COVID-19 Clinical Trials EDA Pandas

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime
import re
plt.style.use('ggplot')
sns.set_palette("husl")
df = pd.read_csv('COVID clinical trials.csv')
print("Dataset Shape:", df.shape)
print("\nColumns:", df.columns.tolist())
print("\nFirst few rows:")
print(df.head())
```

Output:

```
Dataset Shape: (5783, 27)

Columns: ['Rank', 'NCT Number', 'Title', 'Acronym', 'Status', 'Study Results', 'Conditions', 'Interventions', 'Outcome Measures', 'Sponsor/Collaborator
s', 'Gender', 'Age', 'Phases', 'Enrollment', 'Funded Bys', 'Study Type', 'Study Designs', 'Other IDs', 'Start Date', 'Primary Completion Date', 'Complet
ion Date', 'First Posted', 'Results First Posted', 'Last Update Posted', 'Locations', 'Study Documents', 'URL']

First few rows:
   Rank   NCT Number                                          Title  \
0     1  NCT04785898  Diagnostic Performance of the ID Now™ COVID-19...
1     2  NCT04595136  Study to Evaluate the Efficacy of COVID19-0001...
2     3  NCT04395482  Lung CT Scan Analysis of SARS-CoV2 Induced Lun...
3     4  NCT04416061  The Role of a Private Hospital in Hong Kong Am...
4     5  NCT04395924        Maternal-foetal Transmission of SARS-Cov-2

        Acronym                Status        Study Results  \
0   COVID-IDNow  Active, not recruiting  No Results Available
1      COVID-19      Not yet recruiting  No Results Available
2   TAC-COVID19             Recruiting  No Results Available
3      COVID-19  Active, not recruiting  No Results Available
4  TMF-COVID-19             Recruiting  No Results Available

                                    Conditions  \
0                                      Covid19
1                           SARS-CoV-2 Infection
2                                      covid19
3                                        COVID
4  Maternal Fetal Infection Transmission|COVID-19...

                                 Interventions  \
0   Diagnostic Test: ID Now™ COVID-19 Screening Test
1    Drug: Drug COVID19-0001-USR|Drug: normal saline
2  Other: Lung CT scan analysis in COVID-19 patients
3         Diagnostic Test: COVID 19 Diagnostic Test
4  Diagnostic Test: Diagnosis of SARS-Cov2 by RT-...
```

```
                                  Outcome Measures  \
0  Evaluate the diagnostic performance of the ID ...
1  Change on viral load results from baseline aft...
2  A qualitative analysis of parenchymal lung dam...
3  Proportion of asymptomatic subjects|Proportion...
4  COVID-19 by positive PCR in cord blood and / o...

                            Sponsor/Collaborators  ...         Other IDs  \
0             Groupe Hospitalier Paris Saint Joseph  ...       COVID-IDNow
1                       United Medical Specialties  ...  COVID19-0001-USR
2                     University of Milano Bicocca  ...       TAC-COVID19
3                  Hong Kong Sanatorium & Hospital  ...        RC-2020-08
4  Centre Hospitalier Régional d'Orléans|Centre d...  ...      CHRO-2020-10

         Start Date Primary Completion Date   Completion Date  \
0  November 9, 2020       December 22, 2020    April 30, 2021
1  November 2, 2020       December 15, 2020  January 29, 2021
2      May 7, 2020           June 15, 2021     June 15, 2021
3     May 25, 2020           July 31, 2020   August 31, 2020
4      May 5, 2020                May 2021          May 2021

       First Posted Results First Posted Last Update Posted  \
0    March 8, 2021                  NaN      March 8, 2021
1  October 20, 2020                 NaN   October 20, 2020
2     May 20, 2020                  NaN   November 9, 2020
3     June 4, 2020                  NaN       June 4, 2020
4     May 20, 2020                  NaN       June 4, 2020

                                       Locations Study Documents  \
0  Groupe Hospitalier Paris Saint-Joseph, Paris, ...             NaN
1      Cimedical, Barranquilla, Atlantico, Colombia             NaN
2  Ospedale Papa Giovanni XXIII, Bergamo, Italy|P...             NaN
3  Hong Kong Sanatorium & Hospital, Hong Kong, Ho...             NaN
4                      CHR Orléans, Orléans, France             NaN

                                                 URL
```

print("\nMissing values per column:")

print(df.isnull().sum())

Output:

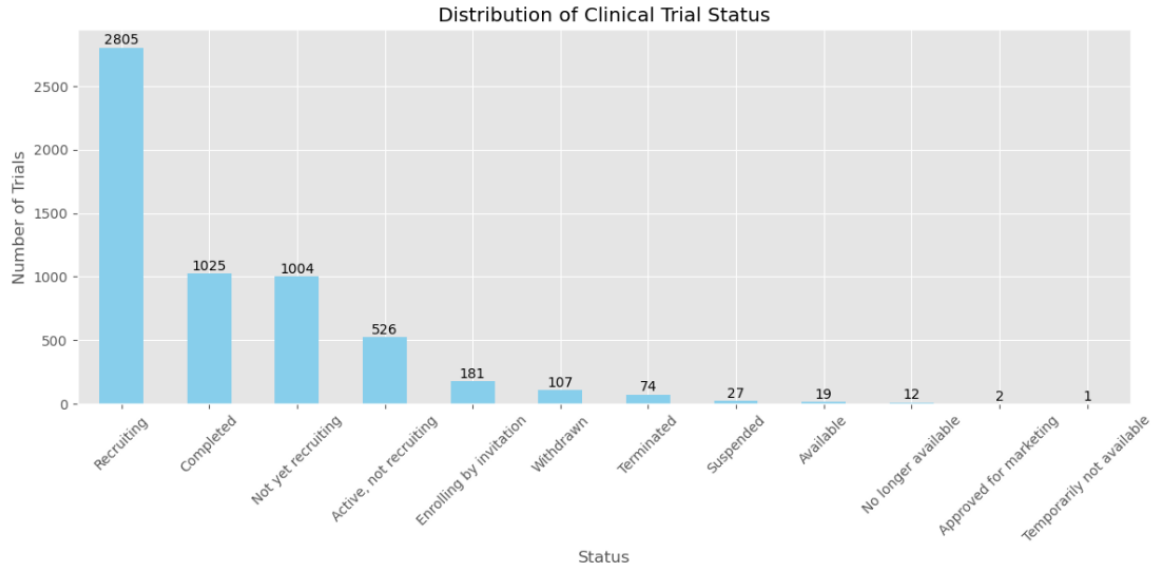
```
Missing values per column:
Rank                          0
NCT Number                    0
Title                         0
Acronym                    3303
Status                        0
Study Results                 0
Conditions                    0
Interventions               886
Outcome Measures             35
Sponsor/Collaborators         0
Gender                       10
Age                           0
Phases                     2461
Enrollment                   34
Funded Bys                    0
Study Type                    0
Study Designs                35
Other IDs                     1
Start Date                   34
Primary Completion Date      36
Completion Date              36
First Posted                  0
Results First Posted       5747
Last Update Posted            0
Locations                   585
Study Documents            5601
URL                           0
dtype: int64
```

```
date_columns = ['Start Date', 'Primary Completion Date', 'Completion Date',
                'First Posted', 'Results First Posted', 'Last Update Posted']
for col in date_columns:
    df[col] = pd.to_datetime(df[col], errors='coerce')
df['Start Year'] = df['Start Date'].dt.year
def clean_phase(phase):
    if pd.isna(phase):
        return 'Not Available'
    phase = str(phase)
    if 'Not Applicable' in phase:
        return 'Not Applicable'
    phases = re.findall(r'Phase\s*\d', phase)
    if phases:
        return ', '.join(sorted(set(phases)))
    return phase

df['Cleaned Phases'] = df['Phases'].apply(clean_phase)
```

```
def clean_enrollment(enrollment):
    if pd.isna(enrollment):
        return 0
    if isinstance(enrollment, str):
        # Remove commas and convert to integer
        enrollment = enrollment.replace(',', '')
        try:
            return int(enrollment)
        except:
            return 0
    return enrollment

df['Cleaned Enrollment'] = df['Enrollment'].apply(clean_enrollment)
def extract_conditions(conditions_str):
    if pd.isna(conditions_str):
        return []
    # Split by | or , and strip whitespace
    conditions = re.split(r'[|,]', conditions_str)
    return [condition.strip() for condition in conditions if condition.strip()]

df['Conditions List'] = df['Conditions'].apply(extract_conditions)
all_conditions = [condition for sublist in df['Conditions List'] for condition in sublist]
condition_counts = pd.Series(all_conditions).value_counts()
plt.figure(figsize=(12, 6))
status_counts = df['Status'].value_counts()
ax = status_counts.plot(kind='bar', color='skyblue')
plt.title('Distribution of Clinical Trial Status')
plt.xlabel('Status')
plt.ylabel('Number of Trials')
plt.xticks(rotation=45)
```

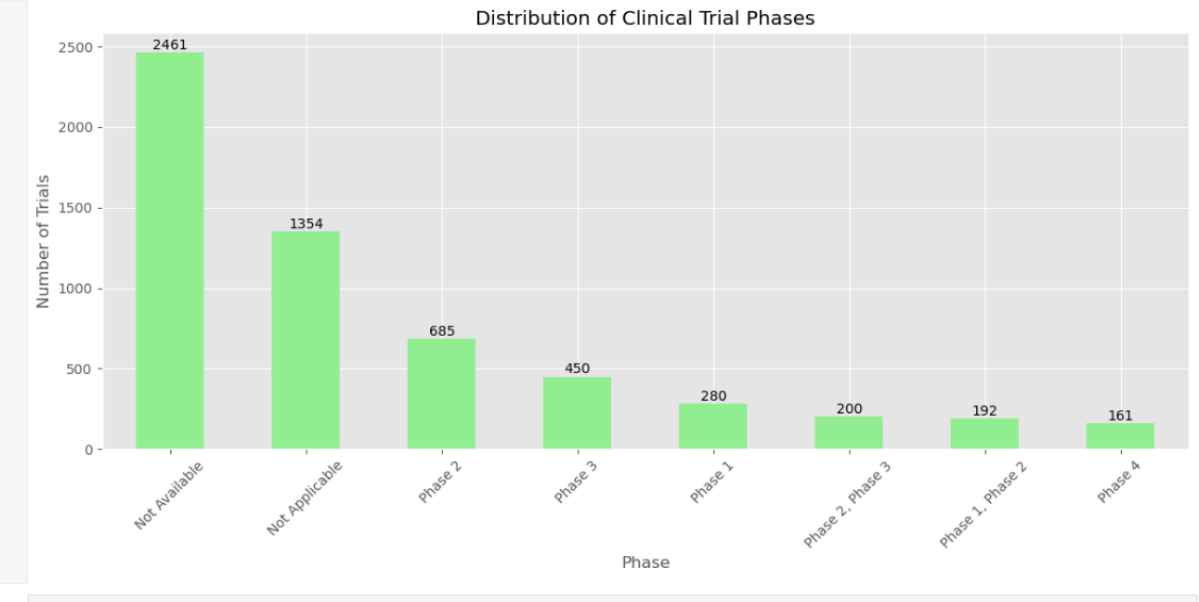
```
for i, v in enumerate(status_counts):
    ax.text(i, v + 0.5, str(v), ha='center', va='bottom')
plt.tight_layout()
plt.savefig('status_distribution.png', dpi=300)
plt.show()
```

Output:



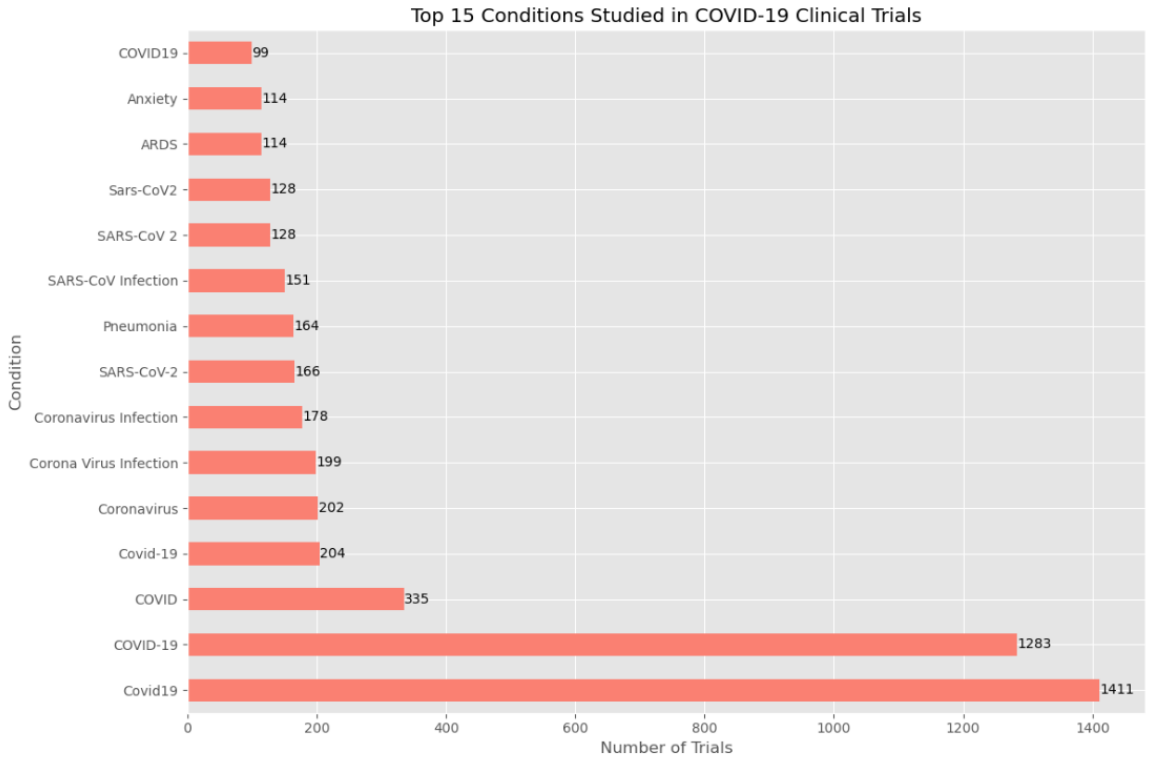

```
plt.figure(figsize=(12, 6))
phase_counts = df['Cleaned Phases'].value_counts()
ax = phase_counts.plot(kind='bar', color='lightgreen')
plt.title('Distribution of Clinical Trial Phases')
plt.xlabel('Phase')
plt.ylabel('Number of Trials')
plt.xticks(rotation=45)
for i, v in enumerate(phase_counts):
    ax.text(i, v + 0.5, str(v), ha='center', va='bottom')
plt.tight_layout()
plt.savefig('phase_distribution.png', dpi=300)
plt.show()
```

Output:



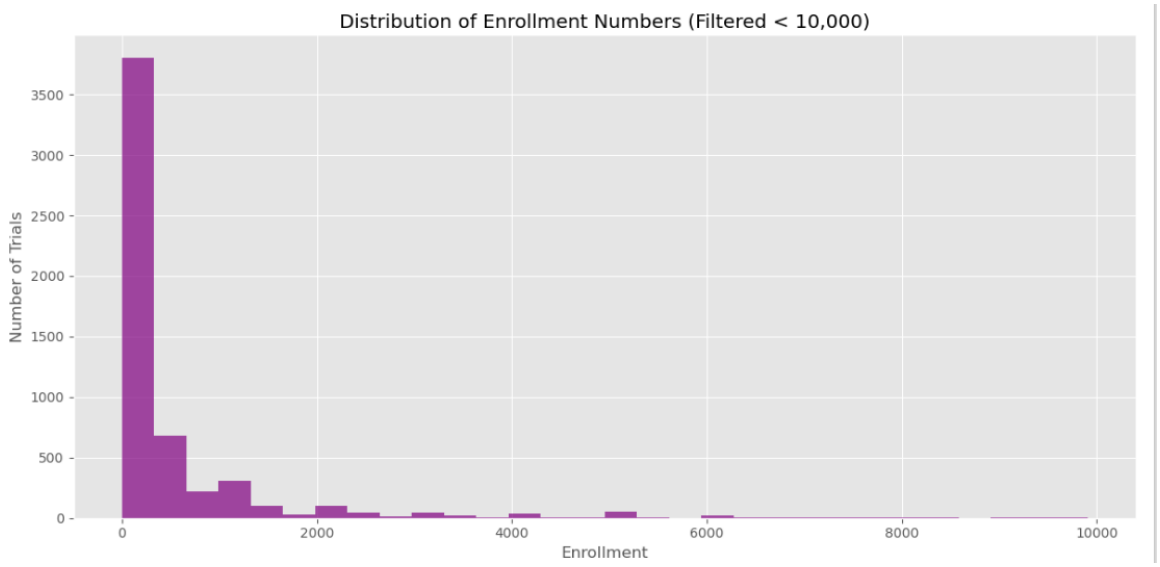

```
plt.figure(figsize=(12, 8))
top_conditions = condition_counts.head(15)
ax = top_conditions.plot(kind='barh', color='salmon')
plt.title('Top 15 Conditions Studied in COVID-19 Clinical Trials')
plt.xlabel('Number of Trials')
plt.ylabel('Condition')
for i, v in enumerate(top_conditions):
    ax.text(v + 0.5, i, str(v), ha='left', va='center')
plt.tight_layout()
plt.savefig('top_conditions.png', dpi=300)
plt.show()
```

Output:



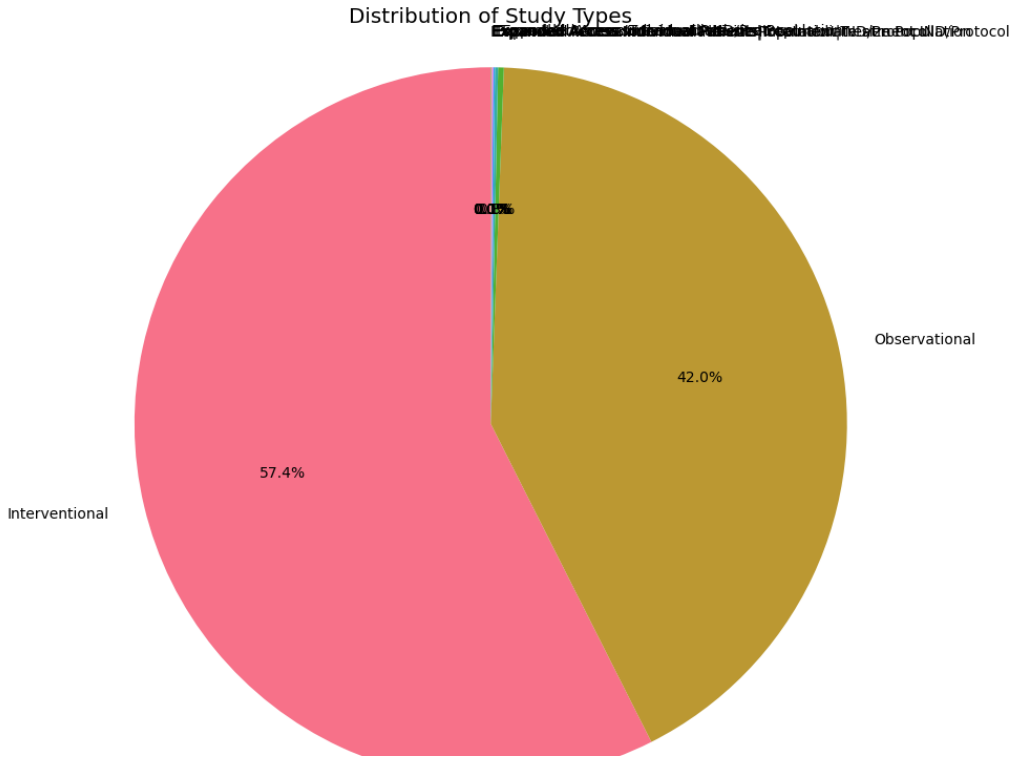

```
plt.figure(figsize=(12, 6))
enrollment_filtered = df[df['Cleaned Enrollment'] < 10000]['Cleaned Enrollment']
plt.hist(enrollment_filtered, bins=30, color='purple', alpha=0.7)
plt.title('Distribution of Enrollment Numbers (Filtered < 10,000)')
plt.xlabel('Enrollment')
plt.ylabel('Number of Trials')
plt.tight_layout()
plt.savefig('enrollment_distribution.png', dpi=300)
plt.show()
```

Output:



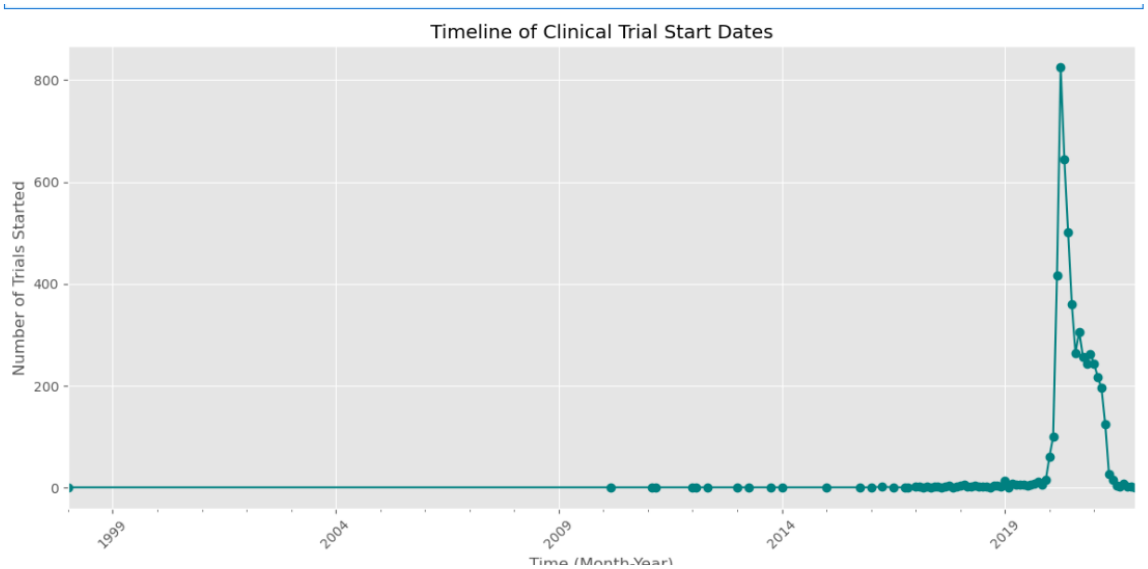

```
plt.figure(figsize=(10, 8))

study_type_counts = df['Study Type'].value_counts()

plt.pie(study_type_counts, labels=study_type_counts.index, autopct='%1.1f%%',
startangle=90)

plt.title('Distribution of Study Types')

plt.axis('equal')

plt.tight_layout()

plt.savefig('study_type_distribution.png', dpi=300)

plt.show()
```

Output:



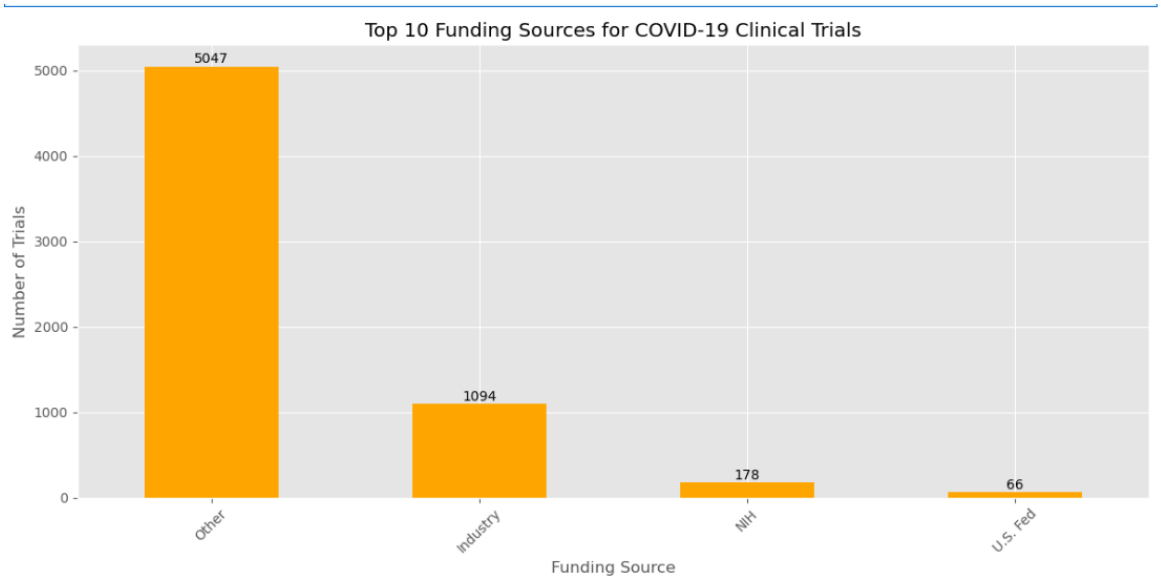

```
plt.figure(figsize=(12, 6))
start_dates = df['Start Date'].dropna()
start_dates_counts = start_dates.dt.to_period('M').value_counts().sort_index()
start_dates_counts.plot(kind='line', marker='o', color='teal')
plt.title('Timeline of Clinical Trial Start Dates')
plt.xlabel('Time (Month-Year)')
plt.ylabel('Number of Trials Started')
plt.xticks(rotation=45)
plt.tight_layout()
plt.savefig('trial_timeline.png', dpi=300)
plt.show()
```

Output:




```
def extract_funders(funders_str):
    if pd.isna(funders_str):
        return ['Not Specified']
    funders = re.split(r'[|]', funders_str)
    return [funder.strip() for funder in funders if funder.strip()]

df['Funders List'] = df['Funded Bys'].apply(extract_funders)
all_funders = [funder for sublist in df['Funders List'] for funder in sublist]
funder_counts = pd.Series(all_funders).value_counts().head(10)

plt.figure(figsize=(12, 6))
ax = funder_counts.plot(kind='bar', color='orange')
plt.title('Top 10 Funding Sources for COVID-19 Clinical Trials')
plt.xlabel('Funding Source')
plt.ylabel('Number of Trials')
plt.xticks(rotation=45)
for i, v in enumerate(funder_counts):
    ax.text(i, v + 0.5, str(v), ha='center', va='bottom')
plt.tight_layout()
plt.savefig('funder_distribution.png', dpi=300)
plt.show()
```

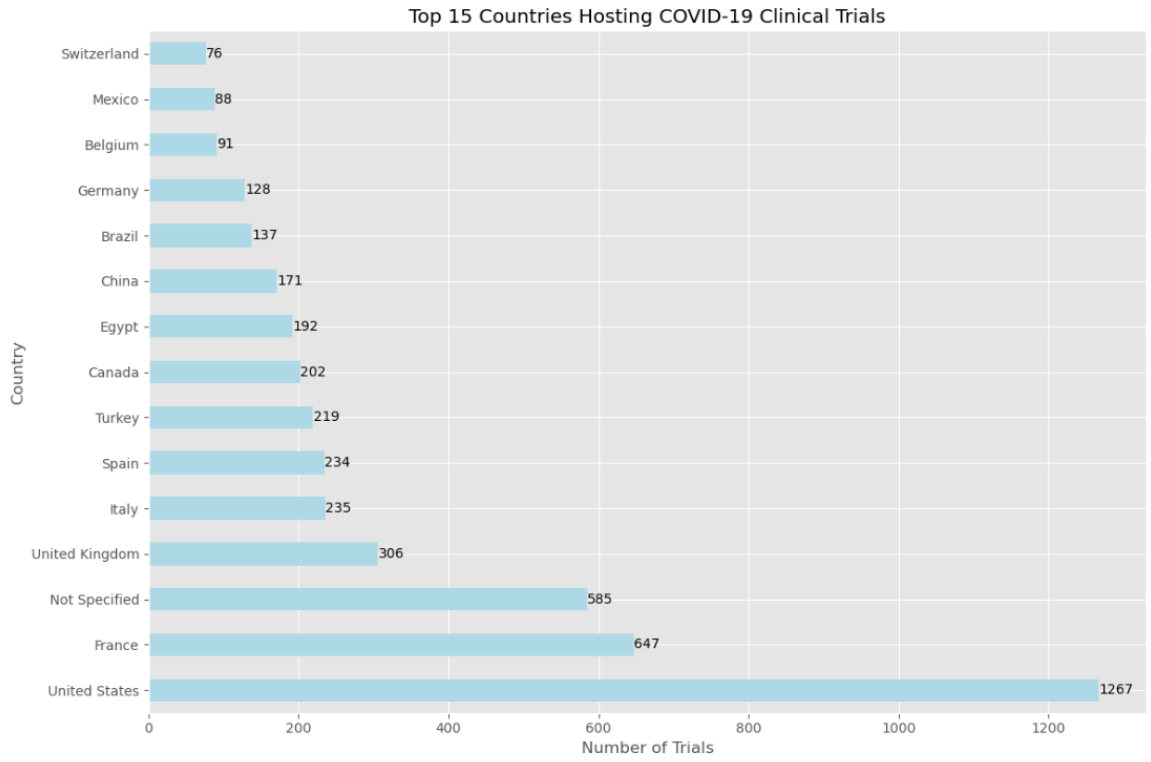
Output:




```
def extract_country(location_str):
    if pd.isna(location_str):
        return 'Not Specified'
    # Extract country from location string (usually the last part)
    parts = location_str.split(',')
    if parts:
        return parts[-1].strip()
    return 'Not Specified'

df['Country'] = df['Locations'].apply(extract_country)
country_counts = df['Country'].value_counts().head(15)

plt.figure(figsize=(12, 8))
ax = country_counts.plot(kind='barh', color='lightblue')
plt.title('Top 15 Countries Hosting COVID-19 Clinical Trials')
plt.xlabel('Number of Trials')
plt.ylabel('Country')
for i, v in enumerate(country_counts):
    ax.text(v + 0.5, i, str(v), ha='left', va='center')
plt.tight_layout()
```

```
plt.savefig('country_distribution.png', dpi=300)
plt.show()
```

Output:

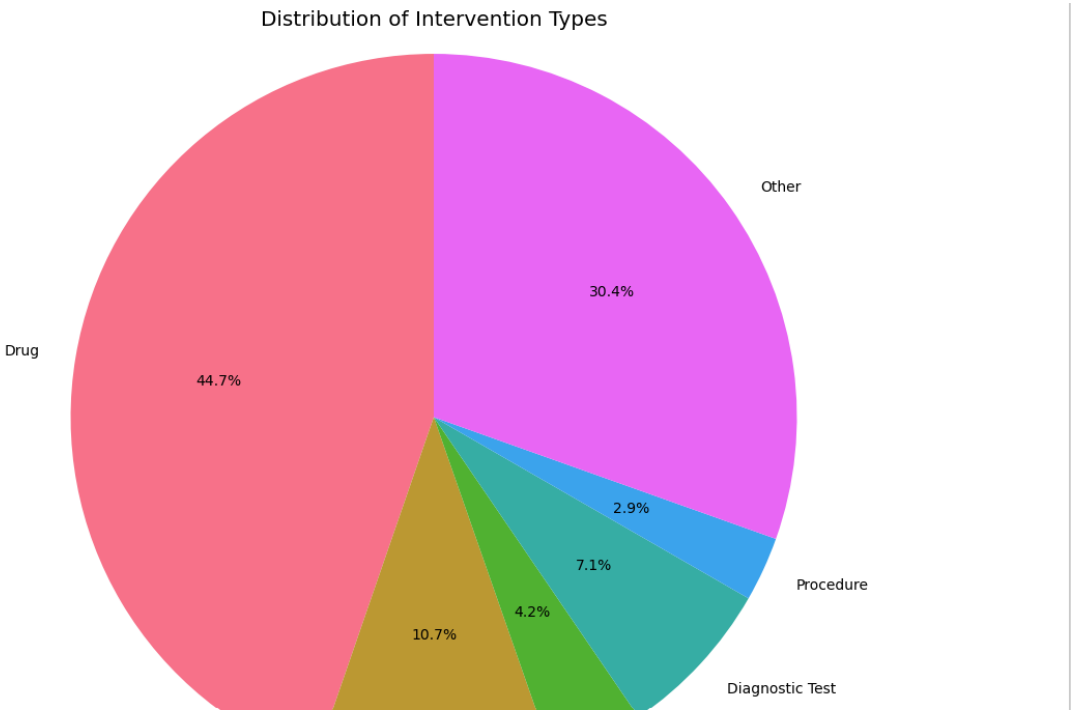



```
def extract_interventions(interventions_str):
    if pd.isna(interventions_str):
        return ['Not Specified']
    interventions = re.split(r'[|]', interventions_str)
    return [intervention.strip() for intervention in interventions if intervention.strip()]

df['Interventions List'] = df['Interventions'].apply(extract_interventions)
all_interventions = [intervention for sublist in df['Interventions List'] for intervention in sublist]
intervention_categories = {
    'Drug': [],
    'Biological': [],
    'Device': [],
    'Diagnostic Test': [],
```

```
    'Procedure': [],
    'Other': []
}

for intervention in all_interventions:
    found = False
    for category, keywords in {
        'Drug': ['drug', 'tablet', 'capsule', 'injection', 'iv', 'oral'],
        'Biological': ['biological', 'plasma', 'vaccine', 'antibody', 'cell'],
        'Device': ['device', 'machine', 'equipment', 'apparatus'],
        'Diagnostic Test': ['diagnostic', 'test', 'assay', 'pcr', 'serology'],
        'Procedure': ['procedure', 'surgery', 'therapy', 'treatment']
    }.items():
        if any(keyword in intervention.lower() for keyword in keywords):
            intervention_categories[category].append(intervention)
            found = True
            break
    if not found:
        intervention_categories['Other'].append(intervention)

# Count interventions by category
intervention_counts = {category: len(interventions) for category, interventions in
intervention_categories.items()}


plt.figure(figsize=(10, 8))
plt.pie(intervention_counts.values(), labels=intervention_counts.keys(), autopct='%1.1f%%',
startangle=90)
plt.title('Distribution of Intervention Types')
plt.axis('equal')
plt.tight_layout()
plt.savefig('intervention_types.png', dpi=300)
```

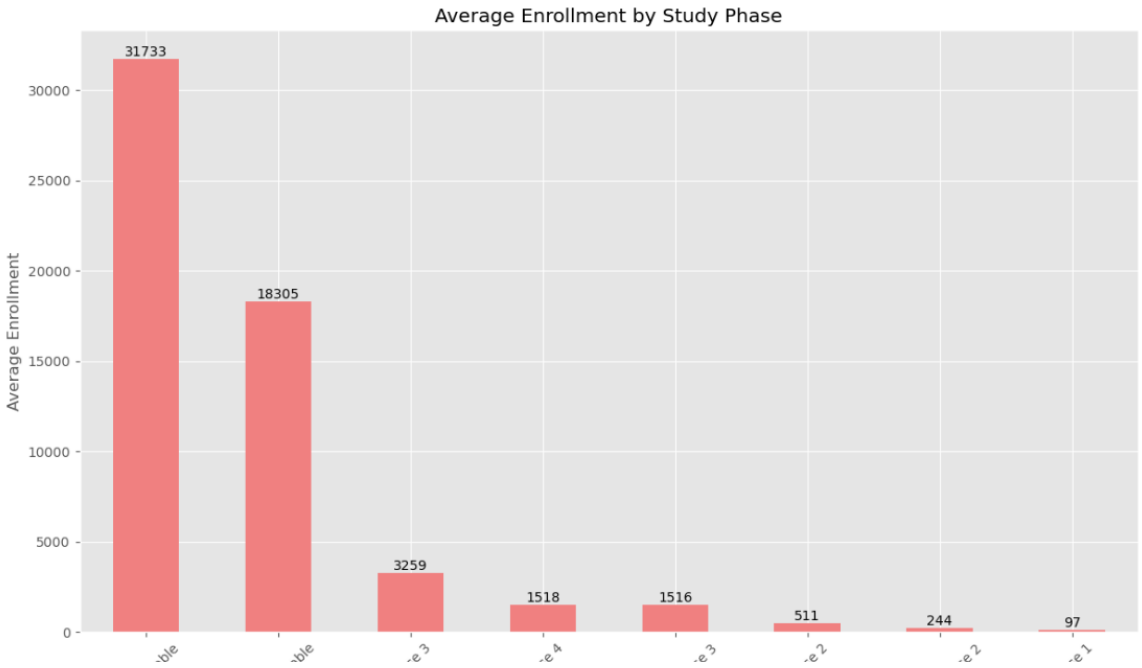
plt.show()

Output:




```
plt.figure(figsize=(12, 8))
phase_enrollment = df.groupby('Cleaned Phases')['Cleaned Enrollment'].mean().sort_values(ascending=False)
ax = phase_enrollment.plot(kind='bar', color='lightcoral')
plt.title('Average Enrollment by Study Phase')
plt.xlabel('Study Phase')
plt.ylabel('Average Enrollment')
plt.xticks(rotation=45)
for i, v in enumerate(phase_enrollment):
    ax.text(i, v + 5, f'{v:.0f}', ha='center', va='bottom')
plt.tight_layout()
plt.savefig('enrollment_by_phase.png', dpi=300)
plt.show()
```

Output :




```
results_available = df['Study Results'].value_counts()
print("\nStudy Results Availability:")
print(results_available)
```

output:

```
Study Results Availability:
Study Results
No Results Available    5747
Has Results               36
Name: count, dtype: int64
```

```
df.to_csv('cleaned_covid_clinical_trials.csv', index=False)

print("\nAnalysis complete! Visualizations have been saved as PNG files.")
print("Cleaned dataset saved as 'cleaned_covid_clinical_trials.csv'")
```

Output:

```
Analysis complete! Visualizations have been saved as PNG files.
Cleaned dataset saved as 'cleaned_covid_clinical_trials.csv'
```