Datos del estudiante

**Nombre y apellidos**

**Fecha de entrega**

Actividad: Uso de HDFS, Spark SQL y MLlib

Objetivos de la actividad

Con esta actividad, los estudiantes pondrán en práctica los conocimientos estudiados en las clases de teoría acerca del manejo de HDFS y Apache Spark. Completarán un ejercicio sencillo, que involucra las dos tecnologías y que los ayudará a entender mejor el propósito de cada una, así como a verlas funcionar en un caso concreto.

A continuación, describimos el trabajo que debe llevar a cabo el alumno y presentaremos la infraestructura disponible en la cual se realizarán las tareas, además de unas orientaciones sobre cómo utilizarla incluidas en el Anexo.

Pautas de elaboración

Se compone de tres partes diferentes.

**PARTE 1**.Manejo de HDFS. Tras acceder a la terminal de Linux en Jupyter Lab, el alumno deberá:

Interfaz de usuario gráfica, Aplicación, Word

Descripción generada automáticamente

Figura 1. HDFS. Fuente: elaboración propia.

* Crear en el directorio raíz de **HDFS** una carpeta llamada <nombre\_apellidos\_ alumno> (sin tildes ni espacios), utilizando el comando de HDFS apropiado. Por ejemplo: /Pepito\_Perez.
* Utilizando el botón **Upload** de JupyterLab (una flecha apuntando hacia arriba que se encuentra en la parte superior del menú lateral), hay que subir el fichero CSV que se indica en el *notebook* a la carpeta **Local Disk** de JupyterLab. Esto se subirá al disco duro de la máquina máster del clúster, que es donde se está ejecutando JupyterLab, y se pondrá en el directorio raíz (/) del sistema de ficheros Linux.
* A continuación, hay que subirlo desde el disco local a la carpeta de **HDFS** creada en el primer apartado, utilizando para ello el comando de HDFS apropiado.
* Una vez subido a HDFS, ejecutar el comando de HDFS que nos da información acerca de cómo está almacenado ese fichero en HDFS. El fichero, por tanto, debe estar ubicado en la siguiente ruta de HDFS: **/<nombre\_apellidos\_alumno>/<nombre\_fichero.csv>**
* Tras este paso, tomar una captura de pantalla que muestre que se han ejecutado dichos comandos, así como sus salidas.

**PARTE 2**.Manejo de Apache Spark con *notebooks* de Jupyter. Se utilizará el archivo /<nombre\_alumno>/<nombre\_fichero.csv> que hemos subido a HDFS para resolver las siguientes cuestiones

* Subir **al directorio GCS** que aparece en el menú lateral de JupyterLab el *notebook* «actividad1.ipynb»*.* **No se debe cambiar, en ningún caso, el nombre del fichero antes de entregarlo**. La plataforma de UNIR ya se encarga de organizarlos automáticamente en carpetas con el nombre del alumno cuando son descargados.
* Responder a las cuestiones indicadas en el *notebook*.
* **NOTA**. Si, al abrir el *notebook* recién subido a Google Cloud Storage, en la parte superior derecha, aparece Python3, pincha sobre dicho texto y escoge la opción **PySpark** para seleccionar el uso de Spark mediante Python.

**PARTE 3**.Spark MLlib. Las instrucciones se encuentran en el *notebook*.

* *Notebook* que funcione, donde todos los <COMPLETAR> hayan sido rellenados.
* Breve documento con las capturas de pantalla que demuestren haber ejecutado con éxito los comandos de HDFS.

Anexo: creación de un clúster en Google Cloud

Cuando se dispone de un clúster en un entorno real, generalmente se instala, en cada uno de los nodos, una distribución del sistema operativo Linux y, sobre este, Spark, Hadoop (HDFS) y el resto de herramientas (Kafka, Hive, etc.). Para que funcionen correctamente, se configura cada una de ellas indicando las direcciones IP del resto de nodos y algunos detalles, como qué nodo es el *namenode* y cuál el *datanode* (HDFS), qué nodo de Spark hace el papel de máster, etc. La figura 2 representa una instalación en un clúster con tres nodos, a los que hemos llamado *unircluster-m* (nodo que hace el papel de «máster», en el sentido de que en él se instalan JupyterLab y el *namenode* de HDFS, y de que no contiene ningún bróker de Kafka para dejarlo más libre), *unircluster-w-0* y *unircluster-w-1*.

Apache Kafka

broker

Hadoop (HDFS) rol de datanode

Apache Spark executor (n cores)

Apache Kafka   
broker

Apache Spark executor (n cores)

Hadoop (HDFS) rol de datanode

unircluster-w-0

unircluster-m

unircluster-w-1

Hadoop (HDFS) rol de namenode

Apache Spark executor (n cores)

JupyterLab

SO Linux (Debian 10)

SO Linux (Debian 10)

SO Linux (Debian 10)

Figura 2. Ejemplo de despliegue en un clúster real con tres nodos físicos que usan el sistema operativo Linux. Fuente: elaboración propia.

Crearemos un clúster exactamente como el de la figura anterior en la plataforma Google Cloud, mediante su herramienta Dataproc. Google nos da numerosas facilidades, ya que despliega automáticamente las herramientas que necesitamos usar (HDFS, Spark, Kafka y JupyterLab) y marca las distintas opciones en una interfaz gráfica. Debemos tener en cuenta que **dicho clúster se crea al vuelo: está levantado durante un cierto período de tiempo, en el que podemos configurarlo, y después se desmantela automáticamente.**

**Paso 1:** entramos en <http://cloud.google.com/free> y pulsamos en «Empezar gratis». Nos solicitará un correo de Gmail, los datos personales y una tarjeta de crédito. Google nos regala $300 de saldo que se pueden gastar en 90 días. Teniendo en cuenta el poco uso que le daremos en esta asignatura, nunca llegaremos a consumir más allá de unos pocos dólares y no se cargará nada.

**Paso 2**: accedemos a <https://console.cloud.google.com/> y activamos la API. Para ello, en la barra del buscador de API de Google Cloud, escribiremos «dataproc» y haremos clic sobre el servicio que aparecerá como resultado de la búsqueda. Una vez que estemos en la página principal del servicio, activamos la API.

**Paso 3**: accedemos al menú lateral y buscamos **STORAGE** 🡪 **Storage** 🡪 **Browser**. Hacemos clic en la parte superior, en **CREATE BUCKET**.

* Elegimos el nombre del *bucket* (el que queramos, siempre que esté libre).
* Seleccionamos dónde almacenar los datos: **Location type 🡪 Region***.* En el menú inferior: **Location 🡪** **europe-west1**
* No hay que modificar ninguna otra opción.
* Pulsamos abajo el botón azul **CREATE**.

Esto creará un *bucket* de Google Cloud Storage (GCS), que es un sistema de almacenamiento persistente gratuito que **sí mantiene el contenido que hayamos almacenado en él, aunque el clúster sea desmantelado**. Por eso, lo usaremos para guardar tanto los *datasets* como los *notebooks* de JupyterLab.

**Paso 4**: accedemos, en el menú lateral, al apartado **BIG DATA (MACRODATOS) 🡪 Dataproc 🡪 Clusters**. Aunque es posible crear el clúster a través de la interfaz gráfica, las opciones que hay que especificar para este curso son muchas y es relativamente fácil equivocarse, por lo que aquí vamos a pulsar directamente en **el botón para abrir el Google Cloud Shell** (terminal de Google para introducir comandos de Google que manejan cualquier componente) que aparece en la **esquina superior derecha** (ver imagen) y pegaremos el comando siguiente.

****

Figura 3. Botón para abrir Google Cloud Shell. Fuente: elaboración propia.

Hemos señalado en rojo y en negrita las opciones **que cada alumno debe** **modificar**:

* Nombre que queramos dar a nuestro clúster.
* **Nombre de nuestro *bucket*** de Google Cloud Storage que hemos creado antes.
* **Identificador** de nuestro proyecto de Google Cloud. **No es el nombre del proyecto** sino el **identificador** que aparece en **letras amarillas** al abrir la terminal. Debe ser del tipo **XXXX-YYYY-12345** (dos palabras separadas por guion, y al final un número, por ejemplo, el ID de mi proyecto es impactful-pilot-269111, pero el vuestro será otro distinto que tenéis que mirar, copiar y pegar aquí)
* Tiempo máximo, en segundos, que queremos que permanezca levantado (por ejemplo 4 horas = 14400 s).

gcloud beta dataproc clusters create **nombrecluster** --enable-component-gateway --bucket **nombrebucket** --region europe-west1 --zone europe-west1-c --master-machine-type n1-standard-2 --master-boot-disk-size 500 --num-workers 2 --worker-machine-type n1-standard-2 --worker-boot-disk-size 500 --image-version 2.0-debian10 --properties spark:spark.jars.packages=org.apache.spark:spark-sql-kafka-0-10\_2.12:3.1.3 --optional-components JUPYTER,ZOOKEEPER --max-age **14400s** --initialization-actions 'gs://goog-dataproc-initialization-actions-europe-west1/kafka/kafka.sh' --project **identificadorproyecto**

Vamos al menú **Clusters** y, cuando el clúster aparezca con una marca en verde, pulsamos sobre su nombre, vamos a **WEB INTERFACES** y pinchamos abajo en JupyterLab.

**IMPORTANTE**: comprobad que el comando se ha pegado correctamente (todo el comando es una sola línea) y no os habéis dejado algún carácter suelto o se ha comido algún espacio en blanco al pegar.