Problem for Covid - 19 Data Analysis Project using Python

~\Downloads\Realproject.py

```python
# covid_analysis.py
# Python 3.8+
# Required packages: pandas, numpy, matplotlib, seaborn
# Install if needed: pip install pandas numpy matplotlib seaborn

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# ========== 1. Import dataset ==========
url = "https://raw.githubusercontent.com/SR1608/Datasets/main/covid-data.csv"
df = pd.read_csv(url)

# ========== 2. High Level Data Understanding ==========
print("=== Shape (rows, columns) ===")
print(df.shape)   # 2.a

print("\n=== Data types ===")
print(df.dtypes)  # 2.b

print("\n=== Info() ===")
print(df.info())  # 2.b (info)pip

print("\n=== Describe() ===")
print(df.describe(include='all'))  # 2.c

# ========== 3. Low Level Data Understanding ==========
# 3.a unique count in 'location'
if 'location' in df.columns:
    print("\nUnique locations count:", df['location'].nunique())
    print("Sample unique locations:", df['location'].unique()[:10])

# 3.b which continent has maximum frequency (value_counts)
if 'continent' in df.columns:
    print("\nContinent frequency (value_counts):")
    print(df['continent'].value_counts())
    most_freq_continent = df['continent'].value_counts().idxmax()
    print("Continent with maximum frequency:", most_freq_continent)

# 3.c max & mean of 'total_cases'
if 'total_cases' in df.columns:
    print("\nTotal_cases max:", df['total_cases'].max())
    print("Total_cases mean:", df['total_cases'].mean())

# 3.d quartiles of 'total_deaths'
if 'total_deaths' in df.columns:
    print("\nTotal_deaths quantiles (25%,50%,75%):")
    print(df['total_deaths'].quantile([0.25, 0.5, 0.75]))

