In [125]:

```
import pandas as pd
import numpy as np
```

In [126]:

```
visitor_park = pd.read_csv("july4_snapshot.csv")
```

In [ ]:

```
#1: Exploratory Data Analysis
```

In [127]:

```
#A:
visitor_park.head()
```

Out[127]:

	visitor	day_pass	season_ticket	domestic	state	country	gender	age	maine_res	stay_t
0	1	1	No	1	NY	USA	0	32	No	
1	2	0	Yes	1	Other	USA	1	43	No	
2	3	1	No	1	ME	USA	1	28	Yes	
3	4	0	Yes	1	NH	USA	1	35	No	
4	5	1	No	0	NaN	MEX	1	44	No	

There are 5 rows of data seen from the head function

In [128]:

```
#Looking at Shape Attribute with .shape
```

In [ ]:

```
visitor_park.shape
```

Out[ ]:

```
(5216, 19)
```

There are 5216 columns and 19 rows

In [ ]:

```
#D:
visitor_park.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5216 entries, 0 to 5215
Data columns (total 19 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   visitor           5216 non-null   int64  
 1   day_pass          5216 non-null   int64  
 2   season_ticket     5216 non-null   object 
 3   domestic          5216 non-null   int64  
 4   state             4127 non-null   object 
 5   country           5160 non-null   object 
 6   gender            5216 non-null   int64  
 7   age               5216 non-null   int64  
 8   maine_res         5216 non-null   object 
 9   stay_four         5216 non-null   int64  
 10  payment_method    5216 non-null   int64  
 11  ice_cream_purch   5216 non-null   int64  
 12  ice_cream_flavor  5216 non-null   object 
 13  sky_chair         5216 non-null   int64  
 14  ferris_wheel      5216 non-null   int64  
 15  lobster_claw      5216 non-null   int64  
 16  lobster_junior    5216 non-null   int64  
 17  merch_spend       5216 non-null   float64
 18  lobsterama_spend  5216 non-null   float64
dtypes: float64(2), int64(12), object(5)
memory usage: 774.4+ KB
```

Gender , age and payment_method are categorical data and other values from 11-18 would be numeric

In [ ]:

```
#E = Rounding of values to 2 decimals
visitor_park.round({"merch_spend":2})
```

Out[ ]:

	visitor	day_pass	season_ticket	state	country	gender	age	main_per	stay_four	pay
0	1	1	No	NY	USA	0	15	No	1	
1	2	0	Yes	Other	USA	1	15	No	1	
2	3	1	No	ME	USA	1	15	Yes	1	
3	4	0	Yes	NH	USA	1	15	No	0	
4	5	1	No	NaN	MEX	1	15	No	1	
...	...	...	...	...	...	...	...	...	...	
5211	5212	0	Yes	NH	USA	0	15	No	1	
5212	5213	1	No	NaN	UK	1	15	No	1	
5213	5214	0	Yes	NH	USA	1	15	No	0	
5214	5215	0	Yes	ME	USA	0	15	Yes	1	
5215	5216	1	No	MA	USA	0	15	No	1	

5216 rows × 18 columns

In [ ]:

```
#F:
visitor_park.isna().sum()
```

Out[ ]:

```
visitor                0
day_pass               0
season_ticket          0
domestic               0
state               1089
country               56
gender                 0
age                    0
maine_res              0
stay_four              0
payment_method         0
ice_cream_purch        0
ice_cream_flavor       0
sky_chair              0
ferris_wheel           0
lobster_claw           0
lobster_junior         0
merch_spend            0
lobsterama_spend       0
dtype: int64
```

Total of 1145 values are missing

In [ ]:

```
#F(a):
visitor_park.isna().sum().sum() * 100 / len(visitor_park)
```

Out[ ]:

```
21.95168711656442
```

In [ ]:

```
#F(b):
visitor_park.isna().sum() * 100 / len(visitor_park)
```

Out[ ]:

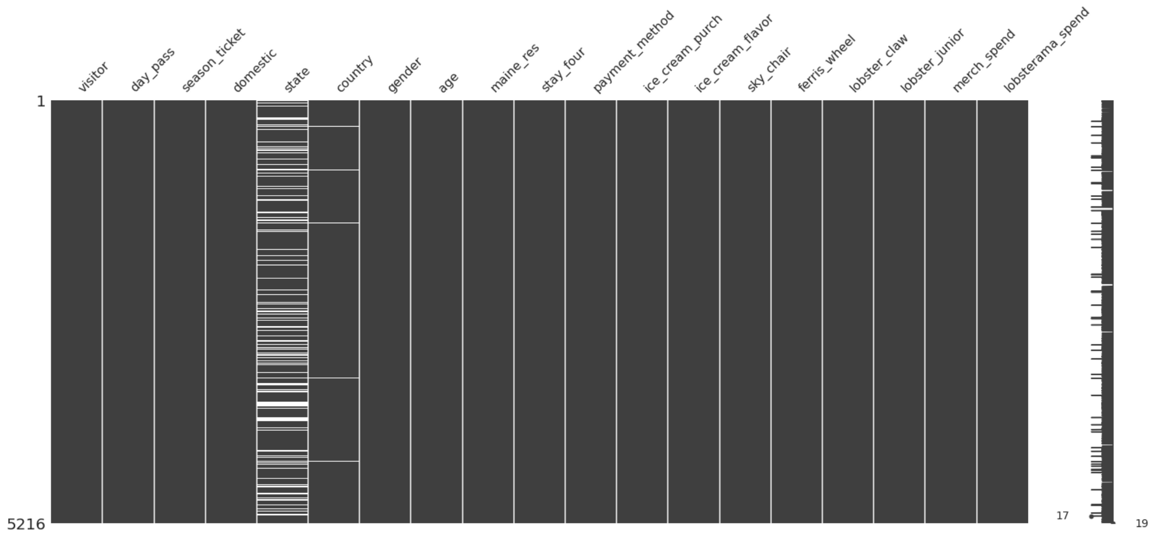
```
visitor              0.000000
day_pass             0.000000
season_ticket        0.000000
domestic             0.000000
state               20.878067
country              1.073620
gender               0.000000
age                  0.000000
maine_res            0.000000
stay_four            0.000000
payment_method       0.000000
ice_cream_purch      0.000000
ice_cream_flavor     0.000000
sky_chair            0.000000
ferris_wheel         0.000000
lobster_claw         0.000000
lobster_junior       0.000000
merch_spend          0.000000
lobsterama_spend     0.000000
dtype: float64
```

In [ ]:

```
#F(c):
import missingno as msno
msno.matrix(visitor_park)
```

Out[ ]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f9448fd6490>
```

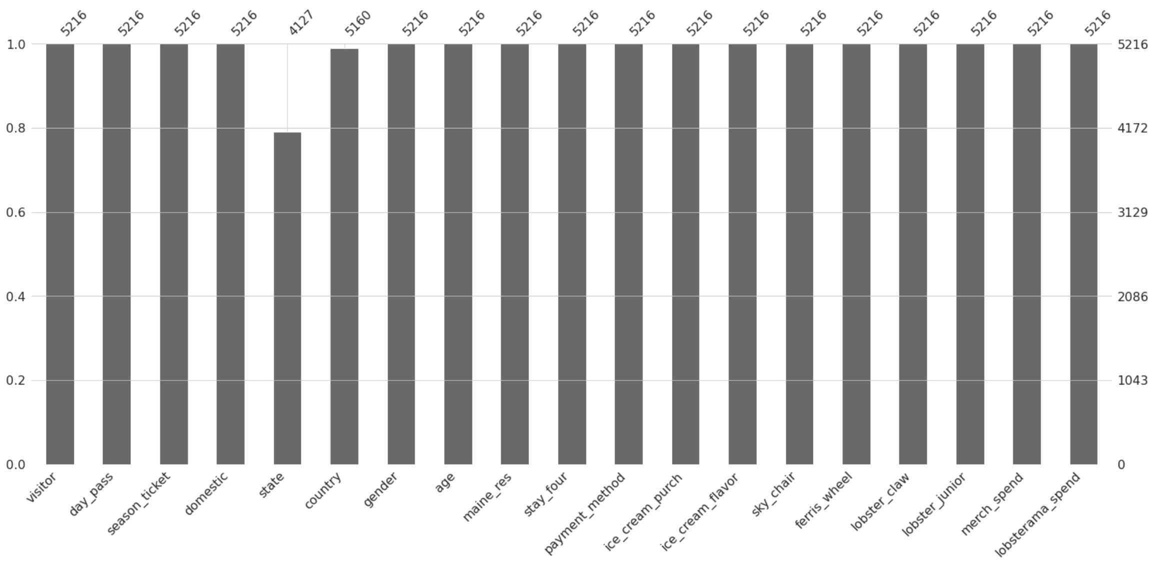


In [ ]:

```
#F(d):
msno.bar(visitor_park)
```

Out[ ]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f944c0ff8b0>
```



In [ ]:

```
#F(e):
visitor_park[visitor_park["state"].isna()]
```

Out[ ]:

	visitor	day_pass	season_ticket	domestic	state	country	gender	age	maine_res	sta
4	5	1	No	0	NaN	MEX	1	44	No	
10	11	1	No	0	NaN	CAN	1	37	No	
12	13	1	No	0	NaN	CHN	1	38	No	
14	15	1	No	0	NaN	BRA	1	40	No	
17	18	1	No	0	NaN	CAN	0	31	No	
...	...	...	...	...	...	...	...	...	...	
5202	5203	1	No	0	NaN	CAN	0	34	No	
5205	5206	1	No	0	NaN	CAN	1	39	No	
5206	5207	1	No	0	NaN	CHN	0	38	No	
5207	5208	1	No	0	NaN	CAN	1	35	No	
5212	5213	1	No	0	NaN	UK	1	30	No	

1089 rows × 19 columns

In [ ]:

```
#G(a):
visitor_park["age"] = np.where(visitor_park["age"] < 15, 15, visitor_park["age"])
visitor_park['age'] = visitor_park['age'].where(visitor_park['age']<15,15)
print(visitor_park)
```

```
      visitor  day_pass season_ticket  state country  gender  age main_per
\
0           1         1            No     NY     USA       0   15       No
1           2         0           Yes  Other     USA       1   15       No
2           3         1            No     ME     USA       1   15      Yes
3           4         0           Yes     NH     USA       1   15       No
4           5         1            No    NaN     MEX       1   15       No
...       ...       ...           ...    ...     ...     ...  ...      ...
5211     5212         0           Yes     NH     USA       0   15       No
5212     5213         1            No    NaN      UK       1   15       No
5213     5214         0           Yes     NH     USA       1   15       No
5214     5215         0           Yes     ME     USA       0   15      Yes
5215     5216         1            No     MA     USA       0   15       No

      stay_four  payment_method  ice_cream_purch ice_cream_flavor  sky_cha
ir  \
0             1               0                1          Vanilla
1
1             1               0                1        Chocolate
0
2             1               0                0             None
1
3             0               0                0             None
0
4             1               0                1          Vanilla
0
...         ...             ...              ...              ...
...
5211          1               0                0             None
0
5212          1               0                1          Vanilla
1
5213          0               0                1        Chocolate
1
5214          1               0                0             None
1
5215          1               0                0             None
0

      ferris_wheel  lobster_claw  lobster_junior  merch_spend  \
0                0             0               1    34.529611
1                0             1               0    23.811135
2                1             0               1    49.231936
3                1             1               0    55.508722
4                1             0               0    61.019885
...            ...           ...             ...          ...
5211             1             0               0    51.132212
5212             1             0               0    43.170379
5213             0             0               0    37.488318
5214             1             0               0    45.340795
5215             0             0               0    46.312087

      lobsterama_spend
0                24.95
1                16.58
2                29.94
3                49.95
4                36.62
...                ...
5211             22.17
5212             23.71
```

```
5213            48.33
5214            34.59
5215            27.02

[5216 rows x 18 columns]
```

In [ ]:

```
#H(a):
visitor_park[visitor_park["stay_four"]==1]
count = (visitor_park["stay_four"] == 1).sum()
total = len(visitor_park["stay_four"])
percentage = count / total * 100
print(percentage)
```

```
59.93098159509203
```

In [ ]:

```
#H(b):
visitor_park_usa = visitor_park[visitor_park["country"] == "USA"]
count = (visitor_park_usa["stay_four"] == 1).sum()
total = len(visitor_park_usa["stay_four"])
percentage_usa = count / total * 100
print(percentage_usa)
```

```
52.047492125030296
```

domestic visitors who stayed more than 4 years

In [ ]:

```
#H(b):
park_other = visitor_park[visitor_park["country"] != "USA"]
count = (park_other["stay_four"] == 1).sum()
total = len(park_other["stay_four"])
percentage_other = count / total * 100
print(percentage_other)
```

```
89.80716253443526
```

Percentage of international visitors stayed more than 4 years

In [129]:

```
#I(a):
park_visitor = visitor_park.drop(columns=["domestic"])
park_visitor.head()
```

Out[129]:

	visitor	day_pass	season_ticket	state	country	gender	age	maine_res	stay_four	paym
0	1	1	No	NY	USA	0	32	No	1	
1	2	0	Yes	Other	USA	1	43	No	1	
2	3	1	No	ME	USA	1	28	Yes	1	
3	4	0	Yes	NH	USA	1	35	No	0	
4	5	1	No	NaN	MEX	1	44	No	1	

Column country already gives us the information regarding domestic and international visitors information

In [ ]:

```
#J(a):
visitor_park = visitor_park.rename(columns={"maine_res" : "main_per"})
visitor_park.head(20)
```

Out[ ]:

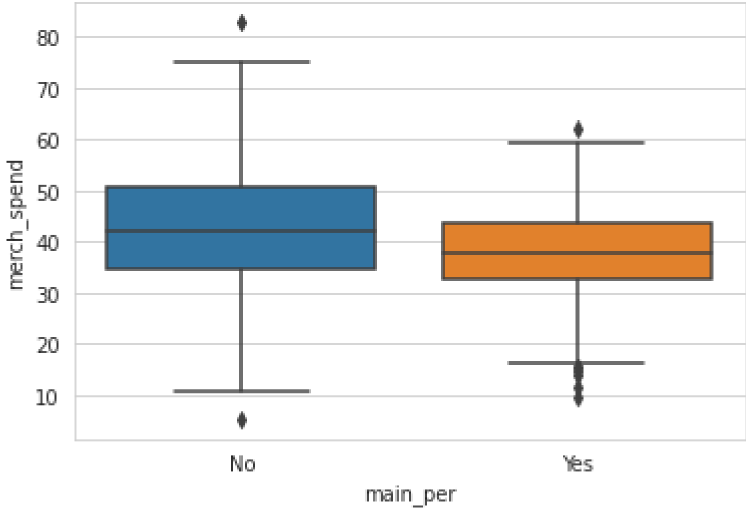
	visitor	day_pass	season_ticket	state	country	gender	age	main_per	stay_four	paym
0	1	1	No	NY	USA	0	15	No	1	
1	2	0	Yes	Other	USA	1	15	No	1	
2	3	1	No	ME	USA	1	15	Yes	1	
3	4	0	Yes	NH	USA	1	15	No	0	
4	5	1	No	NaN	MEX	1	15	No	1	
5	6	1	No	NY	USA	0	15	No	0	
6	7	1	No	Other	USA	1	15	No	0	
7	8	0	Yes	VT	USA	0	15	No	1	
8	9	0	Yes	NH	USA	0	15	No	0	
9	10	1	No	NH	USA	0	15	No	0	
10	11	1	No	NaN	CAN	1	15	No	1	
11	12	1	No	NY	USA	1	15	No	1	
12	13	1	No	NaN	CHN	1	15	No	1	
13	14	0	Yes	ME	USA	0	15	Yes	0	
14	15	1	No	NaN	BRA	1	15	No	1	
15	16	1	No	ME	USA	1	15	Yes	1	
16	17	1	No	NH	USA	0	15	No	0	
17	18	1	No	NaN	CAN	0	15	No	1	
18	19	1	No	MA	USA	1	15	No	0	
19	20	1	No	NaN	FRA	1	15	No	1	

In [ ]:

```
#2: Data Visualization
```

In [ ]:

```
#K:
import seaborn as sns
sns.boxplot(x = "main_per", y = "merch_spend", data = visitor_park);
```

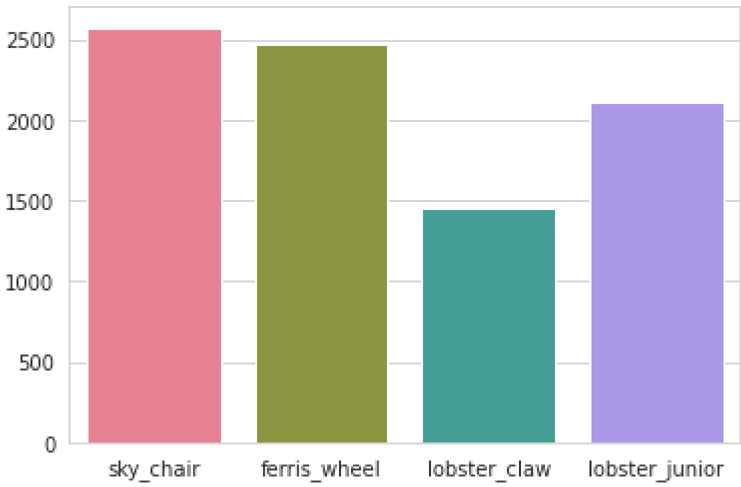


In [ ]:

```
#K(a):
```

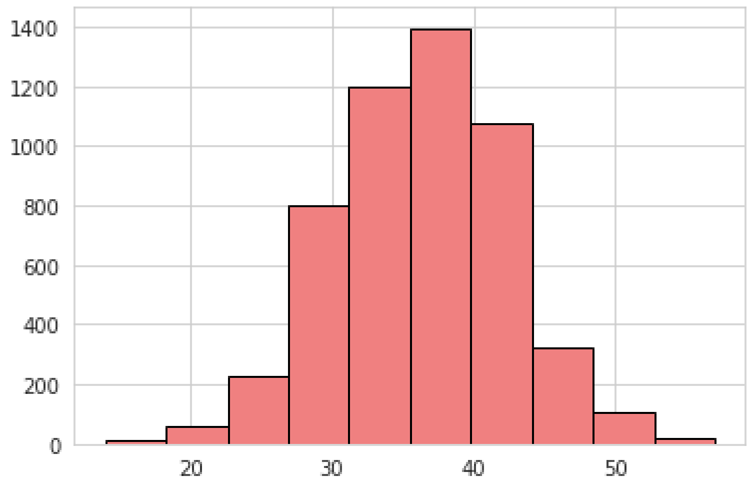
In [ ]:

```
#L:
ride_names = visitor_park[["sky_chair", "ferris_wheel", "lobster_claw", "lobster_junio
r"]]
sns.barplot(x = ride_names.columns, y = ride_names.sum(), palette='husl');
```



In [130]:

```
#M:
import matplotlib.pyplot as plt
plt.hist(visitor_park["age"], color = "lightcoral", edgecolor = "black");
```

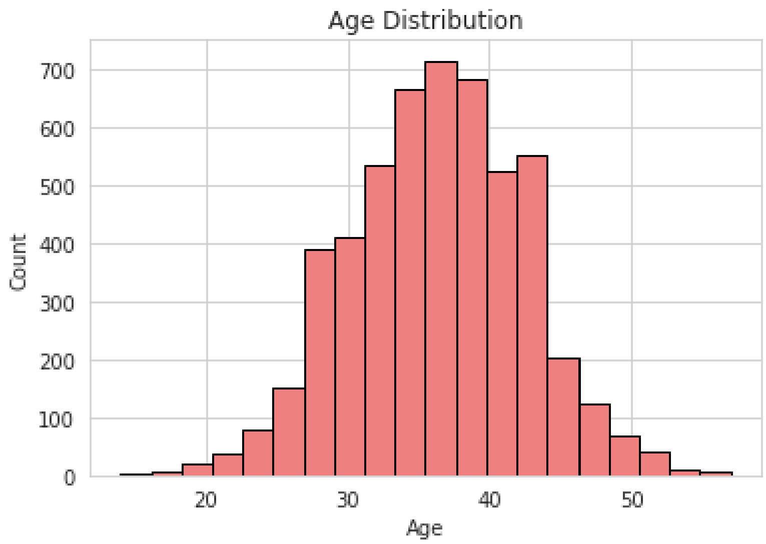


In [ ]:

```

```

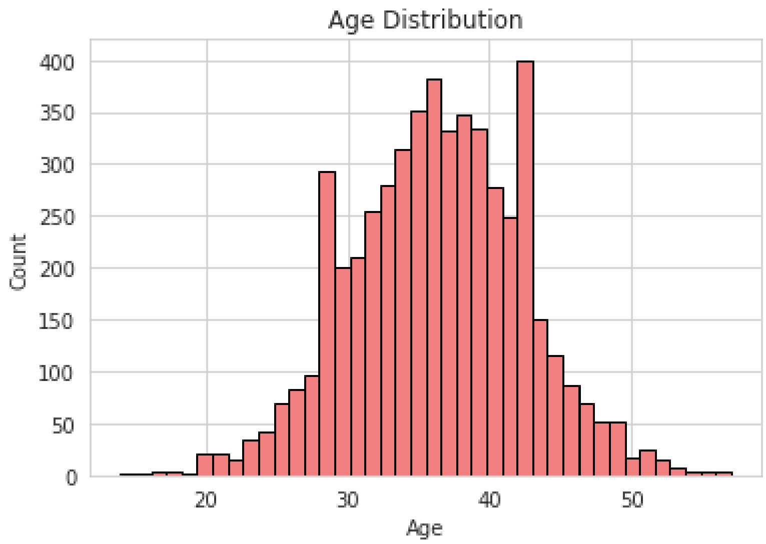
In [131]:

```
#M(b):
plt.hist(visitor_park["age"], bins = 20, color = "lightcoral", edgecolor = "black")
plt.xlabel('Age')
plt.ylabel('Count')
plt.title('Age Distribution');
```



In [132]:

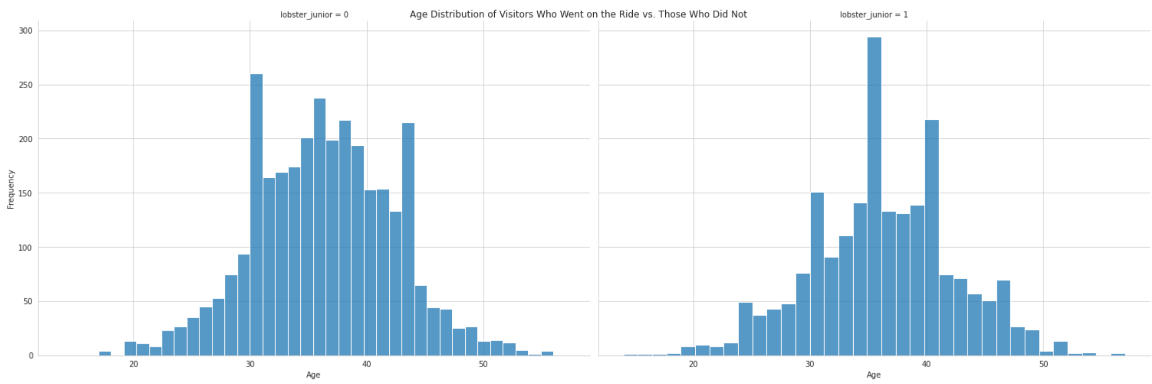
```
plt.hist(visitor_park["age"], bins = 40, color = "lightcoral", edgecolor = "black")
plt.xlabel('Age')
plt.ylabel('Count')
plt.title('Age Distribution');
```



increasing the number of makes the data more concentrated and normally distributed

In [136]:

```
#M(d):
g = sns.FacetGrid(visitor_park, col = "lobster_junior", height = 7, aspect = 1.5)
g.map(sns.histplot, "age", kde = False)
g.set_axis_labels("Age", "Frequency")
g.fig.suptitle("Age Distribution of Visitors Who Went on the Ride vs. Those Who Did No
t");
```

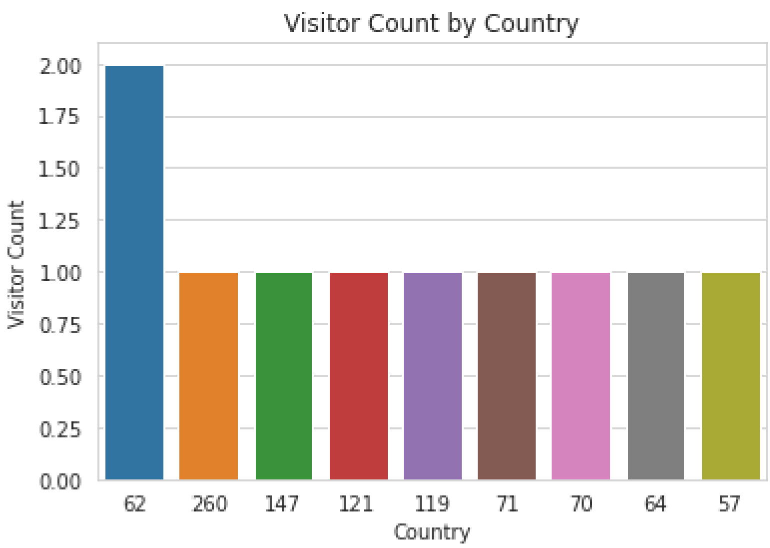


For me it does because both the data have different frequency, especially the ones who went were 35 years and shows the maximum frewuncy compared to ones who didnt . the ones who didnt have a much higher frequency

In [137]:

```
#N:
country_visitor_sum = visitor_park.groupby(["country"])["visitor"].count()
country_visitor_sum = country_visitor_sum[country_visitor_sum.index != "USA"]
print(country_visitor_sum)
country_visitor_sum = country_visitor_sum.sort_values(ascending = False)
country_visitor_sum_df = country_visitor_sum.to_frame()
print(country_visitor_sum_df)
sns.countplot(x=country_visitor_sum_df.index, data= country_visitor_sum, order= country
_visitor_sum.value_counts().index)
plt.xlabel("Country")
plt.ylabel("Visitor Count")
plt.title("Visitor Count by Country");
```

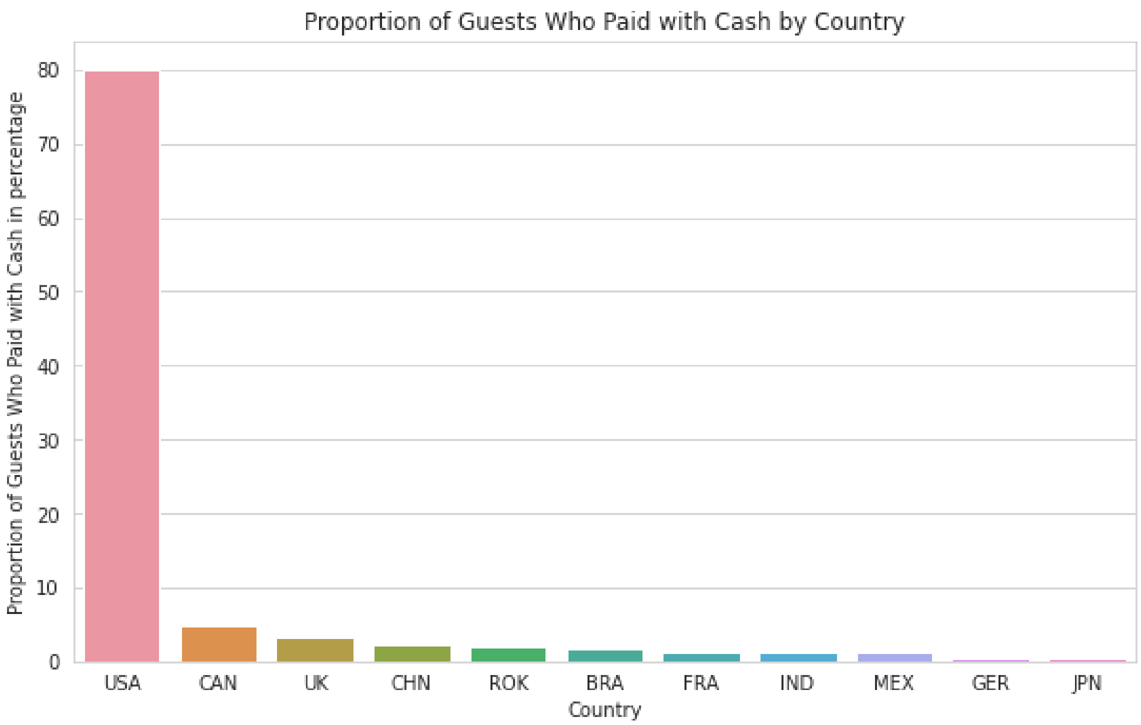
```
country
BRA    121
CAN    260
CHN    147
FRA     70
GER     57
IND     71
JPN     62
MEX     64
ROK     62
UK     119
Name: visitor, dtype: int64
         visitor
country
CAN          260
CHN          147
BRA          121
UK           119
IND           71
FRA           70
MEX           64
JPN           62
ROK           62
GER           57
```



This data shows the number of visitors for the certain international visitor and the visitor_count for the data. South korea has the most visitors compared to other countries.

In [ ]:

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

# Load the data into a DataFrame
df = pd.read_csv('july4_snapshot.csv') # replace with your file name

# Filter the data to only include guests who paid with cash
cash_df = df[df['payment_method'] == 1]



# Group the data by country and calculate the proportion of guests who paid with cash
country_proportions = cash_df.groupby('country').size() / len(cash_df)*100



# Sort the countries in descending order of proportion of guests who paid with cash
country_proportions = country_proportions.sort_values(ascending=False)

# Create a bar plot using seaborn
sns.set_style("whitegrid")
plt.figure(figsize=(10, 6))
sns.barplot(x=country_proportions.index, y=country_proportions.values)
plt.title('Proportion of Guests Who Paid with Cash by Country')
plt.xlabel('Country')
plt.ylabel('Proportion of Guests Who Paid with Cash in percentage')
plt.show()
```



In [ ]:

```
!pip install nbconvert
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/col
ab-wheels/public/simple/
Requirement already satisfied: nbconvert in /usr/local/lib/python3.8/dist-
packages (5.6.1)
Requirement already satisfied: jinja2>=2.4 in /usr/local/lib/python3.8/dis
t-packages (from nbconvert) (2.11.3)
Requirement already satisfied: bleach in /usr/local/lib/python3.8/dist-pac
kages (from nbconvert) (6.0.0)
Requirement already satisfied: pandocfilters>=1.4.1 in /usr/local/lib/pyth
on3.8/dist-packages (from nbconvert) (1.5.0)
Requirement already satisfied: pygments in /usr/local/lib/python3.8/dist-p
ackages (from nbconvert) (2.6.1)
Requirement already satisfied: defusedxml in /usr/local/lib/python3.8/dist
-packages (from nbconvert) (0.7.1)
Requirement already satisfied: jupyter-core in /usr/local/lib/python3.8/di
st-packages (from nbconvert) (5.2.0)
Requirement already satisfied: testpath in /usr/local/lib/python3.8/dist-p
ackages (from nbconvert) (0.6.0)
Requirement already satisfied: entrypoints>=0.2.2 in /usr/local/lib/python
3.8/dist-packages (from nbconvert) (0.4)
Requirement already satisfied: nbformat>=4.4 in /usr/local/lib/python3.8/d
ist-packages (from nbconvert) (5.7.3)
Requirement already satisfied: mistune<2,>=0.8.1 in /usr/local/lib/python
3.8/dist-packages (from nbconvert) (0.8.4)
Requirement already satisfied: traitlets>=4.2 in /usr/local/lib/python3.8/
dist-packages (from nbconvert) (5.7.1)
Requirement already satisfied: MarkupSafe>=0.23 in /usr/local/lib/python3.
8/dist-packages (from jinja2>=2.4->nbconvert) (2.0.1)
Requirement already satisfied: fastjsonschema in /usr/local/lib/python3.8/
dist-packages (from nbformat>=4.4->nbconvert) (2.16.2)
Requirement already satisfied: jsonschema>=2.6 in /usr/local/lib/python3.
8/dist-packages (from nbformat>=4.4->nbconvert) (4.3.3)
Requirement already satisfied: webencodings in /usr/local/lib/python3.8/di
st-packages (from bleach->nbconvert) (0.5.1)
Requirement already satisfied: six>=1.9.0 in /usr/local/lib/python3.8/dist
-packages (from bleach->nbconvert) (1.15.0)
Requirement already satisfied: platformdirs>=2.5 in /usr/local/lib/python
3.8/dist-packages (from jupyter-core->nbconvert) (3.0.0)
Requirement already satisfied: attrs>=17.4.0 in /usr/local/lib/python3.8/d
ist-packages (from jsonschema>=2.6->nbformat>=4.4->nbconvert) (22.2.0)
Requirement already satisfied: pyrsistent!=0.17.0,!=0.17.1,!=0.17.2,>=0.1
4.0 in /usr/local/lib/python3.8/dist-packages (from jsonschema>=2.6->nbfor
mat>=4.4->nbconvert) (0.19.3)
Requirement already satisfied: importlib-resources>=1.4.0 in /usr/local/li
b/python3.8/dist-packages (from jsonschema>=2.6->nbformat>=4.4->nbconvert)
(5.10.2)
Requirement already satisfied: zipp>=3.1.0 in /usr/local/lib/python3.8/dis
t-packages (from importlib-resources>=1.4.0->jsonschema>=2.6->nbformat>=4.
4->nbconvert) (3.13.0)
```

In [135]:

```
!jupyter nbconvert -- to html Aravindrao_assignment1.ipynb
```

```
[NbConvertApp] WARNING | pattern 'to' matched no files
[NbConvertApp] WARNING | pattern 'html' matched no files
[NbConvertApp] Converting notebook Aravindrao_assignment1.ipynb to html
[NbConvertApp] Writing 690440 bytes to Aravindrao_assignment1.html
```

The data shows that there are a lot USA visitors in general when comapred to other countries . USA visitors have paid around 80 percent in cash compared to other country visitors .

Part III: Wildcard: Metrics and “Quantified Self”

I chose to keep track of my daily steps for three days in a row as both a researcher and a subject. I measured and kept track of my daily step totals using the health app on my iphone. According to the results, I walked 2,900 steps on the first day, 3,8456 steps on the second, and 5,198 steps on the third.

My daily physical activity varied substantially from day to day as I discovered by keeping track of my steps. Due to doing errands and finishing home duties, the second day saw a marked increase in the number of steps I took compared to the previous two. I became more aware of my physical activity as a result of seeing the numbers in front of me, and I was inspired to exercise more on the third day. I chose to use the stairs rather than the elevator when I went for a walk in the park. Also, I discovered that keeping track of my steps helped me become more conscious of my daily routine and ways to include more exercise.

No one nearby responded to me tracking my steps throughout the course of the three days. The outcomes, though, were significant to me and gave me a chance to consider my routines. I'd definitely think about continuing this experiment for a while, maybe for a week or a month, to gain a better grasp of my daily physical activity patterns. Overall, this experiment served as a great reminder to emphasize exercise in my daily routine and to keep active.