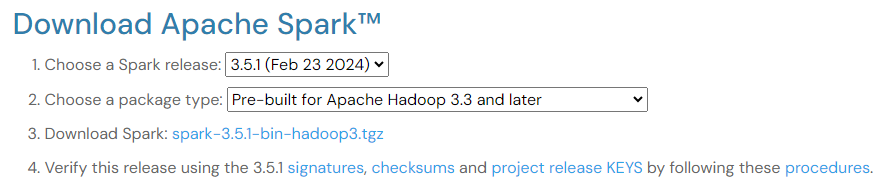
<https://www.oracle.com/java/technologies/downloads/#jdk17-windows>

Se descarga el “x64 Compressed Archive” selecionando opciones JDK17 y Windows.

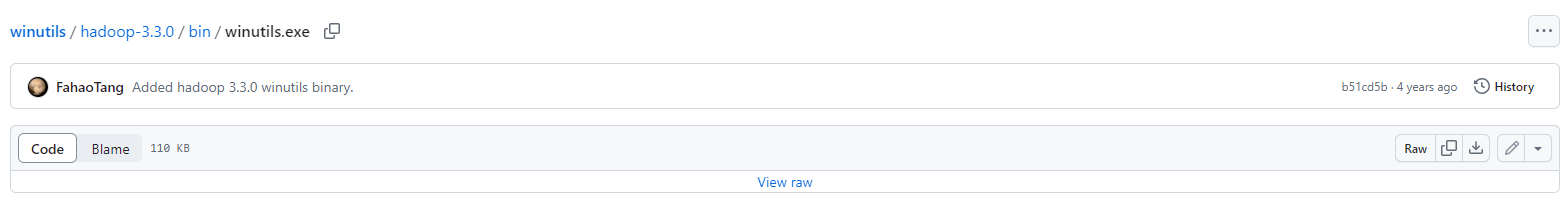


<https://spark.apache.org/downloads.html>

Se descarga spark con enlace de opción 3

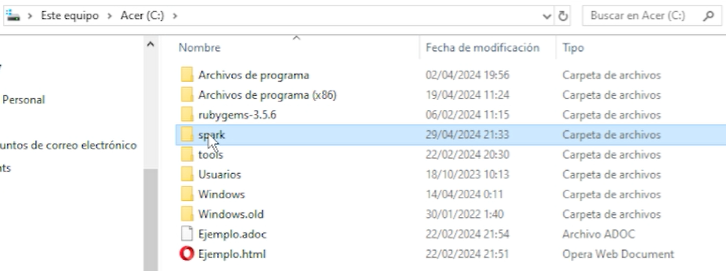


Se descarga winutils con el icono  de la derecha

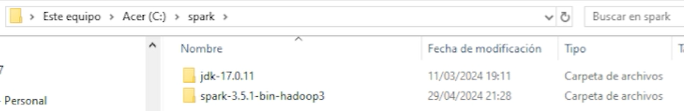


1. **Colocación de las descargas.**

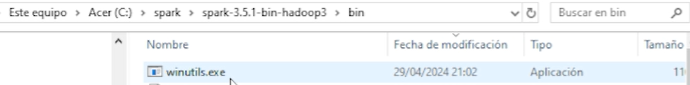
En C: Acer se crea una carpeta llamada “spark”

****

Dentro de esta carpeta “spark” hemos situado las dos carpetas “jdk-17.0.11” y “spark-3.5.1-bin-hadoop3” de las descargas descomprimidas.

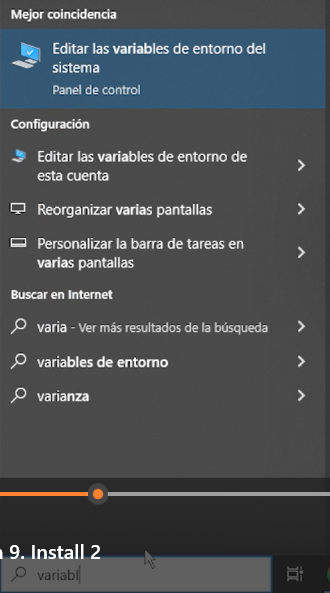
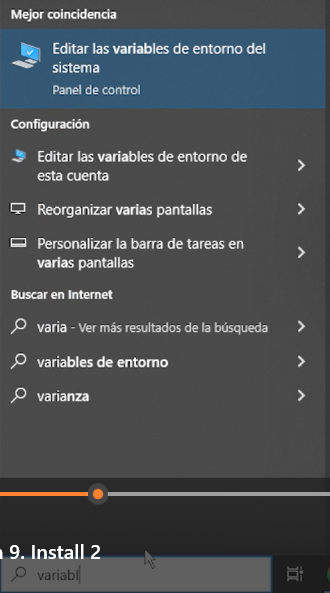
****

Dentro de “spark-3.5.1-bin-hadoop3” y allí dentro de la carpeta “bin” hemos situado el ejecutable “winutils.exe”.

