https://pandas.pydata.org/pandas-docs/stable/userguide/io.html

https://pandas.pydata.org/pandas-docs/stable/user\_guide/io.html

* CSV - Es muy versatil ya que solo tiene comas y saltos de linea
* JSON - Tiene un formato muy similar al de un diccionario de Python
* Excel - Permite guardar el archivo en formato .xls para trabajar con el en Excel o Spreadsheets
* Pickle - Permite comprimir la información, es util cuando se tienen tablas grandes
* Parquet - Permite darle un formato que puede usarse en ambientes de Big Data como Hadoop
* Hadoop

dir\_pandas = './ColabFiles1'

df.to\_csv(f'{dir\_pandas}/test.csv')

df.to\_csv(dir\_pandas, sep = '|', index = False) #sin indice y con separador

df = pd.read\_csv(dir\_pandas.format('test.csv'))

df = pd.read\_csv('poblacion.csv', skiprows=4) #salte las primeras 4 filas

df.to\_excel(dir\_pandas.format('text.xlsx')

df.to\_excel(dir\_pandas.format('text.xlsx'),index=False, sheet\_name='Hoja 1'

df = pd.read\_excel('file\_name.xlsx', sheet\_name='Sheet1')

pd.read\_excel(dir\_pandas.format('test.xlsx'))

pd.read\_excel(dir\_pandas.format('test.xlsx')

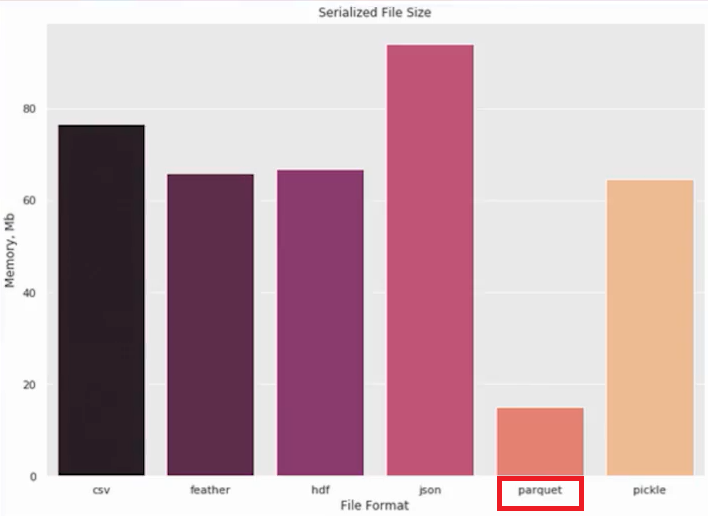
df.to\_json(dir\_pandas.format('text.json')

Formato binario

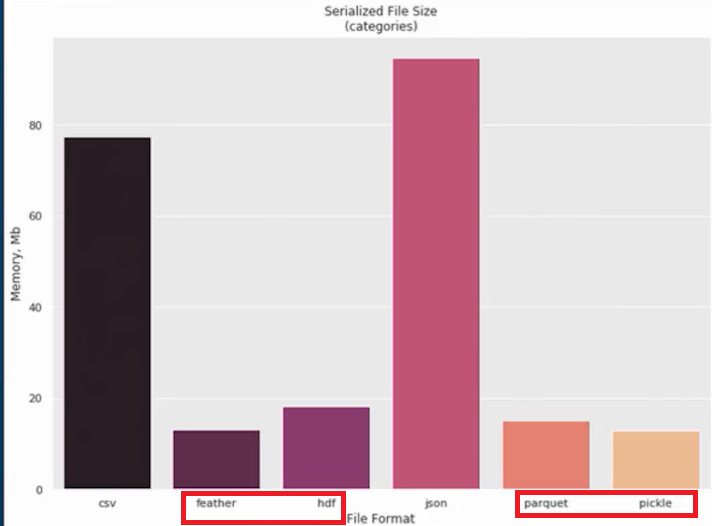
df.to\_pickle(dir\_pandas.format('text.pkl')

df.to\_parquet(dir\_pandas.format('text.parquet')

df.to\_hdf(dir\_pandas.format('text.hdf'),key='data',foamt='table')



SI CATEGORIZAMOS VARIABLES, tipo matrix con indices



CARGA Y LECTURA SON MAS RAPIDOS LOS BINARIOS: parquet, pickle, hdf, feather

EN RAM todos son eficientes, menos JSON

RAM cuando carga y lee son mejores los binaries, pero hdf es el menos eficiente