Space Debris - Exploratory Data Analysis

Blake Rayvid - https://github.com/brayvid

Dataset: https://www.kaggle.com/datasets/kandhalkhandeka/satellites-and-debris-in-earths-orbit

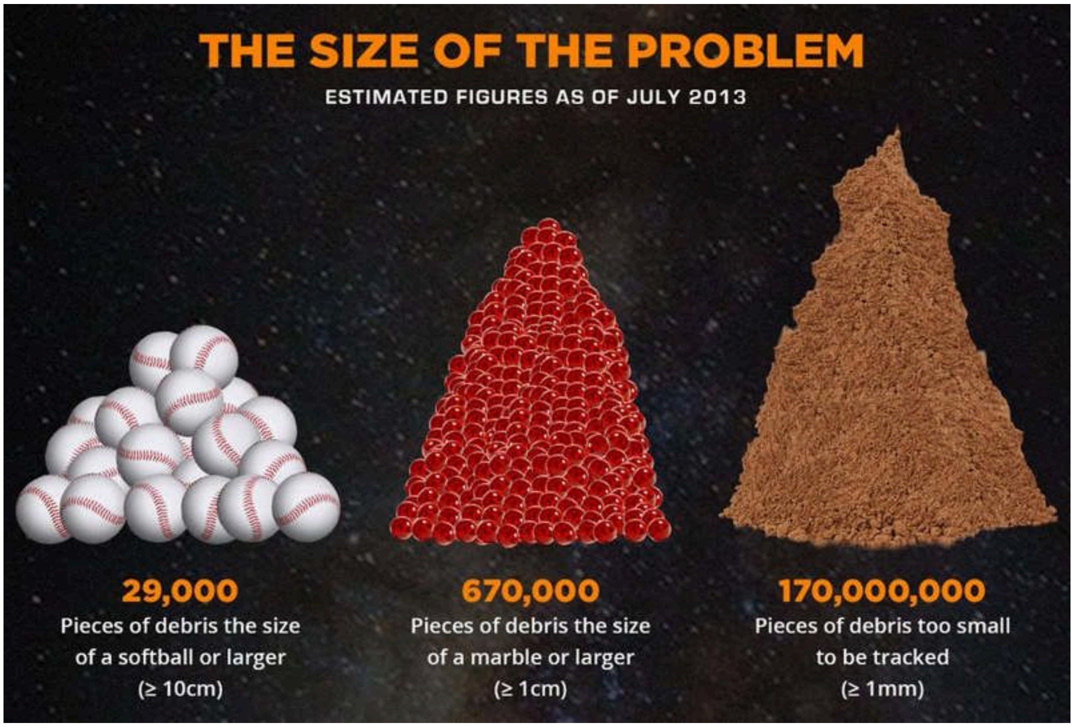


Image link

Dataset overview

This dataset contains roughly 14,000 objects, classified as `DEBRIS`, `PAYLOAD`, `ROCKET` or unknown, and either `LARGE`, `MEDIUM`, `SMALL` or unknown.

According to Google's AI chatbot, space debris range from:
"Larger than 1 mm: 100 million - 170 million objects
Larger than 1 cm: 670,000 objects
Larger than 10 cm: 25,000 - 29,000 objects"

So we know this dataset does not contain all debris objects.

We will investigate the object size and type columns, the orbital parameters, the country of origin and launch date.

Note: This dataset does not contain any mass estimates.

The total mass of all space debris is known to exceed 9,000 metric tons. Source

Read and clean data

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt; plt.rcParams["figure.dpi"] = 144
import seaborn as sns
from matplotlib import ticker
```

```python
# Check out the columns
df = pd.read_csv('space_decay.csv')
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14372 entries, 0 to 14371
Data columns (total 40 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   CCSDS_OMM_VERS       14372 non-null  int64  
 1   COMMENT              14372 non-null  object 
 2   CREATION_DATE        14372 non-null  object 
 3   ORIGINATOR           14372 non-null  object 
 4   OBJECT_NAME          14372 non-null  object 
 5   OBJECT_ID            14333 non-null  object 
 6   CENTER_NAME          14372 non-null  object 
 7   REF_FRAME            14372 non-null  object 
 8   TIME_SYSTEM          14372 non-null  object 
 9   MEAN_ELEMENT_THEORY  14372 non-null  object 
 10  EPOCH                14372 non-null  object 
 11  MEAN_MOTION          14372 non-null  float64
 12  ECCENTRICITY         14372 non-null  float64
 13  INCLINATION          14372 non-null  float64
 14  RA_OF_ASC_NODE       14372 non-null  float64
 15  ARG_OF_PERICENTER    14372 non-null  float64
 16  MEAN_ANOMALY         14372 non-null  float64
 17  EPHEMERIS_TYPE       14372 non-null  int64  
 18  CLASSIFICATION_TYPE  14372 non-null  object 
 19  NORAD_CAT_ID         14372 non-null  int64  
 20  ELEMENT_SET_NO       14372 non-null  int64  
 21  REV_AT_EPOCH         14372 non-null  int64  
 22  BSTAR                14372 non-null  float64
 23  MEAN_MOTION_DOT      14372 non-null  float64
 24  MEAN_MOTION_DDOT     14372 non-null  float64
 25  SEMIMAJOR_AXIS       14372 non-null  float64
 26  PERIOD               14372 non-null  float64
 27  APOAPSIS             14372 non-null  float64
 28  PERIAPSIS            14372 non-null  float64
 29  OBJECT_TYPE          14372 non-null  object 
 30  RCS_SIZE             14174 non-null  object 
 31  COUNTRY_CODE         14333 non-null  object 
 32  LAUNCH_DATE          14333 non-null  float64
 33  SITE                 14333 non-null  object 
 34  DECAY_DATE           0 non-null      float64
 35  FILE                 14372 non-null  int64  
 36  GP_ID                14372 non-null  int64  
 37  TLE_LINE0            14372 non-null  object 
 38  TLE_LINE1            14372 non-null  object 
 39  TLE_LINE2            14372 non-null  object 
dtypes: float64(15), int64(7), object(18)
memory usage: 4.4+ MB
```

