Parquet vs CSV Sacct Data Storage Benchmark

All benchmarks used an identical dataset: one week's worth of sacct user data ranging from June 1st of this year to the 8th, both at time 00:00:00. This week of data resulted in a dataframe with dimensions 241,969 rows x 75 columns.

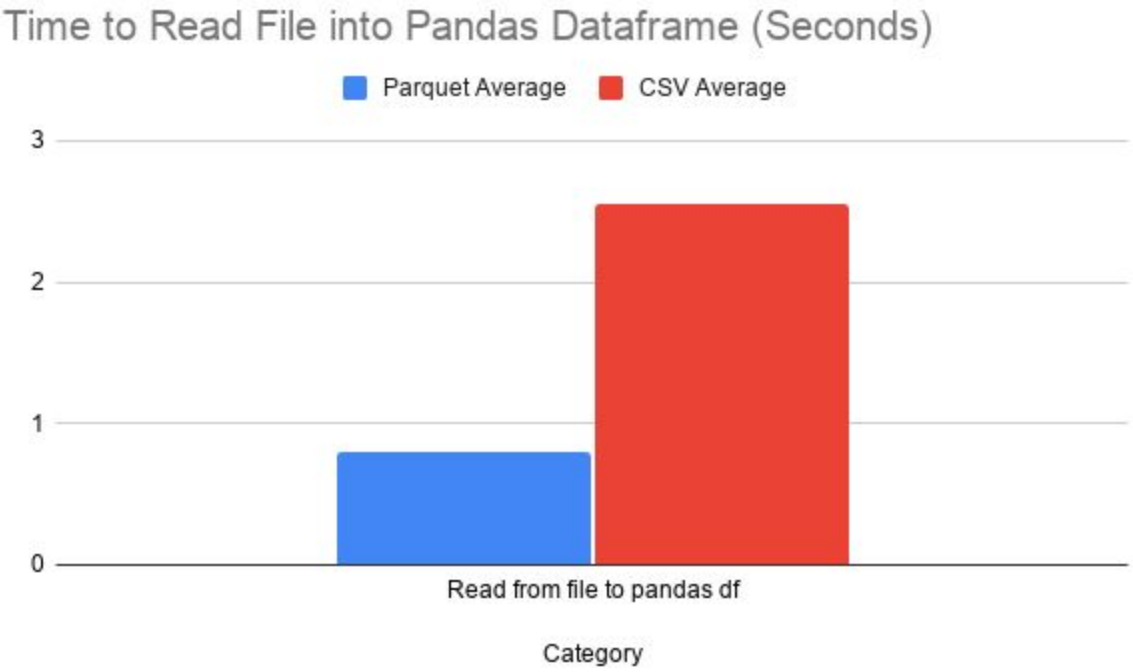
The three metrics measured in the benchmark were as follows:

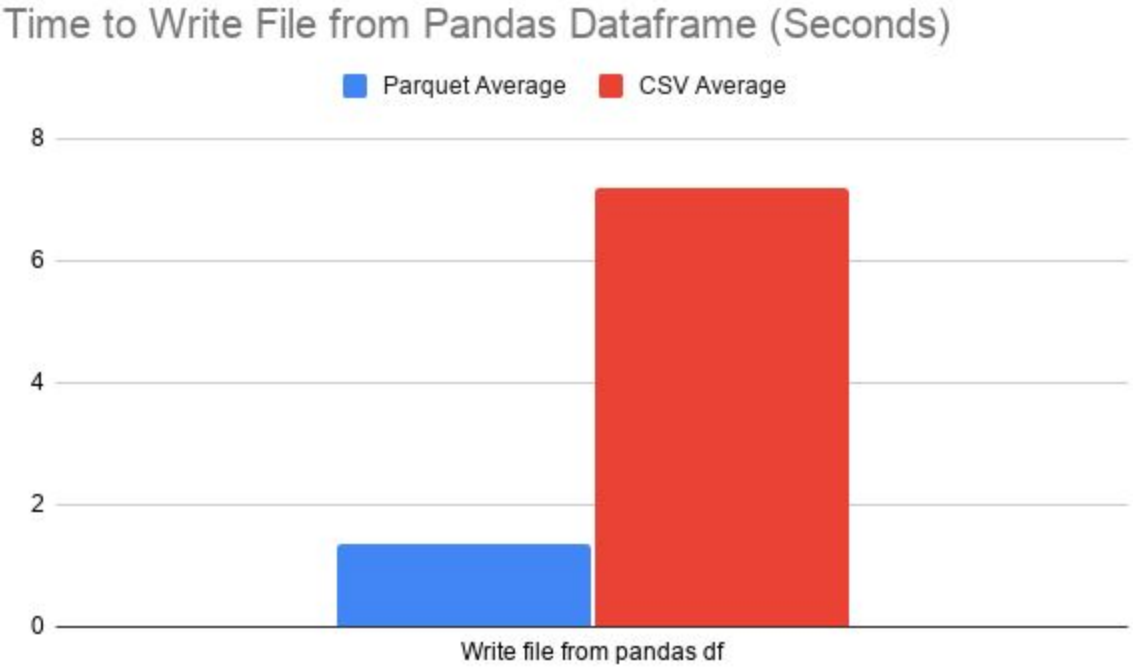
- Time to read file type into a pandas dataframe
- Time to write a file of that type from a pandas dataframe
- Total disk usage of final file

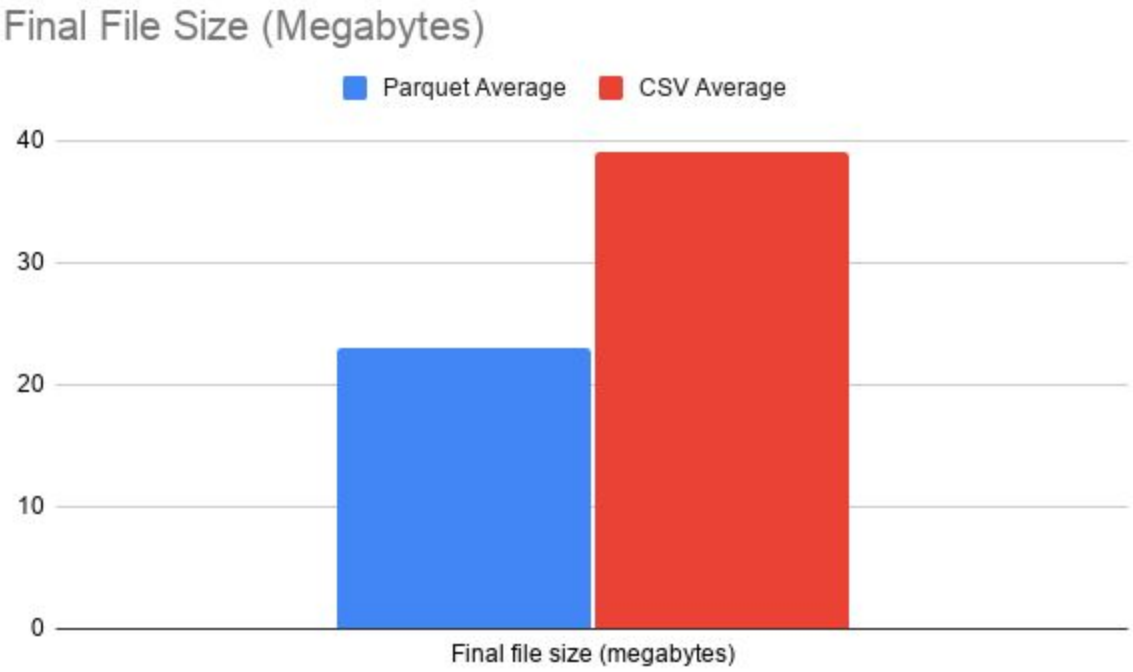
All tests were run twice to hopefully account for and/or catch any errors/outliers in the data. The information was generally very uniform, with only very small differences between attempts.

On average, the final file size when saved as a CSV was approximately 1.7 times larger than the parquet format. The time to read a CSV to a pandas dataframe was also about 3.2 times longer than when reading a parquet file. Finally, it took about 5.3 times longer to write an identical dataframe to a CSV than it did when writing to the parquet format. From what I've seen in general tests and benchmarks online, these numbers only seem to grow as the size of the dataset increases. There is also supposedly a massive advantage for parquet when it comes to querying the dataset, as you can directly and extremely efficiently query a parquet file without loading it into a pandas dataframe.

The complete results are seen in the following graphs:


Time to Read File into Pandas Dataframe (Seconds)
Parquet Average
CSV Average
3
2
1
0
Read from file to pandas df
Category


Time to Write File from Pandas Dataframe (Seconds)
Parquet Average
CSV Average
8
6
4
2
0
Write file from pandas df


Final File Size (Megabytes)
Parquet Average
CSV Average
40
30
20
10
0
Final file size (megabytes)