# Cloud Dataproc

# Comment

Dataproc en GCP nos permitirá tener el mismo ecosistema opensource de Hadoop y Spark. Esto permite correr los mismos códigos para Big Data que tengan hechos directamente en GCP, sin cambios, y con la ventaja de tener un cluster en cosa de minutos.

En este ejemplo, veremos lo sencillo que es crear un cluster en Dataproc, con un nodo maestro y dos workers.

Para ejemplificar el uso del cluster usaremos Spark SQL usando Jupyter Notebooks que ejecutará el trabajo sobre este cluster de Dataproc.

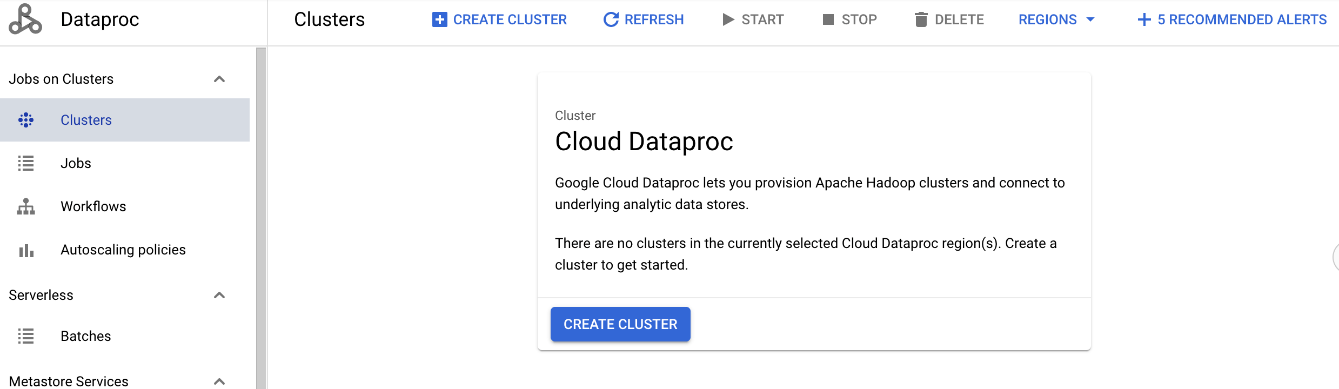
Los datos a analizar los pondremos primero en HDFS y luego en Cloud Storage, para desacoplar la computación del almacenamiento.

Finalmente, pasaremos el trabajo del notebook a un archivo python para que sea lanzado como un trabajo (job) al cluster.

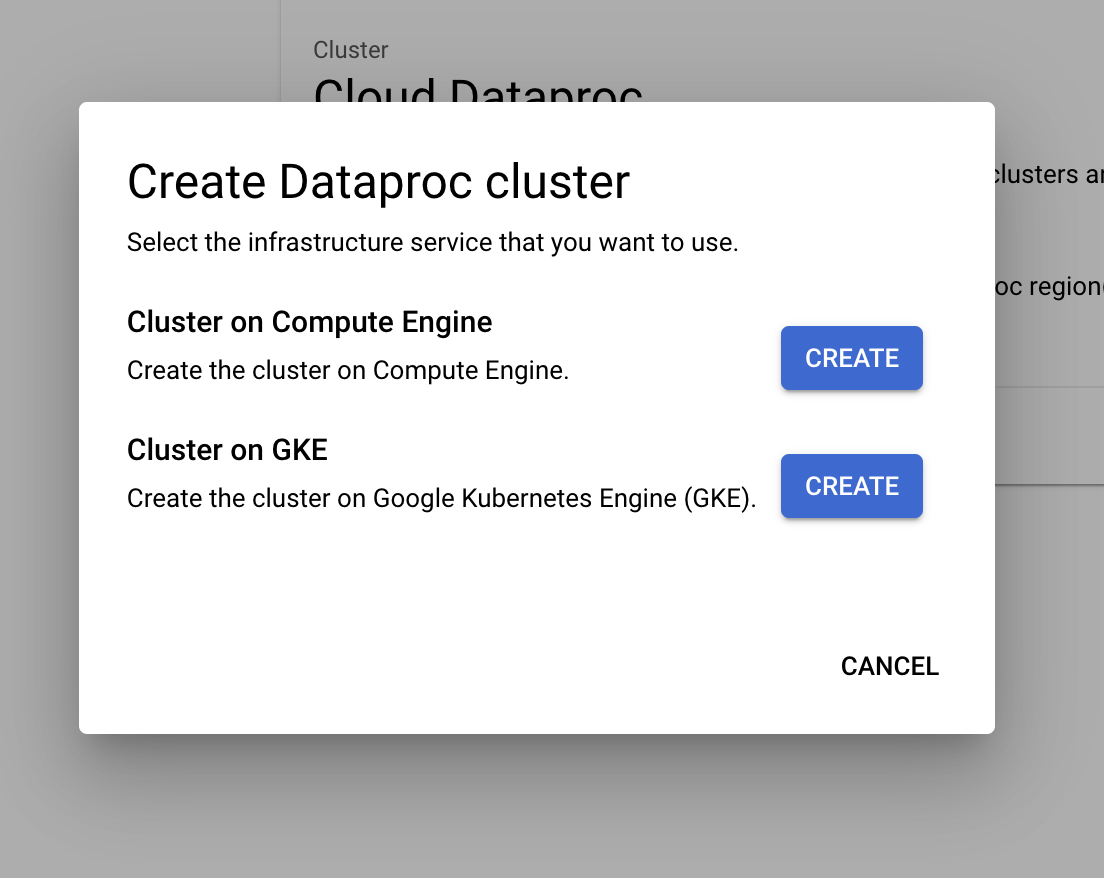
## Cluster de Cloud Dataproc

Buscaremos Dataproc en el menú de navegación, y le pondremos un pin para encontrarlo rápidamente la siguiente vez. Habilite la API en caso que no esté habilitada con anterioridad.

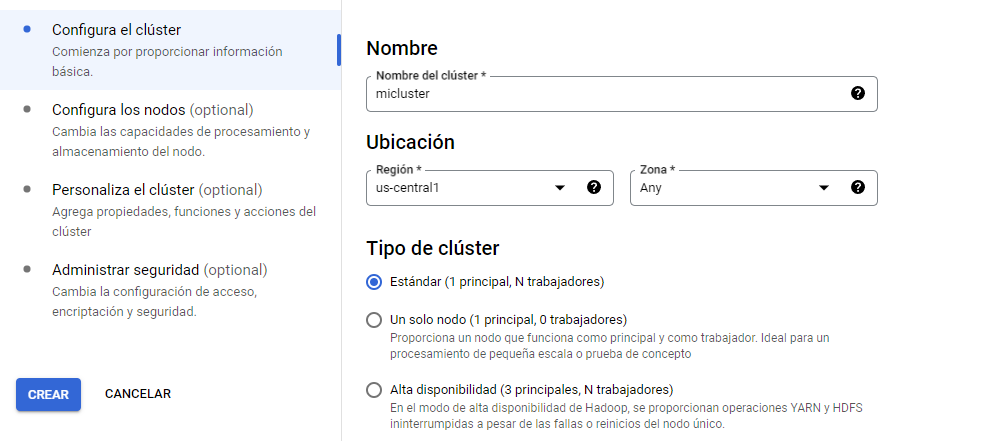
Haz click en **CREATE CLUSTER**.



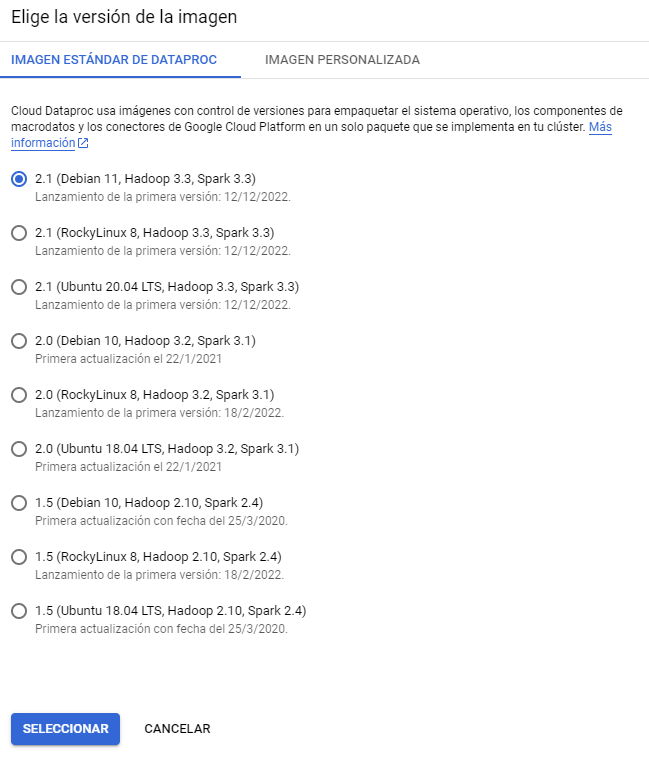
Luego da la opción entre usar Compute Engine o GKE, elegiremos Compute Engine.



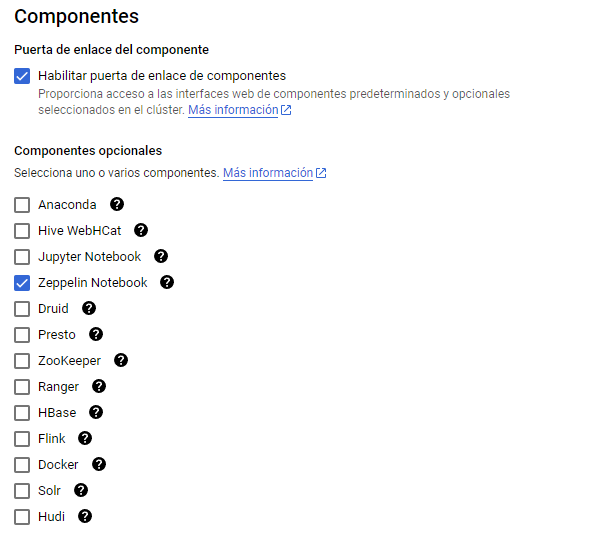
Le pondremos un nombre a nuestro cluster, por ejemplo “micluster”, seleccionamos región, el tipo de cluster dejaremos el **Standard**.



Luego pon **CAMBIAR** en la sección **Control de Versiones** y selecciona 2.1 (Debian 11, Hadoop 3.3, Spark 3.3). Aquí es donde ajustas las versiones de los software que necesitas y si no están puedes proveer una imagen hecha por ti. Pon **SELECT**.



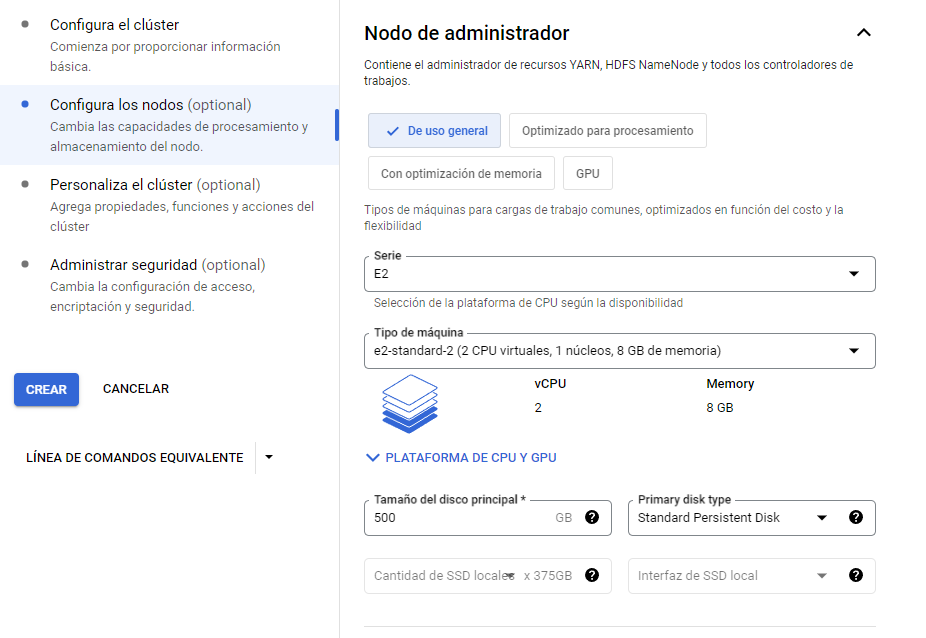
En la sección **Components**, haz click en **Enable component gateway**. Y en **Optional Components** selecciona **Jupyter Notebooks**.

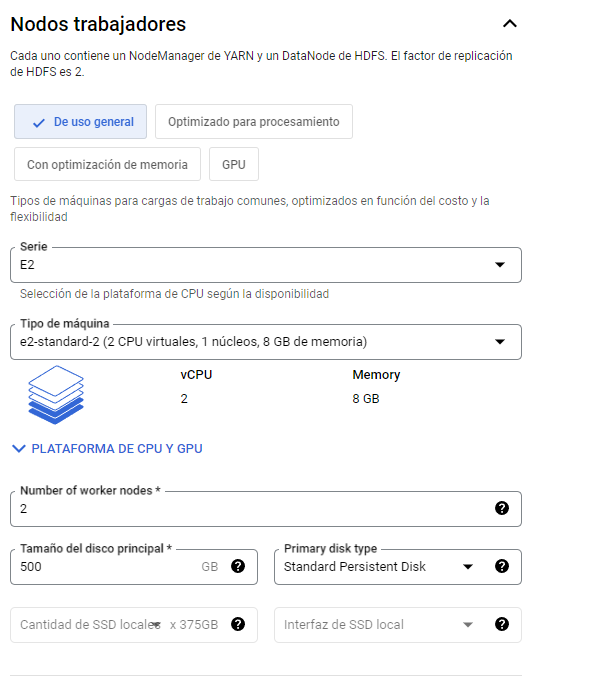


Explora las configuraciones opcionales, pero deja los valores por defecto. Haz click en Equivalent Command Line para ver el gcloud command que crea el cluster que configuraste sin necesidad de tantos clicks. Puedes copiarlo, es útil el caso de que tengas que cerrar el trabajo antes de terminar el lab y luego quieras partir sin perder minutos en configuración.

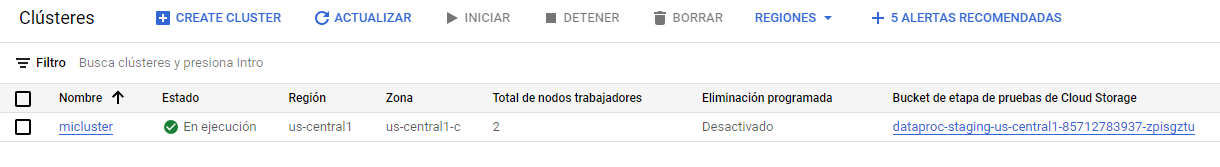


Cambiamos el master y los nodos a los más pequeños:



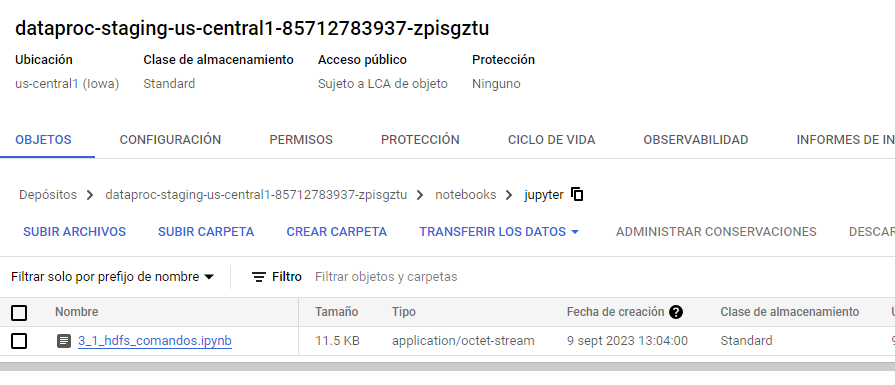


Luego presiona **CREAR** para crear el cluster. Deberías ver luego de unos minutos el cluster en estado Running, como en la siguiente figura.



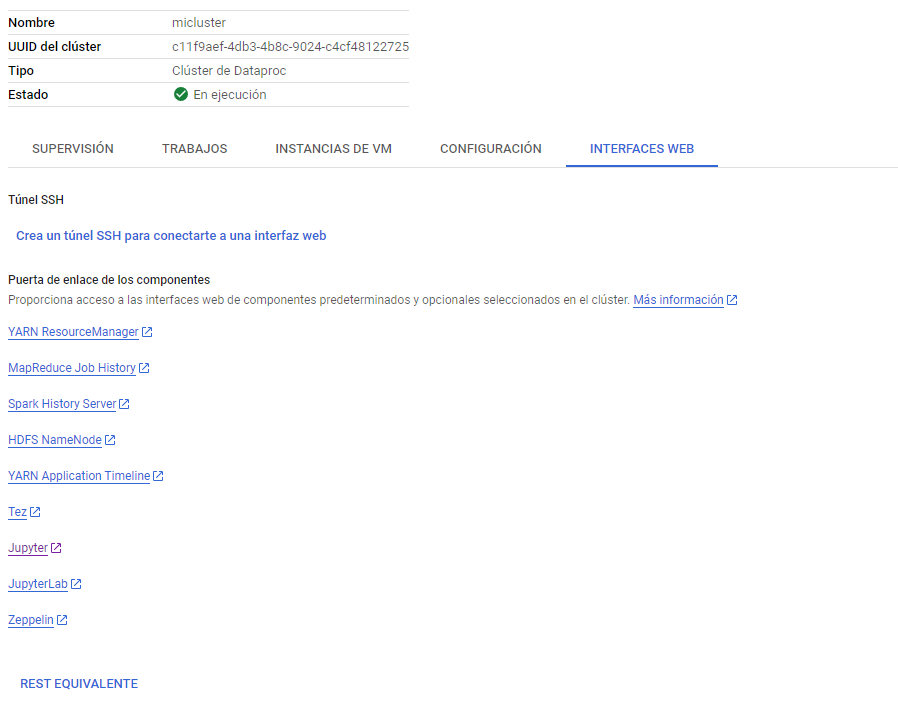
## **Usando software instalado en Dataproc (Jupyter)**

Este notebook tenemos que almacenarlo en el bucket que usa Cloud Dataproc, lo puedes encontrar entre los datos del cluster:

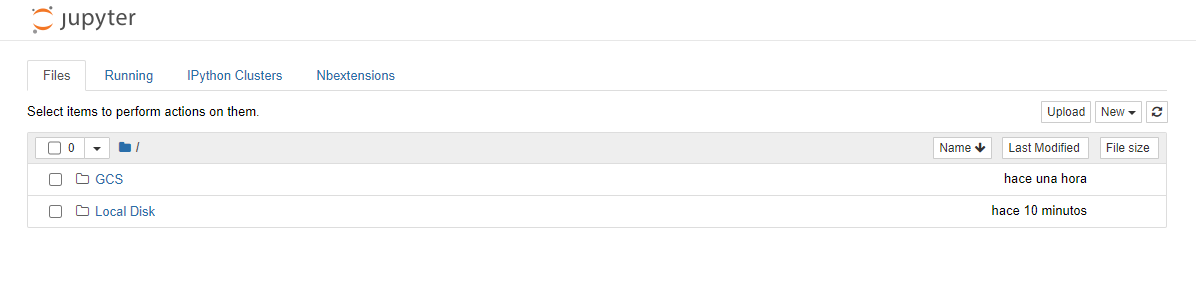


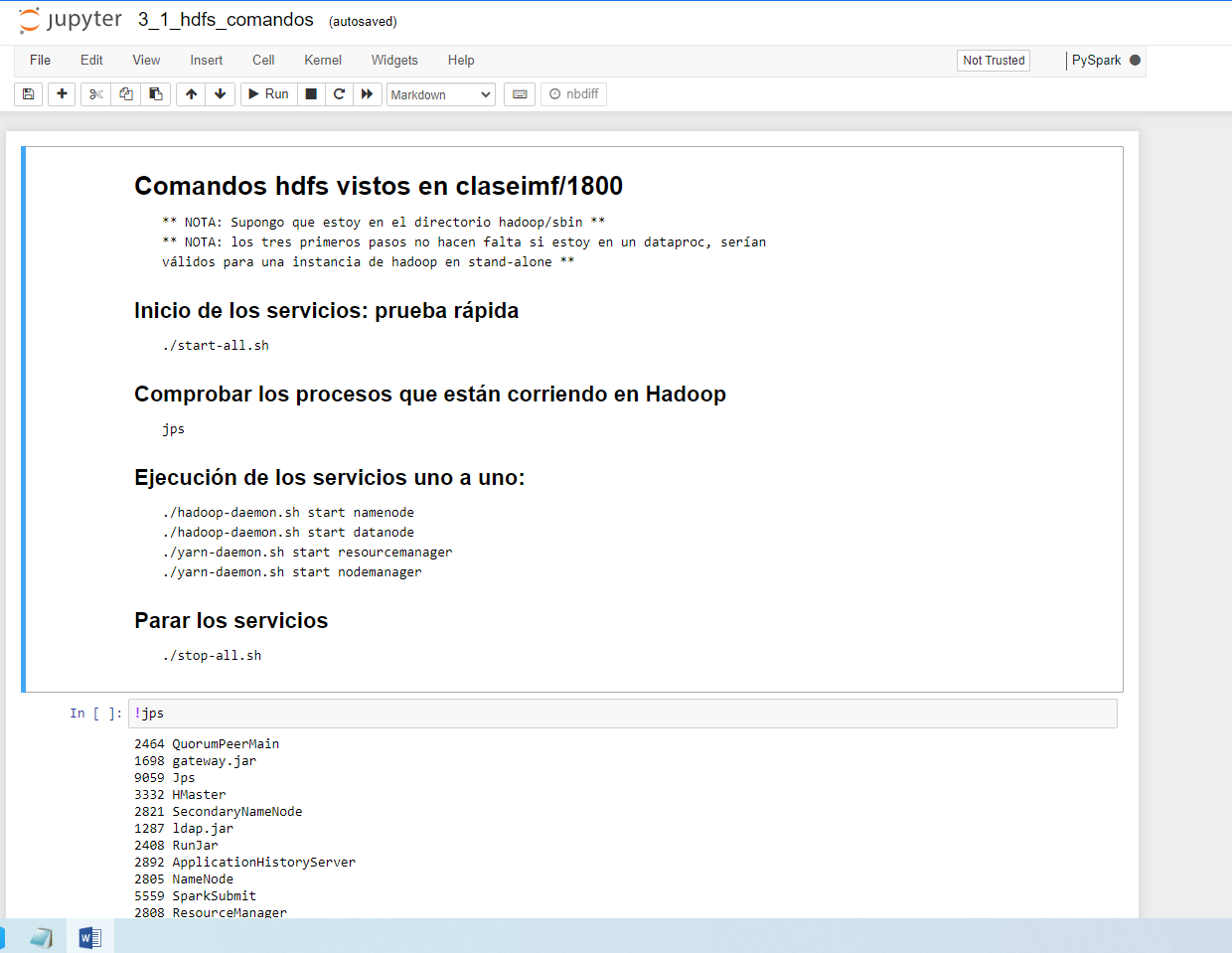
Subir el fichero 3\_1\_hdfs\_comandos.ipynb.

Sabemos que el cluster tiene instalado Hadoop, Spark, entre otros, de hecho tenemos acceso a su interfaz gráfica al entrar a los detalles del cluster, en **Web Interfaces > Component gateway**. Haz click sobre el que dice **Jupyter**.



Te aparecerá algo como, dado que se el contenido de la carpeta notebooks/jupyter que está en Cloud Storage y tiene el notebook que acabas de subir:

Haz click en el notebook dentro de GCS para abrirlo.



/home/javier\_gonzalez\_mendez/imf

mkdir imf

gsutil cp gs://dataproc-staging-us-central1-85712783937-zpisgztu/mapper.py mapper.py

gsutil cp gs://dataproc-staging-us-central1-85712783937-zpisgztu/reducer.py reducer.py

gsutil cp gs://dataproc-staging-us-central1-85712783937-zpisgztu/input.txt input.txt

cat input.txt | python3 mapper.py | sort | python3 reducer.py

mapred streaming -files mapper.py,reducer.py -mapper "python3 mapper.py" -reducer "python3 reducer.py" -input **hdfs:///**user/dataproc/input.txt -output **hdfs:///**user/dataproc/salida.txt

hdfs dfs -ls /user/dataproc

hdfs dfs -cat /user/dataproc/salida.txt

Claro, aquí tienes un ejemplo simple de un trabajo MapReduce en Python que puedes ejecutar en Hadoop en Google Cloud Platform (GCP). Este ejemplo cuenta la cantidad de palabras en un conjunto de documentos de texto. Para ejecutarlo, sigue estos pasos paso a paso:

**Paso 1: Preparación**

Asegúrate de tener un clúster de Hadoop configurado en GCP. Puedes utilizar Google Dataproc para crear un clúster si aún no lo tienes.

**Paso 2: Preparación de datos**

Coloca tus archivos de entrada en Google Cloud Storage (GCS) o en HDFS, según lo que esté configurado en tu clúster. Por ejemplo, puedes tener un archivo de texto llamado **input.txt** en GCS.

**Paso 3: Creación de los scripts de Python**

Crea dos scripts de Python: uno para el mapper y otro para el reducer.

* **mapper.py**:

pythonCopy code

#!/usr/bin/env python import sys # Leer la entrada desde la corriente estándar for line in sys.stdin: # Dividir la línea en palabras words = line.strip().split() # Emitir pares clave-valor (palabra, 1) para cada palabra for word in words: print(f"{word}\t1}")

* **reducer.py**:

pythonCopy code

#!/usr/bin/env python import sys current\_word = None word\_count = 0 # Leer la entrada desde la corriente estándar for line in sys.stdin: word, count = line.strip().split('\t') if current\_word == word: word\_count += int(count) else: if current\_word: # Emitir el resultado (palabra, contador) cuando cambia la palabra print(f"{current\_word}\t{word\_count}") current\_word = word word\_count = int(count) # Asegurarse de emitir el último resultado if current\_word: print(f"{current\_word}\t{word\_count}")

**Paso 4: Cargar los scripts en GCS**

Sube los scripts de Python a Google Cloud Storage para que estén disponibles para tu clúster de Hadoop:

bashCopy code

gsutil cp mapper.py gs://ruta/a/mapper.py gsutil cp reducer.py gs://ruta/a/reducer.py

Reemplaza **gs://ruta/a/** con la ruta adecuada en GCS.

**Paso 5: Ejecutar el trabajo MapReduce**

Ejecuta el trabajo MapReduce en tu clúster de Hadoop utilizando el comando **hadoop-streaming**:

bashCopy code

hadoop jar $HADOOP\_HOME/share/hadoop/tools/lib/hadoop-streaming-\*.jar \ -files gs://ruta/a/mapper.py,gs://ruta/a/reducer.py \ -mapper 'python mapper.py' \ -reducer 'python reducer.py' \ -input gs://ruta/de/entrada/input.txt \ -output gs://ruta/de/salida

Asegúrate de reemplazar **gs://ruta/a/mapper.py**, **gs://ruta/a/reducer.py**, **gs://ruta/de/entrada/input.txt** y **gs://ruta/de/salida** con las rutas y nombres de archivo adecuados.

**Paso 6: Ver resultados**

Puedes ver los resultados en GCS o HDFS, según donde hayas configurado la ruta de salida. Por ejemplo:

bashCopy code

gsutil cat gs://ruta/de/salida/part-\*

Este comando mostrará el conteo de palabras en los documentos de entrada.

Este es un ejemplo simple de un trabajo MapReduce en Python en Hadoop en GCP. Puedes adaptar este ejemplo para realizar análisis más complejos según tus necesidades.



y un ejemplo del fichero input.txt

ChatGPT

Claro, aquí tienes un ejemplo simple del contenido que podría tener el archivo **input.txt** utilizado en el ejemplo anterior. Supongamos que tienes un conjunto de documentos de texto en los que deseas contar la cantidad de palabras:

**input.txt:**

Copy code

Este es un ejemplo de un documento de texto. Este es otro ejemplo de documento. Cada documento tiene varias palabras. Las palabras pueden repetirse en los documentos.

En este ejemplo, el archivo **input.txt** contiene varios párrafos de texto. El trabajo MapReduce procesaría este archivo y contaría la cantidad de veces que aparece cada palabra en todos los documentos. El resultado sería un conteo de palabras y su frecuencia en el conjunto de documentos.

Ten en cuenta que en un escenario real, tu archivo **input.txt** podría ser mucho más grande y contener una variedad de texto. El objetivo del trabajo MapReduce es analizar y procesar estos datos de manera eficiente para obtener información útil.