```python
# Check out the values of important columns
df['RCS_SIZE'].unique()
```

```
array(['MEDIUM', 'SMALL', 'LARGE', nan], dtype=object)
```

```python
df['OBJECT_TYPE'].unique()
```

```
array(['DEBRIS', 'PAYLOAD', 'ROCKET BODY', 'TBA'], dtype=object)
```

```python
df['COUNTRY_CODE'].unique()
```

```
array(['FR', 'CIS', 'IND', 'PRC', 'US', 'SEAL', 'ITSO', 'SES', 'NICO',
       'ESA', 'JPN', 'EUTE', 'GER', 'PAKI', 'EUME', 'SPN', 'ALG', 'AUS',
       'ARGN', 'SAUD', 'IT', 'CA', 'AC', 'GLOB', 'GREC', 'UAE', 'DEN',
       'NIG', 'UK', 'TURK', 'SKOR', 'CHBZ', 'ISRA', 'USBZ', 'ROC', 'IM',
       'THAI', 'IRAN', 'INDO', 'KAZ', 'AB', 'MALA', 'EGYP', 'COL', 'BRAZ',
       'RASC', 'NOR', 'VTNM', 'NETH', 'ORB', 'VENZ', 'SAFR', 'SWTZ',
       'SWED', 'UKR', 'SING', 'STCT', 'MEX', 'LUXE', 'CHLE', 'BELA',
       'NKOR', 'AZER', 'ASRA', 'ECU', 'EST', 'O3B', 'QAT', 'POL', 'PER',
       'BOL', 'FRIT', 'BEL', 'URY', 'IRAQ', 'TMMC', 'LAOS', 'BERM', 'LTU',
       'FIN', 'CZCH', 'BGR', 'TBD', 'MA', 'AGO', 'BGD', 'RP', 'JOR',
```

```
       'ISS', 'SDN', 'RWA', 'SVN', 'NZ', 'PRY', 'TUN', 'HUN', 'MMR',
       'MUS', 'KWT', nan], dtype=object)
```

```
# Show how many elements of each column are NaN
df.isna().sum()
```

```
CCSDS_OMM_VERS            0
COMMENT                   0
CREATION_DATE             0
ORIGINATOR                0
OBJECT_NAME               0
OBJECT_ID                39
CENTER_NAME               0
REF_FRAME                 0
TIME_SYSTEM               0
MEAN_ELEMENT_THEORY       0
EPOCH                     0
MEAN_MOTION               0
ECCENTRICITY              0
INCLINATION               0
RA_OF_ASC_NODE            0
ARG_OF_PERICENTER         0
MEAN_ANOMALY              0
EPHEMERIS_TYPE            0
CLASSIFICATION_TYPE       0
NORAD_CAT_ID              0
ELEMENT_SET_NO            0
REV_AT_EPOCH              0
BSTAR                     0
MEAN_MOTION_DOT           0
MEAN_MOTION_DDOT          0
SEMIMAJOR_AXIS            0
PERIOD                    0
APOAPSIS                  0
PERIAPSIS                 0
OBJECT_TYPE               0
RCS_SIZE                198
COUNTRY_CODE             39
LAUNCH_DATE              39
SITE                     39
DECAY_DATE            14372
FILE                      0
GP_ID                     0
TLE_LINE0                 0
TLE_LINE1                 0
TLE_LINE2                 0
dtype: int64
```

```
# This column is always NaN so drop it entirely
df = df.drop(labels=['DECAY_DATE'], axis=1)
```

```
 # Rename the object type to be more presentable
df['OBJECT_TYPE'] = df['OBJECT_TYPE'].replace(to_replace={'DEBRIS':'Debris','PAYLOAD':'Payload','TBA': 'Unknown','ROCKET BOD
df['OBJECT_TYPE'].unique()
```

```
array(['Debris', 'Payload', 'Rocket', 'Unknown'], dtype=object)
```

```
 # Rename 'TBD' and NaN country codes to 'Unknown'
df['COUNTRY_CODE'] = df['COUNTRY_CODE'].replace(to_replace={'TBD':'Unknown',np.nan:'Unknown'})
df['COUNTRY_CODE'].unique()
```

