

## LABORATORIO 2

**INTEGRANTES:** Galeano Tabares Daniel Felipe - Rivas Luna Andrés Felipe -Salas Barrera Jorge Andrés

## OBSERVACIONES

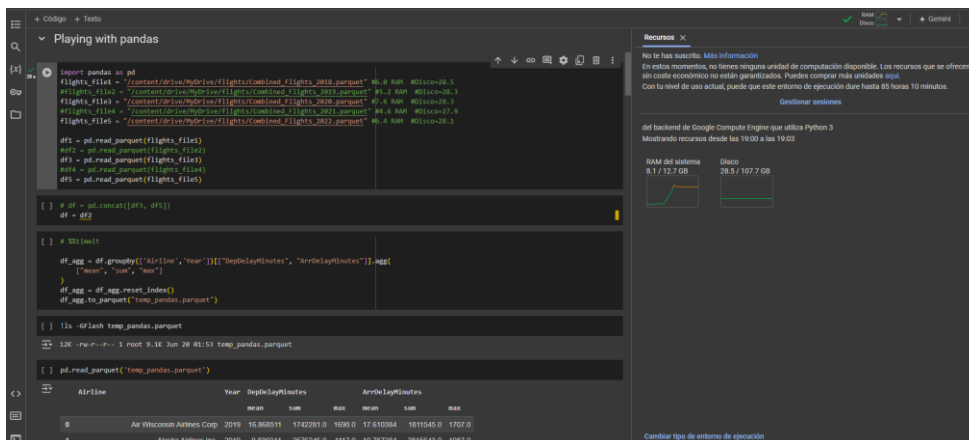
El informe compara el rendimiento de varias herramientas de procesamiento de datos: Pandas, PySpark, Polars y Dask. Pandas demostró ser la menos eficiente, con tiempos de procesamiento entre 19 y 30 segundos y un uso de RAM de 6.6 GB a 10.2 GB. PySpark mostró una mejora notable, completando la prueba en 8 segundos y utilizando 4.6 GB de RAM. Polars y Dask destacaron por su rapidez, completando las pruebas en solo 1 segundo; Dask utilizó un poco más de RAM (6.7 GB) en comparación con Polars (4.2 GB). En cuanto al espacio en disco, las diferencias fueron menores: Pandas y Polars usaron 28.5 GB, mientras que PySpark y Dask usaron 29.4 GB. Estas observaciones resaltan las diferencias en eficiencia y rendimiento entre las herramientas evaluadas.

## EVIDENCIAS

### Pandas

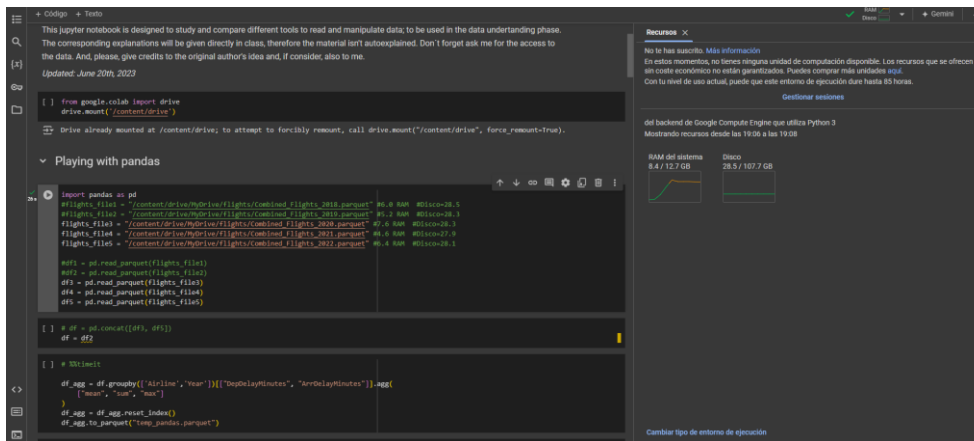
#### Prueba 1:

- **Archivos:** File1 (2018), File3 (2020), File5 (2022)
- **Tiempo:** 28 segundos
- **RAM:** 8.1 GB
- **Espacio en Disco:** 28.5 GB



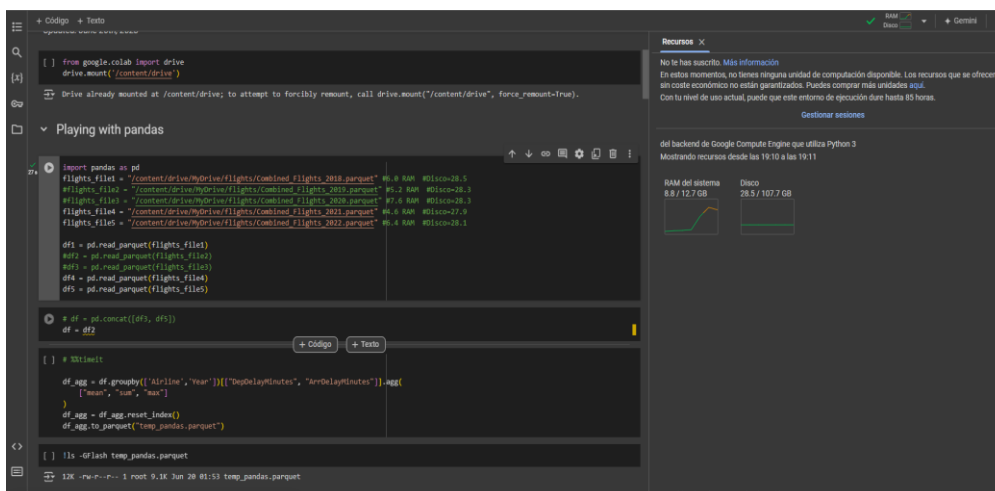
## Prueba 2:

- **Archivos:** File3 (2020), File4 (2021), File5 (2022)
- **Tiempo:** 26 segundos
- **RAM: 8.4 GB**
- **Espacio en Disco:** 28.5 GB



## Prueba 3:

- **Archivos:** File1 (2018), File4 (2021), File5 (2022)
- **Tiempo:** 27 segundos
- **RAM: 8.8 GB**
- **Espacio en Disco:** 28.5 GB



## Prueba 4:

- **Archivos:** File1 (2018), File2 (2019)
- **Tiempo:** 30 segundos

- **RAM: 10.2 GB**

Author: Elias Baltago Bolivar  
Inspired in: <https://www.youtube.com/watch?v=m9F9xChwM8>

Original data: Kaggle  
This Jupyter notebook is designed to study and compare different tools to read and manipulate data; to be used in the data understanding phase. The corresponding explanations will be given directly in class, therefore the material isn't autoexplained. Don't forget ask me for the access to the data. And, please, give credits to the original author's idea and, if consider, also to me.  
Updated: June 20th, 2023

```
[ ] from google.colab import drive
drive.mount('/content/drive')

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

Playing with pandas

```
import pandas as pd
flights_file1 = "/content/drive/MyDrive/Flights/Combined_flights_2018.parquet" #6.8 RAM #0.1s=28.5
flights_file2 = "/content/drive/MyDrive/Flights/Combined_flights_2019.parquet" #6.2 RAM #0.1s=28.3
flights_file3 = "/content/drive/MyDrive/Flights/Combined_flights_2020.parquet" #7.8 RAM #0.1s=28.3
flights_file4 = "/content/drive/MyDrive/Flights/Combined_flights_2021.parquet" #4.8 RAM #0.1s=27.9
flights_file5 = "/content/drive/MyDrive/Flights/Combined_flights_2022.parquet" #6.4 RAM #0.1s=28.3

df1 = pd.read_parquet(flights_file1)
df2 = pd.read_parquet(flights_file2)
df3 = pd.read_parquet(flights_file3)
df4 = pd.read_parquet(flights_file4)
df5 = pd.read_parquet(flights_file5)

[ ] # df = pd.concat([df1, df2])
df = df3

[ ] # df1=df5
```

Resources X

No te has suscrito. Más información  
En este momento, no tienes ninguna unidad de computación disponible. Los recursos que se ofrecen sin coste económico no están garantizados. Puedes comprar más unidades aquí.  
Con tu nivel de uso actual, puede que este entorno de ejecución dure hasta 84 horas 50 minutos.  
Gestionar sesiones

del backend de Google Compute Engine que utiliza Python 3  
Mostrando recursos desde las 19:22 a las 19:23

RAM del sistema 10.2 / 12.7 GB Disco 28.5 / 107.7 GB

## Prueba 5:

- **Archivos:** File1 (2018), File3 (2020)
- **Tiempo:** 19 segundos
- **RAM: 6.6 GB**

Author: Elias Baltago Bolivar  
Inspired in: <https://www.youtube.com/watch?v=m9F9xChwM8>

Original data: Kaggle  
This Jupyter notebook is designed to study and compare different tools to read and manipulate data; to be used in the data understanding phase. The corresponding explanations will be given directly in class, therefore the material isn't autoexplained. Don't forget ask me for the access to the data. And, please, give credits to the original author's idea and, if consider, also to me.  
Updated: June 20th, 2023

```
[ ] from google.colab import drive
drive.mount('/content/drive')

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

Playing with pandas

```
import pandas as pd
flights_file1 = "/content/drive/MyDrive/Flights/Combined_flights_2018.parquet" #6.8 RAM #0.1s=28.5
flights_file2 = "/content/drive/MyDrive/Flights/Combined_flights_2019.parquet" #6.2 RAM #0.1s=28.3
flights_file3 = "/content/drive/MyDrive/Flights/Combined_flights_2020.parquet" #7.8 RAM #0.1s=28.3
flights_file4 = "/content/drive/MyDrive/Flights/Combined_flights_2021.parquet" #4.8 RAM #0.1s=27.9
flights_file5 = "/content/drive/MyDrive/Flights/Combined_flights_2022.parquet" #6.4 RAM #0.1s=28.3

df1 = pd.read_parquet(flights_file1)
df2 = pd.read_parquet(flights_file2)
df3 = pd.read_parquet(flights_file3)
df4 = pd.read_parquet(flights_file4)
df5 = pd.read_parquet(flights_file5)

[ ] # df = pd.concat([df1, df2])
df = df3

[ ] # df1=df5
```

Resources X

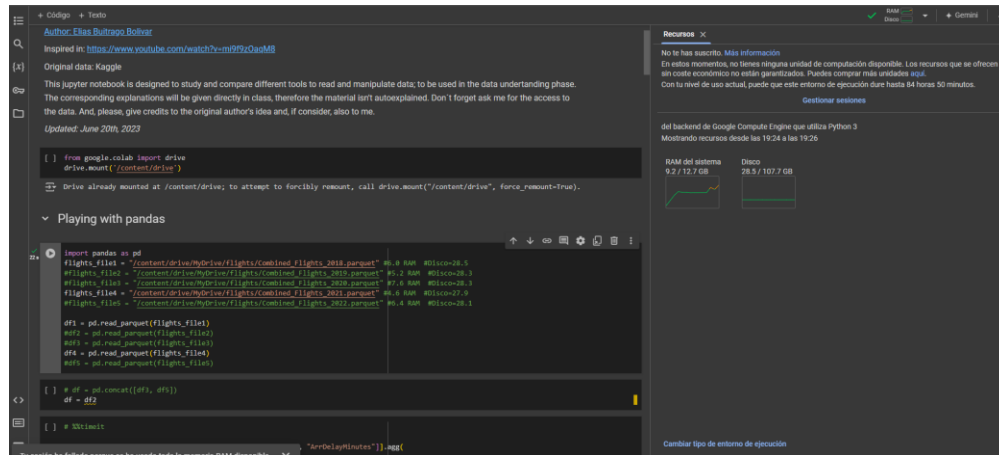
No te has suscrito. Más información  
En este momento, no tienes ninguna unidad de computación disponible. Los recursos que se ofrecen sin coste económico no están garantizados. Puedes comprar más unidades aquí.  
Con tu nivel de uso actual, puede que este entorno de ejecución dure hasta 84 horas 50 minutos.  
Gestionar sesiones

del backend de Google Compute Engine que utiliza Python 3  
Mostrando recursos desde las 19:24

RAM del sistema 6.6 / 12.7 GB Disco 28.5 / 107.7 GB

## Prueba 6:

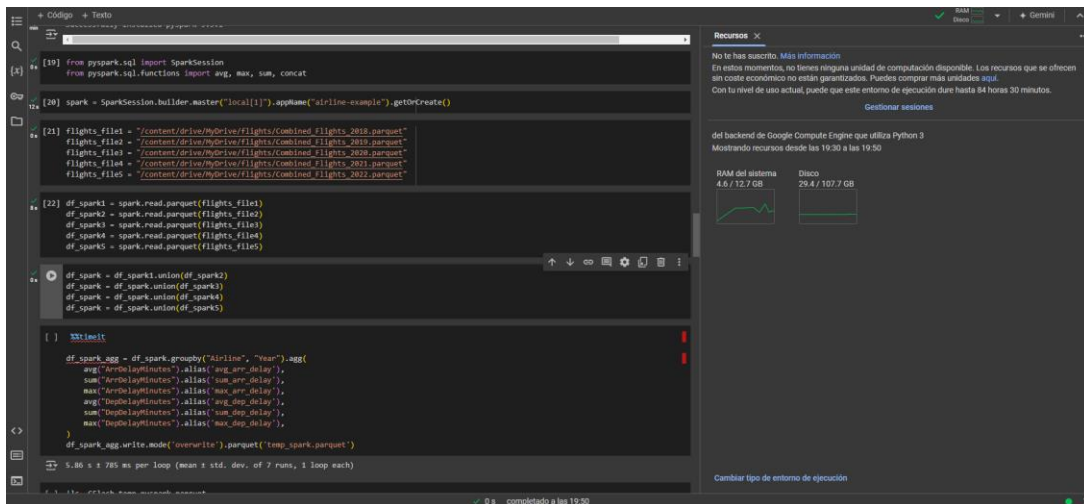
- **Archivos:** File1 (2018), File4 (2021)
- **Tiempo:** 22 segundos
- **RAM: 9.2 GB**



## PySpark

### Prueba 1:

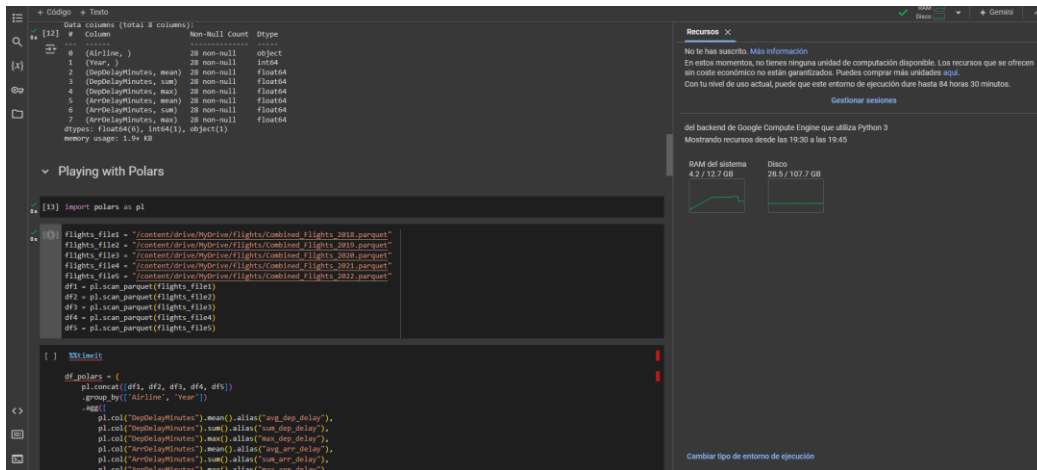
- **Archivos:** File1 (2018), File2(2019), File3 (2020), File (2021), File5 (2022)
- **Tiempo:** 8 segundos
- **RAM: 4.6 GB**
- **Espacio en Disco: 29.4 GB**



## Polars

### Prueba 1:

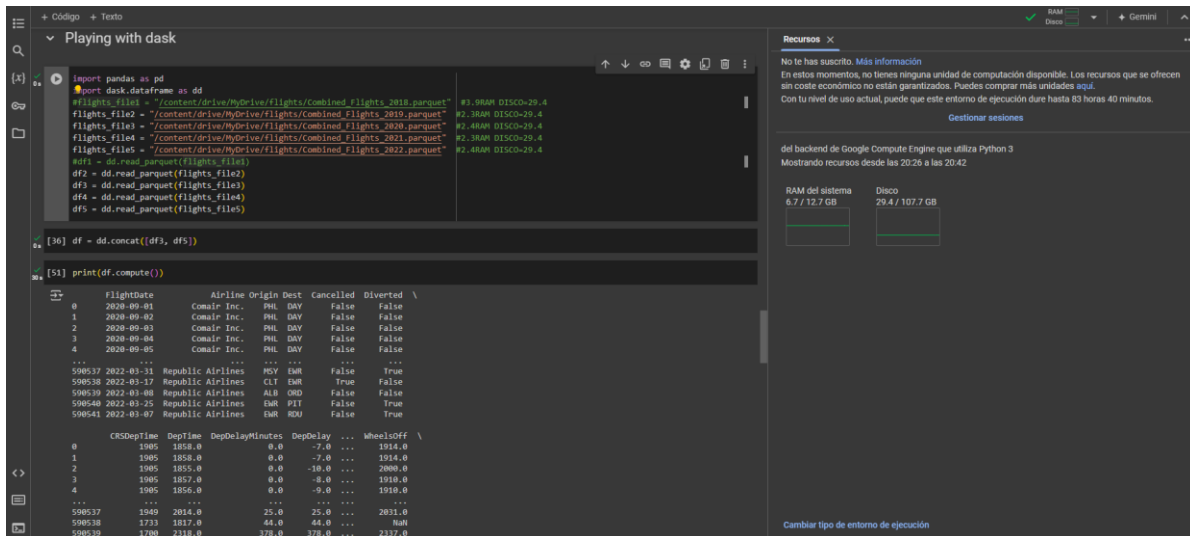
- **Archivos:** File1 (2018), File2(2019), File3 (2020), File4 (2021), File5 (2022)
- **Tiempo:** 1 segundo
- **RAM: 4.2 GB**
- **Espacio en Disco: 28.5 GB**



## Dask

### Prueba 1:

- **Archivos:** File2(2019), File3 (2020), File4 (2021), File5 (2022)
- **Tiempo:** 1 segundos
- **RAM:** 6.7 GB
- **Espacio en Disco:** 29.4 GB



## CONCLUSIONES

### **Polars y Dask:**

- Son las herramientas más eficientes para el procesamiento de datos.
- Completaron las pruebas en 1 segundo.
- Polars destacó por su uso más bajo de RAM (4.2 GB).

### **PySpark:**

- Mostró un rendimiento sólido.
- Completó la prueba en 8 segundos.
- Utilizó 4.6 GB de RAM, lo que lo convierte en una opción viable para manejar grandes volúmenes de datos con eficiencia.

### **Pandas:**

- Fue la menos eficiente.
- Tiempos de procesamiento más largos, entre 19 y 30 segundos.
- Mayor consumo de RAM, de 6.6 GB a 10.2 GB.
- Más adecuado para conjuntos de datos más pequeños o menos críticos en términos de tiempo.