

Raw Data Sheet1				
	Unnamed: 0	Unnamed: 1	Unnamed: 2	Unnamed: 3
1				
2				
3			Source Head	Final Source
4			Digital	Facebook
5				Google
6				Affiliate

is there a problem with the data structure

Analyzed ^

```
python
Always show details  Copy

import pandas as pd

# Load the Excel file
file_path = "/mnt/data/Raw Data.xlsx"
xls = pd.ExcelFile(file_path)

# Display sheet names
xls.sheet_names

Result
['Sheet1']
```

Analyzed ^

```
python
Always show details  Copy

# Load the sheet into a DataFrame
df = pd.read_excel(xls, sheet_name='Sheet1')

# Display basic information about the data structure
df.info(), df.head()

STDOUT/STDERR
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 22 entries, 0 to 21
Data columns (total 18 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   Unnamed: 0    0 non-null     float64
 1   Unnamed: 1    0 non-null     float64
 2   Unnamed: 2    4 non-null     object  
 3   Unnamed: 3    20 non-null    object  
 4   Unnamed: 4    21 non-null    object  
 5   Unnamed: 5    20 non-null    object  
 6   Unnamed: 6    20 non-null    object  
 7   Unnamed: 7    20 non-null    object  
 8   Unnamed: 8    20 non-null    object  
 9   Unnamed: 9    20 non-null    object  
 10  Unnamed: 10   20 non-null    object  
 11  Unnamed: 11   20 non-null    object  
 12  Unnamed: 12   20 non-null    object  
 13  Unnamed: 13   20 non-null    object  
 14  Unnamed: 14   20 non-null    object  
 15  Unnamed: 15   20 non-null    object  
 16  Unnamed: 16   20 non-null    object  
 17  Unnamed: 17   20 non-null    object  
 18  Unnamed: 18   20 non-null    object  
 19  Unnamed: 19   20 non-null    object  
 20  Unnamed: 20   20 non-null    object  
 21  Unnamed: 21   20 non-null    object 
```

```

0    Unnamed: 0    20 non-null    object
1    Unnamed: 1    20 non-null    object
2    Unnamed: 2    15 non-null    object
3    Unnamed: 3    14 non-null    object
4    Unnamed: 4    14 non-null    object
5    Unnamed: 5    14 non-null    object
6    Unnamed: 6    16 non-null    object
7    Unnamed: 7    15 non-null    object
8    Unnamed: 8    15 non-null    object
9    Unnamed: 9    14 non-null    object
10   Unnamed: 10   14 non-null    object
11   Unnamed: 11   14 non-null    object
12   Unnamed: 12   16 non-null    object
13   Unnamed: 13   15 non-null    object
14   Unnamed: 14   14 non-null    object
15   Unnamed: 15   16 non-null    object
16   Unnamed: 16   15 non-null    object
17   Unnamed: 17   15 non-null    object
dtypes: float64(2), object(16)
memory usage: 3.2+ KB

```

Result

(None,

	Unnamed: 0	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	NaN	NaN	NaN	NaN	NaN
1	NaN	NaN	NaN	NaN	Qualified Leads
2	NaN	NaN	Source Head	Final Source	Week 1
3	NaN	NaN	Digital	Facebook	954.475219
4	NaN	NaN	NaN	Google	647.180758

	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8
0	NaN	NaN	NaN	NaN
1	NaN	NaN	NaN	Total landed conversions
2	Week 2	Week 3	Grand Total	Week 1
3	1104.302923	1039.506115	3098.284257	188.153061
4	808.931577	666.852979	2122.965314	172.030612

	Unnamed: 9	Unnamed: 10	Unnamed: 11	Unnamed: 12	Unnamed: 13	Unnamed: 14	Unnamed: 15	Unnamed: 16	Unnamed: 17
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	NaN	NaN	NaN	NaN	Cost	NaN	NaN	NaN	NaN
2	Week 2	Week 3	Grand Total	Week 1	Week 2	Week 3	Week 1	Week 2	Week 3
3	248.88574	240.413999	677.4528	226868	436133	4543	1205.763002	1752.342262	1889.665331
4	221.992914	196.982566	591.006092	361681	502500	2967	2102.422326	2263.585764	1506.529265

The data structure has several issues:

- Empty Columns:** Columns `Unnamed: 0` and `Unnamed: 1` contain only `NaN` values and can be dropped.
- Misaligned Headers:** The actual column headers seem to be in the second or third row instead of the first. You may need to adjust the header row while loading the data.
- Mixing of Different Data Types:** Some columns have a mix of numerical and text values, suggesting improper structure.
- Unclear Column Naming:** The current column names are `Unnamed`, indicating that they are not correctly defined in the dataset.
- Multiple Headers Present:** It appears that the dataset has two levels of headers, one for categories and one for weeks, which might require restructuring.

Would you like me to clean and restructure the dataset for better readability? [\[>\]](#)

before you clean it
let us discuss what the correct data structure should look like

Defining the Correct Data Structure

Before cleaning the data, let's establish the ideal structure based on how you intend to use it. From my initial inspection, the dataset appears to track **qualified leads, conversions, and costs across different sources over multiple weeks**. Here's what we should aim for:



1. Key Data Structure Considerations

- **Column Headers Should Be Clear:** The first row should contain meaningful column names.
- **Each Row Should Represent a Single Observation:** Instead of spreading weekly data across multiple columns, a tidy format should have a separate column for "Week" and another for "Value."
- **Data Types Should Be Consistent:** Columns should have either numerical or categorical data, avoiding mixed types.
- **Hierarchical Headers Should Be Flattened:** If the dataset uses multiple header rows (e.g., `Category -> Subcategory -> Week 1, 2, 3`), it should be normalized.

2. Proposed Data Structure (Tidy Format)

Source Head	Final Source	Metric	Week	Value
Digital	Facebook	Qualified Leads	Week 1	954.47
Digital	Facebook	Qualified Leads	Week 2	1104.30
Digital	Facebook	Qualified Leads	Week 3	1039.50
Digital	Google	Qualified Leads	Week 1	647.18
Digital	Google	Qualified Leads	Week 2	808.93
Digital	Google	Qualified Leads	Week 3	666.85
Digital	Facebook	Total Conversions	Week 1	188.15
Digital	Facebook	Total Conversions	Week 2	248.88
Digital	Facebook	Total Conversions	Week 3	240.41
Digital	Google	Total Conversions	Week 1	172.03
Digital	Google	Total Conversions	Week 2	221.99
Digital	Google	Total Conversions	Week 3	196.98
Digital	Facebook	Cost	Week 1	226868
Digital	Facebook	Cost	Week 2	436133
Digital	Facebook	Cost	Week 3	454302
Digital	Google	Cost	Week 1	361681
Digital	Google	Cost	Week 2	502500
Digital	Google	Cost	Week 3	296760

3. Why This Structure?

- **Easier Analysis & Visualization:** Pivot tables, Power BI, and Python can easily process this format.
- **Eliminates Multiple Headers:** No need to worry about which row contains the actual column names.
- **Enables Time-Series Analysis:** By storing weeks as values rather than columns, it supports trend analysis.
- **Scalable:** If new weeks are added, they appear as rows instead of requiring structural changes.

4. Next Steps

1. Confirm if this structure meets your needs – If ↓ i have any additional columns (like regions or campaign types), we can integrate them.
2. Proceed with Cleaning – Once confirmed, I will transform your dataset into this structure.

What do you think? Any tweaks needed before we proceed?

looks good
but i want the 3 metrics in 3 different columns
rest is okay

Got it! You want Qualified Leads, Total Conversions, and Cost as separate columns, rather than being stacked in a single "Metric" column.

Here's the updated final structure:

Final Data Structure

↓

Source Head	Final Source	week	Qualified Leads	Total Conversions	Cost
Digital	Facebook	Week 1	954.47	188.15	226868
Digital	Facebook	Week 2	1104.30	248.88	436133
Digital	Facebook	Week 3	1039.50	240.41	454302
Digital	Google	Week 1	647.18	172.03	361681
Digital	Google	Week 2	808.93	221.99	502500
Digital	Google	Week 3	666.85	196.98	296760

Why This Works Well

- Each row represents one week for a source, keeping data tidy.
- All three key metrics have dedicated columns, making it easy to compare them directly.
- Easy to use for Power BI, Excel Pivot Tables, or Python-based analysis.

- Adding new weeks or sources is seamless without modifying the structure.



Next Steps

- ✓ Now that the structure is finalized, I'll clean and reshape your data into this format. Let me proceed with the transformation.

Analysis errored ^

```

python                                         Always show details  | ⌂ Copy

# Step 1: Identify the actual header row (appears to be the 3rd row)
df_cleaned = pd.read_excel(xls, sheet_name='Sheet1', skiprows=2)

# Step 2: Rename the columns based on meaningful names
df_cleaned.rename(columns={
    'Unnamed: 2': 'Source Head',
    'Unnamed: 3': 'Final Source',
    'Unnamed: 4': 'Metric',
}, inplace=True)

# Step 3: Drop irrelevant empty columns
df_cleaned.drop(columns=['Unnamed: 0', 'Unnamed: 1'], inplace=True)

# Step 4: Forward fill 'Source Head' to fill merged cells
df_cleaned['Source Head'].fillna(method='ffill', inplace=True)

# Step 5: Remove any empty rows (rows without a 'Final Source' value)
df_cleaned = df_cleaned[df_cleaned['Final Source'].notna()]

# Step 6: Reshape the data into the correct structure
df_melted = df_cleaned.melt(id_vars=['Source Head', 'Final Source', 'Metric'], var_name='Week')

# Step 7: Pivot to get separate columns for each metric
df_pivot = df_melted.pivot_table(index=['Source Head', 'Final Source', 'Week'],
                                    columns='Metric',
                                    values='Value').reset_index()

# Step 8: Rename columns to match the final structure
df_pivot.columns.name = None # Remove the column index name
df_pivot.rename(columns={'Qualified Leads': 'Qualified Leads',
                        'Total landed conversions': 'Total Conversions',
                        'Cost': 'Cost'}, inplace=True)

# Step 9: Ensure weeks are ordered correctly
df_pivot['Week'] = pd.Categorical(df_pivot['Week'], categories=['Week 1', 'Week 2', 'Week 3'])
df_pivot.sort_values(by=['Source Head', 'Final Source', 'Week'], inplace=True)

# Display the cleaned and structured data
import ace_tools as tools
tools.display_dataframe_to_user(name="Cleaned Data", dataframe=df_pivot)

9151     var_name=var_name,
9152     value_name=value_name,
9153     col_level=col_level,
9154     ignore_index=ignore_index,
9155     ).__finalize__(self, method="melt")

File ~/local/lib/python3.11/site-packages/pandas/core/reshape/melt.py:77, in melt(frame, id_vars, value_vars,
   75     missing = Index(com.flatten(id_vars)).difference(cols)
   76     if not missing.empty:
   77         raise KeyError(

```

```

    ,)
    raise KeyError(
78         "The following 'id_vars' are not present "
79         f"in the DataFrame: {list(missing)}"
80     )
81 else:
82     id_vars = []

```

KeyError: "The following 'id_vars' are not present in the DataFrame: ['Metric']"

It looks like the column "Metric" might not have been correctly extracted. Let me inspect the column names and adjust the cleaning process accordingly. [–]

Analyzed ^

python

Always show details | 

```
# Check the actual column names in the dataset
df_cleaned.columns
```

Result

```
Index(['Source Head', 'Final Source', 'Qualified Leads', 'Unnamed: 5',
       'Unnamed: 6', 'Unnamed: 7', 'Total landed conversions', 'Unnamed:
       'Unnamed: 10', 'Unnamed: 11', 'Cost', 'Unnamed: 13', 'Unnamed: 14'
       'COA', 'Unnamed: 16', 'Unnamed: 17'],
      dtype='object')
```

The issue arises because the metrics (Qualified Leads, Total Conversions, and Cost) are already in separate columns, rather than under a "Metric" column. Instead of melting and pivoting, I'll directly restructure the data to fit the required format. Let me fix that now. [–]

Analyze ↓

python

Always show details | 

```
# Step 1: Identify relevant columns
metric_columns = {
    'Qualified Leads': ['Qualified Leads', 'Unnamed: 5', 'Unnamed: 6'],
    'Total Conversions': ['Total landed conversions', 'Unnamed: 9', 'Unnamed: 10'],
    'Cost': ['Cost', 'Unnamed: 13', 'Unnamed: 14']
}
```

```
# Step 2: Reshape the dataset
reshaped_data = []
```

```
# Extract unique source heads and final sources
for index, row in df_cleaned.iterrows():
    source_head = row['Source Head']
    final_source = row['Final Source']

    for week_idx, week in enumerate(['Week 1', 'Week 2', 'Week 3']):
        reshaped_data.append({
            'Source Head': source_head,
            'Final Source': final_source,
            'Week': week,
            'Qualified Leads': row[metric_columns['Qualified Leads'][week_idx]],
            'Total Conversions': row[metric_columns['Total Conversions'][week_idx]],
            'Cost': row[metric_columns['Cost'][week_idx]]
        })

# Convert to DataFrame
df_final = pd.DataFrame(reshaped_data)
```

```

# Ensure proper data types
df_final['Week'] = pd.Categorical(df_final['Week'], categories=['Week 1', 'Week 2', 'Week 3'])
df_final.sort_values(by=['Source Head', 'Final Source', 'Week'], inplace=True)

# Display the cleaned data
tools.display_dataframe_to_user(name="Cleaned Data", dataframe=df_final)

```

```

NameError                                 Traceback (most recent call last)
Cell In[5], line 34
      31 df_final.sort_values(by=['Source Head', 'Final Source', 'Week'], inplace=True)
      32 # Display the cleaned data
--> 34 tools.display_dataframe_to_user(name="Cleaned Data", dataframe=df_final)

NameError: name 'tools' is not defined

```

Analyzed ^

python Always show details Copy

```

# Re-import ace_tools and display the cleaned data again
import ace_tools as tools

tools.display_dataframe_to_user(name="Cleaned Data", dataframe=df_final)

```

Result

	Source Head	Final Source	Week	Qualified Leads	Total Conversions
0	Source Head	Final Source	Week 1	Week 1	Week 1
1	Source Head	Final Source	Week 2	Week 2	Week 2
2	Source Head	Final Source	Week 3	Week 3	Week 3
9	Digital	Affiliate	Week 1	437.661808	127.265306
10	Digital	Affiliate	Week 2	460.126218	124.702391

	Cost
0	Week 1
1	Week 2
2	Week 3
9	303000
10	251600

Cleaned Data				
		Source Head	Final Source	Week
1	0	Source Head	Final Source	Week 1
2	1	Source Head	Final Source	Week 2
3	2	Source Head	Final Source	Week 3
4	9	Digital	Affiliate	Week 1
5	10	Digital	Affiliate	Week 2
6	11	Digital	Affiliate	Week 3

The data is now structured as per the agreed format. However, I noticed that the first few rows still contain column headers instead of actual data. Let's clean those up and ensure all values are correctly formatted. [→]

Analyzed ^

python Always show details Copy

```

# Remove any remaining header-like rows
df_final = df_final[~df_final['Source Head'].str.contains('Source Head', na=False)]

# Convert numerical columns to appropriate data types
df_final['Qualified Leads'] = pd.to_numeric(df_final['Qualified Leads'], errors='coerce')
df_final['Total Conversions'] = pd.to_numeric(df_final['Total Conversions'], errors='coerce')
df_final['Cost'] = pd.to_numeric(df_final['Cost'], errors='coerce')

# Display the cleaned and corrected data
tools.display_dataframe_to_user(name="Final Cleaned Data", dataframe=df_final)

```

STDOUT/STDERR

```

/tmp/ipykernel_12/3127026779.py:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = val instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-dc
df_final['Qualified Leads'] = pd.to_numeric(df_final['Qualified Leads'])
/tmp/ipykernel_12/3127026779.py:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

```

```

See the caveats in the documentation: https://pandas.pydata.org/pandas-dc
df_final['Total Conversions'] = pd.to_numeric(df_final['Total Conversic
/tmp/ipykernel_12/3127026779.py:7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

```

```

See the caveats in the documentation: https://pandas.pydata.org/pandas-dc
df_final['Cost'] = pd.to_numeric(df_final['Cost'], errors='coerce')
/home/sandbox/.local/lib/python3.11/site-packages/pandas/core/internals/b
values = values.astype(str)

```

Result

	Source Head	Final Source	Week	Qualified Leads	Total Conversion
9	Digital	Affiliate	Week 1	661808	127.26530
10	Digital	Affiliate	Week 2	126218	124.78239
11	Digital	Affiliate	Week 3	230.833724	80.95550
12	Digital	Digital Others	Week 1	367.000000	Na
13	Digital	Digital Others	Week 2	368.000000	Na

Cost

9	303000.0
10	251600.0
11	179400.0
12	Nan
13	Nan

Final Cleaned Data

		Source Head	Final Source	Week
1	9	Digital	Affiliate	Week 1
2	10	Digital	Affiliate	Week 2
3	11	Digital	Affiliate	Week 3

12	Digital	Digital Others	Week 1
5 13	Digital	Digital Others	Week 2
6 14	Digital	Digital Others	Week 3

The data is now correctly structured, with proper numerical formatting and unwanted header rows removed. Let me know if you need any further refinements or additional transformations. [x]

thanks

from Source Head remove Reference

from Final Source remove TV Total and Digital Total

Analyzed ^

```
python
# Remove rows where 'Source Head' is 'Reference'
df_final = df_final[df_final['Source Head'] != 'Reference']

# Remove rows where 'Final Source' is 'TV Total' or 'Digital Total'
df_final = df_final[~df_final['Final Source'].isin(['TV Total', 'Digital Total'])]

# Display the updated cleaned data
tools.display_dataframe_to_user(name="Filtered Cleaned Data", dataframe=df_final)

STDOUT/STDERR
/home/sandbox/.local/lib/python3.11/site-packages/pandas/core/internals/b
values = values.astype(str)

```

Result

	Source Head	Final Source	Week	Qualified Leads	Total Conversion
9	Digital	Affiliate	Week 1	437.661808	127.26530
10	Digital	Affiliate	Week 2	460.126218	124.70239
11	Digital	Affiliate	Week 3	230.833724	80.95550
12	Digital	Digital Others	Week 1	367.000000	Na
13	Digital	Digital Others	Week 2	368.000000	Na

Cost

	Cost
9	303000.0
10	251600.0
11	179400.0
12	NaN
13	NaN

		Source Head	Final Source	Week
1	9	Digital	Affiliate	Week 1
2	10	Digital	Affiliate	Week 2
3	11	Digital	Affiliate	Week 3
4	12	Digital	Digital Others	Week 1

5	13	Digital	Digital Others	Week 2
6	14	Digital	Digital Others	Week 3

The requested filters have been applied—"Reference" has been removed from "Source Head," and "TV Total" and "Digital Total" have been removed from "Final Source." Let me know if you need any further refinements. [x]

thanks
working perfectly
can you combine all the python code into 1 single code

Here is the combined Python code that performs all the cleaning and restructuring in a single execution:

⟨/⟩ Clean Transform Data

I've combined all the steps into a single Python script that loads, cleans, and restructures your data in one go. Let me know if you need any modifications or optimizations!

modify my code
it should run on google colab
and save output as an xlsx file

</> Clean Transform Data

```
1 import pandas as pd
2 from google.colab import files
3
4 # Upload the file manually in Google Colab
5 uploaded = files.upload()
6 file_name = list(uploaded.keys())[0] # Get uploaded file name
7
8 # Load the Excel file
9 xls = pd.ExcelFile(file_name)
10
11 # Load the sheet, skipping unnecessary rows
12 df = pd.read_excel(xls, sheet_name='Sheet1', skiprows=2)
13
14 # Rename relevant columns
15 column_mapping = [
```

Edited

I've modified your code so that it runs on **Google Colab**, allowing for manual file uploads and saving the cleaned data as an **Excel file** (`.xlsx`), which will be automatically downloaded after execution. Let me know if you need any further tweaks! 



Ask anything

