In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv('covid_clinical_trials.csv')

print(df.shape)

df.head()
```

```
(5783, 27)
```

Out[1]:

	Rank	NCT Number	Title	Acronym	Status	Study Results	Conditions
0	1	NCT04785898	Diagnostic Performance of the ID Now™ COVID-19...	COVID-IDNow	Active, not recruiting	No Results Available	Covid19
1	2	NCT04595136	Study to Evaluate the Efficacy of COVID19-0001...	COVID-19	Not yet recruiting	No Results Available	SARS-CoV-2 Infection
2	3	NCT04395482	Lung CT Scan Analysis of SARS-CoV2 Induced Lun...	TAC-COVID19	Recruiting	No Results Available	covid19
3	4	NCT04416061	The Role of a Private Hospital in Hong Kong Am...	COVID-19	Active, not recruiting	No Results Available	COVID
4	5	NCT04395924	Maternal-foetal Transmission of SARS-Cov-2	TMF-COVID-19	Recruiting	No Results Available	Maternal Fetal Infection Transmission|COVID-19...

5 rows × 27 columns

In [2]:
```
# Check missing values
```

In [3]:
```
df.isnull().sum()
```

Out[3]:
```
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
```

In [4]:
```
df.drop(['Results First Posted', 'Study Documents'], axis=1, inplace=True)
```

In [5]:
```
categorical_cols = df.select_dtypes(include='object').columns

for col in categorical_cols:
    if df[col].isnull().sum() > 0:
        df[col].fillna(f"Missing {col}", inplace=True)
```

```
C:\Users\krish\AppData\Local\Temp\ipykernel_13732\3483844190.py:5: FutureWarning:
A value is trying to be set on a copy of a DataFrame or Series through chained as
signment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work becau
se the intermediate object on which we are setting values always behaves as a cop
y.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.meth
od({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to pe
rform the operation inplace on the original object.


  df[col].fillna(f"Missing {col}", inplace=True)
```

In [6]:
```
# Handle missing categorical data
```

In [7]:
```
categorical_cols = df.select_dtypes(include='object').columns

for col in categorical_cols:
    if df[col].isnull().sum() > 0:
        df[col] = df[col].fillna(f"Missing {col}")
```

In [8]:
```
# Verify cleaning
```

In [9]:
```
df.isnull().sum()
```

Out[9]:
```
Rank                         0
NCT Number                   0
Title                        0
Acronym                      0
Status                       0
Study Results                0
Conditions                   0
Interventions                0
Outcome Measures             0
Sponsor/Collaborators        0
Gender                       0
Age                          0
Phases                       0
Enrollment                  34
Funded Bys                   0
Study Type                   0
Study Designs                0
Other IDs                    0
Start Date                   0
Primary Completion Date      0
Completion Date              0
First Posted                 0
Last Update Posted           0
Locations                    0
URL                          0
dtype: int64
```

In [10]:
```
# Filling missing numeric data in Enrollment with median
```

In [11]:
```
median_enrollment = df['Enrollment'].median()
df['Enrollment'].fillna(median_enrollment, inplace=True)


df.isnull().sum()
```

```
C:\Users\krish\AppData\Local\Temp\ipykernel_13732\145516709.py:2: FutureWarning:
A value is trying to be set on a copy of a DataFrame or Series through chained as
signment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work becau
se the intermediate object on which we are setting values always behaves as a cop
y.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.meth
od({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to pe
rform the operation inplace on the original object.


  df['Enrollment'].fillna(median_enrollment, inplace=True)
```

Out[11]:
```
Rank                        0
NCT Number                  0
Title                       0
Acronym                     0
Status                      0
Study Results               0
Conditions                  0
Interventions               0
Outcome Measures            0
Sponsor/Collaborators       0
Gender                      0
Age                         0
Phases                      0
Enrollment                  0
Funded Bys                  0
Study Type                  0
Study Designs               0
Other IDs                   0
Start Date                  0
Primary Completion Date     0
Completion Date             0
First Posted                0
Last Update Posted          0
Locations                   0
URL                         0
dtype: int64
```

In [12]:
```
median_enrollment = df['Enrollment'].median()
df['Enrollment'] = df['Enrollment'].fillna(median_enrollment)
```

In [13]:
```
# Extract country name from Locations column
```

In [14]:
```
df['Country'] = df['Locations'].apply(lambda x: str(x).split(',')[-1].strip())
```

In [15]:
```
df['Country'].value_counts().head(10)
```

Out[15]:
```
Country
United States        1267
France                647
Missing Locations     585
United Kingdom        306
Italy                 235
Spain                 234
Turkey                219
Canada                202
Egypt                 192
China                 171
Name: count, dtype: int64
```

In [16]:
```
# Univariate Analysis
```

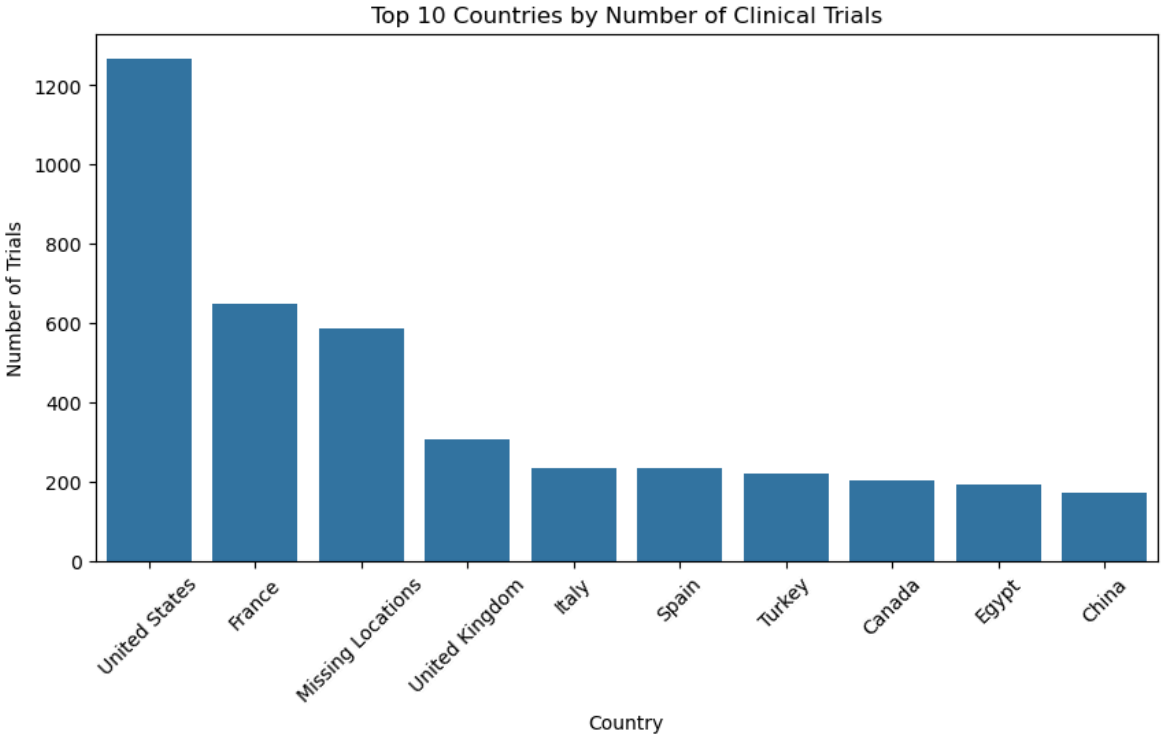
In [17]:
```
# Top 10 contributing countries
```

In [18]:
```
import matplotlib.pyplot as plt
import seaborn as sns

top_countries = df['Cuontry'].value_counts().head(10)
```

```
plt.figure(figsize=(10,5))
sns.barplot(x=top_countries.index, y=top_countries.values)
plt.title('Top 10 Countries by Number of Clinical Trials')
plt.ylabel('Number of Trials')
plt.xticks(rotation=45)
plt.show()
```

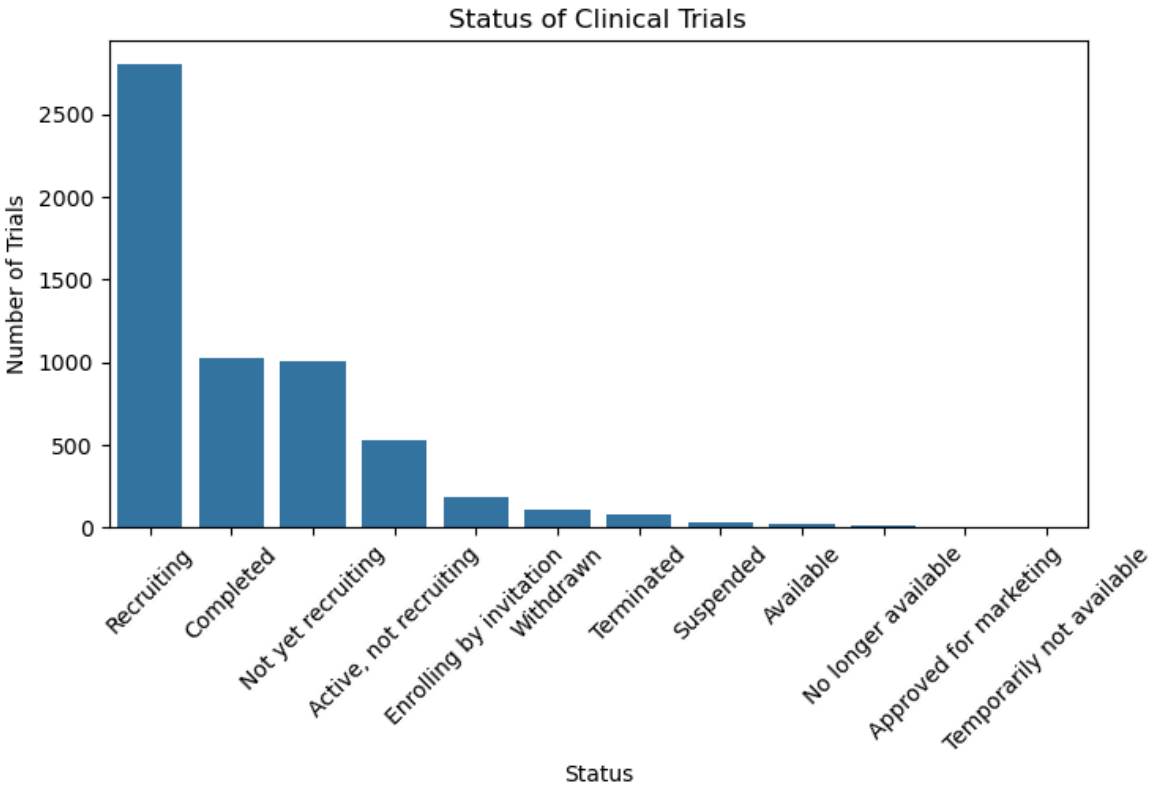
```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:3805,
in Index.get_loc(self, key)
   3804 try:
-> 3805     return self._engine.get_loc(casted_key)
   3806 except KeyError as err:

File index.pyx:167, in pandas._libs.index.IndexEngine.get_loc()

File index.pyx:196, in pandas._libs.index.IndexEngine.get_loc()

File pandas\\_libs\\hashtable_class_helper.pxi:7081, in pandas._libs.hashtable.Py
ObjectHashTable.get_item()

File pandas\\_libs\\hashtable_class_helper.pxi:7089, in pandas._libs.hashtable.Py
ObjectHashTable.get_item()

KeyError: 'Cuontry'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
Cell In[18], line 4
      1 import matplotlib.pyplot as plt
      2 import seaborn as sns
----> 4 top_countries = df['Cuontry'].value_counts().head(10)
      6 plt.figure(figsize=(10,5))
      7 sns.barplot(x=top_countries.index, y=top_countries.values)

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\frame.py:4102, in Dat
aFrame.__getitem__(self, key)
   4100 if self.columns.nlevels > 1:
   4101     return self._getitem_multilevel(key)
-> 4102 indexer = self.columns.get_loc(key)
   4103 if is_integer(indexer):
   4104     indexer = [indexer]

File C:\ProgramData\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:3812,
in Index.get_loc(self, key)
   3807     if isinstance(casted_key, slice) or (
   3808         isinstance(casted_key, abc.Iterable)
   3809         and any(isinstance(x, slice) for x in casted_key)
   3810     ):
   3811         raise InvalidIndexError(key)
-> 3812     raise KeyError(key) from err
   3813 except TypeError:
   3814     # If we have a listlike key, _check_indexing_error will raise
   3815     #  InvalidIndexError. Otherwise we fall through and re-raise
   3816     #  the TypeError.
   3817     self._check_indexing_error(key)

KeyError: 'Cuontry'
```

In [19]:
```
import matplotlib.pyplot as plt
import seaborn as sns

top_countries = df['Country'].value_counts().head(10)

plt.figure(figsize=(10,5))
sns.barplot(x=top_countries.index, y=top_countries.values)
plt.title('Top 10 Countries by Number of Clinical Trials')
plt.ylabel('Number of Trials')
plt.xticks(rotation=45)
plt.show()
```



In [20]:
```
# Status distribution of trials
```

In [21]:
```
status_counts = df['Status'].value_counts()

plt.figure(figsize=(8,4))
sns.barplot(x=status_counts.index, y=status_counts.values)
plt.title('Status of Clinical Trials')
plt.ylabel('Number of Trials')
plt.xticks(rotation=45)
plt.show()
```

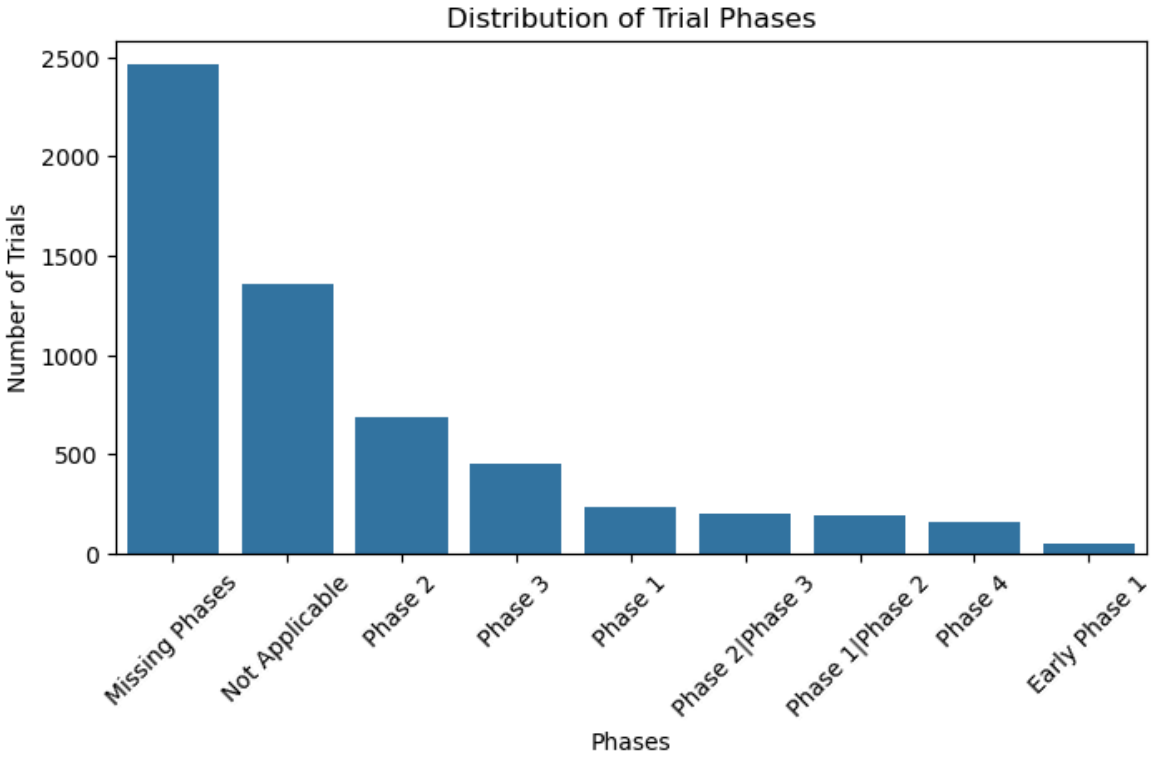


In [22]:
```
Phases distributing
```

```
  Cell In[22], line 1
    Phases distributing
           ^
SyntaxError: invalid syntax
```

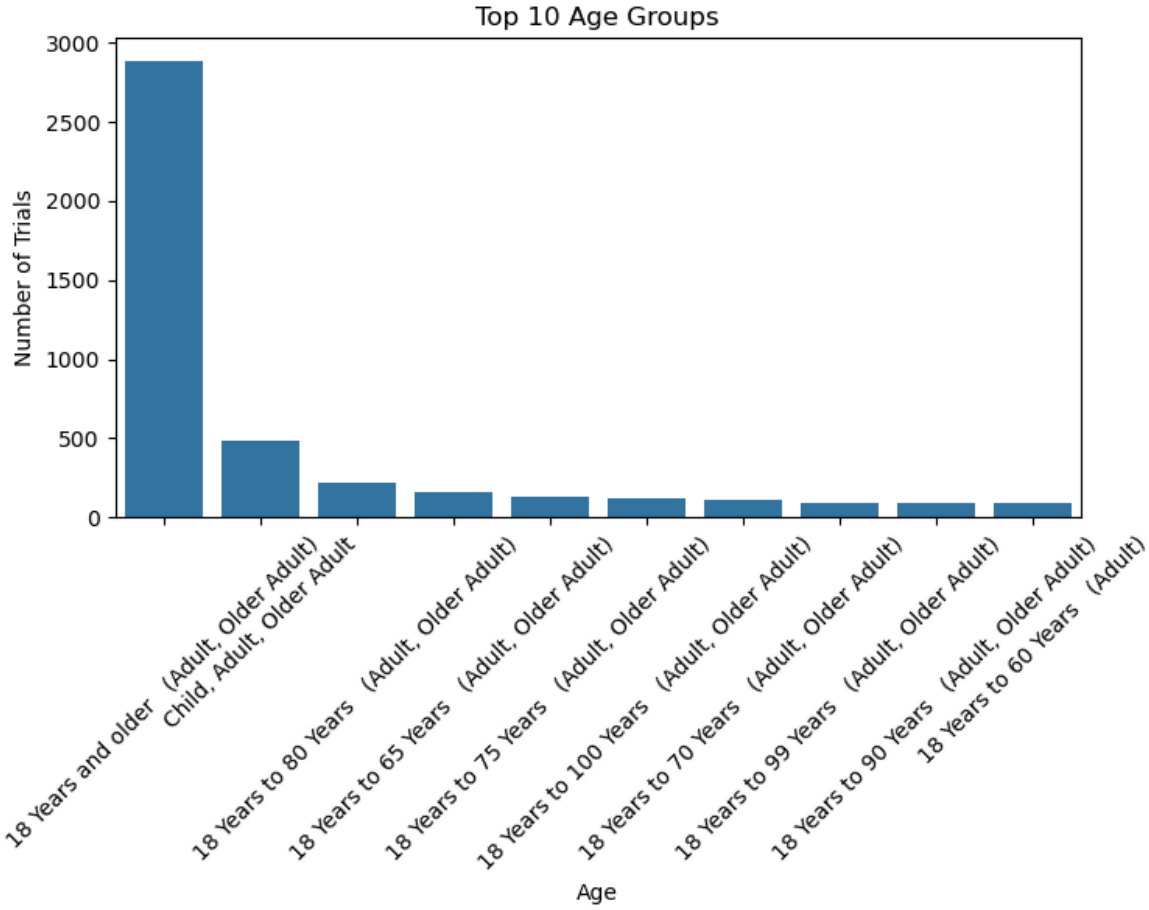
In [23]:
```
# Phases distributing
```

In [24]:
```
phase_counts = df['Phases'].value_counts()

