Laboratorio 2

## Presentado por: Nicolas Morales Galindo, Juan Sebastian Sanchez

# Objetivo del Laboratorio

El objetivo principal de este laboratorio es comparar el desempeño de diferentes librerías de Python utilizadas para la carga y manipulación de datos tabulares. Las librerías en cuestión son Pandas, Polars, Spark y Dask. Se busca identificar cuál de estas librerías es más eficiente en términos de manejo de memoria y capacidad de procesamiento de datos en Google Colaboratory.

# Introducción

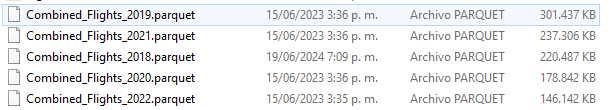
En este laboratorio se evaluará la eficiencia en el procesamiento básico de datos tabulares utilizando las librerías mencionadas. El propósito es determinar cuál puede manejar la mayor cantidad de datos sin saturar la memoria RAM disponible y evitar reinicios del entorno de Google Colaboratory. Los estudiantes deberán experimentar con cada librería y el conjunto de datos propuesto para encontrar la más eficiente en términos de carga de datos y uso de memoria.

# Desarrollo

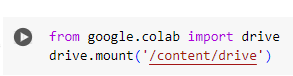
Librería Panda

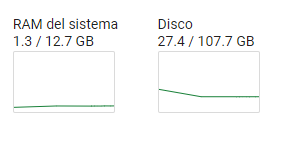
en primer lugar probaremos nuestra carga de archivos con la librería panda.

* A continuación podemos observar el peso de los archivos con los cuales realizaremos la carga de la información, donde el archivo mas pesado que podemos encontrar es el del año 2019 y el menos pesado es el del año 2022.

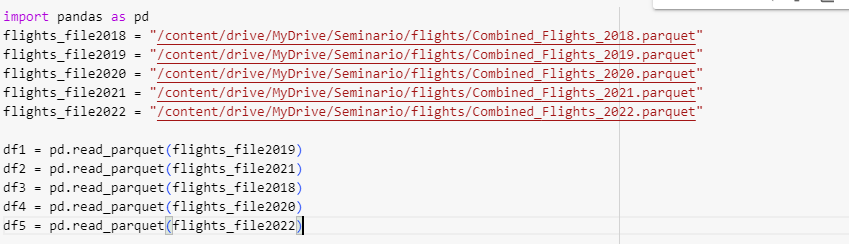


* Al crear el primer cuaderno en colab y ejecutar el primer comando podemos observar que la instancia de nuestro ambiente inicia con 1.3 de RAM y un disco de 27.4.



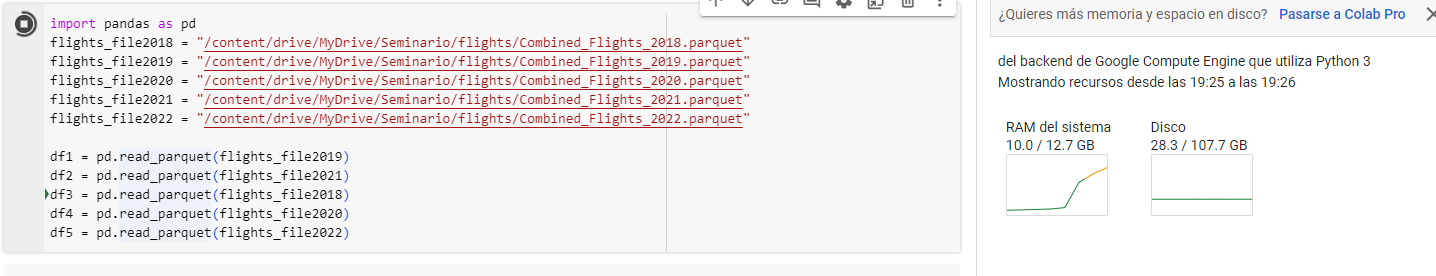


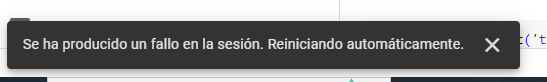
* Inicialmente procedimos a organizar un poco el codigo, dejamos que la lectura de los archivos parquet se ejecuten desde el archivo mas pesado hasta el menos pesado para lograr un mejor analisis y comparación.



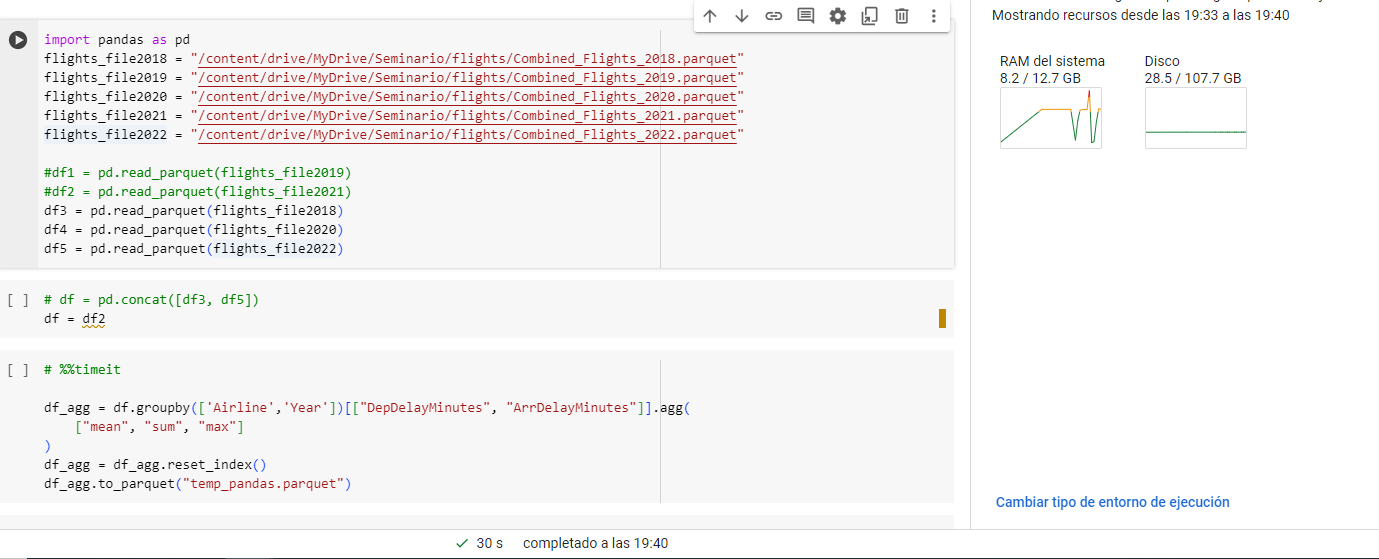
* Ejecutamos el codigo anterior y podemos evidenciar que logro leer con éxito los archivos 2019 y 2021, en la lectura del archivo de 2018 se produce un fallo en la sesión.

El total del peso de esos dos archivos que se leyeron con éxito es de: 537.743 KB usando 10 de RAM en un tiempo de 45 segundos

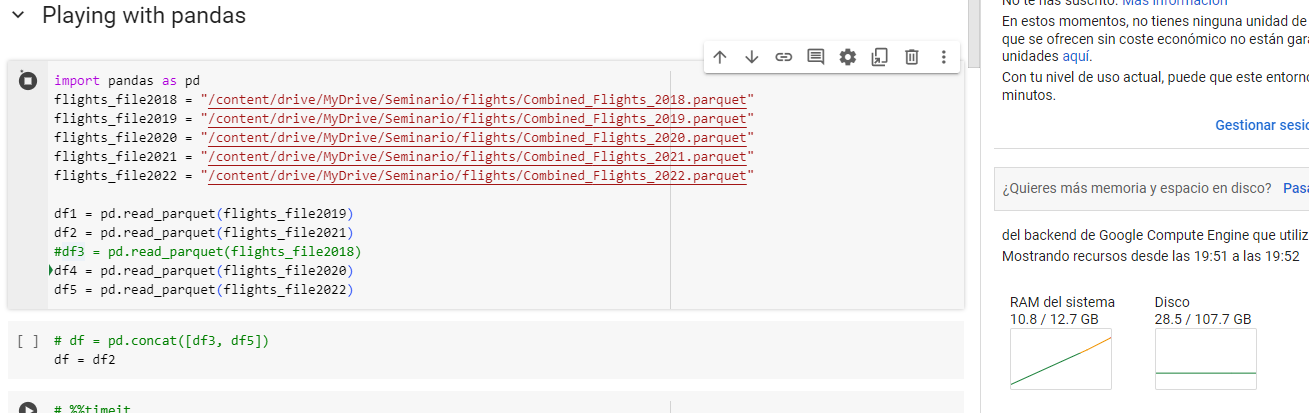




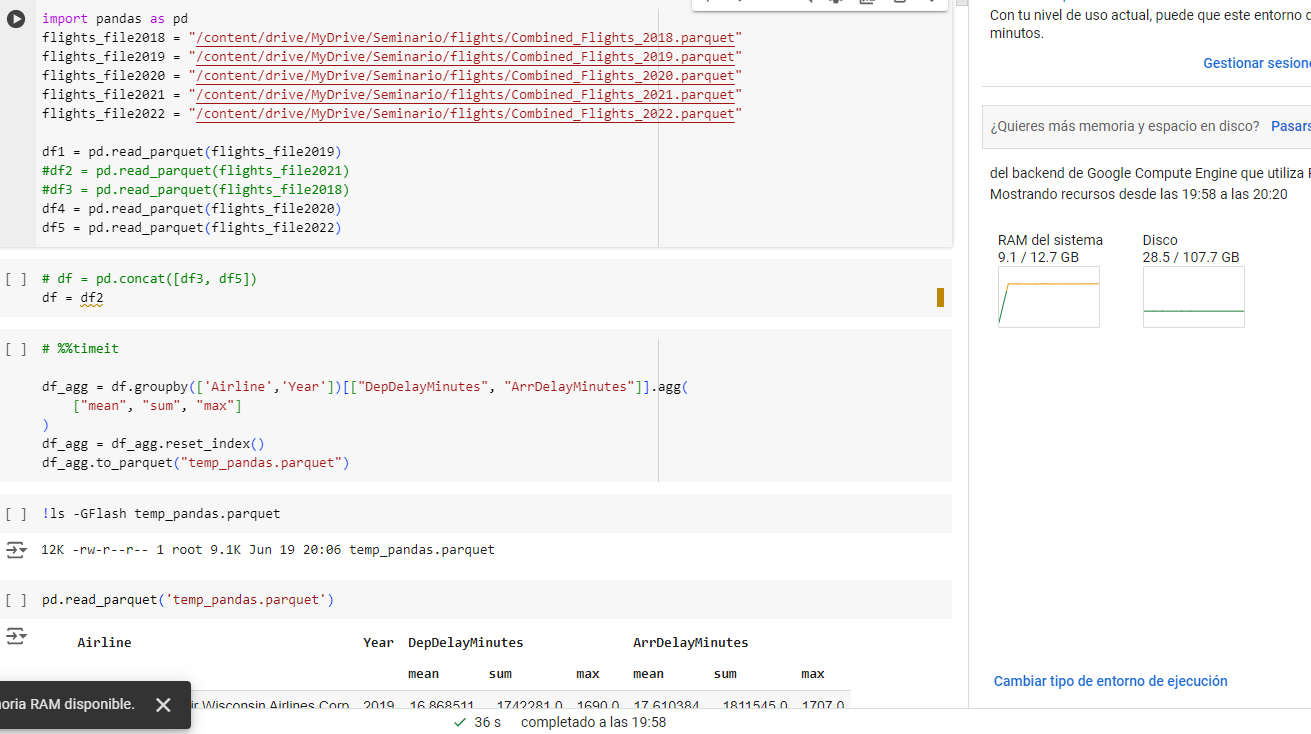
* Ahora la prueba que vamos a realizar es la carga de los tres archivos con menos peso en este caso son los del año 2018, 2020, 2022, en donde se termina la lectura con éxito en un lapso e 30 segundos, con un uso de 8.2 de RAM. El peso total de los archivos leidos es de: 545.471 KB.



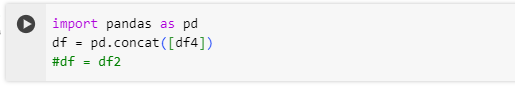
* Ahora lo que hicimos fue ejecutar todas las lecturas de los archivos excepto el archivo del 2018 y obtenemos un error cuando se ejecuta la lectura del archivo 2020 por uso de ram.  
  La lectura termina con un uso de RAM 10.8, a los 50 segundos y un peso de los archvos que se leyeron con éxito: 537.743 KB



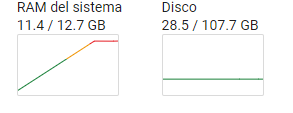
* Ahora lo que hicimos fue ejecutar todas las lecturas de los archivos excepto el archivo del 2021 y 2018 y conseguimos un resultado exitoso.  
  La lectura termina con un uso de RAM 9.1, a los 36 segundos y un peso de los archvos que se leyeron con éxito: 626.421 KB.

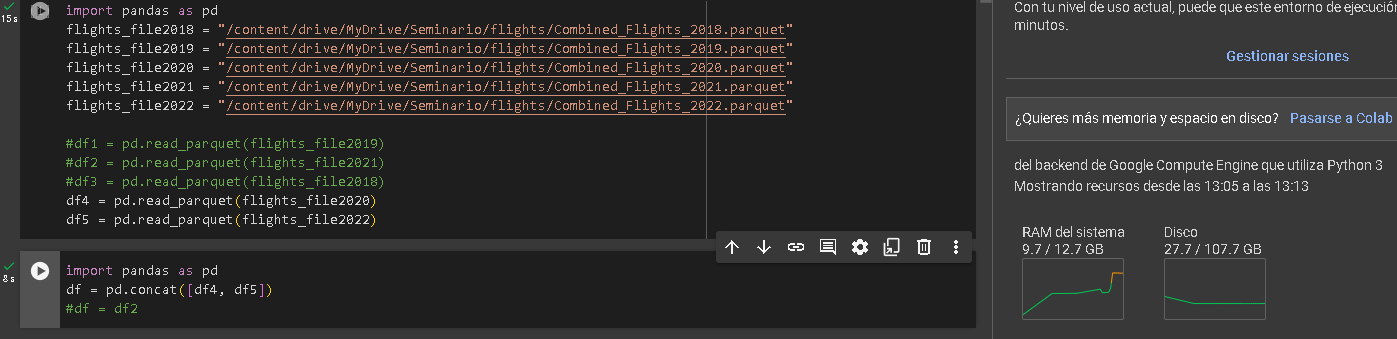


Es la unica combinacion que hasta el momento se ha podido hacer la lectura de tres archivos.

* Leyendo los tres archivos anteriores solo logramos completar la concatenacion con la variable “df4”:  
  

Con los demas sale error por exceso en el uso de la ram. La información de uso de recuros se presenta a continuación.

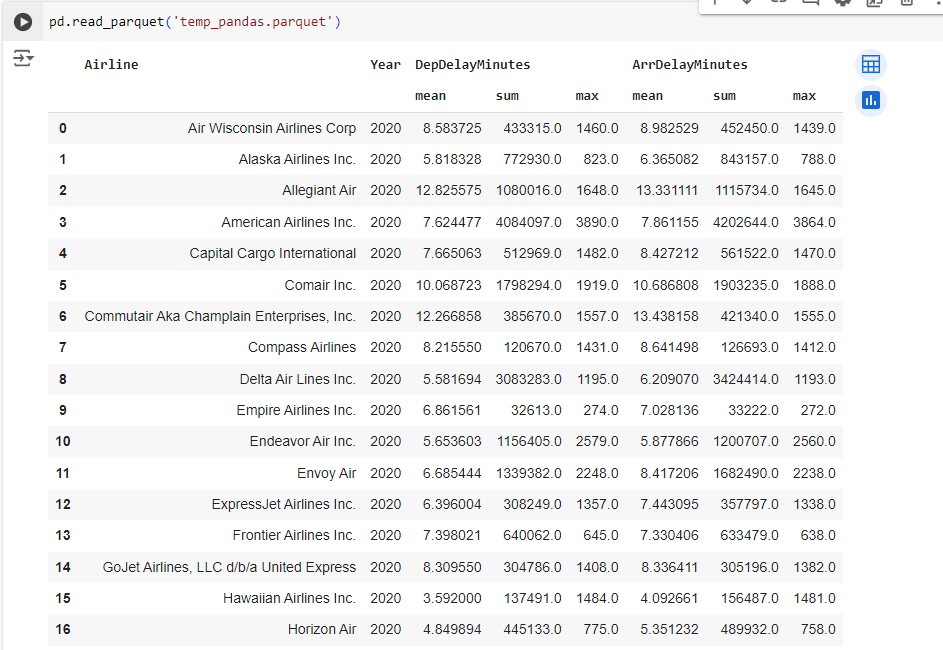


Al leer los archivos que menos pesan y concatenarlos obtenemos un resultado exitoso con un tiempo en la concatenacion de los archivos de 7 segundos: 

* Ya en el siguiente comando :  
  

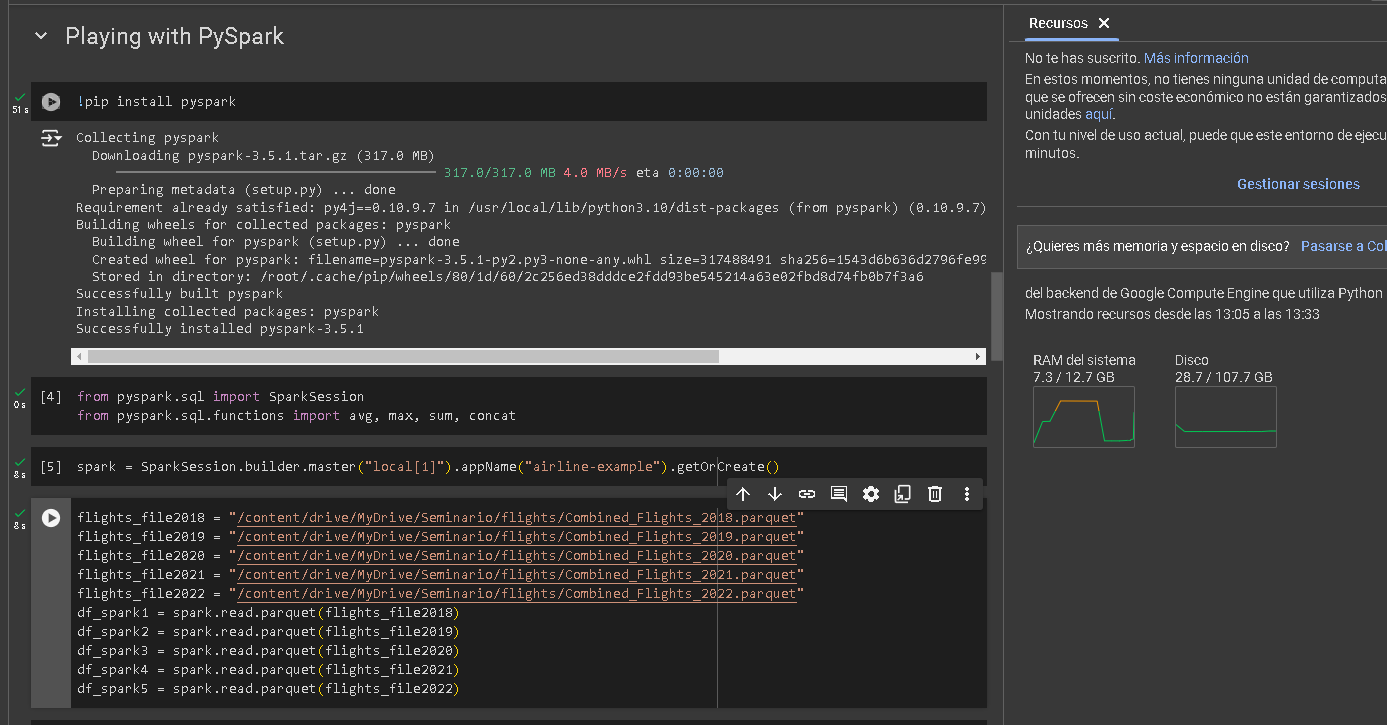
Se ejecuta en un segundo y la ram queda en 11.4.

Al ejecutar la lectura de la agrupacion obtenemos lo siguiente, en un tiempo de 1 segundo.

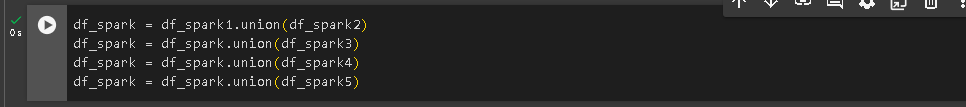


Librería PySpark

* A continuación, vamos a iniciar con el uso de la librería PySpark, el la imagen de a continuación veremos la ejecución de la lectura de los archivos en donde podemos observar que la ejecución de todos los archivos se completo en 8 segundos con un uso de memoria RAM de 7.3.

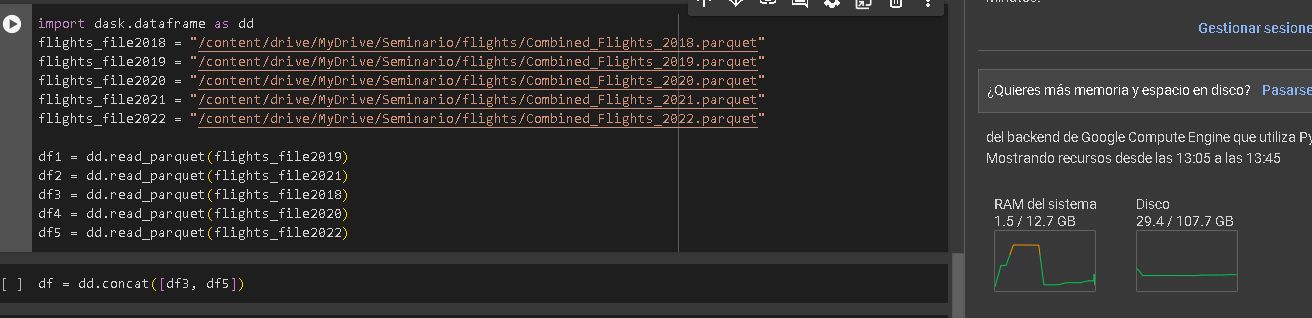


* Una vez finalizada la lectura se realiza la concatenación de los resultados la cual se logra completar en 1 segundo.

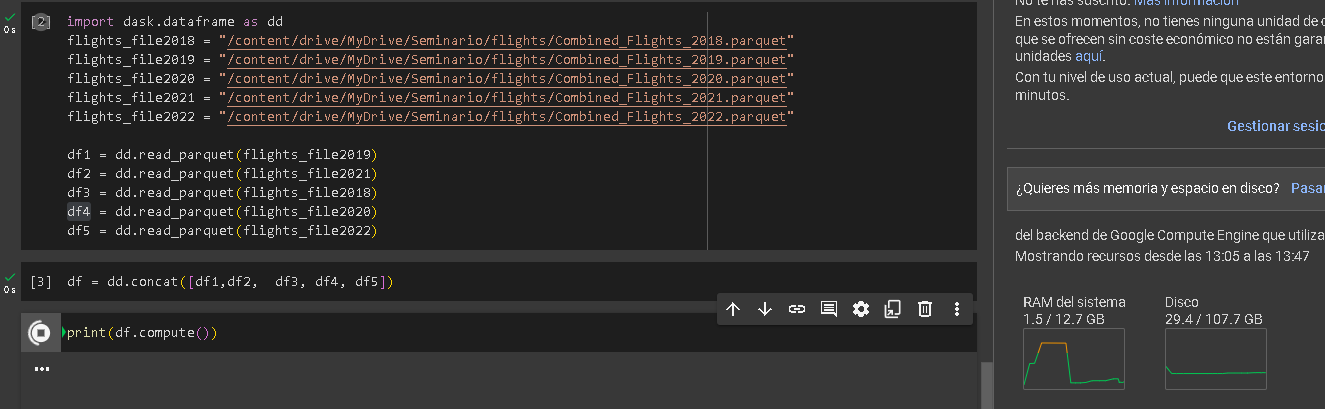


Librería Dask

* A continuación procederemos en probar la librería dask con la lectura de todos los archivos podemos obtener los siguientes datos: se completan la lectura de todos los archivos en 1 segundo con un uso de RAM de 1.5.

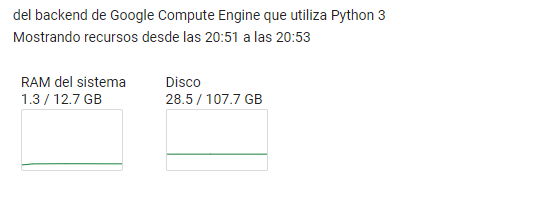


* La concatenación de todos los resultados de las lecturas se logra completar en 1 segundos con un uso de RAM de 1.5.

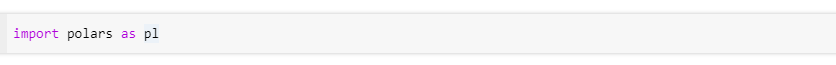


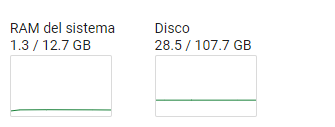
Librería Polar

* Al momento de arrancar google colab , el ambiente muestra que tiene ocupado 1.3 GB de ram y 28.5 GB de disco.

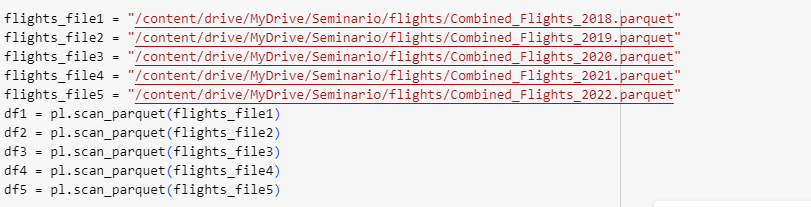


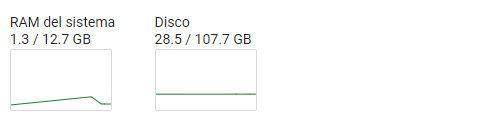
* Al importar la librería polar el ambiente tiene en uso 1.2 GB de ram y 27.4 GB disco.





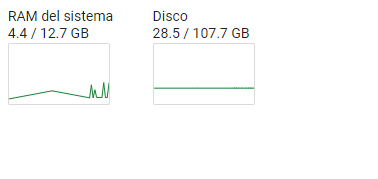
* Al momento de ejecutar esta porcion de codigo la cual obtiene el contenido de los archivos observamos que el ambiente tiene 1.3 GB de ram, y no presenta variación en el disco además de que la tarea la ha conseguido realizar en 0 segundos .





* Al intentar concatenar los archivos ya procesados, se observa un aumento en Ram , llegando a valores de 4.4 GB, el espacio de disco no se ha alterado y el proceso ha tomado 10 segundos en completarse.





# Conclusión y Recomendación

En este laboratorio, se evaluó el desempeño de las librerías Pandas, PySpark, Dask y Polars en términos de uso de memoria y tiempo de procesamiento de datos tabulares en Google Colaboratory.

# Librería Recomendada

Dask es la mejor librería según los datos obtenidos. Esta librería completó la lectura de todos los archivos en solo 1 segundo y usó la menor cantidad de RAM (1.5 GB), lo que la hace la más eficiente en términos de manejo de memoria y capacidad de procesamiento. Por lo tanto, Dask es la opción más recomendada para manejar grandes volúmenes de datos en Google Colaboratory.