```

```python
# 3.e continent with maximum 'human_development_index'
if {'continent', 'human_development_index'}.issubset(df.columns):
    # groupby continent and take idxmax or max
    hdi_by_cont = df.groupby('continent')['human_development_index'].max().dropna()
    print("\nMax human_development_index by continent:")
    print(hdi_by_cont)
    if not hdi_by_cont.empty:
        continent_max_hdi = hdi_by_cont.idxmax()
        print("Continent with maximum human_development_index:", continent_max_hdi)

# 3.f continent with minimum 'gdp_per_capita'
if {'continent', 'gdp_per_capita'}.issubset(df.columns):
    gdp_by_cont = df.groupby('continent')['gdp_per_capita'].min().dropna()
    print("\nMin gdp_per_capita by continent:")
    print(gdp_by_cont)
    if not gdp_by_cont.empty:
        continent_min_gdp = gdp_by_cont.idxmin()
        print("Continent with minimum gdp_per_capita:", continent_min_gdp)

# ========== 4. Filter dataframe to required columns and update df ==========
cols_keep =
['continent','location','date','total_cases','total_deaths','gdp_per_capita','human_development_in-
dex']
missing_cols = [c for c in cols_keep if c not in df.columns]
if missing_cols:
    raise ValueError(f"The dataset is missing required columns: {missing_cols}")

df = df[cols_keep].copy()
print("\nFiltered dataframe shape:", df.shape)

# ========== 5. Data Cleaning ==========
# 5.a Remove duplicates
before_dup = df.shape[0]
df = df.drop_duplicates()
after_dup = df.shape[0]
print(f"\nRemoved duplicates: {before_dup - after_dup}")

# 5.b Missing values in all columns
print("\nMissing values (per column):")
print(df.isnull().sum())

# 5.c Remove observations where continent is missing (dropna subset)
before_drop_cont = df.shape[0]
df = df.dropna(subset=['continent'])
after_drop_cont = df.shape[0]
print(f"\nRows removed due to missing continent: {before_drop_cont - after_drop_cont}")

# 5.d Fill all remaining missing values with 0
df = df.fillna(0)
print("\nMissing values after fillna(0):")
print(df.isnull().sum())

# ========== 6. Date time format ==========
```

```python
# 6.a convert date column to datetime
df['date'] = pd.to_datetime(df['date'], errors='coerce')

# if coercion produced NaT, you may want to check them:
n_invalid_dates = df['date'].isna().sum()
print(f"\nNumber of invalid/NaT dates after to_datetime: {n_invalid_dates}")

# 6.b Create new column 'month' extracted from date
df['month'] = df['date'].dt.month.fillna(0).astype(int)  # 0 means invalid/missing date

# ========== 7. Data Aggregation ==========
# 7.a Find max value in all columns using groupby on 'continent'
df_groupby = df.groupby('continent').max(numeric_only=False).reset_index()
# Note: groupby.max will take max for columns; for non-numeric like date it takes max lexicographically / latest date.
print("\n=== df_groupby (max per continent) ===")
print(df_groupby.head())

# ========== 8. Feature Engineering ==========
# 8.a Create total_deaths_to_total_cases ratio
# Work on df_groupby (as requested further analysis to use df_groupby)
# Ensure numeric type:
df_groupby['total_cases'] = pd.to_numeric(df_groupby['total_cases'], errors='coerce').fillna(0)
df_groupby['total_deaths'] = pd.to_numeric(df_groupby['total_deaths'], errors='coerce').fillna(0)

# compute ratio safely (avoid division by zero)
df_groupby['total_deaths_to_total_cases'] = np.where(
    df_groupby['total_cases'] == 0,
    0,
    df_groupby['total_deaths'] / df_groupby['total_cases']
)

print("\nFeature 'total_deaths_to_total_cases' added:")
print(df_groupby[['continent','total_deaths','total_cases','total_deaths_to_total_cases']])

# ========== 9. Data Visualization ==========
sns.set(style='whitegrid')

# 9.a Univariate analysis on gdp_per_capita: histogram (seaborn)
plt.figure(figsize=(8,5))
# convert to numeric and drop zeros if you want to avoid 0-filled missing values dominating
gdp_series = pd.to_numeric(df_groupby['gdp_per_capita'], errors='coerce').dropna()
sns.histplot(gdp_series, kde=True)
plt.title('Distribution of gdp_per_capita (df_groupby)')
plt.xlabel('gdp_per_capita')
plt.tight_layout()
plt.show()

# 9.b Scatter plot of total_cases & gdp_per_capita (from df_groupby)
plt.figure(figsize=(8,5))
x = df_groupby['gdp_per_capita'].astype(float)
y = df_groupby['total_cases'].astype(float)
```

```
sns.scatterplot(x=x, y=y)
plt.xlabel('gdp_per_capita')
plt.ylabel('total_cases')
plt.title('Total Cases vs GDP per Capita (df_groupby)')
plt.tight_layout()
plt.show()

# 9.c Pairplot on df_groupby dataset
# For pairplot, choose numeric columns only to keep it readable
numeric_cols = df_groupby.select_dtypes(include=[np.number]).columns.tolist()
if len(numeric_cols) >= 2:
    sns.pairplot(df_groupby[numeric_cols])
    plt.suptitle('Pairplot of numeric features in df_groupby', y=1.02)
    plt.show()
else:
    print("Not enough numeric columns for pairplot.")

# 9.d Bar plot of 'continent' vs 'total_cases'
plt.figure(figsize=(10,6))
# Use seaborn catplot (kind='bar') as suggested:
sns.catplot(kind='bar', x='continent', y='total_cases', data=df_groupby, height=5, aspect=1.6)
plt.title('Total Cases by Continent (df_groupby)')
plt.tight_layout()
plt.show()

# ========== 10. Save df_groupby to local drive ==========
out_filename = "df_groupby.csv"
df_groupby.to_csv(out_filename, index=False)
print(f"\nSaved df_groupby to: {out_filename}")

# Optionally, also save the cleaned and filtered df
df.to_csv("covid_cleaned_filtered.csv", index=False)
print("Saved cleaned filtered dataframe to: covid_cleaned_filtered.csv")

```