Exploratory Data Analysis on Google Merchandise Store.

Insights about Dataset

This dataset provides a curated subset of the anonymized Google Analytics event data for three months of the Google Merchandise Store.The full dataset is available as a BigQuery Public Dataset.

The Data includes information on items sold in the store and how much money was spent by users over time.It is both comprehensive enough to invite real analysis yet simple enough to facilitate teaching.

step 1: Import Libraries

In [3]:
```
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
import plotly.express as px
import warnings
warnings.filterwarnings('ignore')
```

Step 2.Load the Dataset

In [7]:
```
df=pd.read_csv('items.csv')
df_users=pd.read_csv('users.csv')
df_items=pd.read_csv('events1.csv')
```

In [9]:
```
df.head()
```

Out[9]:

	id	name	brand	variant	category	price_in_usd
0	0	Google Land & Sea Cotton Cap	Google	Single Option Only	Apparel	14
1	1	Google KeepCup	Google	Single Option Only	New	28
2	2	Google Land & Sea Nalgene Water Bottle	Google	Single Option Only	Drinkware	20
3	3	Google Unisex Eco Tee Black	Google	LG	Uncategorized Items	22
4	4	Google Chicago Campus Bottle	Google	Single Option Only	Campus Collection	11

In [11]:
```
df_users.head()
```

Out[11]:

	id	ltv	date
0	0	0	2020-10-13 05:08:47
1	1	0	2020-11-24 14:26:54
2	2	0	2020-11-24 06:19:54
3	3	231	2020-05-02 11:09:15
4	4	102	2020-11-18 15:54:38

In [13]:
```
df_items.head()
```

Out[13]:

	user_id	ga_session_id	country	device	type	item_id	date
0	2133	16909	US	mobile	purchase	94	2020-11-01 00:27:14
1	2133	16909	US	mobile	purchase	425	2020-11-01 00:27:14
2	5789	16908	SE	desktop	purchase	1	2020-11-01 01:44:44
3	5789	16908	SE	desktop	purchase	62	2020-11-01 01:44:44
4	5808	4267	US	mobile	add_to_cart	842	2020-11-01 03:06:29

In [15]:
```
print(df.shape)
print(df_users.shape)
print(df_items.shape)
```

```
(1381, 6)
(270154, 3)
(758884, 7)
```

Step 3.Data Integration

Here we merging all the datasets together for better acknowledgement

In [17]:
```
merged_data=pd.merge(df,df_items,left_on='id',right_on='item_id',how='inner')

final_data=pd.merge(merged_data,df_users,left_on='user_id',right_on='id',how='inner')
```

In [19]:
```
final_data.head()
```

Out[19]:

	id_x	name	brand	variant	category	price_in_usd	user_id	ga_session_id	country	device	type	item_id	date_x	i
0	0	Google Land & Sea Cotton Cap	Google	Single Option Only	Apparel	14	5115	17001	US	mobile	purchase	0	2020-11-02 12:05:14	5
1	0	Google Land & Sea Cotton Cap	Google	Single Option Only	Apparel	14	10904	16401	TR	desktop	purchase	0	2020-11-03 08:19:14	10
2	0	Google Land & Sea Cotton Cap	Google	Single Option Only	Apparel	14	29457	17113	KR	mobile	purchase	0	2020-11-05 18:02:19	29
3	0	Google Land & Sea Cotton Cap	Google	Single Option Only	Apparel	14	30148	16175	MT	desktop	purchase	0	2020-11-05 18:33:59	30
4	0	Google Land & Sea Cotton Cap	Google	Single Option Only	Apparel	14	32087	15869	US	desktop	purchase	0	2020-11-06 03:39:46	32

In [21]:
```
df=final_data
```

In [23]:
```
df.shape
```

Out[23]:
```
(758884, 16)
```

Step 4.Data Cleaning

4.1 Handling missing Data

In [25]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 758884 entries, 0 to 758883
Data columns (total 16 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   id_x           758884 non-null  int64
 1   name           758884 non-null  object
 2   brand          758884 non-null  object
 3   variant        122624 non-null  object
 4   category       758884 non-null  object
 5   price_in_usd   758884 non-null  int64
 6   user_id        758884 non-null  int64
 7   ga_session_id  758884 non-null  int64
 8   country        754329 non-null  object
 9   device         758884 non-null  object
 10  type           758884 non-null  object
 11  item_id        758884 non-null  int64
 12  date_x         758884 non-null  object
 13  id_y           758884 non-null  int64
 14  ltv            758884 non-null  int64
 15  date_y         758884 non-null  object
dtypes: int64(7), object(9)
memory usage: 92.6+ MB
```

In [27]:
```
df.describe()
```

Out[27]:

	id_x	price_in_usd	user_id	ga_session_id	item_id	id_y	ltv
count	758884.000000	758884.000000	758884.000000	758884.000000	758884.000000	758884.000000	758884.000000
mean	885.629356	24.759137	29541.809665	3632.325492	885.629356	29541.809665	61.428210
std	289.267126	20.892459	55129.337846	3952.875337	289.267126	55129.337846	124.049917
min	0.000000	1.000000	1.000000	0.000000	0.000000	1.000000	0.000000
25%	913.000000	11.000000	8401.000000	625.000000	913.000000	8401.000000	0.000000
50%	972.000000	22.000000	17181.000000	2103.000000	972.000000	17181.000000	0.000000
75%	1040.000000	30.000000	25809.000000	5365.000000	1040.000000	25809.000000	80.000000
max	1380.000000	313.000000	270145.000000	18033.000000	1380.000000	270145.000000	1530.000000

In [29]:
```
df.isnull().sum()
```

Out[29]:
```
id_x                0
name                0
brand               0
variant        636260
category            0
price_in_usd        0
user_id             0
ga_session_id       0
country          4555
device              0
type                0
item_id             0
date_x              0
id_y                0
ltv                 0
date_y              0
dtype: int64
```

In [31]:
```
(df['variant'].isnull().sum()/len(df))*100
```

Out[31]:
```
83.84153572878068
```

In [33]:
```
(df['country'].isnull().sum()/len(df))*100
```

Out[33]:
```
0.6002234860663817
```

for this 'varient' columns we have to drop that because more number of null values we can't fill and do make any analysis on this so,we are removing this

In [35]:
```
df.drop(columns=['variant'],inplace=True)
```

We should delete date_y columns because of we have two date columns there we will confuse so ,we have to delete

In [37]:
```
df.drop(columns=['date_y'],inplace=True)
```

In [39]:
```
df.drop(columns=['id_y'],inplace=True)
```

for this country we have only 0.6% of null values we can remove that,we have no problem for removing there is no much impact while doing modelling.

In [41]:
```
df.dropna(inplace=True)
```

In [43]:
```
df.shape
```

Out[43]: (754329, 13)

4.2 Removing Duplicates

In [ ]:
```
#check is there any duplicates or not
```

In [45]:
```
print(f"Number of Duplicates rows:{df.duplicated().sum()}")
```

```
Number of Duplicates rows:39234
```

In [127…
```
# Removing duplicates rows if found
```

In [47]:
```
df.drop_duplicates(inplace=True)
```

In [49]:
```
df.duplicated().sum()
```

Out[49]: 0

So we have successfully deleted the duplicated records from the dataset without any disturbances for analysisng and while going for Machine Learning models.

In [51]:
```
df.shape
```

Out[51]: (715095, 13)

In [53]:
```
df.head(2)
```

Out[53]:

	id_x	name	brand	category	price_in_usd	user_id	ga_session_id	country	device	type	item_id	date_x	ltv
0	0	Google Land & Sea Cotton Cap	Google	Apparel	14	5115	17001	US	mobile	purchase	0	2020-11-02 12:05:14	85
1	0	Google Land & Sea Cotton Cap	Google	Apparel	14	10904	16401	TR	desktop	purchase	0	2020-11-03 08:19:14	40

In [55]:
```
df.rename(columns={'price_in_usd': 'Price'},inplace=True)
```

In [57]:
```
df.rename(columns={'Item_Transaction_Volume': 'Lifetime Value'},inplace=True)
```

In [59]:
```
df[['brand','category','country','device','type']].nunique()
```

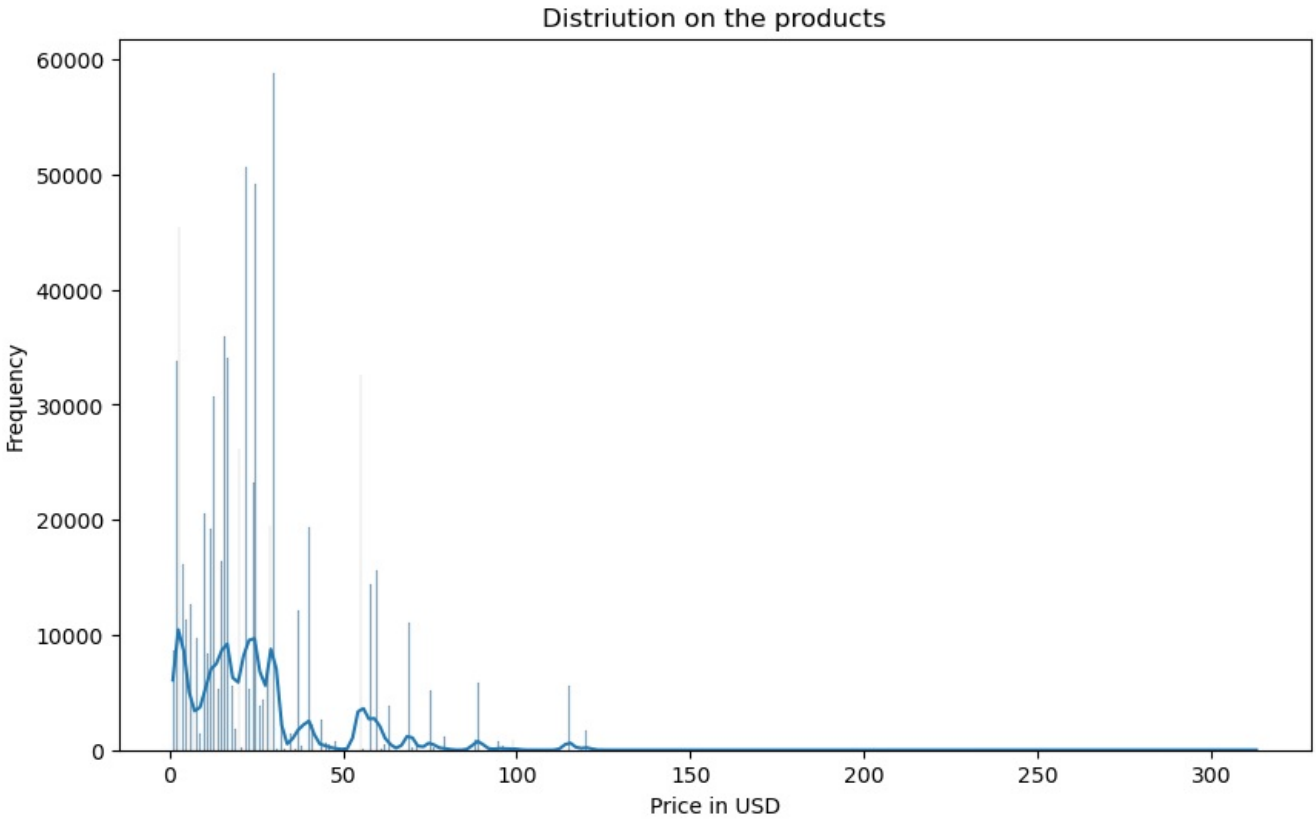
Out[59]:
```
brand         5
category     21
country     108
device        3
type          3
dtype: int64
```

In [61]:
```
for column in ['brand','category','country','device','type']:
    unique_values = df[column].unique()
    print(f"Unique values in {column}: {unique_values}")
    print("**********************************")
```

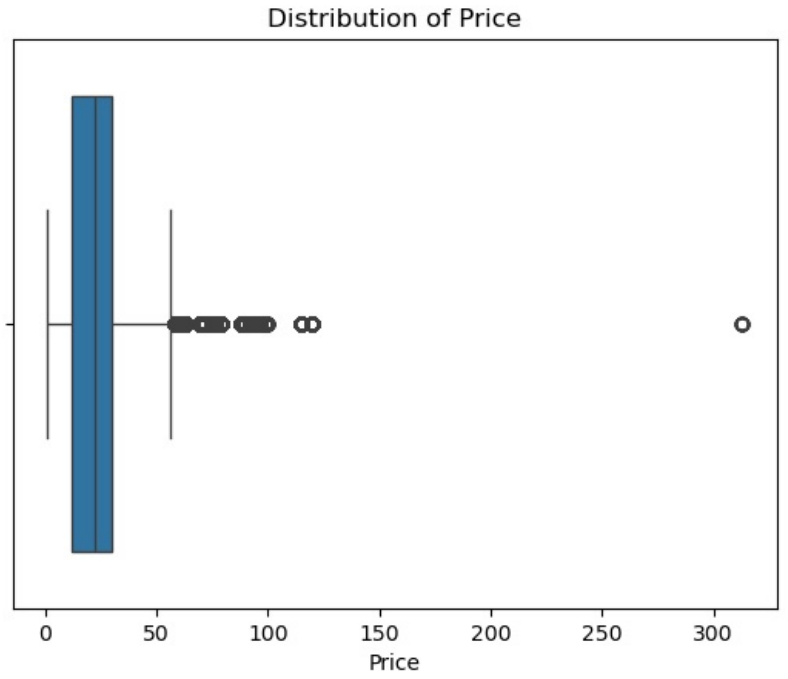
```
Unique values in brand: ['Google' 'Android' 'YouTube' '#IamRemarkable' 'Google Cloud']
**********
Unique values in category: ['Apparel' 'New' 'Drinkware' 'Uncategorized Items' 'Campus Collection'
 'Clearance' 'Shop by Brand' 'Small Goods' 'Black Lives Matter'
 'Electronics Accessories' 'Lifestyle' 'Bags' 'Accessories' 'Office'
 'Stationery' 'Fun' 'Google' 'Writing Instruments' 'Notebooks & Journals'
 'Eco-Friendly' 'Gift Cards']
**********
Unique values in country: ['US' 'TR' 'KR' 'MT' 'ES' 'MX' 'IL' 'IN' 'CA' 'GB' 'PL' 'MY' 'QA' 'TW'
 'GR' 'ID' 'RU' 'DE' 'PE' 'FR' 'PK' 'EG' 'JP' 'BD' 'SE' 'HN' 'RS' 'UY'
 'NL' 'IT' 'SG' 'CN' 'HK' 'IE' 'CO' 'TH' 'HR' 'BR' 'AU' 'LK' 'PT' 'CZ'
 'CY' 'AT' 'BA' 'DK' 'RO' 'DZ' 'SA' 'NZ' 'UA' 'NG' 'SK' 'BY' 'BG' 'VN'
 'CH' 'PH' 'SI' 'AR' 'MA' 'GT' 'IQ' 'NO' 'KZ' 'SV' 'LV' 'CL' 'PR' 'BE'
 'PS' 'FI' 'AE' 'DO' 'KW' 'LU' 'GH' 'LT' 'EC' 'GE' 'AM' 'JO' 'XK' 'MK'
 'PY' 'ZA' 'IS' 'MN' 'TN' 'AZ' 'HU' 'KH' 'CR' 'VE' 'TT' 'BO' 'PA' 'EE'
 'AL' 'JM' 'BS' 'NP' 'MO' 'LB' 'BH' 'MM' 'KE' 'OM']
**********
Unique values in device: ['mobile' 'desktop' 'tablet']
**********
Unique values in type: ['purchase' 'add_to_cart' 'begin_checkout']
**********
```

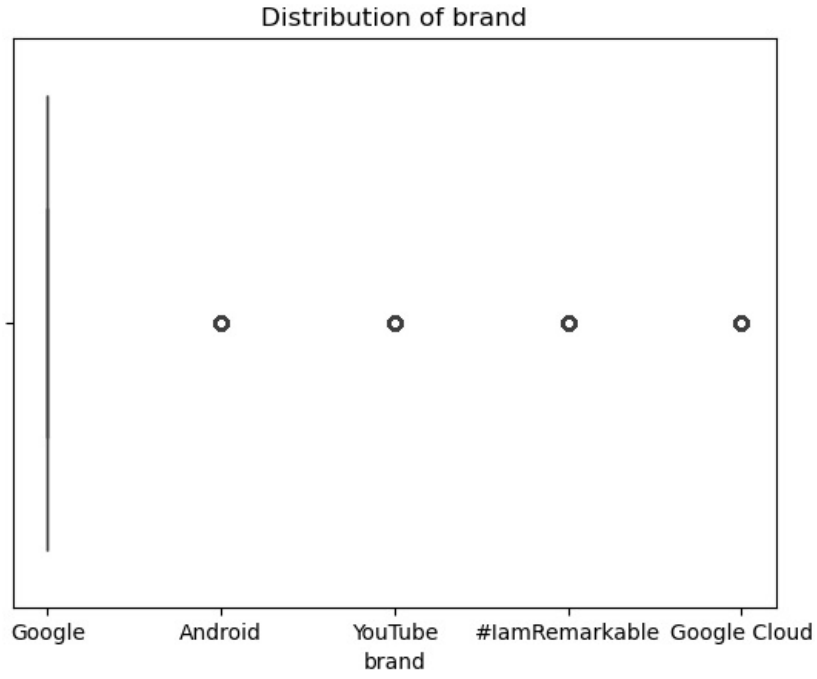
2.3 Handling Outliers

In [63]:
```
plt.figure(figsize=(10,6))
sns.histplot(df['Price'],kde=True)
plt.title('Distriution on the products')
plt.xlabel('Price in USD')
plt.ylabel('Frequency')
plt.show()
```



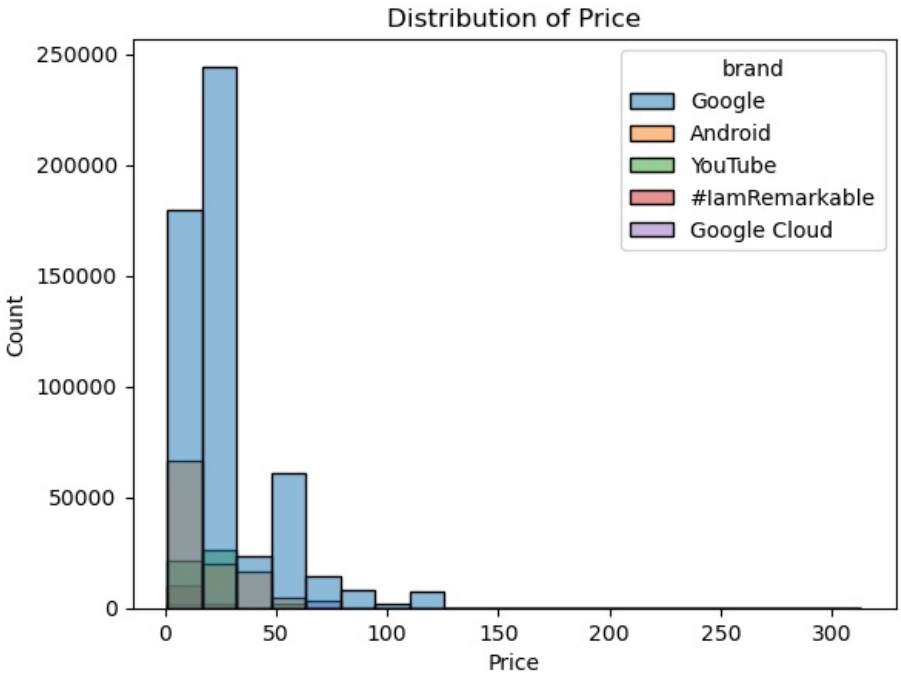
In [74]:
```
for i in ['Price','brand']:
    sns.boxplot(data=df,x=i)
    plt.title(f"Distribution of {i}")
    plt.show()
```





In [291…

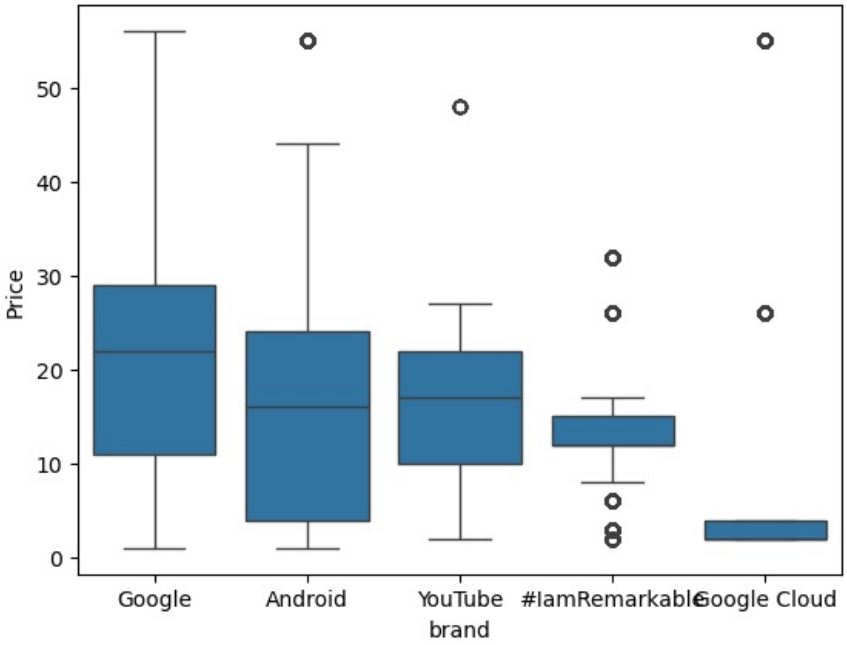
```
sns.histplot(x='Price',hue='brand',data=df,bins=20)
plt.title("Distribution of Price")
plt.show()
```



In [86]:
```
Q1=df['Price'].quantile(0.25)
Q3=df['Price'].quantile(0.75)
IQR=Q3-Q1
lower_bound=Q1-1.5*IQR
upper_bound=Q3+1.5*IQR
```

In [88]:
```
df=df[(df['Price']>=lower_bound)&(df['Price']<=upper_bound)]
```

In [99]:
```
sns.boxplot(data=df,x='brand',y='Price')
plt.show()
```



I think Most of the outliers are removed so we can proceed with nextr step .

In [144…
```
top_countries=df.groupby('country')['Price'].sum().nlargest(10)
top_countries_df=top_countries.reset_index()
top_countries_df
```

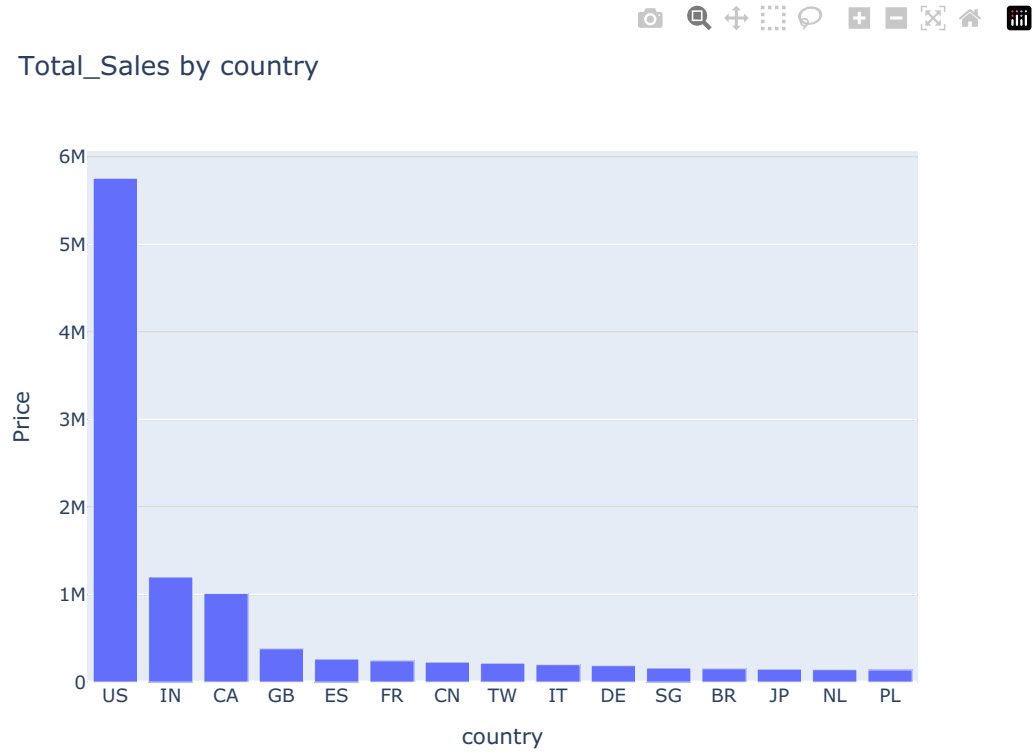
Out[144…

	country	Price
0	US	5757648
1	IN	1201400
2	CA	1013037
3	GB	381401
4	ES	263467
5	FR	243556
6	CN	228323
7	TW	216623
8	IT	199411
9	DE	187953

In [148…

```
Country_Sales=df.groupby('country')['Price'].sum().nlargest(15)

fig=px.bar(Country_Sales,x=Country_Sales.index,y='Price',title='Total_Sales by country')
fig.show()
```



In [150…

```
df.head(1)
```

Out[150…

	id_x	name	brand	category	Price	user_id	ga_session_id	country	device	type	item_id	date_x	ltv
0	0	Google Land & Sea Cotton Cap	Google	Apparel	14	5115	17001	US	mobile	purchase	0	2020-11-02 12:05:14	85

In [152…

```
Total_Sales_brand=df.groupby('brand')['Price'].sum()
```

In [160…

```
fig=px.pie(df,names='brand',values='Price',title='Price of brand')
fig.show()
```

In [162…
```
df['date_x']=pd.to_datetime(df['date_x'],dayfirst=True,errors='coerce')
df.head()
```

Out[162…

	id_x	name	brand	category	Price	user_id	ga_session_id	country	device	type	item_id	date_x	ltv
0	0	Google Land & Sea Cotton Cap	Google	Apparel	14	5115	17001	US	mobile	purchase	0	2020-02-11 12:05:14	85
1	0	Google Land & Sea Cotton Cap	Google	Apparel	14	10904	16401	TR	desktop	purchase	0	2020-03-11 08:19:14	40
2	0	Google Land & Sea Cotton Cap	Google	Apparel	14	29457	17113	KR	mobile	purchase	0	2020-05-11 18:02:19	33
3	0	Google Land & Sea Cotton Cap	Google	Apparel	14	30148	16175	MT	desktop	purchase	0	2020-05-11 18:33:59	517
4	0	Google Land & Sea Cotton Cap	Google	Apparel	14	32087	15869	US	desktop	purchase	0	2020-06-11 03:39:46	55

In [166…
```
df['Year']=df['date_x'].dt.year
df['Month']=df['date_x'].dt.month
df['Day']=df['date_x'].dt.day
df.head()
```

Out[166…

	id_x	name	brand	category	Price	user_id	ga_session_id	country	device	type	item_id	date_x	ltv	Year	Mont
0	0	Google Land & Sea Cotton Cap	Google	Apparel	14	5115	17001	US	mobile	purchase	0	2020-02-11 12:05:14	85	2020.0	2.
1	0	Google Land & Sea Cotton Cap	Google	Apparel	14	10904	16401	TR	desktop	purchase	0	2020-03-11 08:19:14	40	2020.0	3.
2	0	Google Land & Sea Cotton Cap	Google	Apparel	14	29457	17113	KR	mobile	purchase	0	2020-05-11 18:02:19	33	2020.0	5.
3	0	Google Land & Sea Cotton Cap	Google	Apparel	14	30148	16175	MT	desktop	purchase	0	2020-05-11 18:33:59	517	2020.0	5.
4	0	Google Land & Sea Cotton Cap	Google	Apparel	14	32087	15869	US	desktop	purchase	0	2020-06-11 03:39:46	55	2020.0	6.

In [170…

```
Month_mapping={
    1:"January",
    2:"February",
    3:"March",
    4:"Aprile",
    5:"May",
    6:"June",
    7:"July",
    8:"Agust",
    9:"September",
    10:"october",
    11:"November",
    12:"December"}
df['Month']=df['Month'].replace(Month_mapping)
```

In [172…

```
df.head()
```

Out[172…

	id_x	name	brand	category	Price	user_id	ga_session_id	country	device	type	item_id	date_x	ltv	Year	Mo
0	0	Google Land & Sea Cotton Cap	Google	Apparel	14	5115	17001	US	mobile	purchase	0	2020-02-11 12:05:14	85	2020.0	Febru
1	0	Google Land & Sea Cotton Cap	Google	Apparel	14	10904	16401	TR	desktop	purchase	0	2020-03-11 08:19:14	40	2020.0	Ma
2	0	Google Land & Sea Cotton Cap	Google	Apparel	14	29457	17113	KR	mobile	purchase	0	2020-05-11 18:02:19	33	2020.0	M
3	0	Google Land & Sea Cotton Cap	Google	Apparel	14	30148	16175	MT	desktop	purchase	0	2020-05-11 18:33:59	517	2020.0	M
4	0	Google Land & Sea Cotton Cap	Google	Apparel	14	32087	15869	US	desktop	purchase	0	2020-06-11 03:39:46	55	2020.0	Ju

In [174…

```
month_to_quarter_mapping = {
    'January': 'Q1', 'February': 'Q1', 'March': 'Q1',
    'Aprile': 'Q2', 'May': 'Q2', 'June': 'Q2',
    'July': 'Q3', 'Agust': 'Q3', 'September': 'Q3',
    'october': 'Q4', 'November': 'Q4', 'December': 'Q4'
}
```

```
#Replacing month names with quarter values.
df['Quarter']=df['Month'].replace(month_to_quarter_mapping)
```

In [3]:
```
fig=px.bar(group_year,x=group_year.index,y='Price',color='brand')
fig.show()
```