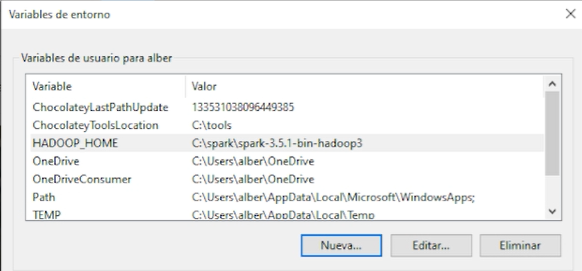


1. **Editar variables de entorno de esta cuenta**

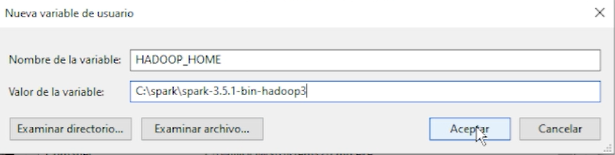
Hemos buscado en Windows la opción “Editar variables de entorno de esta cuenta”

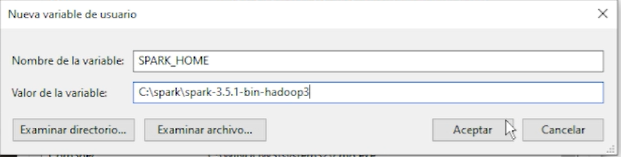
 → 

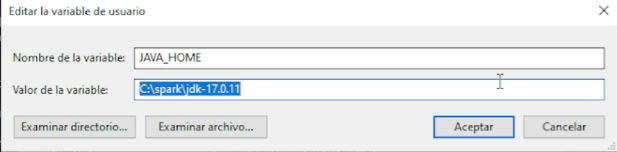
Y allí “Nueva”



Para añadir las distintas variables:







Tal que así:



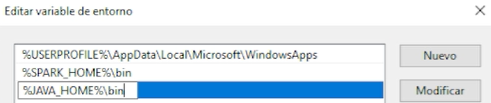


En el path le damos a Editar



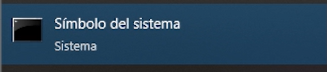


Y en “Nuevo” definimos %SPARK\_HOME%\bin y %JAVA\_HOME%\bin

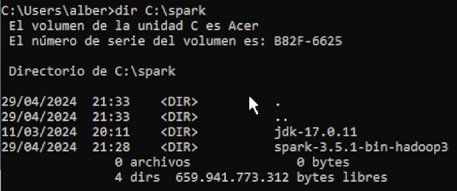


1. **Comprovación de correcta instalación**

Vamos al comand prompt

 → 

Nos situamos en la carpeta



Vamos a bin



Abrimos spark-shell



Comprovamos



Ejecutamos spark-shell



**5) Dataset a utilizar**

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Abre un Terminal (en windows preferiblemente PowerShell) apuntando al directorio *../spark-x.y.z-bin-hadoop/bin/*

El dataset que utilizaremos se llama *Summer-Olympic-medals-1976-to-2008.csv*.

Texto

Descripción generada automáticamente

# Inspección de los datos locales

Carga el dataset de la siguiente manera (con la ruta del dataset que aplique):

*val dataset = spark.read.option("header","true").option("inferSchema", "true").csv("/path/to/Summer-Olympic-medals-1976-to-2008.csv")*

val olym\_dataset = spark.read.option("header","true").option("inferSchema","true").csv("C:/storage/Summer-Olympic-medals-1976-to-2008.csv")

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

1. Explora el dataset

Texto

Descripción generada automáticamente

1. ¿Que han aportado las opciones “header” y “inferSchema”?

olym\_dataset.head()

Este comando permite ver todos los datos de la primera fila del dataset explorado sin el nombre de las columnas.

olym\_dataset.schema

Permite ver las características que tiene cada columna del dataset como el tipo de dato.

Texto

Descripción generada automáticamente

1. ¿Como harías para contar las medallas conseguidas por año y país?

Primero importar la librería de funciones de sql para que el spark pueda interpretar las sentencias que se le van a entregar a través de una variable, donde se agrupara por año y país, contando las medallas que tiene cada uno.

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Escala de tiempo

Descripción generada automáticamente con confianza media

1. Usando SparkSQL muestra alguna métrica interesante.

Para poder ejecutar alguna sentencia select, se debe crear una vista de la variable que contiene el dataset. Y luego crear otra variable que contenta la consulta a realizar. Como sigue:

olym\_dataset.createTempView("agrupacion")

val cantciudad = spark.sql("""SELECT City, Sport, COUNT(\*) AS CantMedallas FROM agrupacion GROUP BY City, Sport ORDER BY CantMedallas DESC""")

Texto

Descripción generada automáticamente

1. ¿Que muestra la SparkUI: [http://localhost:4040](http://localhost:4040/) ?

SPARK\_UI LOCAL: ***http://joseangulo:4040/jobs/***

Observar que cada imagen es la vista de las opciones de arriba de esta pagina: JOBS, STAGES, ENVIROMENT, EXECUTORS, SQL

**VISTA JOBS:** Los Trabajos que se han ejecutado con sus caracteristicas principales de performance.

Tabla

Descripción generada automáticamente

**VISTA DE STAGES:**

Tabla

Descripción generada automáticamente

**VISTA DE LOS ENVIRONMENT:**

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**VISTA DE LOS EXECUTORS:**

Una captura de pantalla de una computadora

Descripción generada automáticamente

**VISTA DE SQL:**

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente