Hadoop

Unidad	Unidad 2
Entrega	Método de entrega de la actividad en el campus virtual

1. ¿Cuáles son los objetivos de la práctica?

1. Entender la Orquestación de Contenedores: Familiarizarse con docker-compose y la manera en que puede ser utilizado para coordinar múltiples contenedores que trabajan juntos.
2. Configurar un Clúster Hadoop Simulado: Aprender a configurar un clúster Hadoop básico utilizando Docker, que incluya un NameNode y varios DataNodes.
3. Practicar con la Interfaz Web de Hadoop: Utilizar la interfaz web de Hadoop para monitorear el estado y rendimiento del clúster de Hadoop.
4. Ejecución de Trabajos MapReduce: Ejecutar trabajos MapReduce en el clúster para entender cómo Hadoop distribuye y procesa los datos.
5. Manejo de Volúmenes en Docker: Aprender cómo los volúmenes de Docker se pueden usar para persistir datos a través de contenedores y cómo esto se traduce en el almacenamiento de datos para Hadoop.

2. Enunciado

Familiarizarse con Docker mediante la creación, configuración y ejecución de un contenedor que ejecute una aplicación web básica:

1. Obtén una imagen de Hadoop para Docker: Puedes buscar en Docker Hub una imagen de Hadoop que alguien más haya creado.
    a. Inicia el contenedor de Hadoop
    b. Verifica la instalación de Hadoop

```bash
# Verificar la versión de Hadoop
hadoop version

# Ejecutar un comando simple de Hadoop (por ejemplo, listar directorios en H
hdfs dfs -ls /
```

2. Practica Hadoop
    a. Crear un nuevo directorio en HDFS

```bash
hdfs dfs -mkdir /test
```

    b. Descarga los datos de la practica final (https://www.kaggle.com/competitions/nfl-big-data-bowl-2024/data)
    c. Copia los archivos desde tu sistema local al sistema de archivos HDFS (Sustituye localfile.txt son los archivos descargados)

```bash
docker cp localfile.txt <container_id>:/localfile.txt
```

d. En el contenedor, copia ese archivo al HDFS:

```bash
hdfs dfs -put /localfile.txt /test/
```

e. Listar los archivos en el directorio HDFS creado:

```bash
hdfs dfs -ls /test
```

f. Leer el contenido de los archivos (Sustituye localfile.txt son los archivos descargados):

```bash
hdfs dfs -cat /test/localfile.txt
```

3. Realizar una operación map-reduce con los datos descargados.
   a. Nota: No es necesario realizar una operación con todos los csv, solo con uno de ellos.

3. Detalles de la entrega

1. Memoria en pdf que contenga:
   a. Capturas de pantalla y descripción de los pasos 1 y 2 descritos anteriormente.
   b. Explicación de la operación map-reduce realizada.
2. Incluir la clase .java o jar de la operación map-reduce realizada

4. Anexo I

Docker hub: https://hub.docker.com/
Apache Maven: https://maven.apache.org/
IntelliJ IDEA: https://www.jetbrains.com/idea/
Documentación Hadoop: https://hadoop.apache.org/docs/stable/

Como configurar un proyecto MAP-REDUCE en InteliJ IDEA

Se recomienda realizar la practica utilizando Maven e IntelliJ IDEA, puedes seguir los siguientes pasos pero antes asegúrate de tener instalado Java en tu sistema, ya que es un requisito previo para Maven y Hadoop.

Instalar Maven:

1. Descargar Maven: Ve al sitio web oficial de Maven y descarga la última versión.
2. Descomprimir: Descomprime el archivo en tu directorio deseado.

3. **Configurar el PATH**:
    - Windows: Agrega la carpeta bin de Maven al PATH en las variables de entorno del sistema.
    - Linux/Mac: Actualiza tu archivo .bashrc o .bash_profile con export PATH=/path/to/maven/bin:$PATH.

Instalar IntelliJ IDEA:

1. Descargar IntelliJ IDEA: Ve al sitio web de JetBrains y descarga la edición Community o Ultimate.
2. Instalar: Ejecuta el instalador y sigue las instrucciones.

Configurar IntelliJ IDEA para Maven:

1. Abrir IntelliJ IDEA y seleccionar New Project.
2. Elegir Maven como tipo de proyecto. Asegúrate de que el JDK correcto esté seleccionado.
3. Crear el proyecto siguiendo las instrucciones del asistente.

Descargar las Dependencias de Maven

Para un proyecto de MapReduce, necesitarás las dependencias de Hadoop. Esto se hace agregando las dependencias en tu archivo pom.xml. Aquí tienes un ejemplo básico para Hadoop:

```
<dependencies>
    <!-- Hadoop -->
    <dependency>
        <groupId>org.apache.hadoop</groupId>
        <artifactId>hadoop-common</artifactId>
        <version>2.7.1</version>
    </dependency>
    <dependency>
        <groupId>org.apache.hadoop</groupId>
        <artifactId>hadoop-mapreduce-client-core</artifactId>
        <version>2.7.1</version>
    </dependency>
    <!-- Otras dependencias necesarias -->
</dependencies>
```

Crear la Clase MapReduce* (Ver ejemplo):

1. Crear Clases Mapper y Reducer:
    1. En IntelliJ, crea nuevas clases Java para tu Mapper y Reducer.
    2. Implementa las clases Mapper y Reducer siguiendo la API de Hadoop.
2. Escribir la Lógica del Mapper y Reducer:
    1. En tu clase Mapper, sobrescribe el método map.

2. En tu clase Reducer, sobrescribe el método reduce.

3. Clase Driver:
    1. Crea una clase principal (Driver) que configure y ejecute el trabajo de MapReduce.

Nota: No es necesario utilizar todos los archivos csv, cuidado ya que el map-reduce se ejecuta sobre todos los ficheros de una carpeta hdfs.
Nota 2: Tened cuidad que los ficheros CSV esta divididos por “salto de línea” y por “coma”.

Crear el JAR

Para empaquetar tu aplicación en un JAR:

1. Configurar Artefacto JAR en IntelliJ:
    1. Ve a File > Project Structure > Artifacts.
    2. Agrega un nuevo artefacto JAR desde módulos con dependencias.
    3. Selecciona tu clase principal en el campo Main Class.
2. Construir el Artefacto:
    1. Ve a Build > Build Artifacts.
    2. Selecciona tu artefacto y haz clic en Build.
3. Ejecutar y Probar:
    1. Puedes probar tu JAR en un entorno de Hadoop o usar un entorno local para pruebas básicas.

Ejecuta tu .jar

Es posible que como hayas configurado tu proyecto, el comando de la ejecución se más sencillo que el visto en la clase virtual síncrona.

- hadoop jar {nombre_jar}.jar {carpeta_hdfs_entrada} {carpeta_hdfs_salida}