```
# pip install yfinance pandas



import pandas as pd

import yfinance as yf



def extract_revenue(fin_df):

    """

    Turn yfinance financials (rows=line items, cols=periods) into a tidy

    Date/Revenue DataFrame, handling PeriodIndex or string dates robustly.

    """

    if fin_df is None or fin_df.empty:

        return None



    # Try common revenue row names

    for name in ["Total Revenue", "Revenue", "TotalRevenue"]:

        if name in fin_df.index:

            s = fin_df.loc[name].dropna()

            break

    else:

        return None  # no revenue row found
```

```
    # Handle date-like columns that might be PeriodIndex / Timestamp / str
    idx = s.index
    if isinstance(idx, pd.PeriodIndex):
        dates = idx.to_timestamp(how="end")
    else:
        dates = pd.to_datetime(idx, errors="coerce")



    out = pd.DataFrame(
        {
            "Date": dates,
            "Revenue": pd.to_numeric(s.values, errors="coerce"),
        }
    ).dropna(subset=["Date", "Revenue"]).sort_values("Date")



    return out



# --- Fetch from yfinance ---
tsla = yf.Ticker("TSLA")



# Try quarterly first, then fall back to annual — avoid "or" with DataFrames
tesla_revenue = extract_revenue(tsla.quarterly_financials)
```

```
if tesla_revenue is None or tesla_revenue.empty:
    tesla_revenue = extract_revenue(tsla.financials)



if tesla_revenue is None or tesla_revenue.empty:
    raise RuntimeError("Could not extract Tesla revenue from yfinance.")



# Display the last five rows
print(tesla_revenue.tail())
```