```
array(['FR', 'CIS', 'IND', 'PRC', 'US', 'SEAL', 'ITSO', 'SES', 'NICO',
       'ESA', 'JPN', 'EUTE', 'GER', 'PAKI', 'EUME', 'SPN', 'ALG', 'AUS',
       'ARGN', 'SAUD', 'IT', 'CA', 'AC', 'GLOB', 'GREC', 'UAE', 'DEN',
       'NIG', 'UK', 'TURK', 'SKOR', 'CHBZ', 'ISRA', 'USBZ', 'ROC', 'IM',
       'THAI', 'IRAN', 'INDO', 'KAZ', 'AB', 'MALA', 'EGYP', 'COL', 'BRAZ',
       'RASC', 'NOR', 'VTNM', 'NETH', 'ORB', 'VENZ', 'SAFR', 'SWTZ',
       'SWED', 'UKR', 'SING', 'STCT', 'MEX', 'LUXE', 'CHLE', 'BELA',
       'NKOR', 'AZER', 'ASRA', 'ECU', 'EST', 'O3B', 'QAT', 'POL', 'PER',
       'BOL', 'FRIT', 'BEL', 'URY', 'IRAQ', 'TMMC', 'LAOS', 'BERM', 'LTU',
       'FIN', 'CZCH', 'BGR', 'Unknown', 'MA', 'AGO', 'BGD', 'RP', 'JOR',
       'ISS', 'SDN', 'RWA', 'SVN', 'NZ', 'PRY', 'TUN', 'HUN', 'MMR',
       'MUS', 'KWT'], dtype=object)
```

```
 # Rename object size for presentation
df['RCS_SIZE'] = df['RCS_SIZE'].replace(to_replace={'LARGE':'Large','MEDIUM':'Medium',"SMALL":"Small"})
df['RCS_SIZE'].unique()
```

```
array(['Medium', 'Small', 'Large', nan], dtype=object)
```

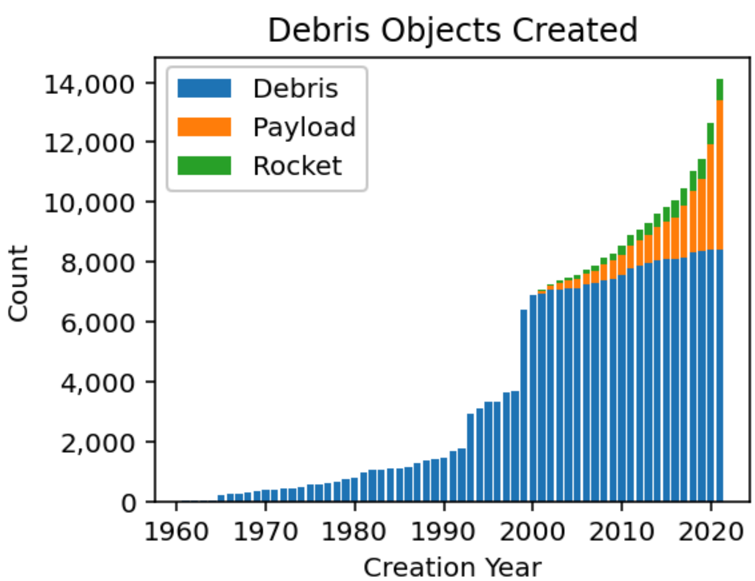
```
# Create a new column for the period in hours
df['PERIOD_HOURS'] = df['PERIOD'] / 60
```

```
# Create new column for the (maximum) altitude above Earth surface (Earth radius ~ 6371km)
df['ALTITUDE_MI'] = (df['SEMIMAJOR_AXIS'] - 6371) * 0.6213
```

Descriptive questions

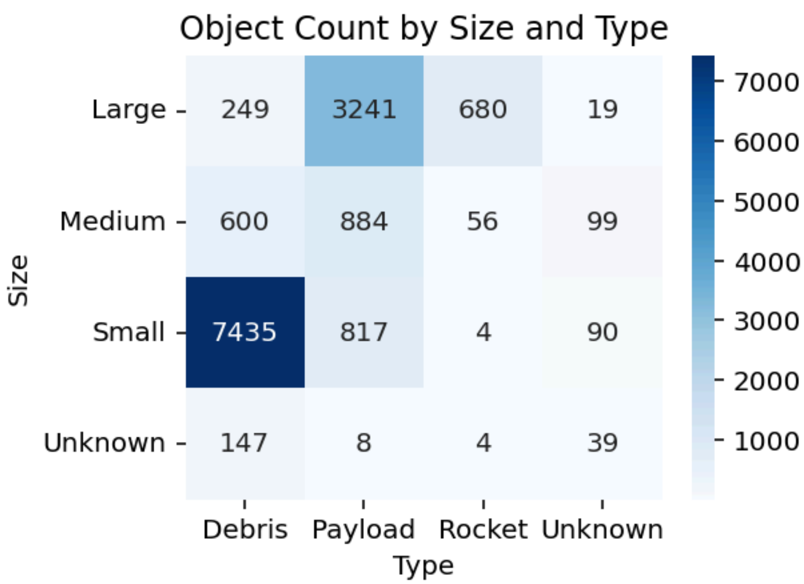
What time range does the dataset cover?

```
df['AggValue'] = 1
has_launch_date_df = df.dropna(subset='LAUNCH_DATE')
has_launch_date_year_counts = has_launch_date_df.pivot_table(values='AggValue', index='LAUNCH_DATE', columns='OBJECT_TYPE',

fig, ax = plt.subplots(figsize=(4, 3))
plt.bar(x=has_launch_date_year_counts.index, height=has_launch_date_year_counts['Debris'])
plt.bar(x=has_launch_date_year_counts.index, height=has_launch_date_year_counts['Payload'], bottom=has_launch_date_year_coun
plt.bar(x=has_launch_date_year_counts.index, height=has_launch_date_year_counts['Rocket'], bottom=(has_launch_date_year_coun
plt.legend(['Debris','Payload','Rocket'], framealpha=1, loc=(0.02,0.7))
plt.xlabel('Creation Year')
plt.ylabel('Count')
plt.title('Debris Objects Created')

# https://stackoverflow.com/questions/25973581/how-to-format-axis-number-format-to-thousands-with-a-comma
thousands_format = ticker.StrMethodFormatter('{x:,.0f}')
ax.yaxis.set_major_formatter(thousands_format)
plt.show()
```



How much of each size and type of space debris is there?

```
# Plot heatmap of size vs type
df_filled = df.copy()
df_filled.loc[:,'RCS_SIZE'] = df_filled['RCS_SIZE'].fillna('Unknown')
class_counts = df_filled.pivot_table(values='AggValue', index='RCS_SIZE', columns='OBJECT_TYPE', aggfunc=np.sum)

fig, ax = plt.subplots(figsize=(4, 3))
sns.heatmap(class_counts, cmap="Blues", annot=True, fmt='g')
plt.ylabel('Size')
plt.xlabel('Type')
plt.title("Object Count by Size and Type")
plt.yticks(rotation=0)
plt.show()
```

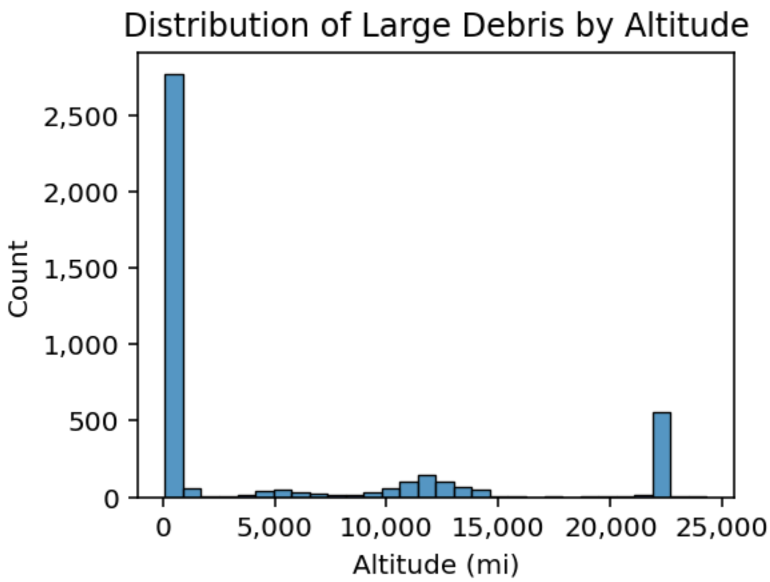


In what orbits are the debris mainly located?

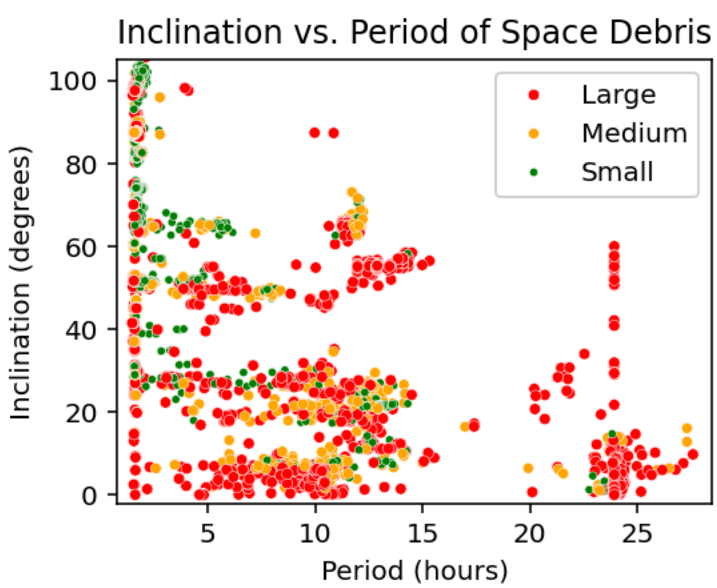
```
# Display statistics of the orbital elements
debris_elements_df = df[['ECCENTRICITY','INCLINATION', 'RA_OF_ASC_NODE','ARG_OF_PERICENTER', 'MEAN_ANOMALY','SEMIMAJOR_AXIS'
debris_elements_df.describe()
```

	ECCENTRICITY	INCLINATION	RA_OF_ASC_NODE	ARG_OF_PERICENTER	MEAN_ANOMALY
count	14372.000000	14372.000000	14372.000000	14372.000000	14372.000000
mean	0.067168	74.354208	182.353111	165.977199	191.015595
std	0.181547	29.626780	116.717713	104.461866	109.873807
min	0.000005	0.001400	0.020000	0.008200	0.004800
25%	0.000725	53.055300	71.987350	77.363475	90.001600
50%	0.003416	86.369200	189.460250	150.483900	206.957400
75%	0.013505	98.565125	292.484750	258.328550	284.163200
max	0.897218	144.586200	359.967900	359.989900	359.981900

```
# Plot count vs altitude histogram for just large debris
PERIOD_NO_OUTLIERS_LARGE_MASK = ((df['PERIOD_HOURS'] < 27) & (df['RCS_SIZE'] == 'Large'))
period_no_outliers_large_df = df[PERIOD_NO_OUTLIERS_LARGE_MASK]

fig, ax = plt.subplots(figsize=(4, 3))
sns.histplot(data=period_no_outliers_large_df, x="ALTITUDE_MI",bins=30)
plt.xlabel('Altitude (mi)')
plt.title('Distribution of Large Debris by Altitude')
ax.yaxis.set_major_formatter(thousands_format)
ax.xaxis.set_major_formatter(thousands_format)
plt.show()
```

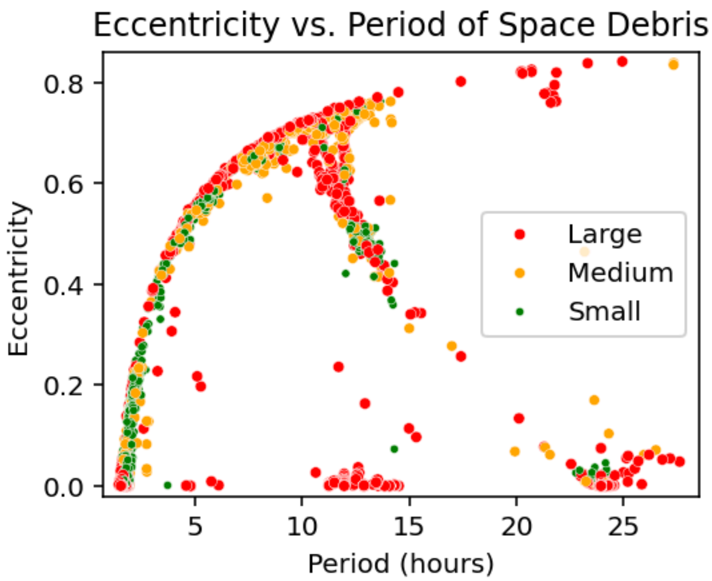



```
# Inclination vs period scatter plot
sizes = ['Large', 'Medium', 'Small']
sizes_palette = {'Small': 'green', 'Medium':'orange', 'Large': 'red'}

fig, ax = plt.subplots(figsize=(4, 3))
sns.scatterplot(x=df['PERIOD_HOURS'], y=df['INCLINATION'],s=10, size=df['RCS_SIZE'], size_order=sizes, sizes=[18,16,10], hue
plt.xlabel('Period (hours)')
plt.ylabel('Inclination (degrees)')
plt.title('Inclination vs. Period of Space Debris')
plt.xlim((0.75,28.5))
plt.ylim((-2,105))
plt.legend()
plt.show()
```

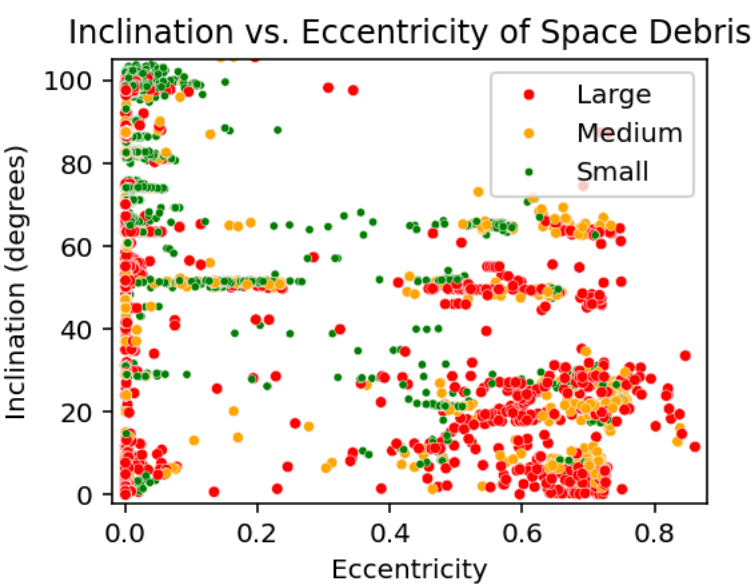


Notice the vertical bands at roughly T= 1.5h and T = 24h. This corresponds to the lowest-earth orbits and geosynchronous orbit respectively.

```
# Eccentricity vs period scatter plot
fig, ax = plt.subplots(figsize=(4, 3))
sns.scatterplot(x=df['PERIOD_HOURS'], y=df['ECCENTRICITY'],s=10, size=df['RCS_SIZE'], size_order=sizes, sizes=[18,16,10], hu
plt.xlabel('Period (hours)')
plt.ylabel('Eccentricity')
plt.title('Eccentricity vs. Period of Space Debris')
plt.xlim((0.75,28.5))
plt.ylim((-0.02,0.86))
plt.legend()
plt.show()
```

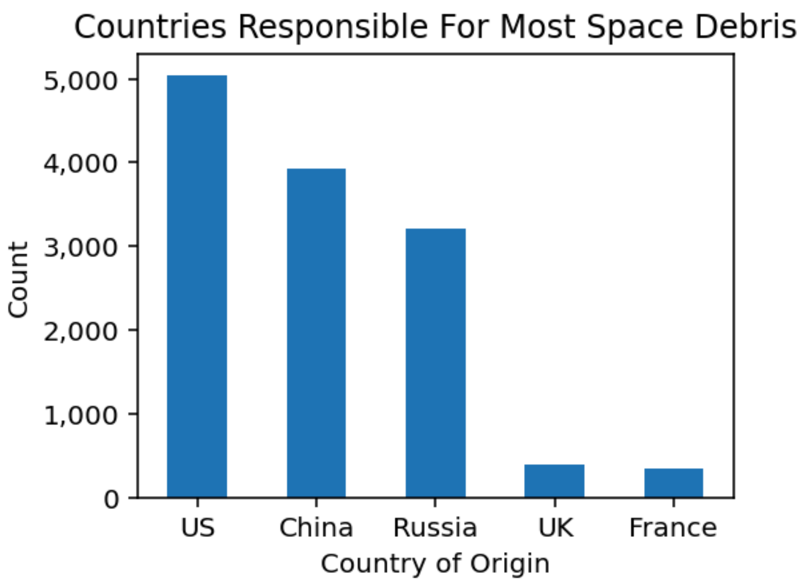



```
# Inclination vs eccentricity scatter plot
fig, ax = plt.subplots(figsize=(4, 3))
sns.scatterplot(x=df['ECCENTRICITY'], y=df['INCLINATION'],s=10, size=df['RCS_SIZE'], size_order=sizes, sizes=[18,16,10], hue
plt.xlabel('Eccentricity')
plt.ylabel('Inclination (degrees)')
plt.title('Inclination vs. Eccentricity of Space Debris')
plt.xlim((-0.02,0.88))
plt.ylim((-2,105))
plt.legend()
plt.show()
```



Which countries are responsible for the most debris?

```
# Bar chart count by country
top_counts = df['COUNTRY_CODE'].value_counts()[:5]

fig, ax = plt.subplots(figsize=(4, 3))
top_counts.plot(kind='bar')
plt.xlabel('Country of Origin')
plt.ylabel('Count')
plt.title('Countries Responsible For Most Space Debris')
plt.xticks(ticks=[0,1,2,3,4], labels=["US",'China','Russia','UK','France'],rotation=0)
ax.yaxis.set_major_formatter(thousands_format)
plt.show()
```

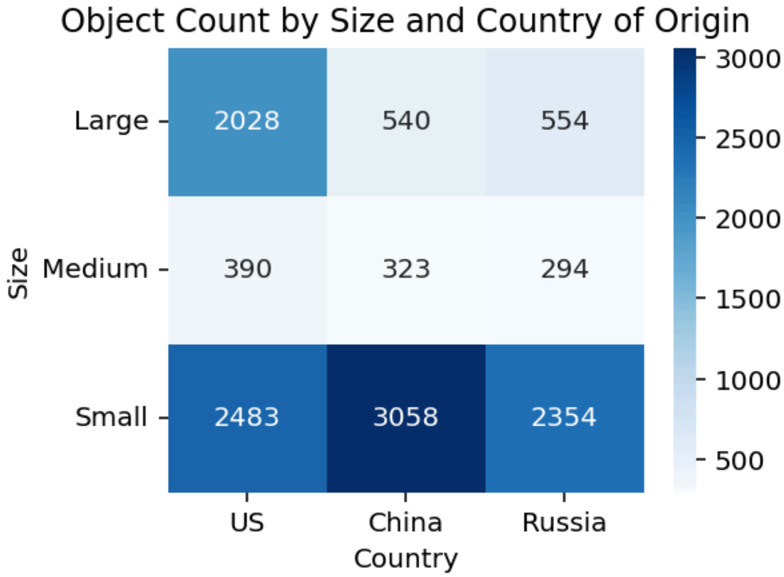



```
responsibility = top_counts / len(df)
print(f'The US is responsible for roughly {int(100 * responsibility[0])}% of space debris.')
```

```
The US is responsible for roughly 35% of space debris.
```

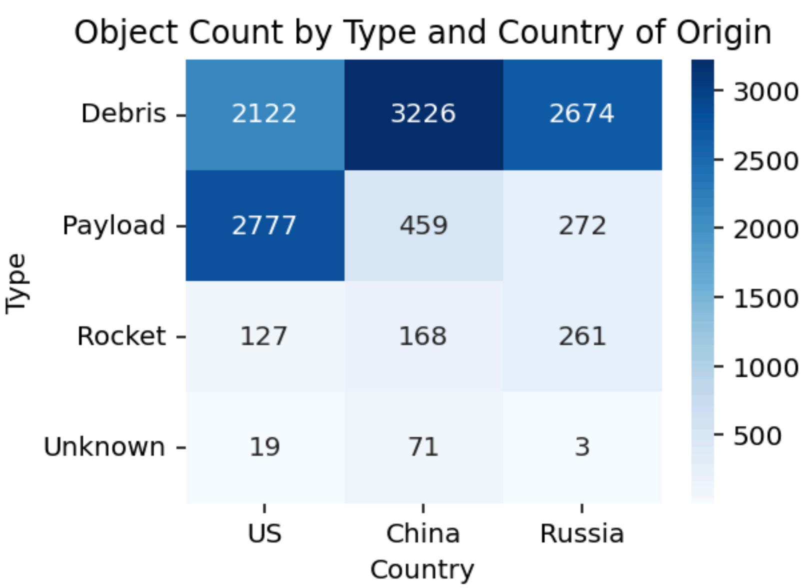
How many small, medium and large objects are there from the top 3 contributing countries?

```
# Just look at top 3 contributors
top_codes = top_counts.keys()[:3]
TOP_MASK = (df['COUNTRY_CODE'].isin(top_codes))
top_df = df[TOP_MASK]
country_names = ["US",'China','Russia']

country_counts = top_df.pivot_table(values='AggValue', index='RCS_SIZE', columns='COUNTRY_CODE', aggfunc=np.sum)

# Specify order of rows and columns
country_counts = country_counts.reindex(index=sizes, columns=top_codes)

fig, ax = plt.subplots(figsize=(4, 3))
sns.heatmap(country_counts, cmap="Blues", annot=True, fmt='g')
plt.ylabel('Size')
plt.xlabel('Country')
plt.title("Object Count by Size and Country of Origin")
plt.xticks(ticks=[0.5,1.5,2.5], labels=country_names)
plt.yticks(ticks=[0.5,1.5,2.5], labels=sizes,rotation=0)
plt.show()
```



How many of each category of object are there from the top 3 contributing countries?

```
top_codes = top_counts.keys()[:3]
TOP_MASK = (df_filled['COUNTRY_CODE'].isin(top_codes))
top_df = df[TOP_MASK]

country_counts = top_df.pivot_table(values='AggValue', index='OBJECT_TYPE', columns='COUNTRY_CODE', aggfunc=np.sum)

# Specify order of rows and columns
obj_types = ['Debris', 'Payload', 'Rocket','Unknown']
country_counts = country_counts.reindex(index=obj_types, columns=top_codes)

fig, ax = plt.subplots(figsize=(4, 3))
sns.heatmap(country_counts, cmap="Blues", annot=True, fmt='g')
plt.ylabel('Type')
plt.xlabel('Country')
plt.title("Object Count by Type and Country of Origin")
plt.xticks(ticks=[0.5,1.5,2.5], labels=country_names)
plt.yticks(ticks=[0.5,1.5,2.5,3.5], labels=obj_types, rotation=0)
plt.show()
```

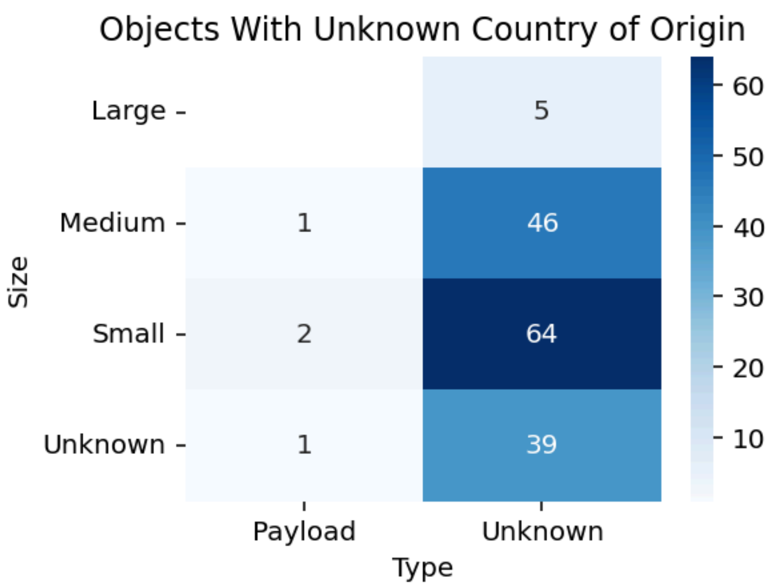


How many objects' country of origin is unknown?

```
NO_COUNTRY_MASK = ((df['COUNTRY_CODE'] == 'Unknown'))
no_country_df = df[NO_COUNTRY_MASK]
print(f"There are {len(no_country_df)} debris objects in this dataset with unknown country of origin.")
```

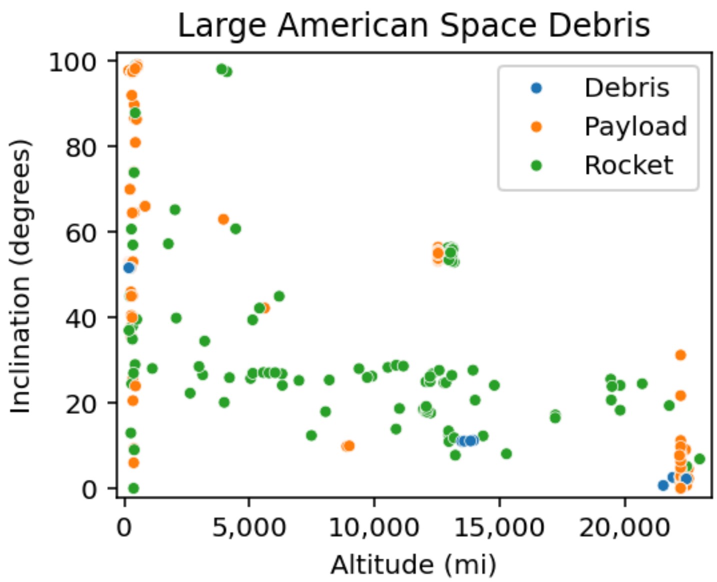
```
There are 158 debris objects in this dataset with unknown country of origin.
```

```
no_country_df.loc[:,'RCS_SIZE'] = no_country_df['RCS_SIZE'].fillna('Unknown')
no_country_class_counts = no_country_df.pivot_table(values='AggValue', index='RCS_SIZE', columns='OBJECT_TYPE', aggfunc=np.s

fig, ax = plt.subplots(figsize=(4, 3))
sns.heatmap(no_country_class_counts, cmap="Blues", annot=True, fmt='g')
plt.ylabel('Size')
plt.xlabel('Type')
plt.title("Objects With Unknown Country of Origin")
plt.xticks(ticks=[0.5,1.5], labels=["Payload", 'Unknown'])
plt.yticks(ticks=[0.5,1.5,2.5,3.5], labels=["Large",'Medium','Small', 'Unknown'],rotation=0)
plt.show()
```

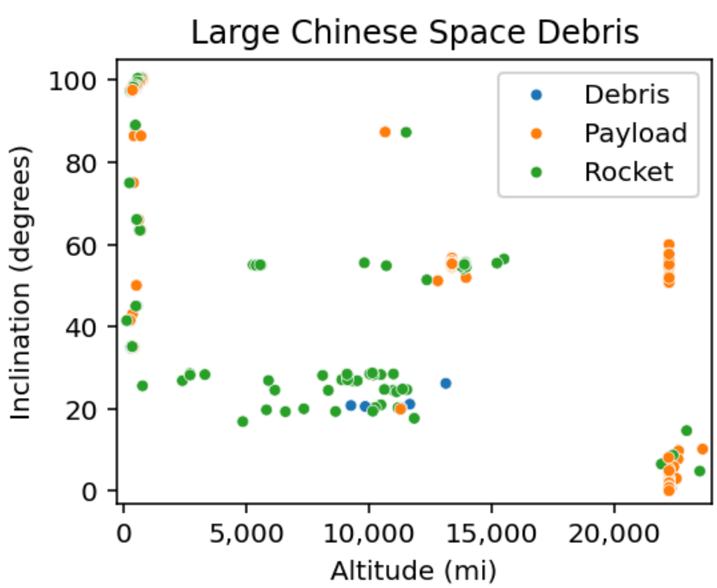


Dive into specific slices and analyze their orbits.

```
AMERICAN_LARGE_MASK = ((df['COUNTRY_CODE'] == 'US') & (df['RCS_SIZE'] == 'Large') & (df['OBJECT_TYPE'] != 'Unknown')) # excl
american_large_df = df[AMERICAN_LARGE_MASK]

obj_type_2 = ['Debris','Payload','Rocket']

fig, ax = plt.subplots(figsize=(4, 3))
sns.scatterplot(x=american_large_df['ALTITUDE_MI'], y=american_large_df['INCLINATION'],s=20,hue=american_large_df['OBJECT_TY
plt.xlabel('Altitude (mi)')
plt.ylabel('Inclination (degrees)')
plt.title('Large American Space Debris')
plt.xlim((-300, 23500))
plt.ylim((-2,102))
plt.legend()
ax.xaxis.set_major_formatter(thousands_format)
plt.show()
```




```
CHINESE_LARGE_MASK = ((df['COUNTRY_CODE'] == 'PRC') & (df['RCS_SIZE']=='Large') & (df['OBJECT_TYPE'] != 'Unknown')) # exclud
chinese_large_df = df[CHINESE_LARGE_MASK]

fig, ax = plt.subplots(figsize=(4, 3))
sns.scatterplot(x=chinese_large_df['ALTITUDE_MI'], y=chinese_large_df['INCLINATION'],s=20,hue=chinese_large_df['OBJECT_TYPE'
plt.xlabel('Altitude (mi)')
plt.ylabel('Inclination (degrees)')
plt.title('Large Chinese Space Debris')
plt.xlim((-300, 24000))
plt.ylim((-3,105))
plt.legend()
ax.xaxis.set_major_formatter(thousands_format)
plt.show()
```



```
LEO_MASK = (df['PERIOD'] <= 130)
leo_df = df[LEO_MASK]

US_CN_RU_MASK = (leo_df['COUNTRY_CODE'].isin(['US','PRC','CIS','Unknown']))
leo_ucr_df = leo_df[US_CN_RU_MASK]

LG_MASK = (leo_ucr_df['RCS_SIZE'] == 'Large')
leo_ucr_lg_df = leo_ucr_df[LG_MASK]

fig, ax = plt.subplots(figsize=(4, 3))
sns.scatterplot(x=leo_ucr_lg_df['ALTITUDE_MI'], y=leo_ucr_lg_df['INCLINATION'],s=20, hue=leo_ucr_lg_df['COUNTRY_CODE'], hue_
plt.xlabel('Altitude (mi)')
plt.ylabel('Inclination (degrees)')
```