Data Cleaning Questions

Based on ARP_0034_3174560763605557342.csv

Group Member Names: Alexia, Ethan, Keoina, Nehemiah

Date & Time Cleaning

1. Convert the Month_Year column into a proper datetime format (YYYY-MM).
   df["Month_Year"] = pd.to_datetime(df["Month_Year"])

2. Extract month name and year into separate columns.
   df['Month'] = df['Month_Year'].dt.month
   df['Year'] = df['Month_Year'].dt.year

3. Check for duplicate months per organization and resolve inconsistencies.
   df = df.drop_duplicates(subset=['Month'], keep='first')

Text Standardization

1. Standardize values in the Organization column (remove trailing spaces, ensure consistent capitalization).
   df['Organization'] = df['Organization'].str.strip().str.title()

2. Clean and format the Address column (separate into street, city, state, and zip).

```
df = df.copy()  # ensure we're not working on a slice

def clean_split_address(address):
    try:
        # Split from the rightmost comma to separate street/city and state/zip
        street_city, state_zip = address.rsplit(',', 1)

        # Clean whitespace
        street_city = street_city.strip()
        state_zip = state_zip.strip()

        # State and ZIP
        state_zip_split = state_zip.split(' ', 1)
        state = state_zip_split[0].upper()
        zip_code = state_zip_split[1] if len(state_zip_split) > 1 else None

        # City is the last word in street_city
        street_parts = street_city.rsplit(' ', 1)
        street = street_parts[0].title()
        city = street_parts[1].title() if len(street_parts) > 1 else None

        return pd.Series({'Street': street, 'City': city, 'State': state, 'Zip': zip_code})

    except Exception:
        # Fallback for malformed addresses
        return pd.Series({'Street': None, 'City': None, 'State': None, 'Zip': None})

# Apply to the DataFrame
df[['Street','City','State','Zip']] = df['Address'].apply(clean_split_address)
```

3. Ensure all ZipCode values are valid 5-digit numeric codes.
   # 1. Truncate 'Zip+4' and remove non-numeric characters
   df['ZipCode_Clean'] = df['ZipCode'].astype(str).str.replace(r'[^\d-]', '', regex=True).str.split('-').str[0].str.zfill(5)

   # 2. Validate: Keep only 5-digit codes, set others to NaN
   df['ZipCode_Clean'] = df['ZipCode_Clean'].apply(
   lambda x: x if pd.notna(x) and len(str(x)) == 5 and str(x).isdigit() else None
)

Numerical Data Quality

1. Identify columns that should be integers (e.g., counts of mediations, interventions) but may contain non-numeric values or blanks. Convert them.
2. Replace missing or invalid numeric entries with 0 (or mark as NaN if appropriate).
3. Detects and removes negative numbers in count columns.
4. Check for outlier values (e.g., unusually high counts of interventions).

Redundancy & Consistency

1. Drop the FID column if it is just an index and not meaningful.
   df.drop("FID", axis=1)
2. Check for duplicate rows (same Month_Year, Organization, and counts).
   duplicates = df.duplicated(subset=['Month_Year', 'Organization', 'Number_of_street_mediations_conducted_',
   'Number_of_high_risk_individuals_receiving_outreach_case_services_',
   'Number_of_hospital_interventions_conducted_',
   'Number_of_high_risk_individuals_contacted_',
   'Number_of_community_meetings__events_',
   'Number_of_violence_reduction_campaigns_'])
   duplicate_rows = df[duplicates]
3. Verify that totals across months make sense (e.g., no sudden large spikes without explanation).
   monthly_activity =
   df.groupby(df["Month_Year"].dt.to_period("M"))[['Number_of_street_mediations_conducted_']].sum()

New Features

1. Create a column for the total outreach activities per row, summing all numeric intervention columns.
   num_data =
   df["Number_of_street_mediations_conducted_"]+df["Number_of_high_risk_individuals_receiving_outreach_case_services_"]+df["Number_of_hospital_interventions_conducted_"]+df["Number_of_high_risk_individuals_contacted_"]+df["Number_of_community_meetings__events_"]+df["Number_of_violence_reduction_campaigns_"]

   df["numeric_total_in"] =
   df["Number_of_street_mediations_conducted_"]+df["Number_of_high_risk_individuals_receiving_outreach_case_services_"]+df["Number_of_hospital_interventions_conducted_"]+df["Number_of_high_risk_individuals_contacted_"]+df["Number_of_community_meetings__events_"]+df["Number_of_violence_reduction_campaigns_"]
2. Calculate the average number of interventions per month per organization.
   df.groupby(["Month_Year","Organization"])["numeric_total_in"].agg(["max","min","mean"])

3. Add a binary flag column indicating whether an organization reported any activity in that month
   import numpy as np

   df["flag"] = np.where (df["numeric_total_in"] > 0, 1, 0)

df.head()