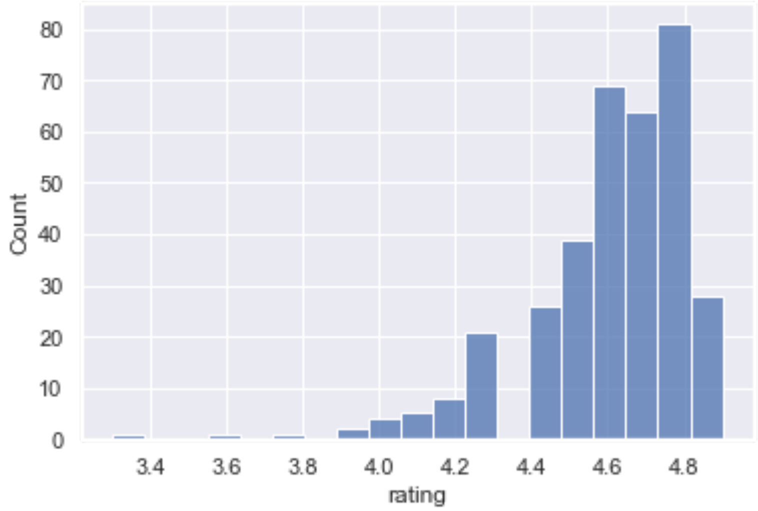
In [388]:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
books = pd.read_csv('clean_books.csv')
sns.histplot(data=books, x='rating')
plt.show()
```



In [389]:

```
books['genre'].value_counts()
books.info()
books
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 350 entries, 0 to 349
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   name    350 non-null    object 
 1   author  350 non-null    object 
 2   rating  350 non-null    float64
 3   year    350 non-null    int64  
 4   genre   350 non-null    object 
dtypes: float64(1), int64(1), object(3)
memory usage: 13.8+ KB
```

Out[389]:

	name	author	rating	year	genre
0	10-Day Green Smoothie Cleanse	JJ Smith	4.7	2016	Non Fiction
1	11/22/63: A Novel	Stephen King	4.6	2011	Fiction
2	12 Rules for Life: An Antidote to Chaos	Jordan B. Peterson	4.7	2018	Non Fiction
3	1984 (Signet Classics)	George Orwell	4.7	2017	Fiction
4	5,000 Awesome Facts (About Everything!) (Natio...	National Geographic Kids	4.8	2019	Childrens
...	...	...	...	...	...
345	Wild: From Lost to Found on the Pacific Crest ...	Cheryl Strayed	4.4	2012	Non Fiction
346	Winter of the World: Book Two of the Century T...	Ken Follett	4.5	2012	Fiction
347	Women Food and God: An Unexpected Path to Almo...	Geneen Roth	4.2	2010	Non Fiction
348	Wonder	R. J. Palacio	4.8	2013	Fiction
349	Wrecking Ball (Diary of a Wimpy Kid Book 14)	Jeff Kinney	4.9	2019	Childrens

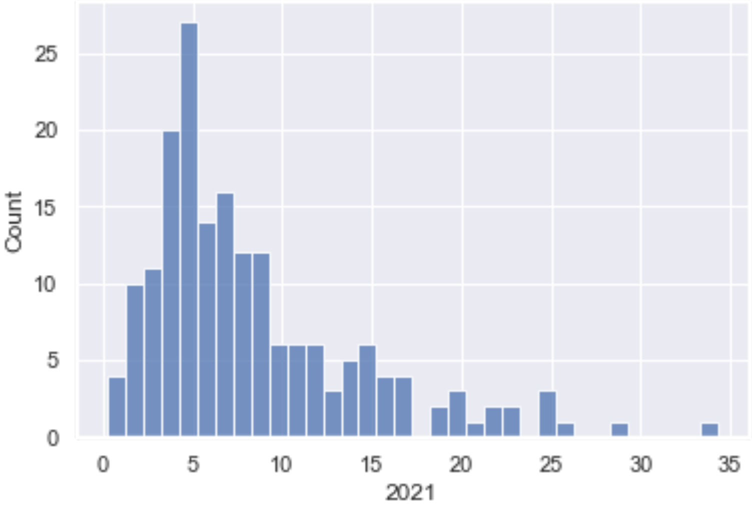
350 rows × 5 columns

In [390]:

```
#Ex1
```

In [391]:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

clean = pd.read_csv('clean_unemployment.csv')
sns.histplot(data=clean, x = '2021', binwidth = 1)
plt.show()
```



In [392]:

```
books["year"].min()
```

Out[392]:

```
2009
```

In [393]:

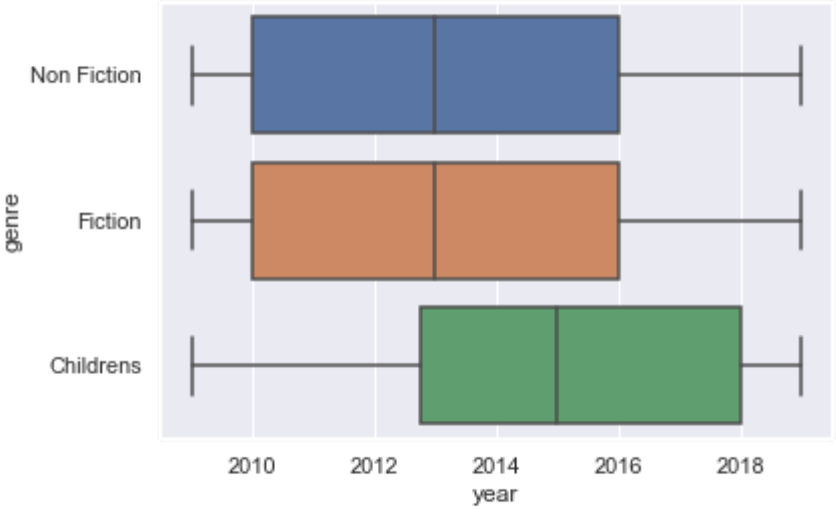
```
books["year"].max()
```

Out[393]:

```
2019
```

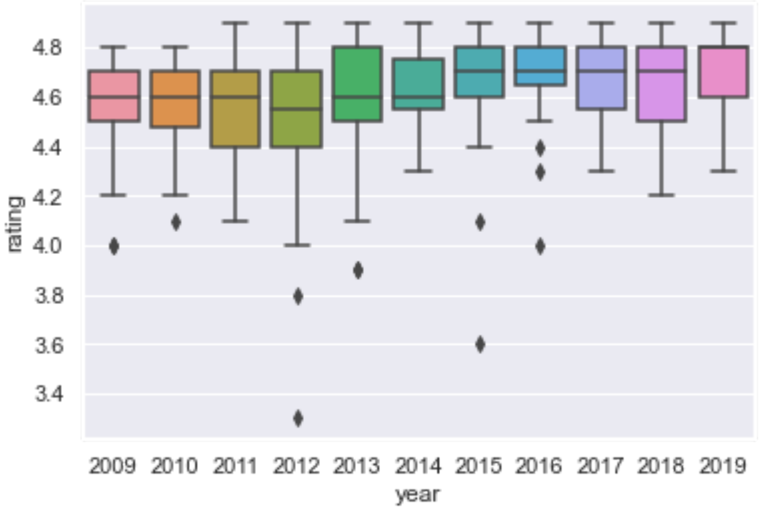
In [394]:

```
sns.boxplot(data = books, x="year", y="genre")
plt.show()
```



In [395]:

```
sns.boxplot(data = books, x="year", y="rating")
plt.show()
```



In [396]:

```
#Ex 2
```

In [397]:

```
unemployment = pd.read_csv("clean_unemployment.csv")

Series_not_oceania = unemployment['continent'].isin(["Oceania"])

unemployment[~unemployment['continent'].isin(["Oceania"])]
```

Out[397]:

	country_code	country_name	continent	2010	2011	2012	2013	2014	2015	2016
0	AFG	Afghanistan	Asia	11.35	11.05	11.34	11.19	11.14	11.13	11.16
1	AGO	Angola	Africa	9.43	7.36	7.35	7.37	7.37	7.39	7.41
2	ALB	Albania	Europe	14.09	13.48	13.38	15.87	18.05	17.19	15.42
3	ARE	United Arab Emirates	Asia	2.48	2.30	2.18	2.04	1.91	1.77	1.64
4	ARG	Argentina	South America	7.71	7.18	7.22	7.10	7.27	7.52	8.11
...	...	...	...	...	...	...	...	...	...	...
175	VNM	Vietnam	Asia	1.11	1.00	1.03	1.32	1.26	1.85	1.85
178	YEM	Yemen, Rep.	Asia	12.83	13.23	13.17	13.27	13.47	13.77	13.43
179	ZAF	South Africa	Africa	24.68	24.64	24.73	24.56	24.89	25.15	26.54
180	ZMB	Zambia	Africa	13.19	10.55	7.85	8.61	9.36	10.13	10.87
181	ZWE	Zimbabwe	Africa	5.21	5.37	5.15	4.98	4.77	4.78	4.79

174 rows × 15 columns

In [398]:

```
#Ex3
```

In [399]:

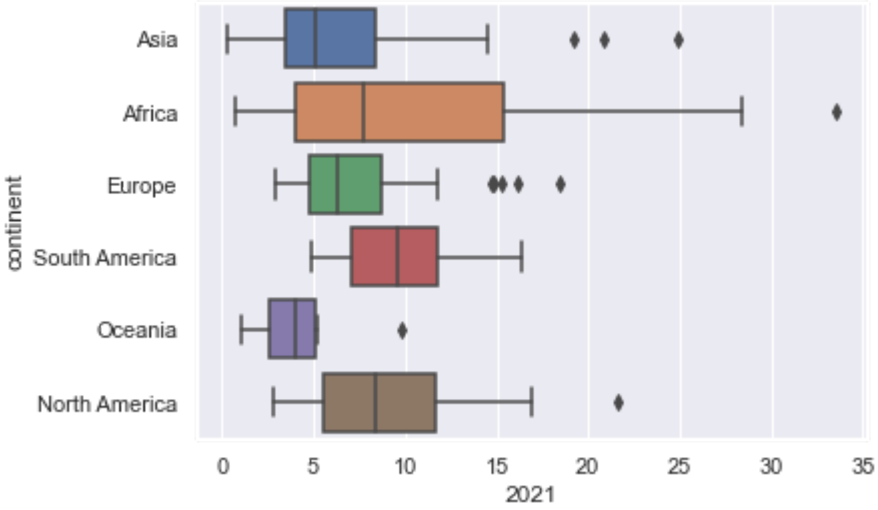
```
unemployment['2021'].min(), unemployment['2021'].max()
```

Out[399]:

```
(0.26, 33.56)
```

In [400]:

```
sns.boxplot(data = unemployment, x="2021", y = 'continent')
plt.show()
```



In [401]:

```
books.groupby('genre').mean()
```

Out[401]:

	rating	year
genre		
Childrens	4.780000	2015.075000
Fiction	4.570229	2013.022901
Non Fiction	4.598324	2013.513966

In [402]:

```
books.agg(["mean", "std"])
```

```
C:\Users\Lut Lat Aung\AppData\Local\Temp\ipykernel_7500\1469691538.py:1:
FutureWarning: ['name', 'author', 'genre'] did not aggregate successfull
y. If any error is raised this will raise in a future version of pandas.
Drop these columns/ops to avoid this warning.
  books.agg(["mean", "std"])
```

Out[402]:

	rating	year
mean	4.608571	2013.508571
std	0.226941	3.284711

In [403]:

```
books.agg({'rating': ["mean", "std"], "year": ["median"]})
```

Out[403]:

	rating	year
mean	4.608571	NaN
std	0.226941	NaN
median	NaN	2013.0

In [404]:

```
#Ex4
```

In [405]:

```
unemployment.agg(["mean", "std"])
```

```
C:\Users\Lut Lat Aung\AppData\Local\Temp\ipykernel_7500\2209990796.py:1:
FutureWarning: ['country_code', 'country_name', 'continent'] did not aggr
egate successfully. If any error is raised this will raise in a future ve
rsion of pandas. Drop these columns/ops to avoid this warning.
  unemployment.agg(["mean", "std"])
```

Out[405]:

	2010	2011	2012	2013	2014	2015	2016	2017	2
mean	8.409286	8.315440	8.317967	8.344780	8.179670	8.058901	7.925879	7.668626	7.42
std	6.248887	6.266795	6.367270	6.416041	6.284241	6.161170	6.045439	5.902152	5.81

In [406]:

```
books.groupby("genre").agg(
    mean_rating= ("rating", "mean"),
    std_rating = ("rating", "std"),
    median_year = ("year", "median"))

books
```

Out[406]:

	name	author	rating	year	genre
0	10-Day Green Smoothie Cleanse	JJ Smith	4.7	2016	Non Fiction
1	11/22/63: A Novel	Stephen King	4.6	2011	Fiction
2	12 Rules for Life: An Antidote to Chaos	Jordan B. Peterson	4.7	2018	Non Fiction
3	1984 (Signet Classics)	George Orwell	4.7	2017	Fiction
4	5,000 Awesome Facts (About Everything!) (Natio...	National Geographic Kids	4.8	2019	Childrens
...	...	...	...	...	...
345	Wild: From Lost to Found on the Pacific Crest ...	Cheryl Strayed	4.4	2012	Non Fiction
346	Winter of the World: Book Two of the Century T...	Ken Follett	4.5	2012	Fiction
347	Women Food and God: An Unexpected Path to Almo...	Geneen Roth	4.2	2010	Non Fiction
348	Wonder	R. J. Palacio	4.8	2013	Fiction
349	Wrecking Ball (Diary of a Wimpy Kid Book 14)	Jeff Kinney	4.9	2019	Childrens

350 rows × 5 columns

In [407]:

```
#Ex4
```

In [408]:
```
unemployment.groupby('continent').agg(['mean', 'std'])
```

```
C:\Users\Lut Lat Aung\AppData\Local\Temp\ipykernel_7500\1942534207.py:1:
FutureWarning: ['country_code', 'country_name'] did not aggregate success
fully. If any error is raised this will raise in a future version of pand
as. Drop these columns/ops to avoid this warning.
  unemployment.groupby('continent').agg(['mean', 'std'])
```

Out[408]:

	2010		2011		2012		2013	
	mean	std	mean	std	mean	std	mean	std
continent								
Africa	9.343585	7.411259	9.369245	7.401556	9.240755	7.264542	9.132453	7.30928
Asia	6.240638	5.146175	5.942128	4.779575	5.835319	4.756904	5.852128	4.66840
Europe	11.008205	6.392063	10.947949	6.539538	11.325641	7.003527	11.466667	6.96920
North America	8.663333	5.115805	8.563333	5.377041	8.448889	5.495819	8.840556	6.08182
Oceania	3.622500	2.054721	3.647500	2.008466	4.103750	2.723118	3.980000	2.64011
South America	6.870833	2.807058	6.518333	2.801577	6.410833	2.936508	6.335000	2.80878

6 rows × 24 columns

In [409]:
```
#Ex5
```

In [410]:
```
continent_summary = unemployment.groupby("continent").agg(mean_rate_2021 =
continent_summary
```

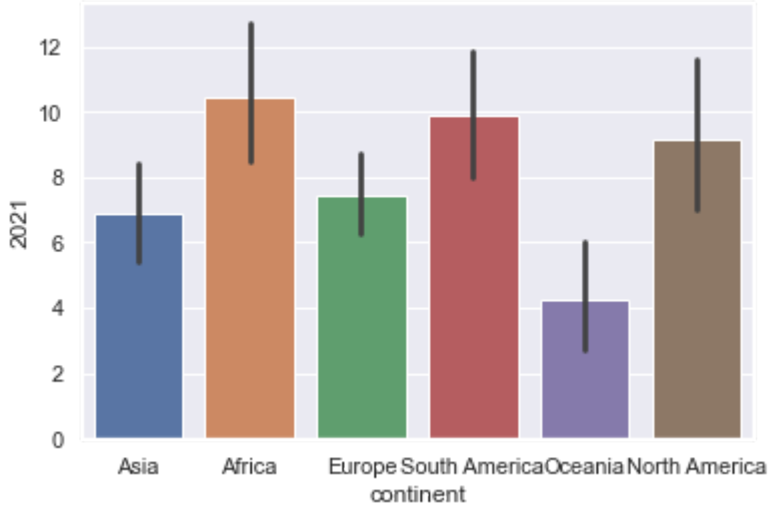
Out[410]:

	mean_rate_2021	std_rate_2021
continent		
Africa	10.473585	8.131636
Asia	6.906170	5.414745
Europe	7.414872	3.947825
North America	9.155000	5.076482
Oceania	4.280000	2.671522
South America	9.924167	3.611624

In [411]:
```
#Ex6
```

In [412]:

```
sns.barplot(data = unemployment, x="continent", y = '2021')
plt.show()
```



In [413]:

```
salaries = pd.read_csv("ds_salaries_clean.csv")
print(salaries.isna().sum())
salaries
```

```
Working_Year            0
Designation             0
Experience              0
Employment_Status       0
Employee_Location       0
Company_Size            0
Remote_Working_Ratio    0
Salary_USD              0
dtype: int64
```

Out[413]:

	Working_Year	Designation	Experience	Employment_Status	Employee_Location	Comp
0	2020	Data Scientist	Mid	FT	DE	
1	2020	Machine Learning Scientist	Senior	FT	JP	
2	2020	Big Data Engineer	Senior	FT	GB	
3	2020	Product Data Analyst	Mid	FT	HN	
4	2020	Machine Learning Engineer	Senior	FT	US	
...	...	...	...	...	...	
602	2022	Data Engineer	Senior	FT	US	
603	2022	Data Engineer	Senior	FT	US	
604	2022	Data Analyst	Senior	FT	US	
605	2022	Data Analyst	Senior	FT	US	
606	2022	AI Scientist	Mid	FT	IN	

607 rows × 8 columns

In [414]:

```
threshold = len(salaries) * 0.05
threshold
```

Out[414]:

```
30.35
```

In [415]:

```
cols_to_drop = salaries.columns[salaries.isna().sum() <= threshold]
print(cols_to_drop)
```

```
Index(['Working_Year', 'Designation', 'Experience', 'Employment_Status',
       'Employee_Location', 'Company_Size', 'Remote_Working_Ratio',
       'Salary_USD'],
      dtype='object')
```

In [416]:

```
salaries.dropna(subset = cols_to_drop, inplace = True)
salaries
```

Out[416]:

	Working_Year	Designation	Experience	Employment_Status	Employee_Location	Comp
0	2020	Data Scientist	Mid	FT	DE	
1	2020	Machine Learning Scientist	Senior	FT	JP	
2	2020	Big Data Engineer	Senior	FT	GB	
3	2020	Product Data Analyst	Mid	FT	HN	
4	2020	Machine Learning Engineer	Senior	FT	US	
...	...	...	...	...	...	
602	2022	Data Engineer	Senior	FT	US	
603	2022	Data Engineer	Senior	FT	US	
604	2022	Data Analyst	Senior	FT	US	
605	2022	Data Analyst	Senior	FT	US	
606	2022	AI Scientist	Mid	FT	IN	

607 rows × 8 columns

In [417]:

```
cols_with_missing_values = salaries.columns[salaries.isna().sum() > 0]
print(cols_with_missing_values)

for col in cols_with_missing_values[:-1]:
    salaries[col].fillna(salaries[col].mode()[0])
```

```
Index([], dtype='object')
```

In [418]:

```
salaries_dict = salaries.groupby("Experience") ["Salary_USD"].median().to_
print(salaries_dict)
```

```
{'Entry': 53948.0, 'Executive': 163694.5, 'Mid': 73465.0, 'Senior': 12938
0.0}
```

In [419]:

```
salaries["Salary_USD"] = salaries["Salary_USD"].fillna(salaries["Experienc
```

In [420]:

```
#Ex7
```

In [421]:

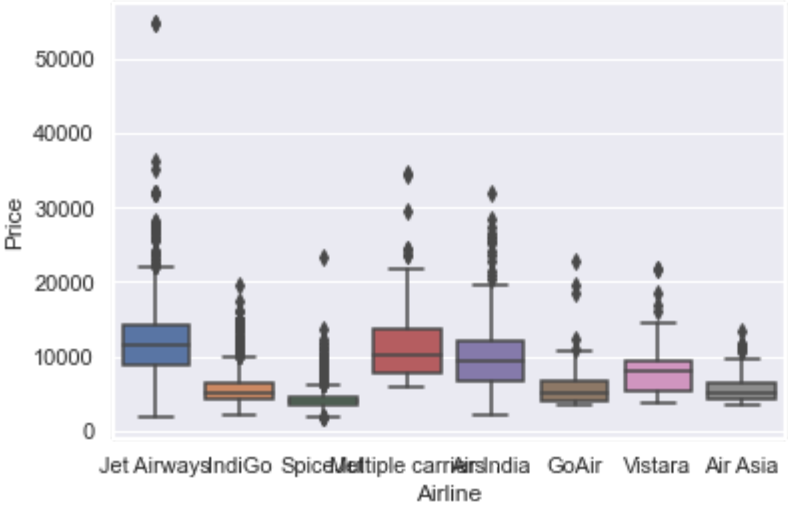
```python
planes = pd.read_csv("airlines_unclean.csv")

print(planes.isna().sum())
print()

threshold = len(planes) * 0.05
print(threshold)
print()

cols_to_drop = planes.columns[planes.isna().sum() <= threshold]
print(cols_to_drop)
print()

planes.dropna(subset = cols_to_drop, inplace = True)
print(planes.isna().sum())
```

```
Unnamed: 0           0
Airline            427
Date_of_Journey    322
Source             187
Destination        347
Route              256
Dep_Time           260
Arrival_Time       194
Duration           214
Total_Stops        212
Additional_Info    589
Price              616
dtype: int64

533.0

Index(['Unnamed: 0', 'Airline', 'Date_of_Journey', 'Source', 'Destination
',
       'Route', 'Dep_Time', 'Arrival_Time', 'Duration', 'Total_Stops'],
      dtype='object')

Unnamed: 0           0
Airline              0
Date_of_Journey      0
Source               0
Destination          0
Route                0
Dep_Time             0
Arrival_Time         0
Duration             0
Total_Stops          0
Additional_Info    300
Price              368
dtype: int64
```

In [422]:

```
print(planes["Additional_Info"].value_counts())

# Create a box plot of Price by Airline
sns.boxplot(data=planes, x='Airline', y='Price')
sns.set(rc={"figure.figsize":(8, 6)}) #width=8, #height=6
plt.show()
```

```
No info                         6399
In-flight meal not included     1525
No check-in baggage included     258
1 Long layover                    14
Change airports                    7
No Info                            2
Business class                     1
Red-eye flight                     1
2 Long layover                     1
Name: Additional_Info, dtype: int64
```



In [423]:

```
#Ex8
```

In [424]:

```
planeairline = planes.groupby('Airline')
Median_price = planeairline['Price'].median()
Add_Dic = Median_price.to_dict()

print(Median_price)
print()
print(Add_Dic)
```

```
Airline
Air Asia              5192.0
Air India             9443.0
GoAir                 5003.5
IndiGo                5054.0
Jet Airways          11507.0
Multiple carriers    10197.0
SpiceJet              3873.0
Vistara               8028.0
Name: Price, dtype: float64

{'Air Asia': 5192.0, 'Air India': 9443.0, 'GoAir': 5003.5, 'IndiGo': 505
4.0, 'Jet Airways': 11507.0, 'Multiple carriers': 10197.0, 'SpiceJet': 38
73.0, 'Vistara': 8028.0}
```

In [425]:

```
    planes["Price"] = planes["Price"].fillna(planes["Airline"].map(Add_Dic
    planes = planes.drop(columns = ['Additional_Info'])
    print(planes.isna().sum())
```

```
Unnamed: 0         0
Airline            0
Date_of_Journey    0
Source             0
Destination        0
Route              0
Dep_Time           0
Arrival_Time       0
Duration           0
Total_Stops        0
Price              0
dtype: int64
```

In [426]:

```
print(salaries.select_dtypes("object").head())
```

```
                  Designation Experience Employment_Status Employee_Locat
ion  \
0              Data Scientist        Mid                FT
DE
1  Machine Learning Scientist     Senior                FT
JP
2           Big Data Engineer     Senior                FT
GB
3        Product Data Analyst        Mid                FT
HN
4   Machine Learning Engineer     Senior                FT
US

  Company_Size
0            L
1            S
2            M
3            S
4            L
```

In [427]:

```
print(salaries["Designation"].value_counts())
```

```
Data Scientist                              143
Data Engineer                               132
Data Analyst                                 97
Machine Learning Engineer                    41
Research Scientist                           16
Data Science Manager                         12
Data Architect                               11
Big Data Engineer                             8
Machine Learning Scientist                    8
Principal Data Scientist                      7
AI Scientist                                  7
Data Science Consultant                       7
Director of Data Science                      7
Data Analytics Manager                        7
ML Engineer                                   6
Computer Vision Engineer                      6
BI Data Analyst                               6
Lead Data Engineer                            6
Data Engineering Manager                      5
Business Data Analyst                         5
Head of Data                                  5
Applied Data Scientist                        5
Applied Machine Learning Scientist            4
Head of Data Science                          4
Analytics Engineer                            4
Data Analytics Engineer                       4
Machine Learning Developer                    3
Machine Learning Infrastructure Engineer      3
Lead Data Scientist                           3
Computer Vision Software Engineer             3
Lead Data Analyst                             3
Data Science Engineer                         3
Principal Data Engineer                       3
Principal Data Analyst                        2
ETL Developer                                 2
Product Data Analyst                          2
Director of Data Engineering                  2
Financial Data Analyst                        2
Cloud Data Engineer                           2
Lead Machine Learning Engineer                1
NLP Engineer                                  1
Head of Machine Learning                      1
3D Computer Vision Researcher                 1
Data Specialist                               1
Staff Data Scientist                          1
Big Data Architect                            1
Finance Data Analyst                          1
Marketing Data Analyst                        1
Machine Learning Manager                      1
Data Analytics Lead                           1
Name: Designation, dtype: int64
```

In [428]:

```
print(salaries["Designation"].nunique())
```

```
50
```

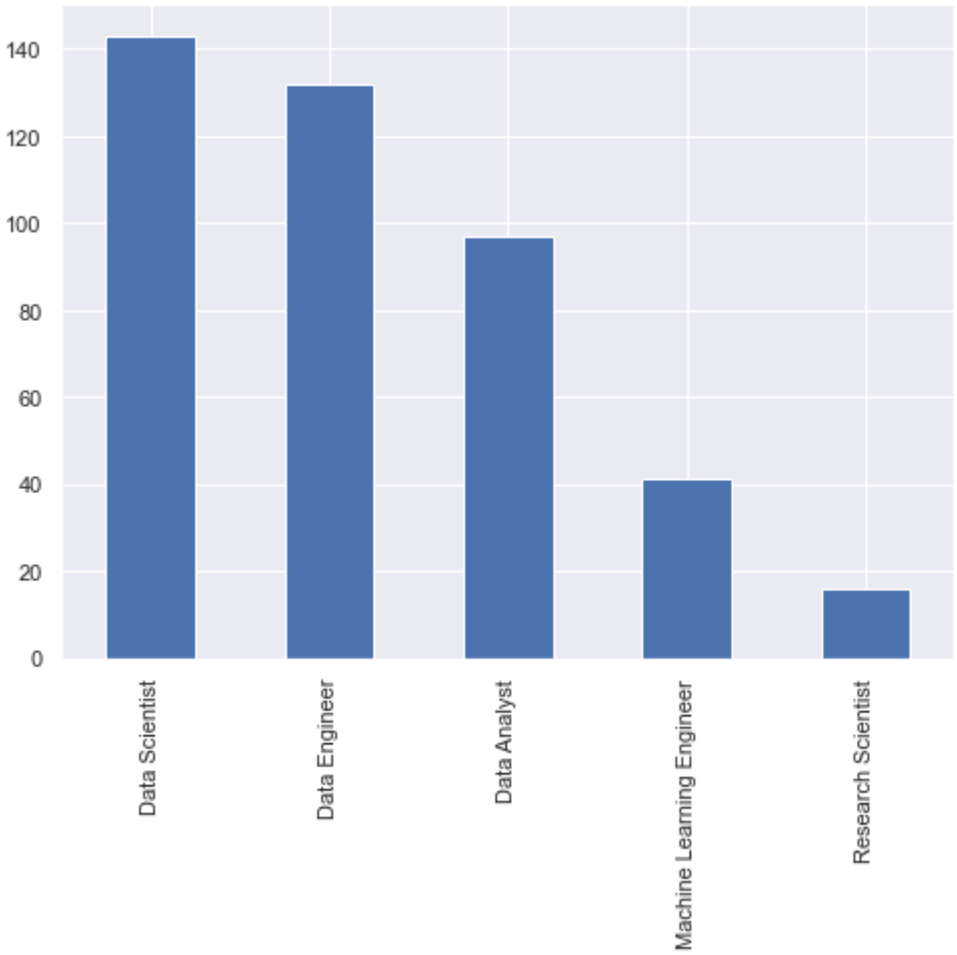
In [429]:

```
salaries_count = salaries['Designation'].value_counts().iloc[0:5]
print(salaries_count.index)
salaries_count.plot(kind='bar')
```

```
Index(['Data Scientist', 'Data Engineer', 'Data Analyst',
       'Machine Learning Engineer', 'Research Scientist'],
      dtype='object')
```

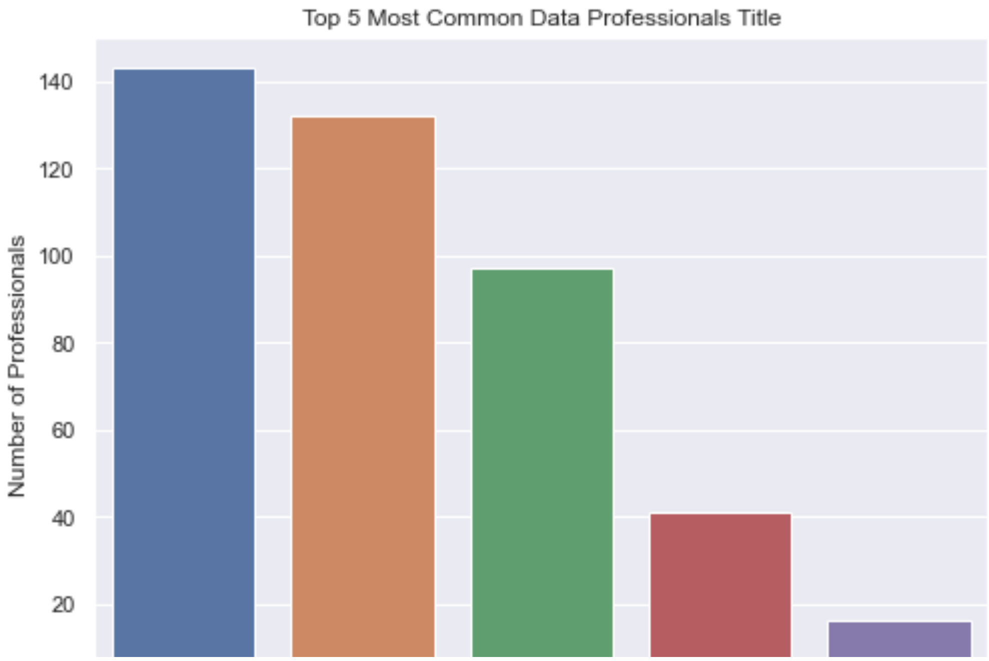
Out[429]:

```
<AxesSubplot:>
```



In [430]:

```
salaries_count = salaries['Designation'].value_counts().iloc[0:5].sort_valu
#print(salaries_count.index)
#print(salaries_count.values)
#salaries_count = salaries_count.sort_values(ascending = False)
plt.xlabel("Professionals")
plt.ylabel("Number of Professionals")
plt.title("Top 5 Most Common Data Professionals Title")
#sns.barplot(x = salaries_count.values, y = salaries_count, order=salaries_
sns.barplot(x= salaries_count.index, y = salaries_count.values)
plt.xticks(rotation=45)
plt.show()
```



In [431]:

```
salaries["Designation"].str.contains("Scientist")
```

Out[431]:

```
0        True
1        True
2       False
3       False
4       False
        ...  
602     False
603     False
604     False
605     False
606      True
Name: Designation, Length: 607, dtype: bool
```

In [432]:

```
salaries["Designation"].str.contains("Machine Learning|AI")
```

Out[432]:

```
0      False
1       True
2      False
3      False
4       True
       ...  
602    False
603    False
604    False
605    False
606     True
Name: Designation, Length: 607, dtype: bool
```

In [433]:

```
job_categories = ["Data Science","Data Analytics",
                 "Data Engineering","Machine Learning",
                 "Managerial","Consultant",]
```

In [434]:

```
data_science = "Scientist|NLP"
data_analyst = "Analyst|Analytics"
data_engineer = "Data Engineer|ETL|Architect|Infrastructure"
ml_engineer = "Machine Learning|ML|Big Data|AI"
manager = "Manager|Head|Director|Lead|Principal|Staff"
consultant = "Consultant|Freelance"
```

In [435]:

```
conditions = [
    (salaries["Designation"].str.contains(data_science)),
    (salaries["Designation"].str.contains(data_analyst)),
    (salaries["Designation"].str.contains(data_engineer)),
    (salaries["Designation"].str.contains(ml_engineer)),
    (salaries["Designation"].str.contains(manager)),
    (salaries["Designation"].str.contains(consultant)),
]
```

In [436]:

```
import numpy as np
salaries["Job_Category"] = np.select(conditions,
                                    job_categories,
                                    default = "Other")
```

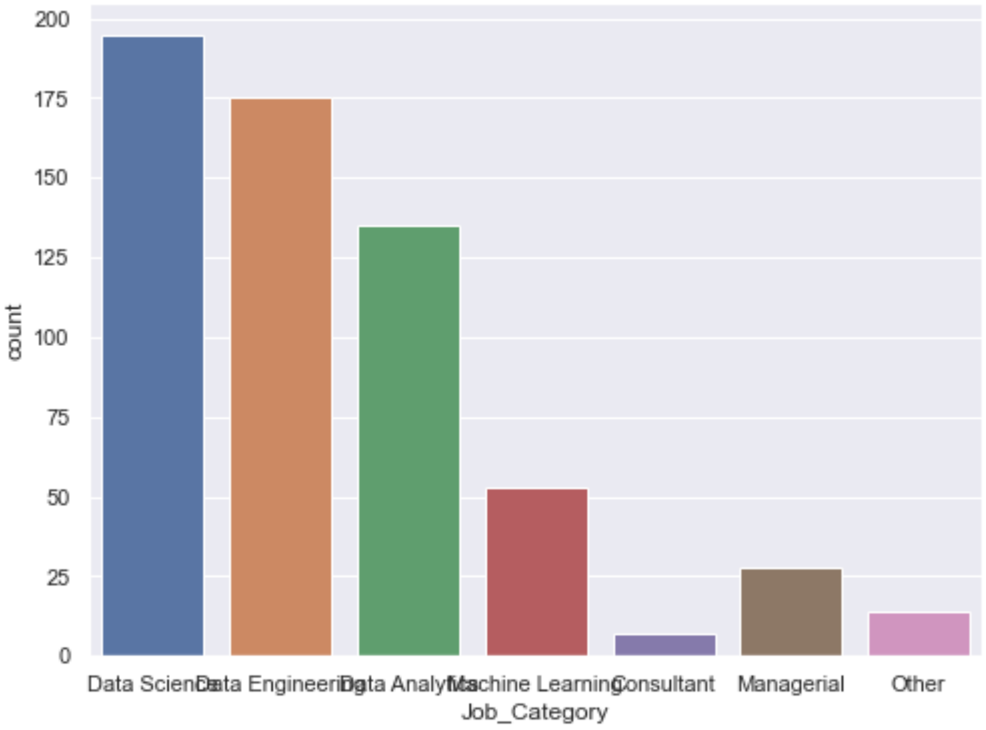
In [437]:

```
print(salaries[["Designation", "Job_Category"]].head())
```

```
                 Designation      Job_Category
0             Data Scientist      Data Science
1  Machine Learning Scientist      Data Science
2          Big Data Engineer  Data Engineering
3       Product Data Analyst    Data Analytics
4   Machine Learning Engineer  Machine Learning
```

In [438]:

```
sns.countplot(data=salaries, x="Job_Category")
plt.show()
```



In [439]:

```
non_numeric = planes.select_dtypes("object")
for col in non_numeric.columns:
    print(f"Number of unique values in {col} column: ",
         non_numeric[col].nunique())
```

```
Number of unique values in Airline column:  8
Number of unique values in Date_of_Journey column:  40
Number of unique values in Source column:  5
Number of unique values in Destination column:  6
Number of unique values in Route column:  122
Number of unique values in Dep_Time column:  218
Number of unique values in Arrival_Time column:  1220
Number of unique values in Duration column:  362
Number of unique values in Total_Stops column:  5
```

In [440]:

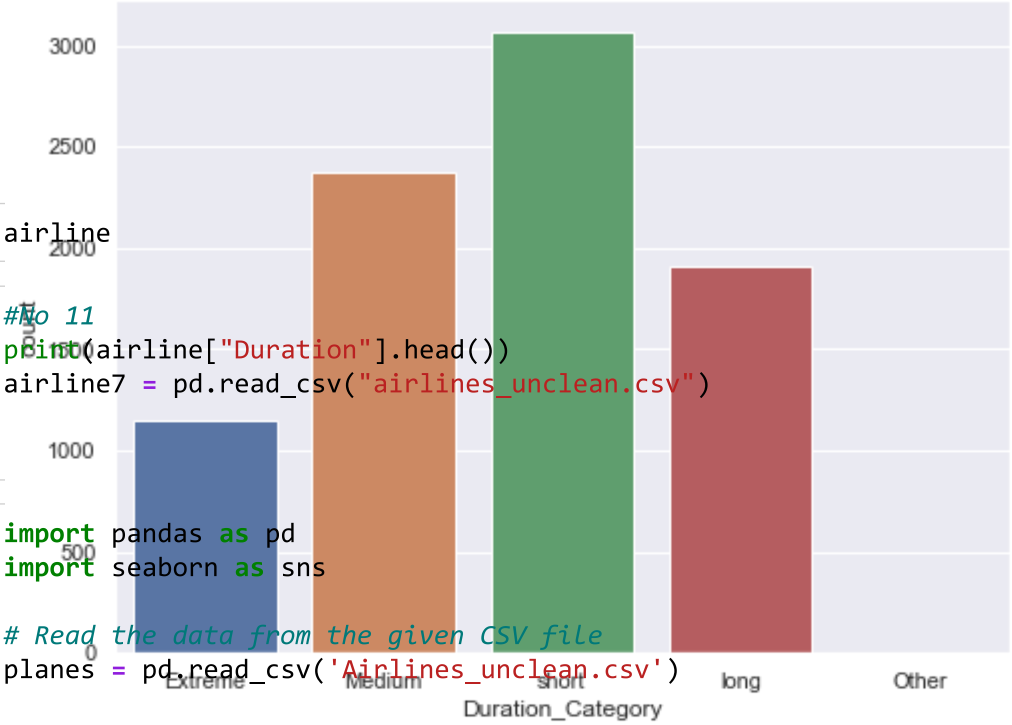
```
planes["Duration"].head()
print(planes.isna().sum())
```

```
Unnamed: 0         0
Airline            0
Date_of_Journey    0
Source             0
Destination        0
Route              0
Dep_Time           0
Arrival_Time       0
Duration           0
Total_Stops        0
Price              0
dtype: int64
```

In [447]:

```
#No.9 Problem and No.10
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

#airline = pd.read_csv("airlines_unclean.csv")


flight_time = ["Extreme", "long", "Medium", "short"]

Extreme_flight = "17h|18h|19h|20h|21h|22h|23h|24h"
short_flights = "0h|1h|2h|3h|4h"
medium_flights = "5h|6h|7h|8h|9h"
long_flights = "10h|11h|12h|13h|14h|15h|16h"

conditionss = [(planes["Duration"].str.contains(Extreme_flight)),
                (planes["Duration"].str.contains(long_flights)),
     (planes["Duration"].str.contains(medium_flights)),
     (planes["Duration"].str.contains(short_flights)),

]

planes["Duration_Category"] = np.select(conditionss,
                                    flight_time,
                                    default = "Other")
print(planes[["Duration", "Duration_Category"]].head())


sns.countplot(data=planes, x="Duration_Category")
plt.show()
```

```
  Duration Duration_Category
0      19h           Extreme
1   5h 25m            Medium
2   4h 45m             short
3   2h 25m             short
4  15h 30m              long
```



In [ ]:

```
airline
```

In [ ]:

```
#No 11
print(airline["Duration"].head())
airline7 = pd.read_csv("airlines_unclean.csv")
```

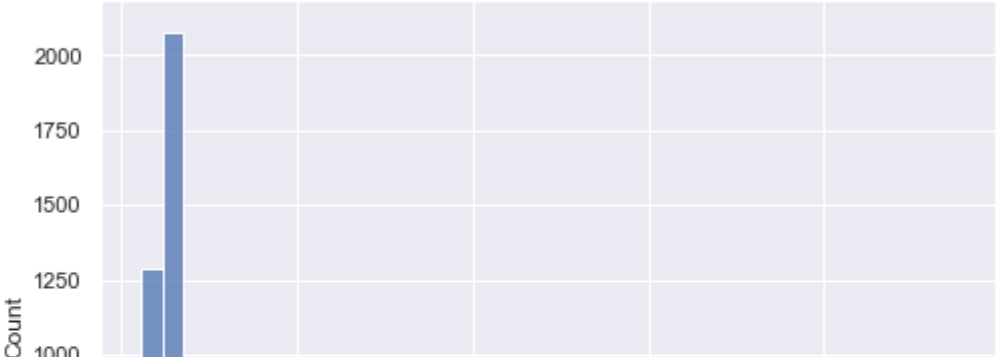
In [458]:

```
import pandas as pd
import seaborn as sns

# Read the data from the given CSV file
planes = pd.read_csv('Airlines_unclean.csv')

# Print the first five values of the "Duration" column
print(planes['Duration'].head())

# Remove "h", "m", and " " from the column and convert the data format to
planes['Duration'] = planes['Duration'].str.replace('h', '').str.replace('

# Convert the column to float data type
planes['Duration'] = planes['Duration'].astype(float)

# Plot a histogram of "Duration" values
sns.histplot(data=planes, x="Duration")
```

```
0        19h
1     5h 25m
2     4h 45m
3     2h 25m
4    15h 30m
Name: Duration, dtype: object
```

Out[458]:

```
<AxesSubplot:xlabel='Duration', ylabel='Count'>
```



In [460]:

```
#No.12
planes["std_dev"] = planes.groupby("Airline")["Price"].transform(lambda x:
planes["median"] = planes.groupby("Airline")["Duration"].transform("median
planes["mean"] = planes.groupby("Destination")["Price"].transform("mean")
print(planes["std_dev"])
print(planes["median"])
print(planes["mean"])
```

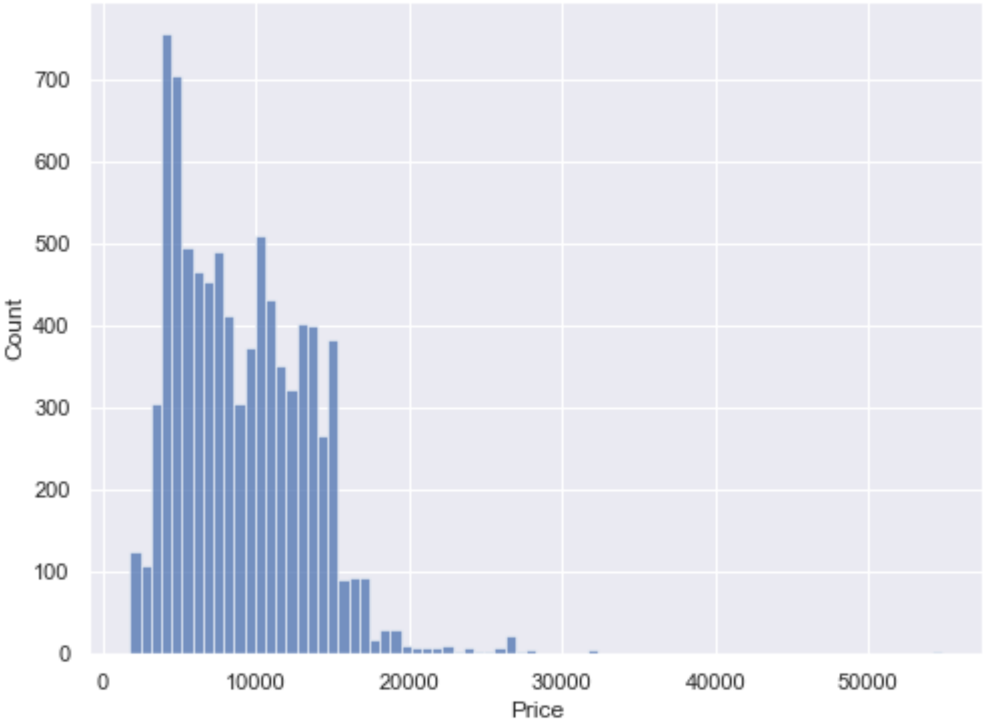
```
0        4230.748840
1        2266.753552
2        2266.753552
3        1790.851944
4        4230.748840
            ...
10655    2016.738954
10656    3865.871975
10657    4230.748840
10658    2864.267802
10659    3865.871975
Name: std_dev, Length: 10660, dtype: float64
0        13.20
1         2.55
2         2.55
3         2.30
4        13.20
         ...
10655     2.50
10656    15.55
10657    13.20
10658     3.10
10659    15.55
Name: median, Length: 10660, dtype: float64
0        10506.993486
1         9132.225153
2        11738.589499
3         9132.225153
4        11738.589499
             ...
10655     9132.225153
10656     9132.225153
10657     5157.794118
10658    11738.589499
10659    10506.993486
Name: mean, Length: 10660, dtype: float64
```

In [ ]:

In [448]:

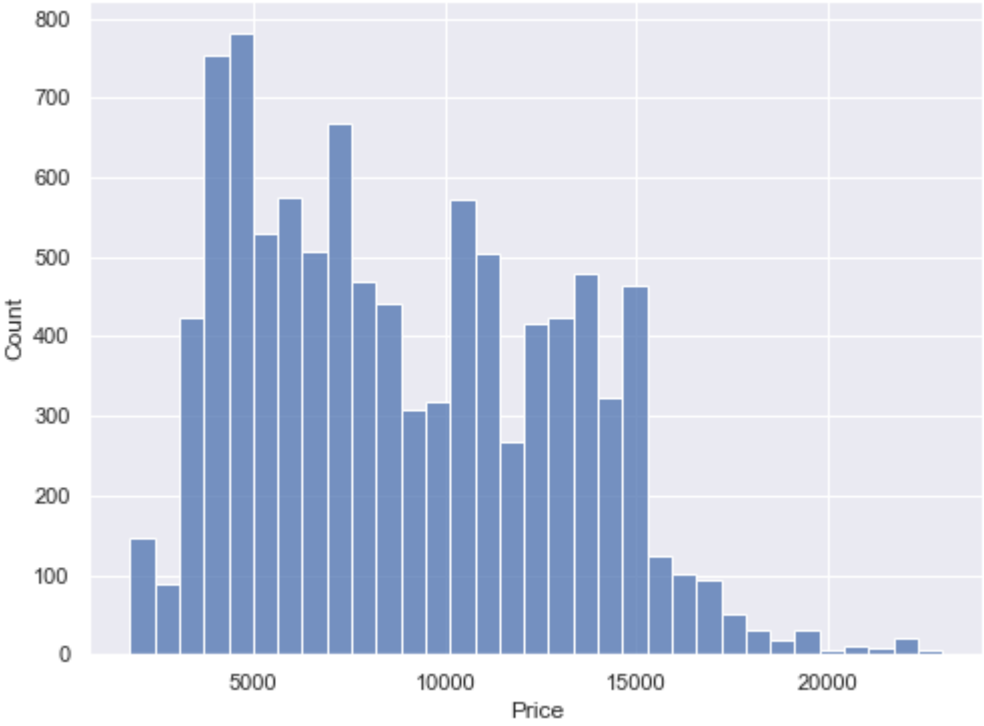
```
#No.13
sns.histplot(data=planes, x='Price')
duration_stats = planes['Duration'].describe()
print(duration_stats)
```

```
count       8508
unique       362
top       2h 50m
freq         425
Name: Duration, dtype: object
```



In [457]:

```
#No.14

import pandas as pd

planes = pd.read_csv('airlines_unclean.csv')

price_seventy_fifth = planes['Price'].quantile(0.75)
price_twenty_fifth = planes['Price'].quantile(0.25)

prices_iqr = price_seventy_fifth - price_twenty_fifth

upper_threshold = price_seventy_fifth + (1.5 * prices_iqr)
lower_threshold = price_twenty_fifth - (1.5 * prices_iqr)

planes = planes[(planes['Price'] >= lower_threshold) & (planes['Price'] <=


sns.histplot(data=planes, x= "Price")
plt.show()
```



In [ ]: