02_eda_clinical_trial

March 16, 2025

1 RARE Diseases

[18]:
```
import pandas as pd

# Load data
data = pd.read_csv('data/CT_all_rare_disease_processed.csv')
df = pd.DataFrame(data)

# Group by 'disease' column and count unique values in 'Overall Status'
# unique_counts = df.groupby('Disease')['Overall Status'].value_counts()

# finding trial status
trial_status = df.groupby(['Disease', 'Overall Status']).size().
 ↪reset_index(name='count')


# Print result
print(trial_status)
print("TRIAL STATUS PRINTED")
```

```
                          Disease           Overall Status  count
0                CT_chagas_disease    ACTIVE_NOT_RECRUITING      2
1                CT_chagas_disease                COMPLETED     33
2                CT_chagas_disease       NOT_YET_RECRUITING      1
3                CT_chagas_disease                RECRUITING      2
4                CT_chagas_disease                TERMINATED      2
5                CT_chagas_disease                   UNKNOWN     14
6                CT_chagas_disease                 WITHDRAWN      1
7   CT_drug_resistant_tuberculosis    ACTIVE_NOT_RECRUITING      4
8   CT_drug_resistant_tuberculosis   APPROVED_FOR_MARKETING      1
9   CT_drug_resistant_tuberculosis                COMPLETED     40
10  CT_drug_resistant_tuberculosis       NOT_YET_RECRUITING      4
11  CT_drug_resistant_tuberculosis                RECRUITING      5
12  CT_drug_resistant_tuberculosis                   UNKNOWN      5
13  CT_drug_resistant_tuberculosis                 WITHDRAWN      1
14  CT_duchenne_muscular_dystrophy    ACTIVE_NOT_RECRUITING     30
15  CT_duchenne_muscular_dystrophy   APPROVED_FOR_MARKETING      3
16  CT_duchenne_muscular_dystrophy                AVAILABLE      1
```

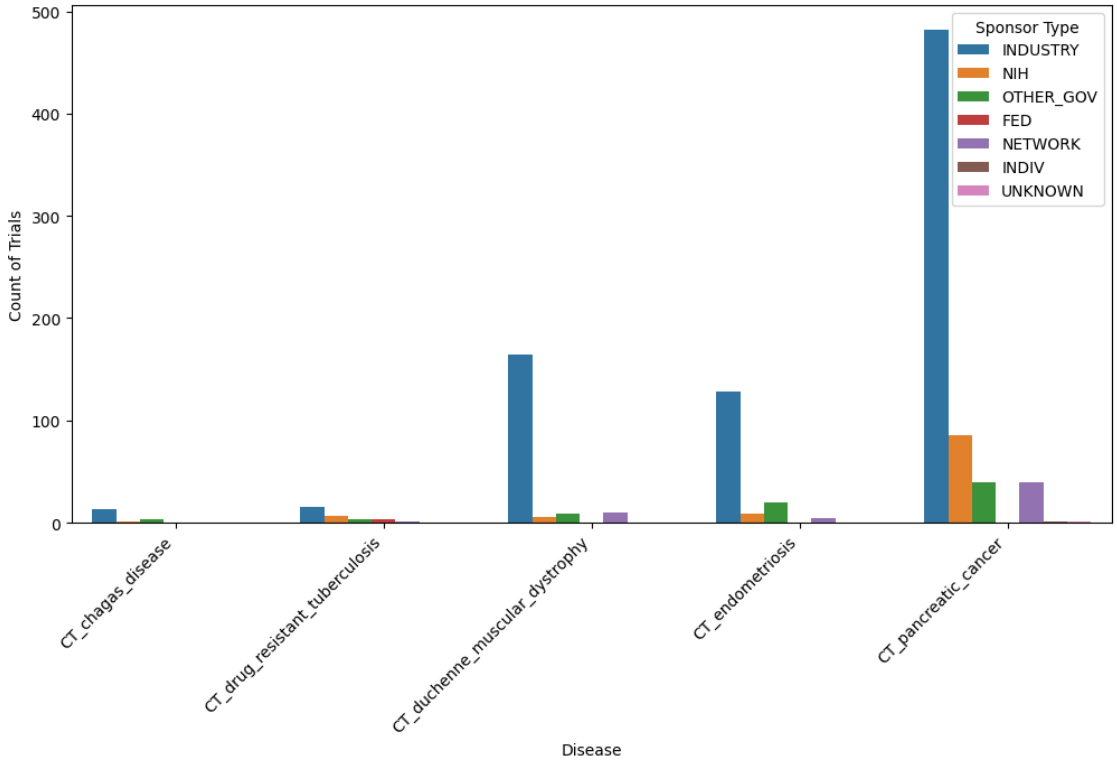
```
17  CT_duchenne_muscular_dystrophy                COMPLETED    178
18  CT_duchenne_muscular_dystrophy  ENROLLING_BY_INVITATION      6
19  CT_duchenne_muscular_dystrophy       NOT_YET_RECRUITING     14
20  CT_duchenne_muscular_dystrophy      NO_LONGER_AVAILABLE      2
21  CT_duchenne_muscular_dystrophy                RECRUITING     47
22  CT_duchenne_muscular_dystrophy                TERMINATED     42
23  CT_duchenne_muscular_dystrophy                   UNKNOWN     33
24  CT_duchenne_muscular_dystrophy                 WITHDRAWN      8
25                CT_endometriosis    ACTIVE_NOT_RECRUITING     25
26                CT_endometriosis                COMPLETED    266
27                CT_endometriosis  ENROLLING_BY_INVITATION      9
28                CT_endometriosis       NOT_YET_RECRUITING     30
29                CT_endometriosis                RECRUITING    110
30                CT_endometriosis                TERMINATED     28
31                CT_endometriosis                   UNKNOWN    107
32                CT_endometriosis                 WITHDRAWN     14
33            CT_pancreatic_cancer    ACTIVE_NOT_RECRUITING    165
34            CT_pancreatic_cancer   APPROVED_FOR_MARKETING      1
35            CT_pancreatic_cancer                AVAILABLE      2
36            CT_pancreatic_cancer                COMPLETED   1010
37            CT_pancreatic_cancer  ENROLLING_BY_INVITATION     14
38            CT_pancreatic_cancer       NOT_YET_RECRUITING    143
39            CT_pancreatic_cancer      NO_LONGER_AVAILABLE      5
40            CT_pancreatic_cancer                RECRUITING    481
41            CT_pancreatic_cancer                 SUSPENDED     10
42            CT_pancreatic_cancer                TERMINATED    264
43            CT_pancreatic_cancer                   UNKNOWN    373
44            CT_pancreatic_cancer                 WITHDRAWN    112
TRIAL STATUS PRINTED
```

[19]:
```
# TRIAL SPONSOR PRINTED

trial_sponsor = df.groupby(['Disease', 'Sponsor Type']).size().
  ↪reset_index(name='count')
print(trial_sponsor)
print("TRIAL SPONSOR PRINTED")
```

```
                          Disease Sponsor Type  count
0                CT_chagas_disease     INDUSTRY     13
1                CT_chagas_disease          NIH      1
2                CT_chagas_disease        OTHER     38
3                CT_chagas_disease    OTHER_GOV      3
4   CT_drug_resistant_tuberculosis          FED      3
5   CT_drug_resistant_tuberculosis     INDUSTRY     15
6   CT_drug_resistant_tuberculosis      NETWORK      1
7   CT_drug_resistant_tuberculosis          NIH      7
8   CT_drug_resistant_tuberculosis        OTHER     31
9   CT_drug_resistant_tuberculosis    OTHER_GOV      3
```

```
10  CT_duchenne_muscular_dystrophy     INDUSTRY    164
11  CT_duchenne_muscular_dystrophy      NETWORK     10
12  CT_duchenne_muscular_dystrophy          NIH      6
13  CT_duchenne_muscular_dystrophy        OTHER    175
14  CT_duchenne_muscular_dystrophy    OTHER_GOV      9
15                CT_endometriosis     INDUSTRY    128
16                CT_endometriosis      NETWORK      4
17                CT_endometriosis          NIH      9
18                CT_endometriosis        OTHER    428
19                CT_endometriosis    OTHER_GOV     20
20            CT_pancreatic_cancer        INDIV      1
21            CT_pancreatic_cancer     INDUSTRY    482
22            CT_pancreatic_cancer      NETWORK     39
23            CT_pancreatic_cancer          NIH     85
24            CT_pancreatic_cancer        OTHER   1933
25            CT_pancreatic_cancer    OTHER_GOV     39
26            CT_pancreatic_cancer      UNKNOWN      1
TRIAL SPONSOR PRINTED
```

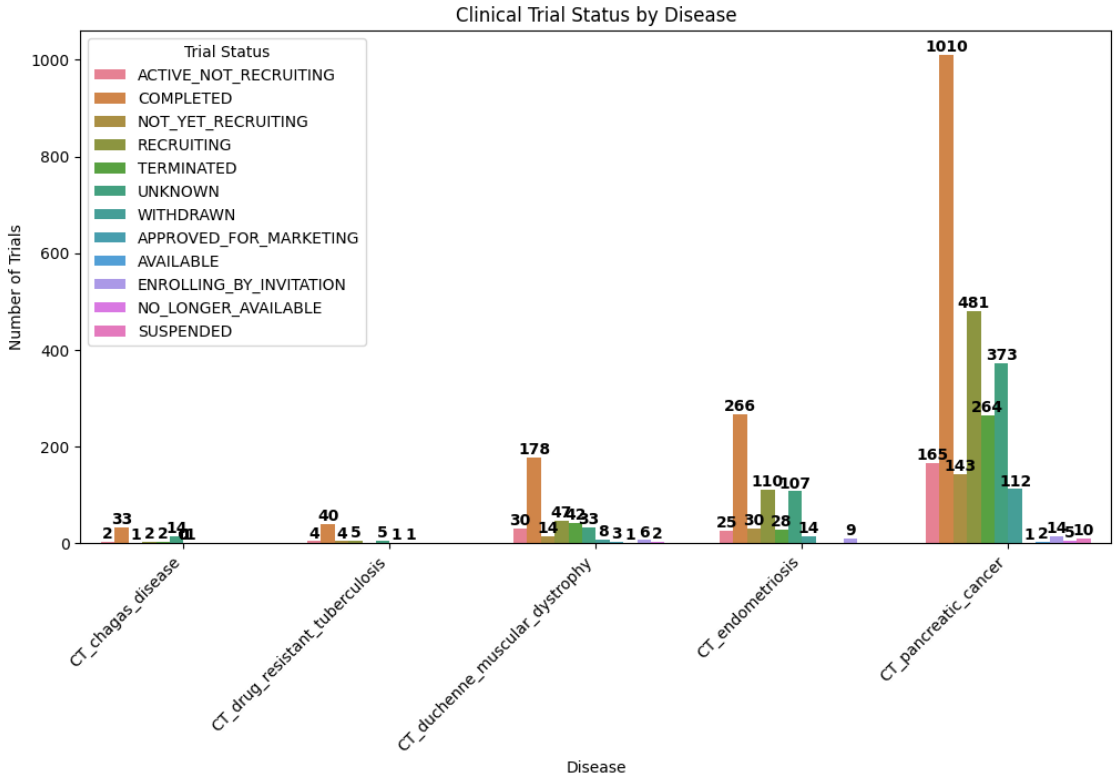
[20]:
```python
import seaborn as sns
import matplotlib.pyplot as plt

# Filter out 'OTHER' sponsor type
filtered_df = df[df["Sponsor Type"] != "OTHER"]

# Group data by Disease and Sponsor Type after filtering
trial_counts_filtered = filtered_df.groupby(["Disease", "Sponsor Type"]).size().
 ↪reset_index(name="Count")

# Create the balanced bar plot without 'OTHER' sponsor type
plt.figure(figsize=(12, 6))
sns.barplot(data=trial_counts_filtered, x="Disease", y="Count", hue="Sponsor␣
 ↪Type")

# Customize the plot
# plt.title("Balanced Clinical Trials Count by Disease and Sponsor Type␣
 ↪(Excluding 'OTHER')")
plt.xlabel("Disease")
plt.ylabel("Count of Trials")
plt.xticks(rotation=45, ha="right")
plt.legend(title="Sponsor Type")

# Show the plot
plt.show()
```



[21]:
```
study_type = df.groupby(['Disease', 'Study Type']).size().
  ↪reset_index(name='count')

# Print result
print(study_type)
print("STUDY TYPE PRINTED")
```

```
                         Disease       Study Type  count
0                CT_chagas_disease   INTERVENTIONAL     37
1                CT_chagas_disease    OBSERVATIONAL     18
2   CT_drug_resistant_tuberculosis  EXPANDED_ACCESS      1
3   CT_drug_resistant_tuberculosis   INTERVENTIONAL     39
4   CT_drug_resistant_tuberculosis    OBSERVATIONAL     20
5   CT_duchenne_muscular_dystrophy  EXPANDED_ACCESS      6
6   CT_duchenne_muscular_dystrophy   INTERVENTIONAL    266
7   CT_duchenne_muscular_dystrophy    OBSERVATIONAL     92
8                CT_endometriosis   INTERVENTIONAL    358
9                CT_endometriosis    OBSERVATIONAL    231
10            CT_pancreatic_cancer  EXPANDED_ACCESS      8
11            CT_pancreatic_cancer   INTERVENTIONAL   2116
12            CT_pancreatic_cancer    OBSERVATIONAL    456
STUDY TYPE PRINTED
```

[33]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# Set figure size
plt.figure(figsize=(12, 6))

trial_status = df.groupby(['Disease', 'Overall Status']).size().
 ↪reset_index(name='count')



# Create bar plot
ax = sns.barplot(data=trial_status, x="Disease", y="count", hue="Overall␣
 ↪Status")

# Add labels and title
plt.xlabel("Disease")
plt.ylabel("Number of Trials")
plt.title("Clinical Trial Status by Disease")

# Rotate x-axis labels for better readability
plt.xticks(rotation=45, ha="right")

# Show legend
plt.legend(title="Trial Status")

# Add numbers on top of bars
for p in ax.patches:
    ax.annotate(
        f'{int(p.get_height())}',  # Convert to int for cleaner display
        (p.get_x() + p.get_width() / 2., p.get_height()),  # Positioning
        ha='center', va='bottom',  # Alignment
        fontsize=10, fontweight='bold', color='black'  # Styling
    )

# Show the plot
plt.show()
```



[22]:
```
study_phases = df.groupby(['Disease', 'Interventions']).size().
  ↪reset_index(name='count')

# Print result
print(study_phases)
print("STUDY Phases PRINTED")
```

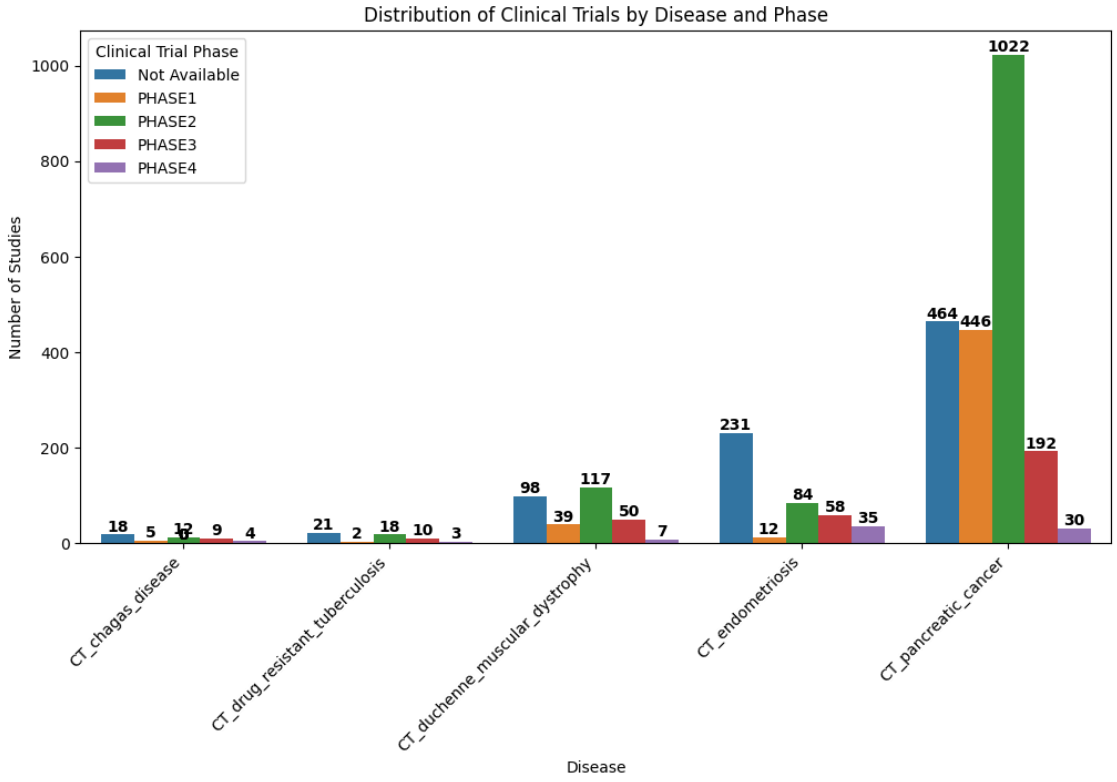
```
                 Disease                                        Interventions  \
0        CT_chagas_disease  40 mg Atorvastatin/day for 120 days P.O., Ator…
1        CT_chagas_disease                                             Aspirin
2        CT_chagas_disease                                         Benznidazole
3        CT_chagas_disease  Benznidazole, E1224, E1224 Placebo, Benznidazo…
4        CT_chagas_disease                         Benznidazole, Nifurtimox
…                      …                                                …
3140  CT_pancreatic_cancer                 without APA program, APA program
3141  CT_pancreatic_cancer       wound care management, wound care management
3142  CT_pancreatic_cancer     wrist-worn accelerometer, Auto-questionnaires
3143  CT_pancreatic_cancer                         zolbetuximab, mFOLFIRINOX
3144  CT_pancreatic_cancer          zolbetuximab, nab-paclitaxel, gemcitabine

      count
0         1
```

```
1          1
2          6
3          1
4          1
…      …
3140       1
3141       1
3142       1
3143       1
3144       1

[3145 rows x 3 columns]
STUDY Phases PRINTED
```

[23]:
```
study_phases = df.groupby(['Disease', 'Phases']).size().
  ↪reset_index(name='count')

# Print result
print(study_phases)
print("STUDY Phases PRINTED")
```

```
                          Disease         Phases  count
0                CT_chagas_disease  Not Available     18
1                CT_chagas_disease         PHASE1      5
2                CT_chagas_disease         PHASE2     12
3                CT_chagas_disease         PHASE3      9
4                CT_chagas_disease         PHASE4      4
5   CT_drug_resistant_tuberculosis  Not Available     21
6   CT_drug_resistant_tuberculosis         PHASE1      2
7   CT_drug_resistant_tuberculosis         PHASE2     18
8   CT_drug_resistant_tuberculosis         PHASE3     10
9   CT_drug_resistant_tuberculosis         PHASE4      3
10  CT_duchenne_muscular_dystrophy  Not Available     98
11  CT_duchenne_muscular_dystrophy         PHASE1     39
12  CT_duchenne_muscular_dystrophy         PHASE2    117
13  CT_duchenne_muscular_dystrophy         PHASE3     50
14  CT_duchenne_muscular_dystrophy         PHASE4      7
15                CT_endometriosis  Not Available    231
16                CT_endometriosis         PHASE1     12
17                CT_endometriosis         PHASE2     84
18                CT_endometriosis         PHASE3     58
19                CT_endometriosis         PHASE4     35
20            CT_pancreatic_cancer  Not Available    464
21            CT_pancreatic_cancer         PHASE1    446
22            CT_pancreatic_cancer         PHASE2   1022
23            CT_pancreatic_cancer         PHASE3    192
24            CT_pancreatic_cancer         PHASE4     30
STUDY Phases PRINTED
```

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



# Set figure size
plt.figure(figsize=(12, 6))

# Create bar plot
ax = sns.barplot(data=study_phases, x="Disease", y="count", hue="Phases")

# Add labels and title
plt.xlabel("Disease")
plt.ylabel("Number of Studies")
plt.title("Distribution of Clinical Trials by Disease and Phase")

# Rotate x-axis labels for better readability
plt.xticks(rotation=45, ha="right")

# Show legend
plt.legend(title="Clinical Trial Phase")

# Add numbers on top of bars
for p in ax.patches:
    ax.annotate(
        f'{int(p.get_height())}',  # Convert to int for cleaner display
        (p.get_x() + p.get_width() / 2., p.get_height()),  # Positioning
        ha='center', va='bottom',  # Alignment
        fontsize=10, fontweight='bold', color='black'  # Styling
    )

# Show the plot
plt.show()
```



[ ]:
```
import pandas as pd
import folium
import time
from geopy.geocoders import Nominatim
from geopy.exc import GeocoderTimedOut
from folium.plugins import MarkerCluster

# Load the dataset
file_path = "data/CT_all_rare_disease_procesed.csv"  # Update with your file␣
 ↪path
df = pd.read_csv(file_path)

# Extract unique locations
df["Cleaned_Location"] = df["Locations"].str.split(",").str[-2:].apply(lambda x:
 ↪ ", ".join(x).strip() if len(x) == 2 else None)
df = df.dropna(subset=["Cleaned_Location"])

# Initialize geocoder
geolocator = Nominatim(user_agent="clinical_trials_mapper")

# Dictionary to store geocoded locations
location_coords = {}
```

```python
# Function to get coordinates with retry handling
def get_coordinates(location):
    try:
        loc = geolocator.geocode(location, timeout=10)
        if loc:
            return (loc.latitude, loc.longitude)
    except GeocoderTimedOut:
        time.sleep(1)
        return get_coordinates(location)  # Retry once if timeout occurs
    return (None, None)

# Check if geocoded locations already exist to avoid redundant API calls
geocoded_file = "geocoded_locations.csv"
try:
    geocoded_df = pd.read_csv(geocoded_file)
    location_coords = dict(zip(geocoded_df["Location"],␣
 ↪zip(geocoded_df["Latitude"], geocoded_df["Longitude"])))
except FileNotFoundError:
    pass  # Proceed with geocoding if file does not exist

# Geocode only missing locations
unique_locations = df["Cleaned_Location"].dropna().unique()
for location in unique_locations:
    if location not in location_coords:
        lat, lon = get_coordinates(location)
        location_coords[location] = (lat, lon)
        time.sleep(1)  # Avoid API rate limits

# Save geocoded locations to prevent re-fetching
geo_df = pd.DataFrame(location_coords.items(), columns=["Location", "Coords"])
geo_df["Latitude"], geo_df["Longitude"] = zip(*geo_df["Coords"])
geo_df.drop(columns=["Coords"], inplace=True)
geo_df.to_csv(geocoded_file, index=False)

# Merge geocoded coordinates back into the dataset
df["Latitude"] = df["Cleaned_Location"].map(lambda loc: location_coords.
 ↪get(loc, (None, None))[0])
df["Longitude"] = df["Cleaned_Location"].map(lambda loc: location_coords.
 ↪get(loc, (None, None))[1])

# Remove rows with missing coordinates
df_cleaned = df.dropna(subset=["Latitude", "Longitude"])

# Define colors for each disease
disease_colors = {
    "pancreatic_cancer": "red",
```

```
    "chagas_disease": "blue",
    "endometriosis": "purple",
    "drug_resistant_tuberculosis": "green",
    "duchenne_muscular_dystrophy": "orange",
}

# Create a folium map centered globally
map_center = [df_cleaned["Latitude"].mean(), df_cleaned["Longitude"].mean()]
m = folium.Map(location=map_center, zoom_start=2)

# Create a dictionary for feature groups (for filtering by disease)
disease_layers = {disease: folium.FeatureGroup(name=disease.replace("_", " ").
 ↪title()) for disease in disease_colors.keys()}

# Add markers for each disease to its respective feature group
for _, row in df_cleaned.iterrows():
    disease = row["Disease"].lower().replace(" ", "_")
    if disease in disease_layers:
        color = disease_colors[disease]
        folium.Marker(
            location=[row["Latitude"], row["Longitude"]],
            popup=f"{row['Disease']} - {row['Locations']}",
            icon=folium.Icon(color=color),
        ).add_to(disease_layers[disease])

# Add each disease layer to the map
for layer in disease_layers.values():
    layer.add_to(m)

# Add layer control to toggle diseases on/off
folium.LayerControl().add_to(m)

# Save and display the map
m.save("clinical_trials_map.html")
m
```

2 Common Diseases

[24]:
```
import pandas as pd

# Load data
data = pd.read_csv('data/CT_all_common_disease_processed.csv')
df1 = pd.DataFrame(data)

# Group by 'disease' column and count unique values in 'Overall Status'
# unique_counts = df.groupby('Disease')['Overall Status'].value_counts()
```

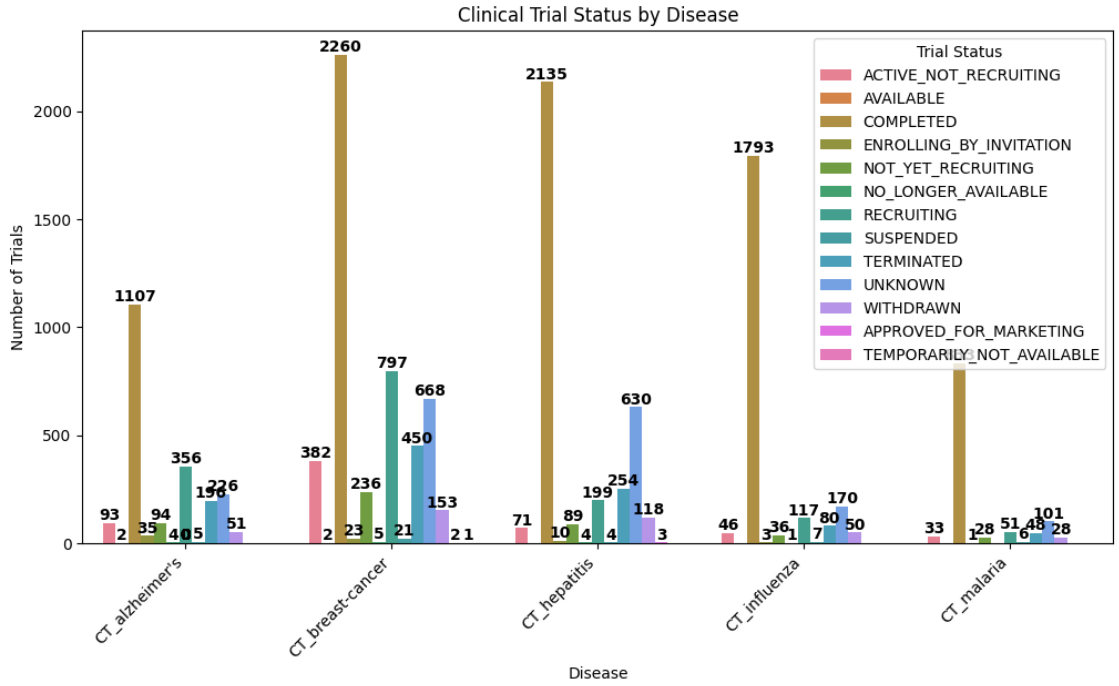
```
# finding trial status
trial_status1 = df1.groupby(['Disease', 'Overall Status']).size().
 ↪reset_index(name='count')


# Print result
print(trial_status1)
print("TRIAL STATUS PRINTED")
```

```
             Disease             Overall Status  count
0     CT_alzheimer's      ACTIVE_NOT_RECRUITING     93
1     CT_alzheimer's                 AVAILABLE      2
2     CT_alzheimer's                 COMPLETED   1107
3     CT_alzheimer's    ENROLLING_BY_INVITATION     35
4     CT_alzheimer's         NOT_YET_RECRUITING     94
5     CT_alzheimer's        NO_LONGER_AVAILABLE      4
6     CT_alzheimer's                 RECRUITING    356
7     CT_alzheimer's                 SUSPENDED      5
8     CT_alzheimer's                 TERMINATED    196
9     CT_alzheimer's                    UNKNOWN    226
10    CT_alzheimer's                 WITHDRAWN     51
11  CT_breast-cancer      ACTIVE_NOT_RECRUITING    382
12  CT_breast-cancer     APPROVED_FOR_MARKETING      2
13  CT_breast-cancer                 AVAILABLE      2
14  CT_breast-cancer                 COMPLETED   2260
15  CT_breast-cancer    ENROLLING_BY_INVITATION     23
16  CT_breast-cancer         NOT_YET_RECRUITING    236
17  CT_breast-cancer        NO_LONGER_AVAILABLE      5
18  CT_breast-cancer                 RECRUITING    797
19  CT_breast-cancer                 SUSPENDED     21
20  CT_breast-cancer  TEMPORARILY_NOT_AVAILABLE      1
21  CT_breast-cancer                 TERMINATED    450
22  CT_breast-cancer                    UNKNOWN    668
23  CT_breast-cancer                 WITHDRAWN    153
24      CT_hepatitis      ACTIVE_NOT_RECRUITING     71
25      CT_hepatitis     APPROVED_FOR_MARKETING      3
26      CT_hepatitis                 COMPLETED   2135
27      CT_hepatitis    ENROLLING_BY_INVITATION     10
28      CT_hepatitis         NOT_YET_RECRUITING     89
29      CT_hepatitis        NO_LONGER_AVAILABLE      4
30      CT_hepatitis                 RECRUITING    199
31      CT_hepatitis                 SUSPENDED      4
32      CT_hepatitis                 TERMINATED    254
33      CT_hepatitis                    UNKNOWN    630
34      CT_hepatitis                 WITHDRAWN    118
35      CT_influenza      ACTIVE_NOT_RECRUITING     46
```

```
36      CT_influenza                  COMPLETED   1793
37      CT_influenza    ENROLLING_BY_INVITATION      3
38      CT_influenza         NOT_YET_RECRUITING     36
39      CT_influenza        NO_LONGER_AVAILABLE      1
40      CT_influenza                  RECRUITING    117
41      CT_influenza                   SUSPENDED      7
42      CT_influenza                  TERMINATED     80
43      CT_influenza                     UNKNOWN    170
44      CT_influenza                   WITHDRAWN     50
45        CT_malaria      ACTIVE_NOT_RECRUITING     33
46        CT_malaria                  COMPLETED    833
47        CT_malaria    ENROLLING_BY_INVITATION      1
48        CT_malaria         NOT_YET_RECRUITING     28
49        CT_malaria                  RECRUITING     51
50        CT_malaria                   SUSPENDED      6
51        CT_malaria                  TERMINATED     48
52        CT_malaria                     UNKNOWN    101
53        CT_malaria                   WITHDRAWN     28
TRIAL STATUS PRINTED
```

[32]:
```python
import matplotlib.pyplot as plt
import seaborn as sns

# Set figure size
plt.figure(figsize=(12, 6))

# Create bar plot
ax = sns.barplot(data=trial_status1, x="Disease", y="count", hue="Overall␣
 ↪Status")

# Add labels and title
plt.xlabel("Disease")
plt.ylabel("Number of Trials")
plt.title("Clinical Trial Status by Disease")

# Rotate x-axis labels for better readability
plt.xticks(rotation=45, ha="right")

# Show legend
plt.legend(title="Trial Status")

# Add numbers on top of bars
for p in ax.patches:
    ax.annotate(
        f'{int(p.get_height())}',  # Convert to int for cleaner display
        (p.get_x() + p.get_width() / 2., p.get_height()),  # Positioning
        ha='center', va='bottom',  # Alignment
```

```
        fontsize=10, fontweight='bold', color='black'  # Styling
    )

# Show the plot
plt.show()
```



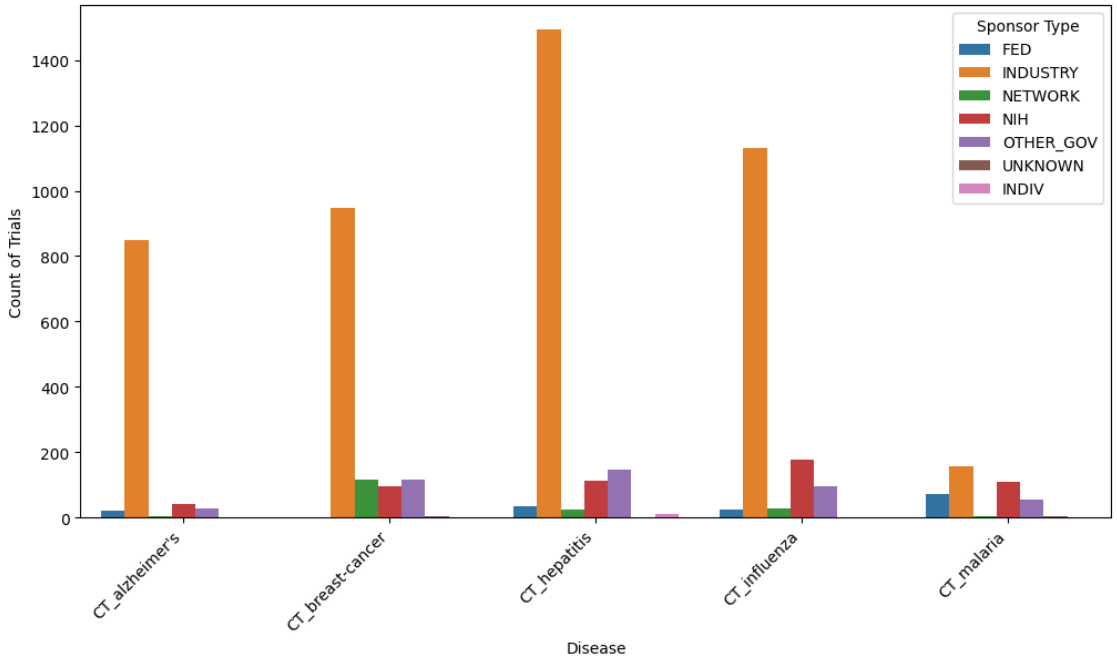
[25]:
```
trial_sponsor1 = df1.groupby(['Disease', 'Sponsor Type']).size().
  ↪reset_index(name='count')
print(trial_sponsor1)
print("TRIAL SPONSOR PRINTED")
```

```
             Disease Sponsor Type  count
0     CT_alzheimer's          FED     20
1     CT_alzheimer's     INDUSTRY    848
2     CT_alzheimer's      NETWORK      5
3     CT_alzheimer's          NIH     41
4     CT_alzheimer's        OTHER   1226
5     CT_alzheimer's    OTHER_GOV     29
6   CT_breast-cancer          FED      1
7   CT_breast-cancer     INDUSTRY    947
8   CT_breast-cancer      NETWORK    115
9   CT_breast-cancer          NIH     96
10  CT_breast-cancer        OTHER   3722
11  CT_breast-cancer    OTHER_GOV    117
12  CT_breast-cancer      UNKNOWN      2
```

```
13      CT_hepatitis          FED     34
14      CT_hepatitis        INDIV      9
15      CT_hepatitis     INDUSTRY   1494
16      CT_hepatitis      NETWORK     23
17      CT_hepatitis          NIH    112
18      CT_hepatitis        OTHER   1699
19      CT_hepatitis    OTHER_GOV    146
20      CT_influenza          FED     24
21      CT_influenza        INDIV      1
22      CT_influenza     INDUSTRY   1130
23      CT_influenza      NETWORK     29
24      CT_influenza          NIH    178
25      CT_influenza        OTHER    847
26      CT_influenza    OTHER_GOV     94
27        CT_malaria          FED     72
28        CT_malaria     INDUSTRY    157
29        CT_malaria      NETWORK      5
30        CT_malaria          NIH    110
31        CT_malaria        OTHER    726
32        CT_malaria    OTHER_GOV     56
33        CT_malaria      UNKNOWN      3
TRIAL SPONSOR PRINTED
```

[26]:
```python
import seaborn as sns
import matplotlib.pyplot as plt

# Filter out 'OTHER' sponsor type
filtered_df = df1[df1["Sponsor Type"] != "OTHER"]

# Group data by Disease and Sponsor Type after filtering
trial_counts_filtered = filtered_df.groupby(["Disease", "Sponsor Type"]).size().
 ↪reset_index(name="Count")

# Create the balanced bar plot without 'OTHER' sponsor type
plt.figure(figsize=(12, 6))
sns.barplot(data=trial_counts_filtered, x="Disease", y="Count", hue="Sponsor␣
 ↪Type")

# Customize the plot
# plt.title("Balanced Clinical Trials Count by Disease and Sponsor Type␣
 ↪(Excluding 'OTHER')")
plt.xlabel("Disease")
plt.ylabel("Count of Trials")
plt.xticks(rotation=45, ha="right")
plt.legend(title="Sponsor Type")

# Show the plot
```

```
plt.show()
```



[27]:
```
study_type1 = df1.groupby(['Disease', 'Study Type']).size().
  ↪reset_index(name='count')

# Print result
print(study_type1)
print("STUDY TYPE PRINTED")
```

```
             Disease       Study Type  count
0     CT_alzheimer's  EXPANDED_ACCESS      6
1     CT_alzheimer's   INTERVENTIONAL   1778
2     CT_alzheimer's    OBSERVATIONAL    385
3   CT_breast-cancer  EXPANDED_ACCESS     10
4   CT_breast-cancer   INTERVENTIONAL   3990
5   CT_breast-cancer    OBSERVATIONAL   1000
6       CT_hepatitis  EXPANDED_ACCESS      7
7       CT_hepatitis   INTERVENTIONAL   2617
8       CT_hepatitis    OBSERVATIONAL    893
9       CT_influenza  EXPANDED_ACCESS      1
10      CT_influenza   INTERVENTIONAL   1963
11      CT_influenza    OBSERVATIONAL    339
12        CT_malaria   INTERVENTIONAL    948
13        CT_malaria    OBSERVATIONAL    181
STUDY TYPE PRINTED
```

[15]:
```
study_phases1 = df1.groupby(['Disease', 'Phases']).size().
 ↪reset_index(name='count')

# Print result
print(study_phases1)
print("STUDY Phases PRINTED")
```

```
             Disease         Phases  count
0     CT_alzheimer's  Not Available    391
1     CT_alzheimer's         PHASE1    240
2     CT_alzheimer's         PHASE2    516
3     CT_alzheimer's         PHASE3    273
4     CT_alzheimer's         PHASE4     85
5   CT_breast-cancer  Not Available   1010
6   CT_breast-cancer         PHASE1    458
7   CT_breast-cancer         PHASE2   1517
8   CT_breast-cancer         PHASE3    589
9   CT_breast-cancer         PHASE4     96
10      CT_hepatitis  Not Available    900
11      CT_hepatitis         PHASE1    275
12      CT_hepatitis         PHASE2    725
13      CT_hepatitis         PHASE3    597
14      CT_hepatitis         PHASE4    610
15      CT_influenza  Not Available    340
16      CT_influenza         PHASE1    307
17      CT_influenza         PHASE2    510
18      CT_influenza         PHASE3    453
19      CT_influenza         PHASE4    369
20        CT_malaria  Not Available    181
21        CT_malaria         PHASE1    175
22        CT_malaria         PHASE2    202
23        CT_malaria         PHASE3    180
24        CT_malaria         PHASE4    177
STUDY Phases PRINTED
```

[30]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# Set figure size
plt.figure(figsize=(12, 6))

# Create bar plot
ax = sns.barplot(data=study_phases1, x="Disease", y="count", hue="Phases")

# Add labels and title
plt.xlabel("Disease")
plt.ylabel("Number of Studies")
```

```
plt.title("Distribution of Clinical Trials by Disease and Phase")

# Rotate x-axis labels for better readability
plt.xticks(rotation=45, ha="right")

# Show legend
plt.legend(title="Clinical Trial Phase")

# Add numbers on top of bars
for p in ax.patches:
    ax.annotate(
        f'{int(p.get_height())}',  # Convert to int for cleaner display
        (p.get_x() + p.get_width() / 2., p.get_height()),  # Positioning
        ha='center', va='bottom',  # Alignment
        fontsize=10, fontweight='bold', color='black'  # Styling
    )

# Show the plot
plt.show()
```