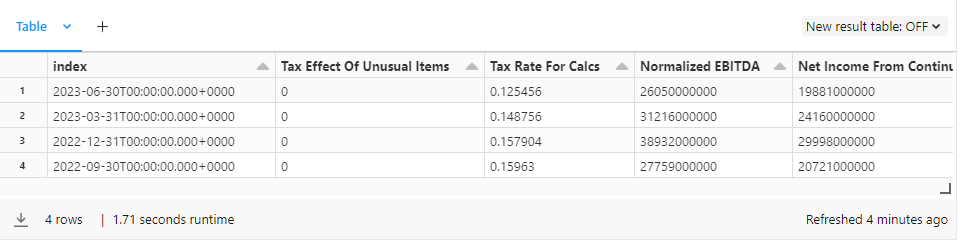
**Exploratory Data Analysis**

For my financial report data, the columns are included as below (obtained using the df.describe()) function on the Apple’s parquet file:

DataFrame[summary: string, Tax Effect Of Unusual Items: string, Tax Rate For Calcs: string, Normalized EBITDA: string, Net Income From Continuing Operation Net Minority Interest: string, Reconciled Depreciation: string, Reconciled Cost Of Revenue: string, EBITDA: string, EBIT: string, Net Interest Income: string, Interest Expense: string, Interest Income: string, Normalized Income: string, Net Income From Continuing And Discontinued Operation: string, Total Expenses: string, Total Operating Income As Reported: string, Diluted Average Shares: string, Basic Average Shares: string, Diluted EPS: string, Basic EPS: string, Diluted NI Availto Com Stockholders: string, Net Income Common Stockholders: string, Net Income: string, Net Income Including Noncontrolling Interests: string, Net Income Continuous Operations: string, Tax Provision: string, Pretax Income: string, Other Income Expense: string, Other Non Operating Income Expenses: string, Net Non Operating Interest Income Expense: string, Interest Expense Non Operating: string, Interest Income Non Operating: string, Operating Income: string, Operating Expense: string, Research And Development: string, Selling General And Administration: string, Gross Profit: string, Cost Of Revenue: string, Total Revenue: string, Operating Revenue: string]

These columns are enforced across every single financial report, which was something that wasn’t the case for my first choice of API for obtaining financial reports (SEC EDGAR API). For each of the parquet files, there are four rows that hold the 4 most recent 10-Q reports from the business.

Every column is stored as a string, so If I am going to do any summation or averaging between the rows or columns, I will have to cast them as their respective data type before operation. This is demonstrated by my practice operations on the data where I have to cast the net income and total revenue to an integer before doing any comparisons. ['BRK.B', 'BF.B', 'VLTO'] financial reports could not be resolved, meaning I will have to find something to display on those pages instead of the financial report from this API.



For my news and sentiment data, I was unable to use the df.describe method to describe the function, and I think that is attributed to the fact that there are nested data structures stored in each respective column. Instead, I simply took a screenshot of the display(df) output. Since there are different data structures in each column, I will have to iterate through these datastructures and make sure to cast when needed. Overall, enough information is given to give a brief summary of the article, a link to the article and a sentiment score to display on each of the ticker pages, which makes it perfect for my project.

['AJG', 'BRK.B', 'BF.B', 'CE', 'CHD', 'CMCSA', 'EQR', 'EL', 'FOXA', 'HBAN', 'LOW', 'MMC', 'MKC', 'MRK', 'MPWR', 'NDAQ', 'NWSA', 'NWS', 'TROW', 'USB']' could not resolve news and sentiment data through the API, meaning I will have to find something else to display on these webpages statically.