plt.figure(figsize=(8,4))
sns.barplot(x=phase_counts.index, y=phase_counts.values)
plt.title('Distribution of Trial Phases')
plt.ylabel('Number of Trials')
plt.xticks(rotation=45)
plt.show()
```



In [25]:
```
# Age group distribution
```

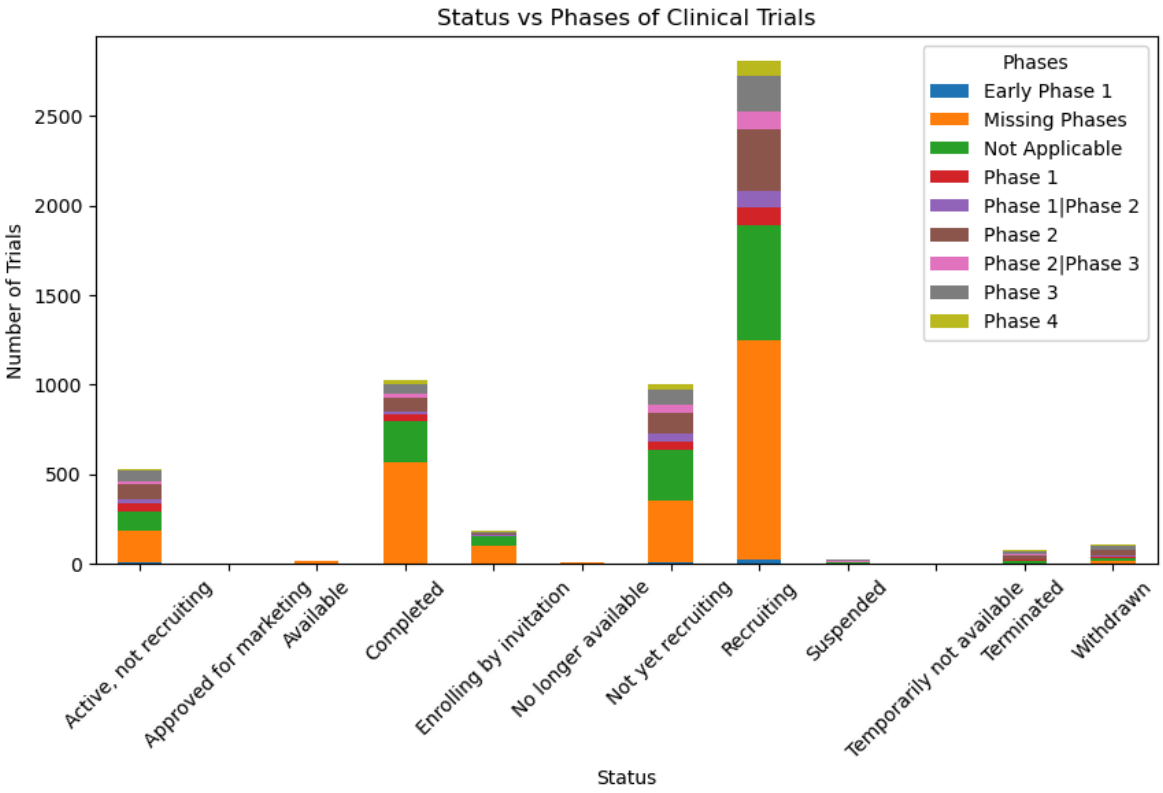
In [26]:
```
age_counts = df['Age'].value_counts().head(10)

plt.figure(figsize=(8,4))
sns.barplot(x=age_counts.index, y=age_counts.values)
plt.title('Top 10 Age Groups')
plt.ylabel('Number of Trials')
plt.xticks(rotation=45)
plt.show()
```



In [27]:
```
# Status vs. Phases
```

In [28]:
```
status_phase = pd.crosstab(df['Status'], df['Phases'])

plt.figure(figsize=(10,5))
status_phase.plot(kind='bar', stacked=True, figsize=(10,5))
plt.title('Status vs Phases of Clinical Trials')
plt.ylabel('Number of Trials')
plt.xticks(rotation=45)
plt.show()
```

```
<Figure size 1000x500 with 0 Axes>
```



In [29]:
```
conditions_outcomes = df.groupby('Conditions')['Outcome Measures'] \
                        .apply(lambda x: ', '.join(x.astype(str))) \
                        .reset_index()

conditions_outcomes.head()
```

Out[29]:

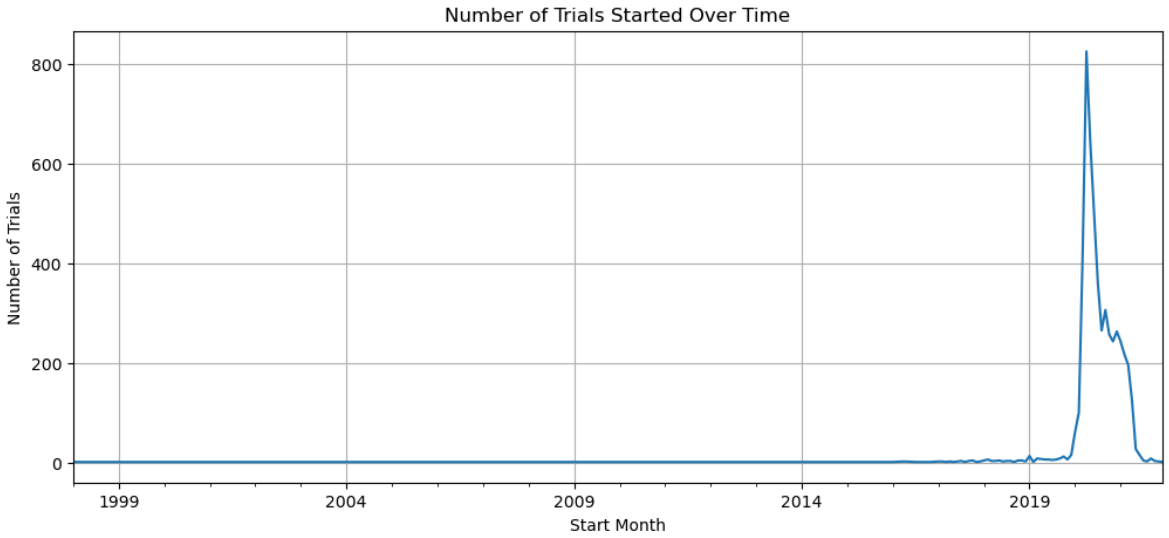
	Conditions	Outcome Measures
0	2019 Novel Coronavirus	Proportion of participants who improve by at l...
1	2019 Novel Coronavirus Infection	new-onset COVID-19|Number of Participants with...
2	2019 Novel Coronavirus Infection|COVID-19 Viru...	Number of participants with treatment emergent...
3	2019 Novel Coronavirus Pneumonia	Clinical recovery time|Complete fever time|Cou...
4	2019 Novel Coronavirus Pneumonia|COVID-19	Pneumonia severity index|Oxygenation index (Pa...

In [30]:
```
# Time Series Analysis
```

In [31]:
```
df['Start Date'] = pd.to_datetime(df['Start Date'], errors='coerce')
```

In [32]:
```
trials_over_time = df['Start Date'].dt.to_period('M').value_counts().sort_index(

plt.figure(figsize=(12,5))
trials_over_time.plot(kind='line')
plt.title('Number of Trials Started Over Time')
plt.ylabel('Number of Trials')
plt.xlabel('Start Month')
```

```
plt.grid(True)
plt.show()
```



In [33]:
```
trials_over_time.sort_values(ascending=False).head(10)
```

Out[33]:
```
Start Date
2020-04    825
2020-05    645
2020-06    502
2020-03    417
2020-07    361
2020-09    306
2020-08    265
2020-12    263
2020-10    257
2020-11    243
Freq: M, Name: count, dtype: int64
```

In [34]:
```
# Conclusion:

# The majority of clinical trials are concentrated in a few countries – United S

# Most trials are in the "Completed" and "Recruiting" status, with a wide variat

# Adult populations are the primary target group for most trials.

# There was a significant increase in trial activity during the early months of

# Data cleaning was essential – we handled missing values for categorical and nu
```

In [35]:
```
df.to_csv('cleaned_covid_clinical_trials.csv', index=False)
print("Cleaned dataset saved successfully!")
```

```
Cleaned dataset saved successfully!
```

In [ ]: