Define EDA?

Exploratory data analysis is the analysis of the data and brings out insights. It’s storytelling, a story that data is trying to tell. EDA is an approach to analyzing the data with the help of various tools and graphical techniques like barplot, histogram, etc.

In [89]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore")
```

Importing libraries

In [90]:
```
df = pd.read_csv(r"C:\Users\divya\OneDrive\Desktop\Mandava\tourism_dataset.c
```

In [91]:
```
df
```

Out[91]:

	Location	Country	Category	Visitors	Rating	Revenue	Accommoc
0	kuBZRkVsAR	India	Nature	948853	1.32	84388.38	
1	aHKUXhjzTo	USA	Historical	813627	2.01	802625.60	
2	dlrdYtJFTA	Brazil	Nature	508673	1.42	338777.11	
3	DxmlzdGkHK	Brazil	Historical	623329	1.09	295183.60	
4	WJCCQlepnz	France	Cultural	124867	1.43	547893.24	
...	...	...	...	...	...	...	
5984	xAzwnVKAqz	USA	Urban	828137	1.97	132848.78	
5985	IfKotyaJFC	France	Nature	276317	3.53	325183.96	
5986	bPyubCWGgA	Egypt	Beach	809198	3.37	927336.50	
5987	kkWIucpBnu	Egypt	Cultural	808303	2.52	115791.43	
5988	gHXUrdticm	France	Cultural	40939	4.65	957026.85	

5989 rows × 7 columns

Data loading

In [92]:
```
df.shape
```

Out[92]:
```
(5989, 7)
```

Loading [MathJax]/extensions/Safe.js

Shape of the data

In [93]:
```
df.head()
```

Out[93]:

	Location	Country	Category	Visitors	Rating	Revenue	Accommodatio
0	kuBZRkVsAR	India	Nature	948853	1.32	84388.38	
1	aHKUXhjzTo	USA	Historical	813627	2.01	802625.60	
2	dlrdYtJFTA	Brazil	Nature	508673	1.42	338777.11	
3	DxmlzdGkHK	Brazil	Historical	623329	1.09	295183.60	
4	WJCCQlepnz	France	Cultural	124867	1.43	547893.24	

Head() method is used to display Top 5 rows of the dataset

In [94]:
```
df.tail()
```

Out[94]:

	Location	Country	Category	Visitors	Rating	Revenue	Accommoc
5984	xAzwnVKAqz	USA	Urban	828137	1.97	132848.78	
5985	IfKotyaJFC	France	Nature	276317	3.53	325183.96	
5986	bPyubCWGgA	Egypt	Beach	809198	3.37	927336.50	
5987	kkWIucpBnu	Egypt	Cultural	808303	2.52	115791.43	
5988	gHXUrdticm	France	Cultural	40939	4.65	957026.85	

Tail() method is used to display Last 5 rows of the dataset

In [95]:
```
df.isna().sum()
```

Out[95]:
```
Location                  0
Country                   0
Category                  0
Visitors                  0
Rating                    0
Revenue                   0
Accommodation_Available   0
dtype: int64
```

This method prints information about the DataFrames. It is used to returns an index object and can be used to view or assign new values to the column lables. Isna() method to create a boolean mask and then use the sum() method to count the number of True values.

In [96]:
```
df.duplicated().sum()
```

Out[96]:
```
np.int64(0)
```

Loading [MathJax]/extensions/Safe.js

This method returns series with True and False values that describe which rows in the DataFrame are duplicated and not

In [97]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5989 entries, 0 to 5988
Data columns (total 7 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   Location                 5989 non-null   object 
 1   Country                  5989 non-null   object 
 2   Category                 5989 non-null   object 
 3   Visitors                 5989 non-null   int64  
 4   Rating                   5989 non-null   float64
 5   Revenue                  5989 non-null   float64
 6   Accommodation_Available  5989 non-null   object 
dtypes: float64(2), int64(1), object(4)
memory usage: 327.6+ KB
```

It gives information about the dataset

In [98]:
```
df.describe()
```

Out[98]:

	Visitors	Rating	Revenue
count	5989.000000	5989.000000	5989.000000
mean	501016.089497	3.009347	499479.367253
std	289783.294978	1.155980	286743.225211
min	1108.000000	1.000000	1025.810000
25%	252789.000000	2.010000	251410.450000
50%	500831.000000	3.000000	494169.350000
75%	751371.000000	4.010000	742241.240000
max	999982.000000	5.000000	999999.490000

Describe() method is used to returns the description of the data in the DataFrame. It isused for calculating some statistical data like mean,sd of numerical values of the df.

In [99]:
```
df.describe(include=["O"])
```

Loading [MathJax]/extensions/Safe.js

Out[99]:

	Location	Country	Category	Accommodation_Available
count	5989	5989	5989	5989
unique	5989	7	6	2
top	gHXUrdticm	Egypt	Adventure	Yes
freq	1	912	1037	3013

Describe(include=["O"]) is used to generate descriptive statistics for columns that contain datatypes like strings etc. The output includes counts,unique values,topvalues and frequency of the top values for each categorial column in the Dataframe.

In [100…
```
df.columns
```

Out[100…
```
Index(['Location', 'Country', 'Category', 'Visitors', 'Rating', 'Revenue',
       'Accommodation_Available'],
      dtype='object')
```

How many columns are there in the dataset

In [101…
```
df.Location.nunique()
```

Out[101…
```
5989
```

Here,the NUnique method() returns the number of unique values for each column by specifying axis

In [102…
```
df.Country.unique()
```

Out[102…
```
array(['India', 'USA', 'Brazil', 'France', 'Egypt', 'China', 'Australia'],
      dtype=object)
```

Here, The Unique method() returns the number of of the unique values for each column.

In [103…
```
locations_in_india = df.Location[df.Country == "India"]

locations_in_india
```

Loading [MathJax]/extensions/Safe.js

```
Out[103…  0       kuBZRkVsAR
          9       pXDJPYzTeU
          20      fBNHCwBuah
          21      gtHGXBHVIR
          25      QHzRNEjJep
                     ...
          5956    WXUfandjUr
          5957    wObJUWCFfv
          5960    ahIRHLEpUE
          5969    JCDXHjzueB
          5970    VgKQvxsZsN
          Name: Location, Length: 896, dtype: object
```

It shows a list of locations in India, through the values appear to be encoded

```
In [104…  for location in locations_in_india:
              print(location,end=" ,  ")
```

Loading [MathJax]/extensions/Safe.js

kuBZRkVsAR , pXDJPYzTeU , fBNHCwBuah , gtHGXBHVIR , QHzRNEjJep , YaykNh tAtN , IhdYrxBzdF , YNxOOsnhWp , vWuJdinaxV , gSGQJYJgbm , BiZZbRNQfu , oGBBtYfEWA , nnRHZzrtgR , rmaqyzxLMw , paGxLGQguG , VleoeMkDnz , tHkTbd zseF , nlPYcKOhga , HwAgVIacsv , kCCgBHBvBQ , zrKgcGqRpk , DVlHsSQUct , mJCfjUiDVK , MOCuVgZYGV , ZaOkXhaLdT , oCFSxJULgp , IMsAVknzrL , WlULch ARwr , dzbjzRHsdy , SnwfJyONMt , iOgichMLcb , tNhrEJwhyX , UEWDMbIRyJ , nppiylEpHM , EWwWubBrhF , ClMnDeZiRw , uNKHOIedpR , lpxprkgQgP , IxSKtd VRYj , SiezkRGbyu , xGfTmAZarc , qfTxISomoN , vxSJGawDdw , fNljFrVtqk , EHIlBqKROp , bpZIxNaBIu , pLdqWtcrYz , eaceBXGBuR , tSYdJmXEXL , yujuAB hSpA , furoUEdWXc , NDVPVbyIcr , aaqVpBnaNz , QpqdSIAnqD , PHqHbNoiSd , zBnsZceFyF , jNcfrqEKAO , OVIIQdFdpo , FsLQrDZUPA , xoAsslwobC , Tahudi QwqT , fryJSoCzEr , aoqtjRQAru , jdOUqjxhpM , ECIhAnwPnf , CuWrasCAEC , VqAyPQvUYf , mAFQmNnmQx , bDTLejbeVD , GwoyBeGlNQ , KUtUdviAWX , XIpxbs XFjI , cUIBGxawhj , OWZgWiThSm , NotsyUNTsz , qZQbQicFnV , guwRhIgPwu , SsTsnIDdqg , kwvhjzwqQF , pxDzMfSsqL , NzveQVwFhz , vGedyibSvJ , VMZRxs YsrL , xMbxASeSgd , qRbqaDtDAq , FXNWksOPgn , KHzgAGvaWz , sAHGjEjEwN , eGFkZkxftC , ZjjiVLSoxV , kRBcMAWXYw , qkgaBdtNGw , nEKKdpkpoO , FJMgOz KygS , VOKFNtMMTd , MQSgEIWxaa , bIQJyBHftf , yrLNKqbqYW , txqOnkLlwF , jhlZsfZROh , kFfKwBAZfE , rbSREiDjhd , upsoDxPduM , mUTWaQuYoX , xuYVEW Cbua , yUSzftHRnr , EXCFAcUBgv , cjNSmBvXYs , zMWQxHrjkm , zawakjKBih , SmnPvJJinb , xTzzQBSoMS , RHITRLLNIb , BZjArXxLZE , WtSgVCgzEC , NktcFg joyg , ReZZOJpxqM , KpKlBlrKAu , rLuQEtYDKx , RQFujpefaU , MIEVMOiJBI , VEYQPAiBeO , EubccVkzlk , QFXdaWPjtT , BElvlNeiNT , hLVekecbIL , ONdnPl AelT , HnUcRjnqcx , cDKneeOmrC , zefpXhsWmJ , nqksaLAfLr , TyiTFCmYku , cATRCRdmRZ , TUUCqLpcQs , rsHUqUmhYp , bbkGRrhPDp , zUuOFbTpzj , jQFSeN azRY , aHaOTGydqZ , PdWTsCuakb , PeWVvxHyBy , tmvRyTfImF , RPBNjTyWsq , hceIFpnLgG , dBTQyteDGw , tvDeHbuohZ , cCLnHfxyGP , xtwDQIFndE , DrgtEX jeha , jPlgCmwJdM , wvWmfdcgpC , WOtSqeijDF , kAqtJDYUGw , rTuBSTFSOJ , XKhtHRBhBF , geZmfaKqlz , zMsEBIzdcu , xdPPvBAqNL , pTaejJwbVQ , vdxYoG JbjE , XXtGCrvVwX , wfjxcPGftw , BUSCAvUGsx , LFieAuvLFe , HLtZNCJJgo , ZOXzXwjFPG , cncJMVkfKH , jLApmTuDdY , EHWUmbIgZJ , KEGHRlejmc , DloisX ivQn , QJZEHzclDK , LpSriRQJYe , PtOVsZCRQL , poClQtzkPd , zBxGNQMWmL , rLNKxUjNEj , iwCOzRDmmi , PfdiDABnoV , knevcrblFE , tMSgrXlcJh , sZtwVx auLH , kUGoaAqQXs , MnUIiMORSI , OWKyjlPFPZ , JTxxFoUxPq , irBRKUncXs , JGedPYhZHl , DPxQZxTnel , zQSSzYfnjG , xZgxwuaSCb , IOSpWyHgdm , riWaLV ynQE , fZBQiYTyDt , frTyUTBPpI , wvrGhgsEMq , XSmEKxdKKH , oVlZfTmClh , rpUsNAfddy , pxvsqXJSAv , hilaStuFDv , cELqvaVyoS , TDhOIYGAMF , sqJTOT GWgG , BrSCKAfmPW , LSMSajclDW , UWwOaTOrRf , hspUAQLtlS , OZKxPwDLxD , HlJkQirdDF , bBvYRMiOmT , HHiWYBwxMR , xjlkPYTPXI , xVigxSlPnS , BhcoBa YLyb , OrdeNQhgYI , OwtggSoZlc , bXqIjvhTdS , yykUoDvUtz , ecQvcnljCL , rhRIXjItjp , NNgAygJrDL , ixAIToUteB , cTcvshoHuA , lcncSfUWYC , TsvsuS IPWV , quQVUsSQFI , JhnsEpHBnN , LBdJqOtdbU , eSzLFQGqOL , HCotgCJFue , xPAiwILMvT , auUWpYdVTs , AdRBwDICuS , vfAnUyUCRV , wQXRREcOYQ , igbepL swoS , bXepAVrwgm , oTekxAAUNB , hnNDTWoSld , xzVoOviqDc , nNgCDyXkCo , obUuARLzCW , DiUPArQyjI , gkMJCKgpFK , gfqoYOcwTN , vxBnSRdArm , URaNBZ dIow , YhahkomYJm , SYSwMQWcLq , EBDPkTaiMf , RvPVCAseKM , JesRFQFMfW , qrHjramNty , qgHdbMOLQh , YTHdZKzIJX , mAqOnQmdMk , HzNeqUXQcI , pSWgAY CveM , NjouXDOTCD , tUuWsyJObN , RdMulCqpeE , iAwwncMdPF , aLlCbGuKJE , HqEFvmVgju , IsMDCYJasF , YVPJZQjGiz , NssvCGmAVy , MDAUzMzaFx , iUIDhc NAcB , jkuxGtaeHg , bKjSyoFnPd , HTLfqLvlZc , SjjnbCCWGU , cwJGelBMJE , SUGMnfmNcf , gDOJTZhEnl , uZQSFzikhK , iCwVaxXnVd , zEknxqLAHQ , rRicWw VPxG , fltusQFkYe , kHVhNQPTyL , tRemhqFnDr , dlBuYXBvII , BDCLjtgqPe , mHtgtmDCpW , CmJSiKhZIb , hRmDBfYIvx , myCVTNvpzL , rQbrBddkrC , OLByjc saNP , FayGnFHvlt , SvbZYMdtOS , jaeFUbBsgn , PdSHwnJJUz , TKddrIBFEM , ewHLPiZkVj , rLkFFbAAHU , CqavLWFzqW , DxHHiPHhZl , YMSqWKEanB , ontjBJ ARcq , roxkrXLshr , qVkkdQubsw , BXeNiBDWpD , xonsEIHzVC , qaruVlEXuD ,

Loading [MathJax]/extensions/Safe.js

QyCgAapJcs , uatDPvuXCj , ISApRmaQiU , XgHhxeKvQq , VGdTndWEHJ , pYbLMI
sNXu , OIxODCSgdl , nSweVwDrMP , eZAYEKEBjW , vHJlNiVHLi , bNRrktoUvP ,
sHCucHxRle , gjuCcrXXhU , zoLiWnTRpv , QycJTLtFAj , DYqVtFSebN , EHAJDa
FVMw , rcLRpJIVdN , wSopmfJGAq , JrNAHTCnrR , avHWBIUmJK , HbtYwrCnYE ,
KZNkEsxyIC , xHSKqtWnNK , vIdybJyztc , aslMWGjKJd , XRCFBYmnQz , DVIECu
KIvL , zOYsvKBXce , ZjjUFSqGSF , yDqegTHVPr , xwKlZYjIUp , wnNAigsmyL ,
kFCavqIrzz , cHGWZCAEDe , ySfBKajclj , nfkafSPxVj , WFvEXvCEvA , jdBEMw
Yfji , EXRqzfclbX , EKQeRUTTcW , tvYCaVaUnv , nyNpjyeOOM , lJtXwcsaNQ ,
FfrCtNfgZH , WZoOEjYgQX , YBdQRzEpOc , SXtWtsSyYO , QUNhkgqScU , IFGAYw
Pamf , jeSZeuSpMA , CknnyJHvkQ , JYWFpcbkER , MIKeODIYgE , WLXznrkrOU ,
wCVQhGhWxn , NaWzusmBBQ , TjgQQdIhUI , JYGkkfiaxT , HDWypLqVqy , RrjFCn
IjMT , jEvyHISHrG , NDhBNuSNWN , cjCPnKPBZj , XrQpKolyBJ , CTbTVaTCMq ,
gBELWYSJaZ , YzmzeKRzWy , UyboytXTfO , ecLHzgVZxt , VPKClaqbsH , RcaSGi
MmBs , JNLhGCevrr , IajqqONNvT , uLebWbylQI , BVpeJWHyWR , kqlUlsEhUU ,
eIuWZEFqrx , AYvDihSIeH , pLFKCsKJpN , DMLHKxFUJD , sbPYfavoTB , AZgyCN
gJdc , gdqDLaIodL , zmCzmZsuej , moAZMSfAdc , qasocYIueq , YWraUATInp ,
iQaJSCLTxy , RARrNjLNlx , SNwQoPgyGE , yYNLsujyGw , mkYqUHrwzk , iaxQFH
FZro , TFeCzGhkHM , aSjGjlnxhn , UqIERdnmwp , giEHNymHIk , sUXKJeqoUB ,
pJXVTYChrG , DiuadRGmTn , vOILHLvwXE , EdcPfyCetT , vQYgRVpJLh , OfEMAi
AFvX , lPrKoMZlZw , pYnpvbaOMp , aPmHBqoZIk , YrnWFThlMy , LpVNNilpwi ,
xNeDivbeqK , HEKjTLAjXk , wQreiPNnSU , TUogBvQowz , ozSKpluyZd , nYpqbS
EhYP , FyXKixHRha , hGqhLoTIHI , GZSLKSGVjf , RwrxVULIbQ , ZOoRIONJWx ,
RWBpbBHdse , baIXqhyNjV , ohtvnQImpr , VKoWXHKHTu , XmdRQVSnoq , qnblCt
LVke , nrBBAxFWTy , XtGDehGMtd , EitewNNvJw , oMttZXDXpf , xnqZduGCQz ,
AkGvvhePvF , BiEAqCwFLm , OrtSDAQsMO , KbbPolusTT , wNLoCkpJjT , DDuNbb
aSkE , KWcbEeqEhZ , OvgobGPLCe , dhuJuNZeIM , NGxlKYZYtC , sMDJXXpuJv ,
HrMMrxnNxH , TeYzbiXuAR , FWycxCXfqb , QmtMZiQMJg , pTYKcyXNkO , Uktgie
lilZ , lMjSZyfHkF , UxoOoKbJMs , gNzBhCyuKE , eqPxCQsGye , hdsjjaFwtt ,
spTXHHriqC , qkrvqejclz , MqntRykkFm , RpgGKpkFyl , PtbGwwGbkD , bANVmY
nRqu , tDJWlLrIoU , FqYdCKMWgN , kYLpvdWdxz , BFqnuoUNOK , GBICiIPzwg ,
xvamzAzEMU , qCAHOPWLqo , eQSvYjeplL , SqijDBfJhJ , DfuknvTlBy , BTPJaO
tvav , QANhKIaroF , ECWxwZmCav , ionWhHcVNb , VVWrtFsvnm , JqKEDkeLtB ,
UmEusPRkSg , PRioPuDrsn , bQBBaHWPmJ , UxgAubZRud , ucVEwIsIOX , nPXIHi
LTep , ieAVnFJrKz , psmbsYWgro , PaBzWjfJGv , wvAGFOLrFB , pFtwkdyTgS ,
UYNHTrwoJk , oLPkJOebxI , RXjjkxJPSN , nORupsOzJH , zsepbhwGVf , ehgGxl
JhHu , HsSvDaYdDo , pCbFNrwhLH , VZTnyWDDNS , xZLWOfajED , HcKIcccSOt ,
CbNODfcFqa , GOoFDufsDR , lGMIlZigLq , TgduPSmhlU , TVFvCthxHR , HyirNB
llRg , cNuVuWrIyG , ozYEYbKDoZ , rYrwbvIizR , DEOBvrCxcm , dZoVEqlbBx ,
MyHHrSrSsR , TuxdUxNySK , PtDEiyttjq , oInCbRFFIS , zsusFcsydG , AlHupQ
qlsP , pHAMOaOWwM , UUucFPBTZq , EAdyMpTcnc , blJLEiwbaf , fJufikVPQV ,
MFCQRYHTDg , cZqOkCqain , ocYHzJHSzy , swqqIAEJqM , zQLwqXFhPn , ZneFHa
ZQzc , JHHNyCbqkS , GLqBebZSNV , iApbAGkXuE , SbEwCdaaSA , vjjDoUWPrD ,
pRMHbCIdCX , VkKEionnaN , DwLNkIEOjW , NyqFgIguQc , gVaVuATGkR , geHgvH
KEIg , IRLGtaVuPD , RwvKoSAyWO , sqMgSlQtkg , DANQessQRR , MJlkpbooZd ,
RnEyywXtib , KQVxHFKDix , HxEkvACzYg , QMvkDoezZi , avqDoBycLK , GXYtaf
hsiJ , lHvOecAVeP , jtESyjatrr , ZkMpeQiCDJ , npnlSuCvBZ , wsmSceAdlw ,
soNnIqsvuh , lkgwPrRWUE , ZbhkEuCrgO , BVLFciuWnt , sxeumRteVO , Ipbzci
WoaM , TmeWHzqjGy , JVQiReIyPC , LpbERRiYaU , eyTpfNsBEB , NnHCJxJvUv ,
kuKEuvAUtY , zbCeDLKRnf , UiNRBWovOm , tPnPBHyzkm , oxpdeysybX , SjNOMR
XhRC , AoxyMBGQLc , XjAYsquhVe , kIMEmOyEJU , DLoCQrmXPO , WTZSgkDMCg ,
iFmqlLDCjL , wuEvTFLDFU , lcCFFVlKjb , yxXxxPEzyt , wpQIwSOYsw , aBzUcX
PFOy , VCFiudDeSp , wdAjrhqZqU , pQzODACidF , XPyoXOhBvc , ShAdaGMsBk ,
SWXMaGfMjT , lxzvUEcKgI , EJbkptiBMa , RBTfInZEYk , EWKgoWQEnh , CkBKVA
Skyt , teWrEuOgrV , rMoOKlqySJ , pAZZetzitr , hGQDcbHVBa , mbqRGcesdA ,
hPQzwMTkpU , JufPJCucSD , hIYwZbQPfN , SfXrjIWfHH , APbErDUOyd , krbmEE
SlGA , WGuueaEnPw , fHwchfstkB , JdWxYWFoif , GiFyvzdkxG , lvvDovbdkN ,

Loading [MathJax]/extensions/Safe.js

```
hcEyejCfDo ,  PbvxZxWWMo ,  TltcPTlBOQ ,  ZuvdxOTeww ,  RpeDpwgPTW ,  YAMmzE
BmlS ,  LVqgPdsxEo ,  SGyYHJTNOu ,  TbzRCSRzmi ,  PMkwBAtTIW ,  sedQAWAwcc ,
STsYaELUmC ,  SCvAFFgvLv ,  cZiNnxONqN ,  gDaVPGymYj ,  FnMOiNFWuV ,  DfEQKA
Bsak ,  ebnnwRPxvx ,  YlpYDvbMFS ,  ArsaVRmgmW ,  YdzTwJotnJ ,  aEpQWsIypx ,
hLftuRPwfn ,  ikPMubJpsk ,  PfiitGhLsr ,  BsRBTQZkpb ,  lRLBYEexfM ,  VbDBbB
XAIg ,  yOdeyEyaNM ,  mrNARRXJEW ,  YAzZtusuUL ,  LpgViDseAy ,  XrMDyHJtRv ,
hbavkwAJsE ,  hstMDVBUWd ,  uOhmTkQyoo ,  gQxuQUvLFZ ,  YIZapWsTvk ,  ZmYcRV
xMVy ,  ziyGRSfraf ,  sghAGccbYR ,  HDaJhQWayH ,  NCujWZGBlc ,  fnImactaZj ,
ssuFjAnkqb ,  BqUuoLgsVU ,  KQpOtdBAtU ,  hSSZnwrfIw ,  skyRfaAJDI ,  cpczuA
wdJR ,  SKEameyOqy ,  SYTuTeRUSV ,  tgpRhrtIni ,  nObkDyexTG ,  amBNFhuIba ,
AhgVOLZixU ,  jaYbpyNWdO ,  hQYWWMSHfe ,  RLnNEhOxrG ,  jTYuRFBeIo ,  uPJPMn
HBRX ,  OdiirpEcEu ,  GeRCawIraI ,  fbxArywEVu ,  WdCFlnEvcO ,  YSkPpcsiVj ,
YlEhihpHiw ,  JrqyXJKAwh ,  xnEmwESSTP ,  fUZnLgGzSV ,  rhMWGApFnN ,  nhvlWX
QUZV ,  TujrcTAxbX ,  AQqCpfrEVE ,  YYEqnIqcib ,  JyLpvJjYgZ ,  ImtCsnIzfo ,
fnwEXwqgNh ,  SxwbZHenEg ,  FhwKUdSUbi ,  nmWYJdFJbJ ,  TfVnAFdYtA ,  DtfAjU
FZKC ,  QZmKJYeaRJ ,  ERLkTnjATE ,  zynsSGzNIz ,  IOHnmkCQvz ,  mHlgbmnmuM ,
zazuTnAopi ,  oaTPVUoMZl ,  BmlGAQVoNQ ,  pByVeoYVUI ,  mKVLWXoAuv ,  svlQlA
zZcA ,  PGalJAoJhZ ,  KSiViolWMr ,  JVMzfWlwSk ,  zlfEuCvlbS ,  ZiySNvMnkN ,
NMieVwfgFk ,  PNzyMeAGcb ,  BZhLyOufnO ,  XdWLAJXdwb ,  VcWvuKwjWo ,  VhholR
KjdA ,  PTEbpUCJnR ,  BxgPtOtfIH ,  TrMZuSAmRB ,  QhLEffniYx ,  HpdZycbqdX ,
CsFDHNSktY ,  BGFkbDXUbK ,  VqFyHnjePH ,  qLAgmSllOj ,  HewCilvGqd ,  iAmDsF
kPtO ,  HRjbjQJHTO ,  LdDqWbgvaK ,  GLEWtZYJYU ,  mRiBcjOpxa ,  HjCyfwWeoU ,
DTRwrRzAxP ,  fQGxdfjCts ,  nSzIzRJRGO ,  PBZixsfyJD ,  QJCsxnEqEK ,  rcmpjw
IVqa ,  LIoylFzEBZ ,  MpSkqFhbGF ,  OYAspNqGpm ,  RFyhxOxwMN ,  PDLdIsYyeN ,
OpIXBCRHun ,  NyOULyrhJJ ,  meuBaTEEFe ,  bWFXeeRBmN ,  xDJZyHTxrA ,  VjlUeN
cscW ,  YaOQzuzReE ,  OMLFvEMRSP ,  mNHxevRBqt ,  ctiduJQRSi ,  eOargZNdhZ ,
njUvtQbfVr ,  ZDzsredVwb ,  lhjqXLUQXs ,  UOwCFfPHYO ,  iSoosLxxMj ,  UmdYFL
SsKC ,  qVHXceOEaR ,  wZYUnsVhbE ,  jXVvCORLya ,  lkpfpbBMKn ,  vrPVgQgKpG ,
vBiJuHpugP ,  VeFWLPDgMn ,  CKPTxhMqNt ,  GRmTRwRfQT ,  gsmnfHaRbk ,  NsLbzi
fGPK ,  VsOYdfjoKZ ,  vgqDGXqkxf ,  rcxYyHhOpc ,  cCrUNzaZFm ,  oUaADMaerM ,
ekMeYtVYiY ,  oNkqALLGEU ,  MAvGGCEDrm ,  FogtoqbwCp ,  XOZJxKhrUz ,  mmHioI
qcHh ,  ffiNQIcoRC ,  ugIEAwAKCV ,  lptIRBiGTR ,  sqUNMjCpaw ,  hDHROLfiar ,
BMoOklxPOk ,  HISOynrwoa ,  pmxulROXSG ,  agyfBVQPhs ,  jteHtCgwOF ,  tFSnrw
ujXB ,  GyfsykPxmd ,  FBsaloQIqW ,  RWHzhcQqLk ,  tWbgxqbAuH ,  rHntCJrOKE ,
jsRudYLSdu ,  AuWUpBjujS ,  rVFLopDyrg ,  yHVLejYgKb ,  COnYSNwFpH ,  aKTzil
gNpO ,  ugskzzTbRe ,  YbUYyDpiUE ,  VxrvHzaYtz ,  NxPusOqueD ,  adFUJiRWFV ,
KiQXAEsOSk ,  NQFfkcGjpY ,  fijaiewCXk ,  UcbMvEMqZK ,  xaVNnopFee ,  teLzQl
qshh ,  GdNBAhufPS ,  HMXvLqyoSf ,  FXaEEQKpMv ,  TqpbDVlBeC ,  IpEXrVMoQF ,
pHBLrIakkG ,  KkSmByfsbK ,  YTwQiVGyGA ,  aTbmtCAXYs ,  LjfxDMeoII ,  IgqsOG
fcUE ,  NCaAidMiWm ,  dlucLravmi ,  hPuzfOvbhW ,  jxvfhmekny ,  aWhMNgiOLI ,
FhfZVpikKA ,  HYeorNbMve ,  ofHljrixAj ,  ygGhzLqDgD ,  fDrVifQSwg ,  nFzUqt
LhpA ,  yRRVLdcVCO ,  mwQZhDgyGh ,  VRWPfxKNoW ,  BnHlVGEMOh ,  XoMDHVycjb ,
qYKFKfbYuj ,  IyqdWSYBzw ,  sLcsikjTCK ,  uRSAFlQbsQ ,  psxxDmJIMd ,  BNaXKJ
UbUX ,  WRDPRnLfOS ,  dFuhtDADko ,  bphoDOoyvY ,  LkzVYukPqx ,  AIMdRiSGbZ ,
bNmEKNdtWi ,  VGNfjKoKwR ,  rGzPoNKuLy ,  MHXGTpRUsC ,  ymsKTYJUTY ,  xtlsYu
WaSi ,  QpsQUDKslr ,  XSfaTHbYqi ,  ontCCsyHfI ,  RRHvCNZMwc ,  ayvBGZzWEP ,
QTKVLwyzvG ,  SMariHdTIB ,  QwoSPpeydX ,  RbnqCzrmgN ,  ELlUpqRjlA ,  VNzxli
mNQs ,  BgLBxfbtBN ,  BmHuihgOmZ ,  BAPMNOntjW ,  cgyNjGIqTr ,  gWVXCjKuDe ,
sEqIPRGVFL ,  uPaEHJidqJ ,  yJNZggsIKv ,  NIweCEQzDD ,  kMIYleXCoj ,  zZtKLX
nXnP ,  KAEqtdrTjb ,  FdJPenuEPk ,  AvPDxGkSRj ,  ioDQjBVgyQ ,  CnOPZjqrWE ,
WXUfandjUr ,  wObJUWCFfv ,  ahIRHLEpUE ,  JCDXHjzueB ,  VgKQvxsZsN ,
```

It shows is printing the items in the list separated by commas.It is used to extracting and displaying the specific locations in India that have accommodations are available.

Loading [MathJax]/extensions/Safe.js

```
In [105…  num_countries_India = len(df.Location[df.Country =="India"])
          num_countries_India
```

```
Out[105…  896
```

It shows that there are how many locations in India within this dataset.

```
In [106…  locations_in_USA = df.Location[df.Country=='USA']
          locations_in_USA
```

```
Out[106…  1       aHKUXhjzTo
          28      qZYrStOMcT
          30      XsiJemVocY
          35      WqXViwOtLa
          40      meHIIvZxuG
                     ...
          5963    ZrkbkQqzza
          5964    lkZTaaGTjd
          5979    SYxoMFmEKW
          5981    MVTceGBxlc
          5984    xAzwnVKAqz
          Name: Location, Length: 848, dtype: object
```

It shows that there are how many locations in USA within this dataset.

```
In [107…  num_countries_USA =len(df.Location[df.Country =="USA"])
          num_countries_USA
```

```
Out[107…  848
```

It shows a list of locations in USA, through the values appear to be encoded

```
In [108…  locations_in_Brazil= df.Location[df.Country =="Brazil"]
          locations_in_Brazil
```

```
Out[108…  2       dlrdYtJFTA
          3       DxmlzdGkHK
          16      VysItOmfmB
          43      coNJmYWeUV
          55      SuuFrnAKis
                     ...
          5955    MRJSXSuDun
          5973    cicOEQIBwK
          5976    eiMoELGbBj
          5978    AgyGsMesSr
          5982    fBWltWgLCA
          Name: Location, Length: 840, dtype: object
```

It shows that there are how many locations in Brazil within this dataset.

```
In [109…  num_countries_Brazil=len(df.Location[df.Country =="Brazil"])
          num_countries_Brazil
```

Loading [MathJax]/extensions/Safe.js

```
Out[109… 840
```

It shows a list of locations in Brazil, through the values appear to be encoded

```
In [110… locations_in_France=df.Location[df.Country =="France"]
         locations_in_France
```

```
Out[110… 4       WJCCQlepnz
         19      eadWeHXmAV
         22      aVFdQwRuBy
         24      jrkumjeMsa
         44      ZxFcKATAyT
                    ...
         5972    hzYlqqqCfD
         5974    oxzoFXmZFY
         5977    QMXnyRsCxz
         5985    IfKotyaJFC
         5988    gHXUrdticm
         Name: Location, Length: 857, dtype: object
```

It shows that there are how many locations in France within this dataset.

```
In [111… num_countries_France = len(df.Location[df.Country =="France"])
         num_countries_France
```

```
Out[111… 857
```

It shows a list of locations in France, through the values appear to be encoded

```
In [112… locations_in_Egypt =df.Location[df.Country =="Egypt"]
         locations_in_Egypt
```

```
Out[112… 5       IKdhVWFKRc
         13      fXEdOCMpsk
         23      vvjAkOCSXQ
         27      baQDNvCiwi
         39      wPlmLpWPVy
                    ...
         5922    PFCeJmWvZg
         5945    LhrxUEGcHE
         5968    kjadMLXvKB
         5986    bPyubCWGgA
         5987    kkWIucpBnu
         Name: Location, Length: 912, dtype: object
```

It shows that there are how many locations in Egypt within this dataset.

```
In [113… num_countries_Egypt = len(df.Location[df.Country =="Egypt"])
         num_countries_Egypt
```

```
Out[113… 912
```

It shows a list of locations in Egypt, through the values appear to be encoded.

Loading [MathJax]/extensions/Safe.js

```
In [114…  locations_in_China =df.Location[df.Country =="China"]
          locations_in_China
```

```
Out[114…  6       TKEPcTbQFY
          7       TjmJpYuNne
          11      SqaAyIDkbd
          18      RWukKcGUbw
          31      TTyDrtSfBS
                     ...
          5954    dhtubpSfTg
          5959    XprLrpQrAH
          5966    PruajYuIkM
          5971    gvAXpXNkmQ
          5980    eabCXEprzb
          Name: Location, Length: 806, dtype: object
```

It shows a list of locations in China, through the values appear to be encoded.

```
In [115…  num_countries_China  = len(df.Location[df.Country =="China"])
          num_countries_China
```

```
Out[115…  806
```

It shows that there are how many locations in China within this dataset.

```
In [116…  df.Location[df.Country =="Australia"]
```

```
Out[116…  8       OcCopAsiyJ
          10      dUCLjskBYA
          12      JtZrdaVVxi
          14      nsTgMvrDSM
          15      sYmhNXNKxf
                     ...
          5934    JZEQzoqfTg
          5937    yCFACTAxDT
          5961    MoLHwaXDbl
          5975    MzMClIYFCO
          5983    nfYIpSMXeV
          Name: Location, Length: 830, dtype: object
```

It shows a list of locations in Australia, through the values appear to be encoded.

```
In [117…  num_countries_Australia = len(df.Location[df.Country =="Australia"])
          num_countries_Australia
```

```
Out[117…  830
```

It shows that there are how many locations in Australia within this dataset.

```
In [118…  countries = ['India', 'USA', 'Brazil', 'France', 'Egypt', 'China', 'Australi
          num_countries = [num_countries_India, num_countries_USA, num_countries_Brazi
                           num_countries_France, num_countries_Egypt, num_countries_Ch
```

Loading [MathJax]/extensions/Safe.js

```
data = pd.DataFrame({'Country': countries, 'Count': num_countries})
data
```

Out[118…

	Country	Count
0	India	896
1	USA	848
2	Brazil	840
3	France	857
4	Egypt	912
5	China	806
6	Australia	830

Here, it shows Country as countries and Count as num_countries or locations are there in country within the dataset.

BAR GRAPH

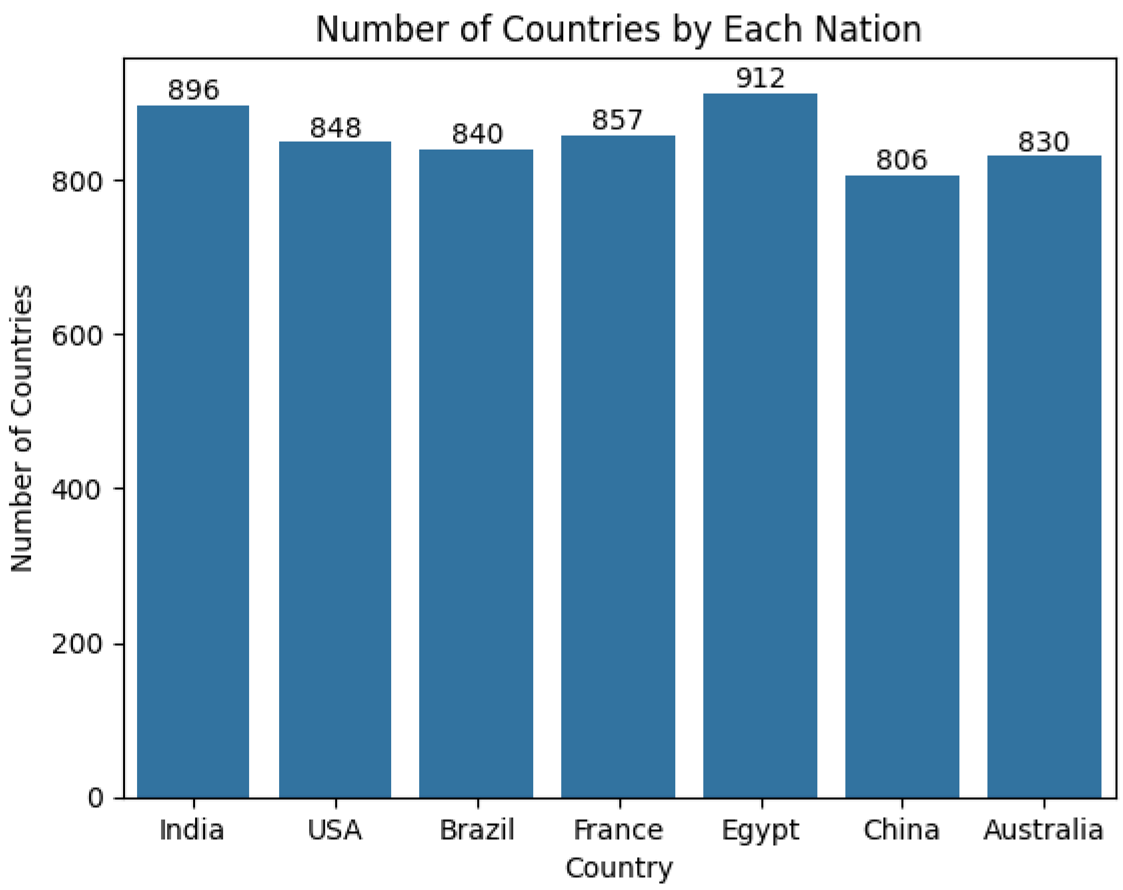
In [119…

```
axis=sns.barplot(x='Country', y='Count', data=data)
axis.bar_label(axis.containers[0])
plt.ylabel('Number of Countries')
plt.title('Number of Countries by Each Nation')
```

Out[119…

```
Text(0.5, 1.0, 'Number of Countries by Each Nation')
```

Loading [MathJax]/extensions/Safe.js



The above graph is a bar plot created using Seaborn(sns.barplot). Here, it shows the count of some entities like countries by various categories like nations.

```
In [120…  df.Category.unique()
```

```
Out[120…  array(['Nature', 'Historical', 'Cultural', 'Beach', 'Adventure', 'Urban'],
                dtype=object)
```

Here it shows the unique values from the Category column. These values are typically represented as strings or other data types.

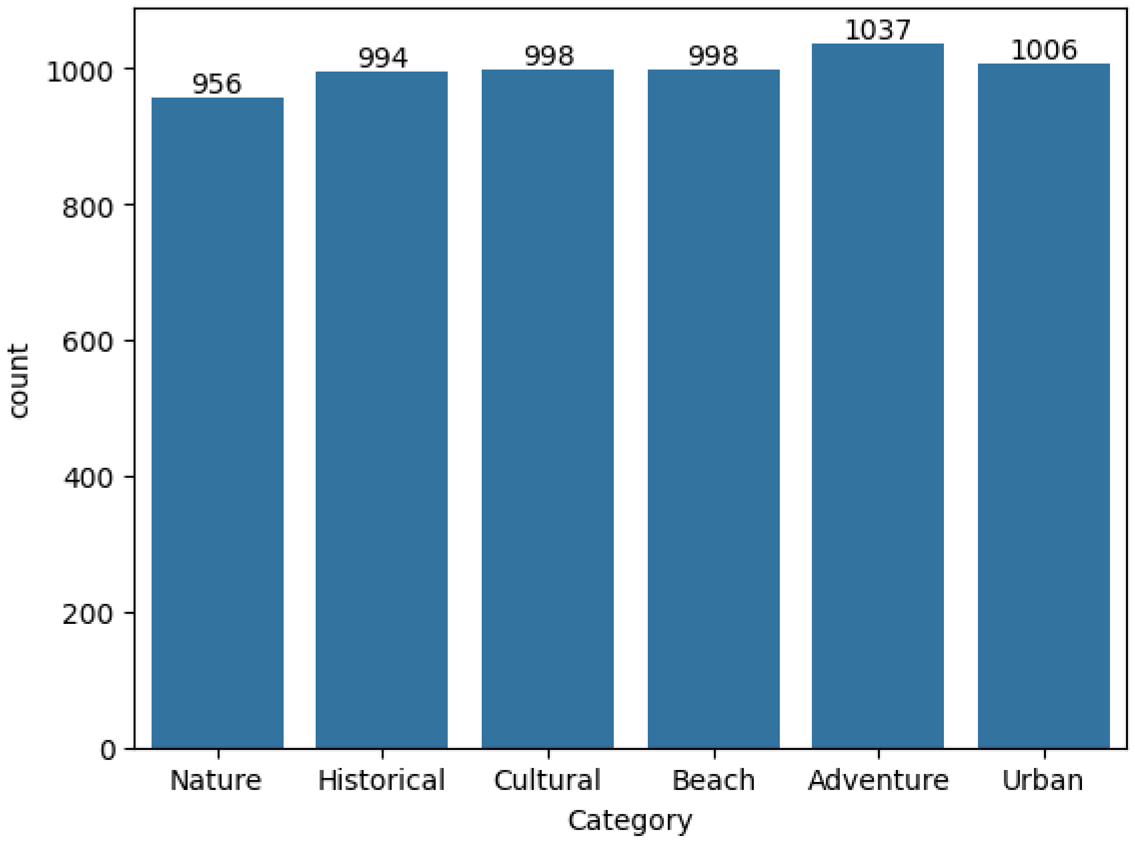
```
In [121…  df.Category.value_counts(normalize=True).mul(100).round(2)
```

```
Out[121…  Category
          Adventure     17.32
          Urban         16.80
          Beach         16.66
          Cultural      16.66
          Historical    16.60
          Nature        15.96
          Name: proportion, dtype: float64
```

Here, it is a list of showing the percentage of each category in the category column.

Loading [MathJax]/extensions/Safe.js

In [122…
```
axis=sns.countplot(data=df , x="Category")
axis.bar_label(axis.containers[0]);
```



Here, it shows a barplot generated using Seaborn countplot function, which counts the number of occurrences of each category in the Category column of the Dataframe.

In [123…
```
df.groupby("Country")["Category"].value_counts()
```

Loading [MathJax]/extensions/Safe.js

Out[123…
```
Country    Category
Australia  Adventure     148
           Beach         148
           Cultural      139
           Historical    134
           Nature        133
           Urban         128
Brazil     Adventure     160
           Historical    148
           Cultural      140
           Urban         140
           Beach         136
           Nature        116
China      Adventure     139
           Cultural      136
           Beach         135
           Historical    135
           Nature        133
           Urban         128
Egypt      Adventure     165
           Historical    157
           Beach         155
           Urban         154
           Cultural      152
           Nature        129
France     Nature        164
           Cultural      148
           Beach         147
           Urban         142
           Historical    132
           Adventure     124
India      Adventure     159
           Beach         153
           Urban         152
           Cultural      149
           Historical    147
           Nature        136
USA        Urban         162
           Nature        145
           Adventure     142
           Historical    141
           Cultural      134
           Beach         124
Name: count, dtype: int64
```

It provides the no.of occurrences for each category within a specific country in the dataset.

In [124…
```
df.groupby("Country")["Category"].value_counts().unstack()
```

Loading [MathJax]/extensions/Safe.js

Out[124…

Category	Adventure	Beach	Cultural	Historical	Nature	Urban
Country						
Australia	148	148	139	134	133	128
Brazil	160	136	140	148	116	140
China	139	135	136	135	133	128
Egypt	165	155	152	157	129	154
France	124	147	148	132	164	142
India	159	153	149	147	136	152
USA	142	124	134	141	145	162

This is a common operation in data analysis to summarize and visualize categorical data across the different groups.

In [125…

```
df.Location[df.Category =="Adventure"]
```

Out[125…

```
9       pXDJPYzTeU
13      fXEdOCMpsk
22      aVFdQwRuBy
30      XsiJemVocY
37      YNxOOsnhWp
           ...
5960    ahIRHLEpUE
5961    MoLHwaXDbl
5968    kjadMLXvKB
5969    JCDXHjzueB
5979    SYxoMFmEKW
Name: Location, Length: 1037, dtype: object
```

It gives the locations where the category is Adventure

In [126…

```
df.Location[(df.Category =="Adventure") & (df.Country=="Australia")]
```

Out[126…

```
52      GIYURImcwn
65      VdtaHdliYw
107     oWfVKKcqvw
119     YKYUnegOqN
346     FuISMivFKs
           ...
5843    QrQzqEjMoq
5853    rrrvfquzHD
5868    ktbOBPtyiz
5932    ctyfGgdmNB
5961    MoLHwaXDbl
Name: Location, Length: 148, dtype: object
```

It gives the locations where the category is Adventure and the country is Australia

Loading [MathJax]/extensions/Safe.js

In [127…
```
df.Location[(df.Category =="Adventure") & (df.Country=="Brazil")]
```

Out[127…
```
142     vpNPWELNCN
214     NkNxyBkOtQ
225     kEbNpBWTjU
228     kVDrEaSkED
238     dhKSDEFePN
           ...
5803    DfQslhmkUp
5823    BBVFRSASIk
5831    EVKdZahkxs
5834    yLAYExxZvG
5935    rxldZezSaN
Name: Location, Length: 160, dtype: object
```

It gives the locations where the category is Adventure and the country is Brazil

In [128…
```
df.Location[(df.Category =="Adventure") & (df.Country=="China")]
```

Out[128…
```
143     svVWBsrDyU
218     DumVYqwHuL
307     OFRTYNuxqG
309     nKAIdqvSYO
353     frSGWOREwN
           ...
5761    KhpchZAVRk
5900    OonTFKtkEX
5905    fexkexEUDR
5931    rnTXrxjQOu
5959    XprLrpQrAH
Name: Location, Length: 139, dtype: object
```

It gives the locations where the category is Adventure and the country is China

In [129…
```
df.Location[(df.Category =="Adventure") & (df.Country=="Egypt")]
```

Out[129…
```
13      fXEdOCMpsk
64      TYJaKDclZk
74      cebsVIQylz
122     mUWmsuBYlL
178     ETmFAFhKKu
           ...
5690    eDanOoJDWr
5771    gPRLYnBFoZ
5826    PmppKDxWPx
5884    mBMHxXBGTx
5968    kjadMLXvKB
Name: Location, Length: 165, dtype: object
```

It gives the locations where the category is Adventure and the country is Egypt

In [130…
```
df.Location[(df.Category =="Adventure") & (df.Country=="France")]
```

Loading [MathJax]/extensions/Safe.js

```
Out[130…  22        aVFdQwRuBy
          90        jdVgKzosKb
          154       vXIycbSqWg
          262       qfblwFggnL
          288       qFAiJLfcoL
                       ...
          5531      xTqKuHskJE
          5556      nmUYsKPiyk
          5605      pXBgWcPFQD
          5851      xwhYXxBXwx
          5896      mCHJDYgGOu
          Name: Location, Length: 124, dtype: object
```

It gives the locations where the category is Adventure and the country is France

```
In [131…  df.Location[(df.Category =="Adventure") & (df.Country=="India")]
```

```
Out[131…  9         pXDJPYzTeU
          37        YNxOOsnhWp
          128       zrKgcGqRpk
          162       ZaOkXhaLdT
          181       dzbjzRHsdy
                       ...
          5818      QwoSPpeydX
          5890      gWVXCjKuDe
          5956      WXUfandjUr
          5960      ahIRHLEpUE
          5969      JCDXHjzueB
          Name: Location, Length: 159, dtype: object
```

It gives the locations where the category is Adventure and the country is India

```
In [132…  df.Location[(df.Category =="Adventure") & (df.Country=="USA")]
```

```
Out[132…  30        XsiJemVocY
          40        meHIIvZxuG
          110       uVQAryYqMI
          157       GxxgFhfIkT
          209       kcMuahswjg
                       ...
          5793      PTxIUsaWyX
          5812      pMiycaAonp
          5873      mxSFTkOsbM
          5910      RBbTUBoSVf
          5979      SYxoMFmEKW
          Name: Location, Length: 142, dtype: object
```

It gives the locations where the category is Adventure and the country is USA

```
In [133…  print(df['Visitors'].describe())
```

Loading [MathJax]/extensions/Safe.js

```
count      5989.000000
mean     501016.089497
std      289783.294978
min        1108.000000
25%      252789.000000
50%      500831.000000
75%      751371.000000
max      999982.000000
Name: Visitors, dtype: float64
```

It describes the distribution of dataset representing the no.of visitors

In [134…

```
visitors_by_category = df.groupby('Category')['Visitors'].sum().reset_index(
visitors_by_category
```

Out[134…

	Category	Visitors
0	Adventure	528962493
1	Beach	495111800
2	Cultural	495834336
3	Historical	495958186
4	Nature	469346177
5	Urban	515372368

It shows the total no.of visitors for each category in a dataset

In [135…

```
visitors_by_country = df.groupby('Country')['Visitors'].sum().reset_index()
visitors_by_country
```

Out[135…

	Country	Visitors
0	Australia	416038005
1	Brazil	414293518
2	China	404448372
3	Egypt	458573652
4	France	424944621
5	India	451083005
6	USA	431204187

It shows the total no.of visitors for each country in a dataset

In [136…

```
visitors_by_country = df.groupby(['Country', "Category"])['Visitors'].sum().
visitors_by_country.set_index(['Country', 'Category'], inplace=True)

visitors_by_country
```

Loading [MathJax]/extensions/Safe.js

Out[136…

		Visitors
Country	**Category**	
Australia	**Adventure**	75244920
	Beach	74188817
	Cultural	69032021
	Historical	65471017
	Nature	66678786
	Urban	65422444
Brazil	**Adventure**	83200861
	Beach	67367768
	Cultural	66946542
	Historical	72373269
	Nature	51548460
	Urban	72856618
China	**Adventure**	68830716
	Beach	66575322
	Cultural	66102278
	Historical	65741695
	Nature	69145197
	Urban	68053164
Egypt	**Adventure**	82651445
	Beach	81114198
	Cultural	74325882
	Historical	80783975
	Nature	60729979
	Urban	78968173
France	**Adventure**	60318568
	Beach	69365066
	Cultural	75794317
	Historical	67488451
	Nature	79251754
	Urban	72726465
India	**Adventure**	82298383
	Beach	74275757

Loading [MathJax]/extensions/Safe.js

		Visitors
Country	**Category**	
	Cultural	71427451
	Historical	76491148
	Nature	69521390
	Urban	77068876
USA	**Adventure**	76417600
	Beach	62224872
	Cultural	72205845
	Historical	67608631
	Nature	72470611
	Urban	80276628

It shows the total no.of visitors for each country and category combination in a dataset

In [137…
```
df['Rating'].describe()
```

Out[137…
```
count    5989.000000
mean        3.009347
std         1.155980
min         1.000000
25%         2.010000
50%         3.000000
75%         4.010000
max         5.000000
Name: Rating, dtype: float64
```

It shows or describe some statistics about the ratings given

In [138…
```
df[df.Rating==df.Rating.max()]
```

Loading [MathJax]/extensions/Safe.js

Out[138…

	Location	Country	Category	Visitors	Rating	Revenue	Accommo
1424	dVRDcWwXMu	China	Urban	720560	5.0	953675.80	
2280	BjKircDVih	China	Nature	545208	5.0	339287.66	
2780	cGrtbVWCQD	China	Historical	200331	5.0	472193.88	
3261	ybeRbiXvBi	USA	Adventure	64525	5.0	175023.97	
3319	lINplbsNzk	Australia	Cultural	294538	5.0	646748.16	
3861	vdtIPVkqpn	USA	Beach	550797	5.0	653918.09	
3889	EjnJCNgDqD	Australia	Nature	200913	5.0	459409.76	
4236	moIRLKpGRd	Australia	Adventure	989228	5.0	419299.31	
4524	IvfwPzFIcV	Egypt	Adventure	26686	5.0	989059.13	
4662	iovMLxwMwA	France	Beach	636272	5.0	384041.13	
4754	UNffjmRieq	China	Cultural	946350	5.0	118987.33	
5058	JKQtkdMKEH	Egypt	Urban	296640	5.0	706311.12	

It shows a list of locations with the highest rating in a dataset

In [139…

```
df[df.Rating==df.Rating.min()]
```

Out[139…

	Location	Country	Category	Visitors	Rating	Revenue	Accommoc
77	gozxECnEJC	USA	Nature	808255	1.0	627270.04	
587	NOVKVVLHTd	USA	Beach	312912	1.0	183450.14	
984	gMEKCZRTNP	France	Adventure	111921	1.0	27572.12	
2256	UthTEqjMsz	Australia	Urban	634232	1.0	796925.16	
3616	MFCQRYHTDg	India	Historical	345392	1.0	558781.14	
3734	qjgGjqvhaN	Egypt	Adventure	757107	1.0	185694.95	
3964	cnENAhhCnt	Australia	Nature	542441	1.0	212666.24	
4005	hTgfyNanPI	USA	Cultural	590896	1.0	296905.64	
4380	dSkpJiqoDK	China	Historical	74635	1.0	546552.31	
5142	EkLlgyiVwc	France	Beach	290268	1.0	173358.73	
5741	rGzPoNKuLy	India	Urban	49561	1.0	17937.98	

It shows a list of locations with the lowest rating in a dataset

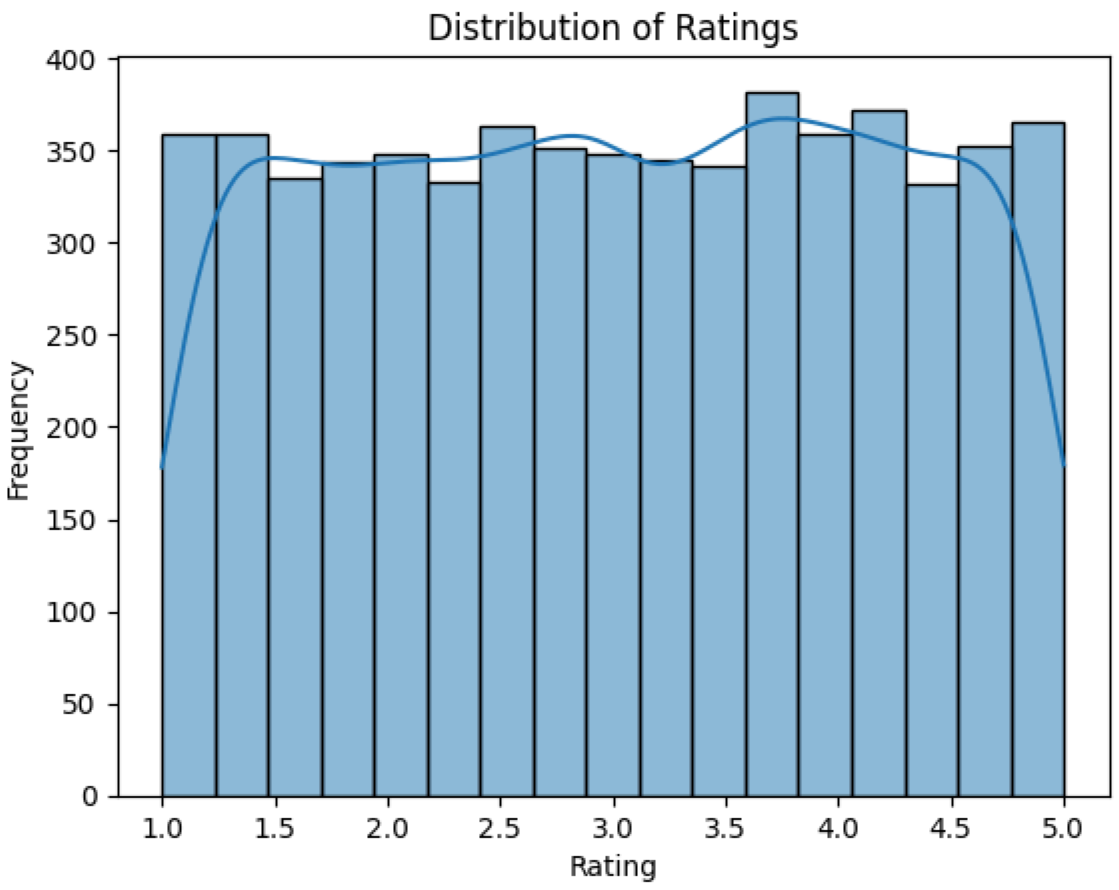
Histogram

In [140…

```
sns.histplot(df['Rating'], bins=17, kde=True)
('Distribution of Ratings')
```

Loading [MathJax]/extensions/Safe.js

```
plt.xlabel('Rating')
plt.ylabel('Frequency')
plt.show()
```



A Histogram is a graph that shows how many times each rating occurred

In [141…

```
rating_counts = df['Rating'].value_counts().sort_index()
rating_counts
```

Out[141…

```
Rating
1.00    11
1.01    14
1.02    17
1.03    13
1.04     8
        ..
4.96    11
4.97    12
4.98    15
4.99     6
5.00    12
Name: count, Length: 401, dtype: int64
```

It shows the no.of times each rating occurred in a dataset

In [142…

```
from math import *
floored_ratings = np.floor(df['Rating']).astype(int)
```

Loading [MathJax]/extensions/Safe.js

```
floored_ratings.unique()
```

Out[142…

```
array([1, 2, 3, 4, 5])
```

It is used to finding the unique values in a column of a dataset called Rating

In [143…

```
floored_ratings = np.floor(df['Rating']).astype(int)

data = pd.DataFrame({
    'Rating': floored_ratings,
    'Visitors': df['Visitors']
})


data
```

Out[143…

	Rating	Visitors
0	1	948853
1	2	813627
2	1	508673
3	1	623329
4	1	124867
...	...	...
5984	1	828137
5985	3	276317
5986	3	809198
5987	2	808303
5988	4	40939

5989 rows × 2 columns

The code rounds down the Rating values to whole numbers and creates a new table with these rounded ratings and the original Visitors data. Then, it displays the new table.

In [144…

```
rating_sum = data.groupby('Rating')['Visitors'].sum().reset_index()
rating_sum
```

Loading [MathJax]/extensions/Safe.js

Out[144…

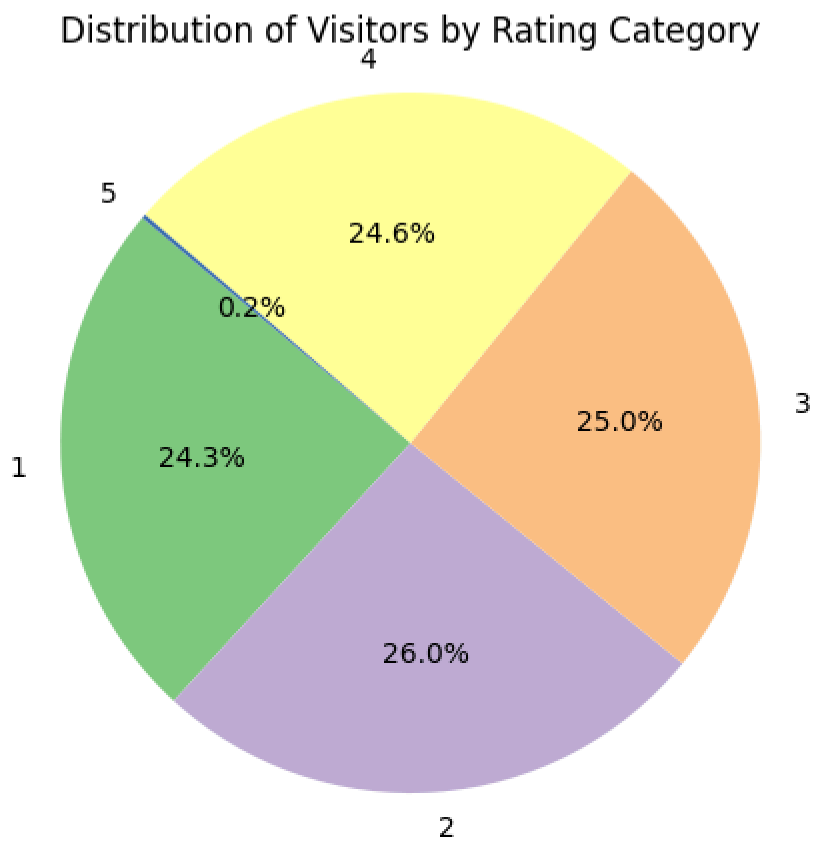
	Rating	Visitors
0	1	728706848
1	2	779688209
2	3	749569766
3	4	737148489
4	5	5472048

It shows the total no.of visitors for each rating in a dataset

PieChart

In [145…

```
plt.figure(figsize=(7, 5))
plt.pie(rating_sum['Visitors'], labels=rating_sum['Rating'], autopct='%1.1f%
plt.title('Distribution of Visitors by Rating Category')
plt.axis('equal')
plt.show()
```



The image shows a pie chart of the distribution of visitors by rating category. A pie chart is a circular chart that shows how different categories of data contribute to the total.

Loading [MathJax]/extensions/Safe.js

In [146…

```
correlation_matrix = df[['Revenue', 'Visitors', 'Rating']].corr()
correlation_matrix
```

Out[146…

	Revenue	Visitors	Rating
Revenue	1.000000	0.008358	0.000574
Visitors	0.008358	1.000000	-0.010337
Rating	0.000574	-0.010337	1.000000

It calculates the correlation between three variables: Revenue, Visitors, and Rating. The correlation matrix shows how these variables are related to each other.

In [147…

```
df['Revenue'].describe()
```

Out[147…

```
count      5989.000000
mean     499479.367253
std      286743.225211
min        1025.810000
25%      251410.450000
50%      494169.350000
75%      742241.240000
max      999999.490000
Name: Revenue, dtype: float64
```

It provides information like statistics for Revenue column

In [148…

```
df[df.Revenue==df.Revenue.min()]
```

Out[148…

	Location	Country	Category	Visitors	Rating	Revenue	Accommoda
3664	pwszmvbODY	France	Historical	533532	2.25	1025.81	

It will find the rows in dataset with lowest Revenue value and return only those rows.

In [149…

```
df[df.Revenue==df.Revenue.max()]
```

Out[149…

	Location	Country	Category	Visitors	Rating	Revenue	Accommoda
2705	zQtYCpWsMs	France	Urban	649167	4.69	999999.49	

It will find the rows in dataset with highest Revenue value and return only those rows.

In [150…

```
revenue_per_country = df.groupby('Country')['Revenue'].sum().reset_index()
revenue_per_country
```

Loading [MathJax]/extensions/Safe.js

Out[150…

	Country	Revenue
0	Australia	4.126633e+08
1	Brazil	4.267832e+08
2	China	3.984324e+08
3	Egypt	4.605948e+08
4	France	4.105266e+08
5	India	4.546763e+08
6	USA	4.277053e+08

It calculate the total revenue for each country in the dataframe and store the results in a new dataframe named revenue_per_country.

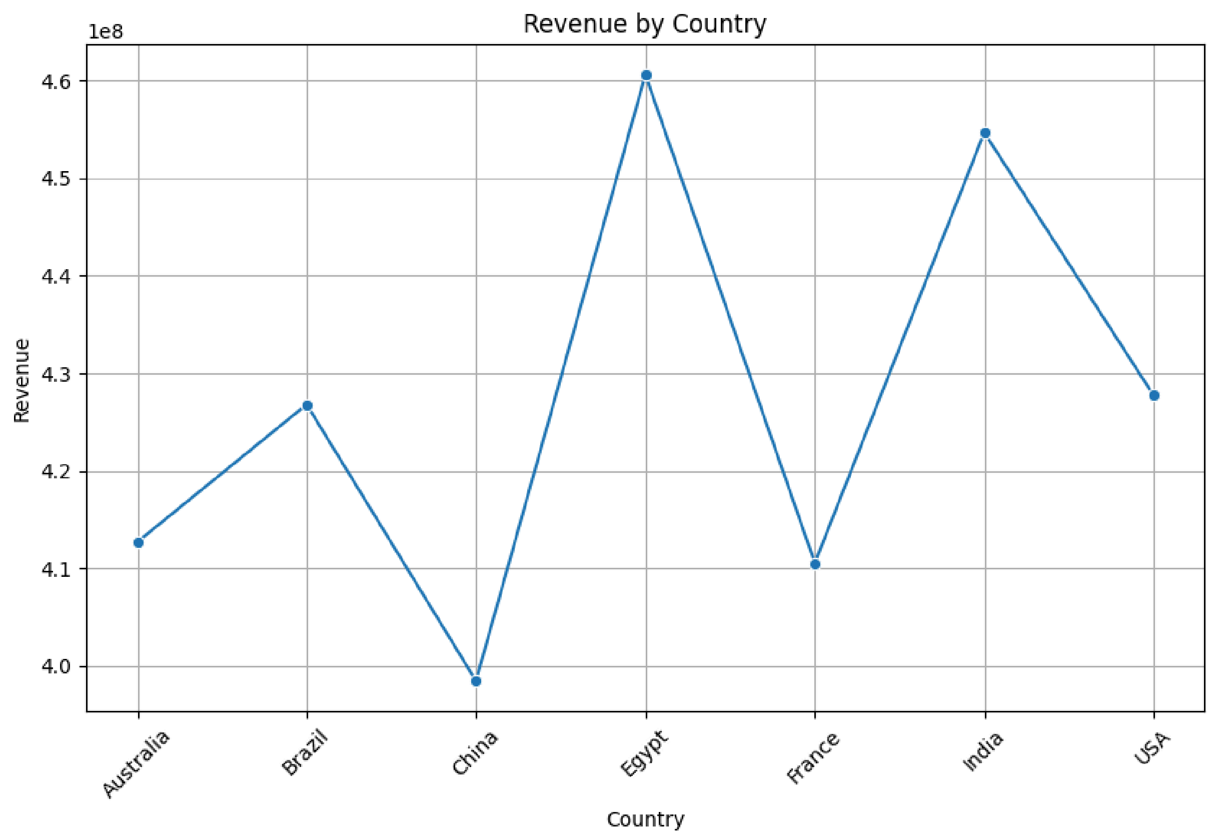
Line Chart

In [151…

```
plt.figure(figsize=(10,6))
sns.lineplot(x='Country', y='Revenue', data=revenue_per_country, marker='o')

plt.title('Revenue by Country')
plt.xlabel('Country')
plt.ylabel('Revenue')
plt.xticks(rotation=45)
plt.grid(True)
plt.show()
```

Loading [MathJax]/extensions/Safe.js



It is a line chart to visualize the revenue per country and it shows how revenue varies across different countries.

In [152…
```
revenue_per_Category = df.groupby('Category')['Revenue'].sum().reset_index()
revenue_per_Category
```

Out[152…

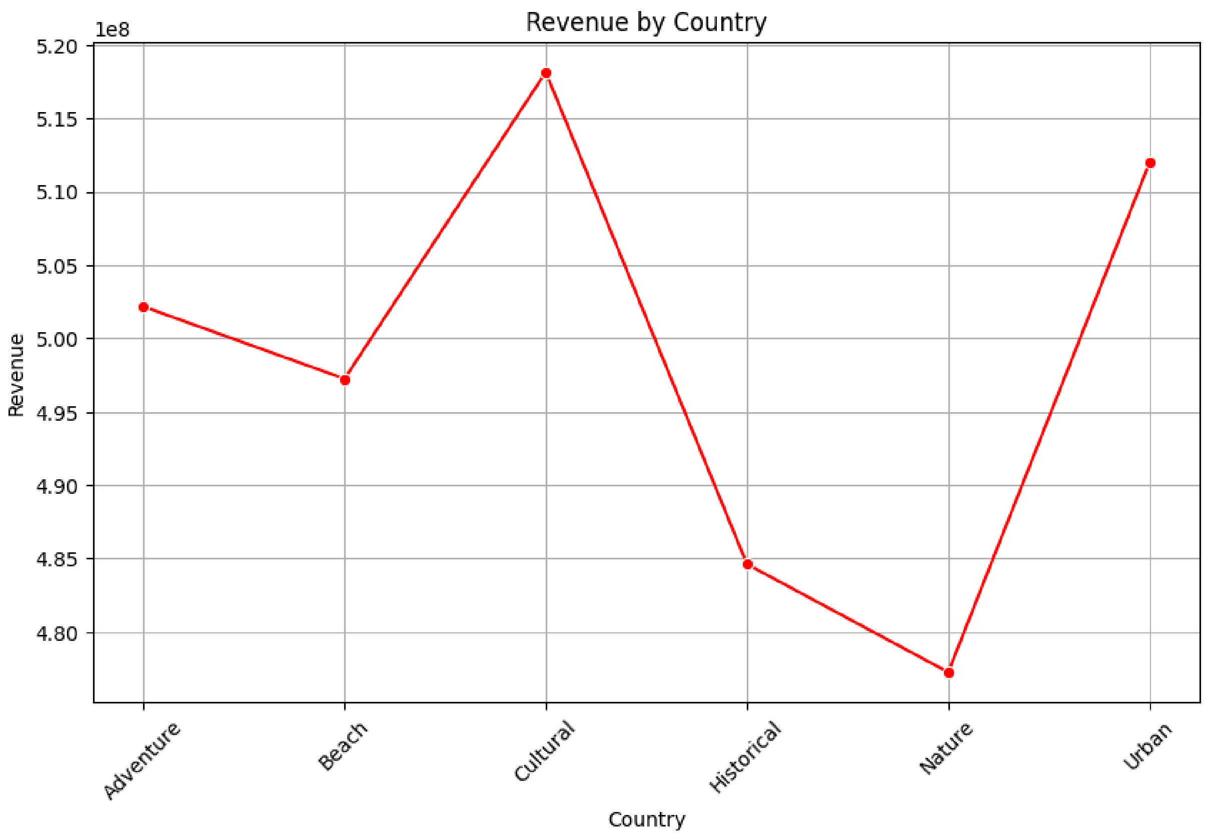
	Category	Revenue
0	Adventure	5.021662e+08
1	Beach	4.972478e+08
2	Cultural	5.181320e+08
3	Historical	4.846126e+08
4	Nature	4.772601e+08
5	Urban	5.119633e+08

It calculate the total revenue for each category in the dataframe and store the results in a new dataframe named revenue_per_category.

In [153…
```
plt.figure(figsize=(10,6))
sns.lineplot(x='Category', y='Revenue', data=revenue_per_Category, marker='c

plt.title('Revenue by Country')
plt.xlabel('Country')
plt.ylabel('Revenue')
Loading [MathJax]/extensions/Safe.js s(rotation=45)
```

```
plt.grid(True)
plt.show()
```

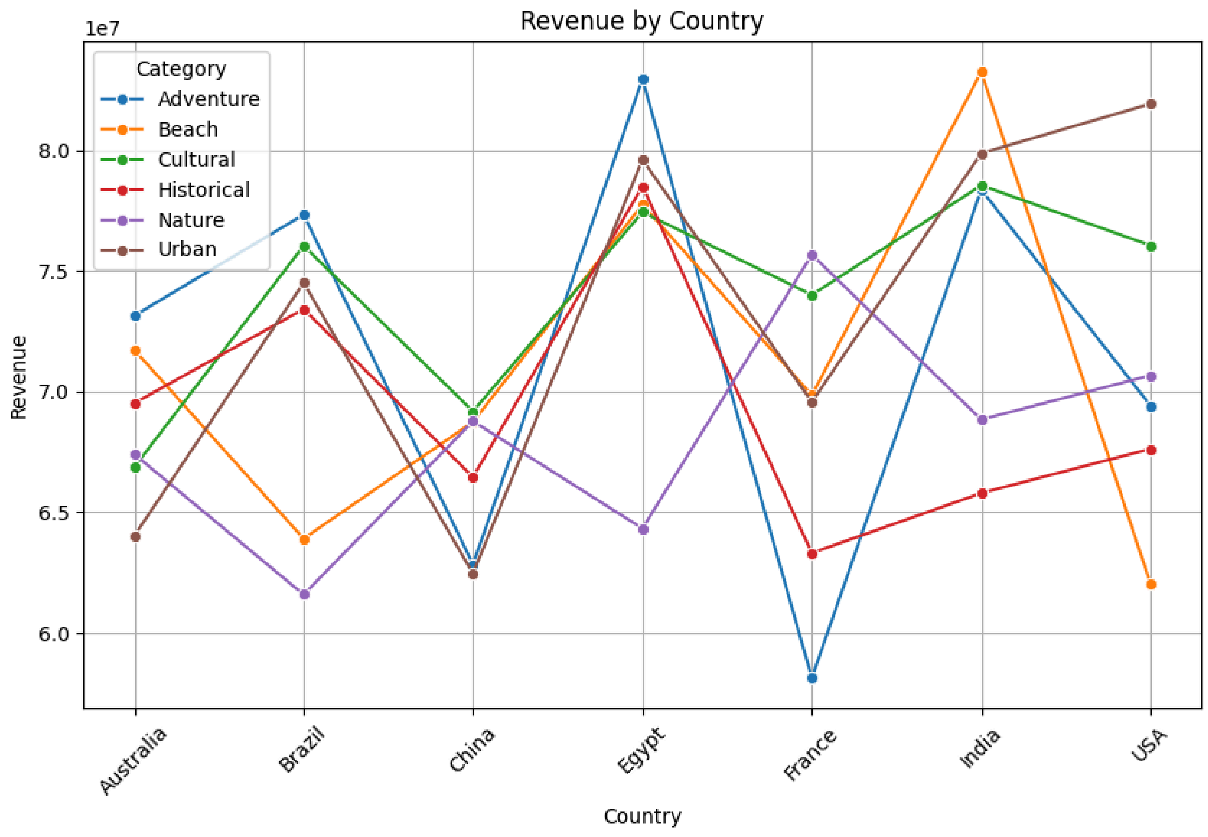


It is a line chart to visualize the revenue per categories and it shows how revenue varies across different categories.

In [154…
```
revenue_per_country = df.groupby(['Country',"Category"])['Revenue'].sum().re
plt.figure(figsize=(10,6))
sns.lineplot(x='Country', y='Revenue',hue="Category", data=revenue_per_count

plt.title('Revenue by Country')
plt.xlabel('Country')
plt.ylabel('Revenue')
plt.xticks(rotation=45)
plt.grid(True)
plt.show()
```

Loading [MathJax]/extensions/Safe.js



It is a line chart to visualize the revenue per country, categorized by different categories. It shows how revenue varies across different countries, with separate lines for each category.

In [164…

```
revenue_per_category = df.groupby(['Country', 'Category'])['Revenue'].sum().
revenue_per_category.set_index(['Country', 'Category'], inplace=True)
revenue_per_category
```

Loading [MathJax]/extensions/Safe.js

Out[164…

		Revenue
Country	**Category**	
Australia	**Adventure**	73143074.22
	Beach	71734255.42
	Cultural	66860675.16
	Historical	69513402.10
	Nature	67405458.04
	Urban	64006467.32
Brazil	**Adventure**	77333822.40
	Beach	63899164.12
	Cultural	76042841.46
	Historical	73418486.47
	Nature	61582499.48
	Urban	74506386.34
China	**Adventure**	62835395.80
	Beach	68741200.34
	Cultural	69177870.10
	Historical	66453400.60
	Nature	68768453.44
	Urban	62456057.97
Egypt	**Adventure**	82950318.40
	Beach	77755196.13
	Cultural	77438684.31
	Historical	78510790.60
	Nature	64323505.07
	Urban	79616298.02
France	**Adventure**	58126792.90
	Beach	69845116.72
	Cultural	74008400.22
	Historical	63304545.83
	Nature	75674952.86
	Urban	69566833.74
India	**Adventure**	78370335.87
	Beach	83256415.38

Loading [MathJax]/extensions/Safe.js

		Revenue
Country	Category	
	Cultural	78545467.98
	Historical	65788358.97
	Nature	68835947.02
	Urban	79879774.13
USA	Adventure	69406465.01
	Beach	62016423.75
	Cultural	76058080.68
	Historical	67623602.79
	Nature	70669239.05
	Urban	81931474.27

It will calculate the total revenue for each country and category combination. It organize the results in a Dataframe where the rows are labelled by country and category. This makes it easier to work with and analyze the data based on these two dimensions.

In [157…
```
df['Accommodation_Available'].value_counts()
```

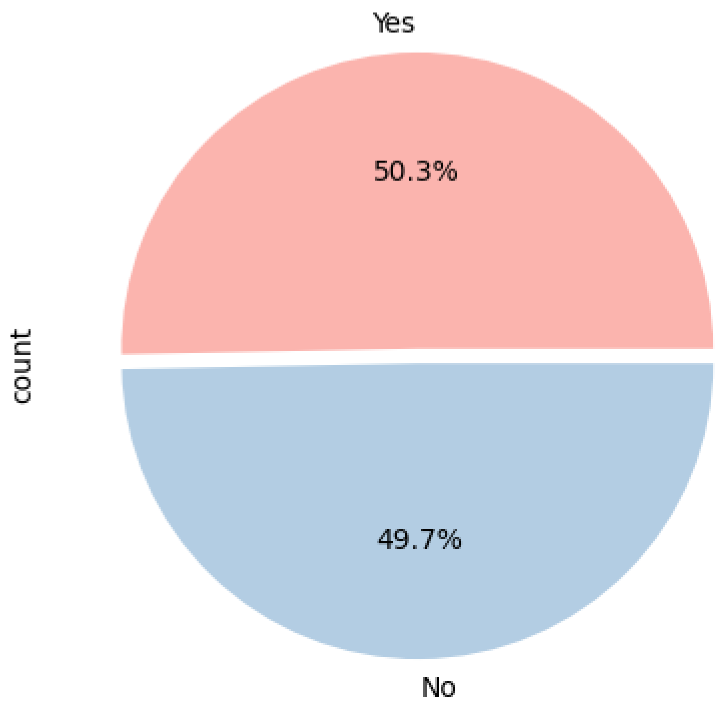
Out[157…
```
Accommodation_Available
Yes    3013
No     2976
Name: count, dtype: int64
```

It tells, how many times the values both Yes and No appear in Accommodation_Available column of my dataframe.

In [158…
```
colors = plt.get_cmap('Pastel1').colors
df['Accommodation_Available'].value_counts().plot(kind="pie", autopct='%1.1f
```

Out[158…
```
<Axes: ylabel='count'>
```

Loading [MathJax]/extensions/Safe.js



It shows the proportion of both Yes and No values in the Accommodation_Availabla column,with the colored using pastel colors.

In [159…

```
stats = df.groupby('Accommodation_Available')['Revenue'].describe()
stats
```

Out[159…

	count	mean	std	min	
Accommodation_Available					
No	2976.0	500830.503891	285201.423110	1227.89	2581
Yes	3013.0	498144.822735	288299.120411	1025.81	2454

It tells information of revenue data for both Yes and No values in the Accommodation_Available column including statistics of my dataframe.

In [160…

```
df.groupby('Accommodation_Available')['Revenue'].sum().reset_index()
```

Out[160…

	Accommodation_Available	Revenue
0	No	1.490472e+09
1	Yes	1.500910e+09

It will calculate the total revenue for both Yes and No in the Accommodation_Available column of my dataframe.

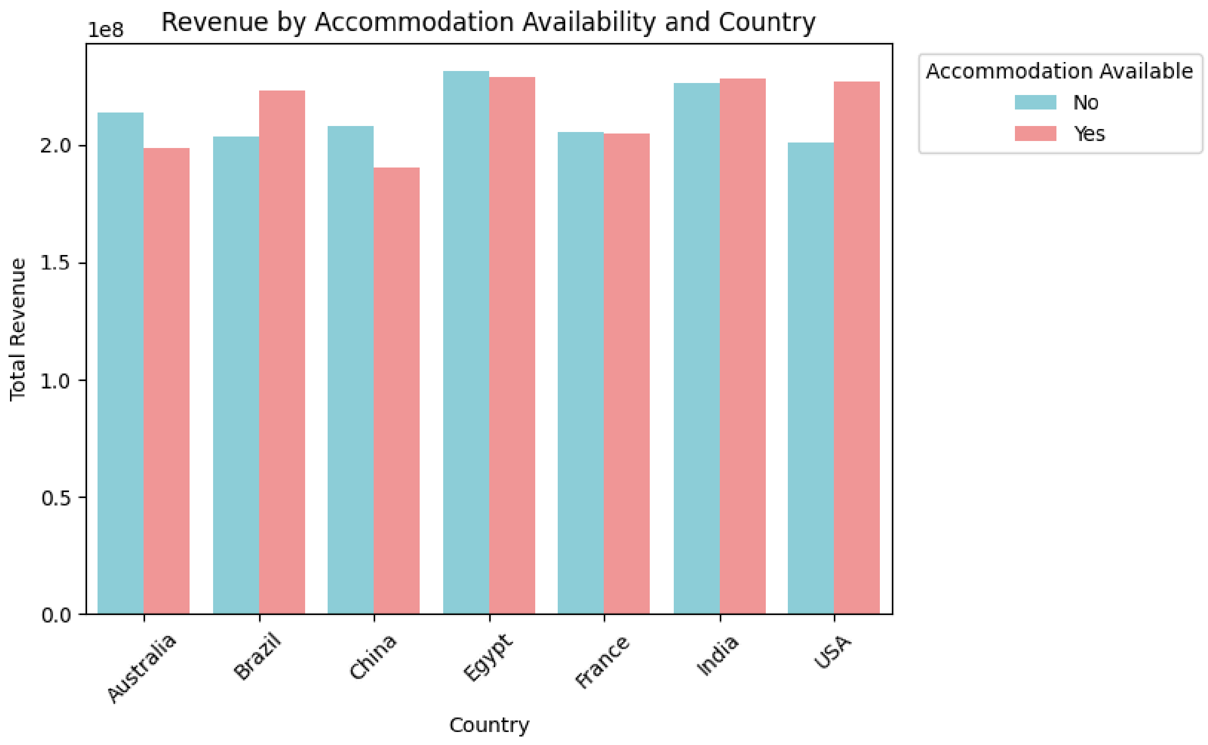
BarChart

Loading [MathJax]/extensions/Safe.js

In [161…

```
revenue_by_accommodation_country = df.groupby(['Accommodation_Available', 'C

plt.figure(figsize=(7, 5))
sns.barplot(data=revenue_by_accommodation_country, x='Country', y='Revenue',
plt.title('Revenue by Accommodation Availability and Country')
plt.xlabel('Country')
plt.ylabel('Total Revenue')
plt.xticks(rotation=45)
plt.legend(title = "Accommodation Available", bbox_to_anchor=(1.02, 1), loc=
plt.show()
```



It is used to visualizes how revenue generated by different countries based on whether or not accommodations are available.

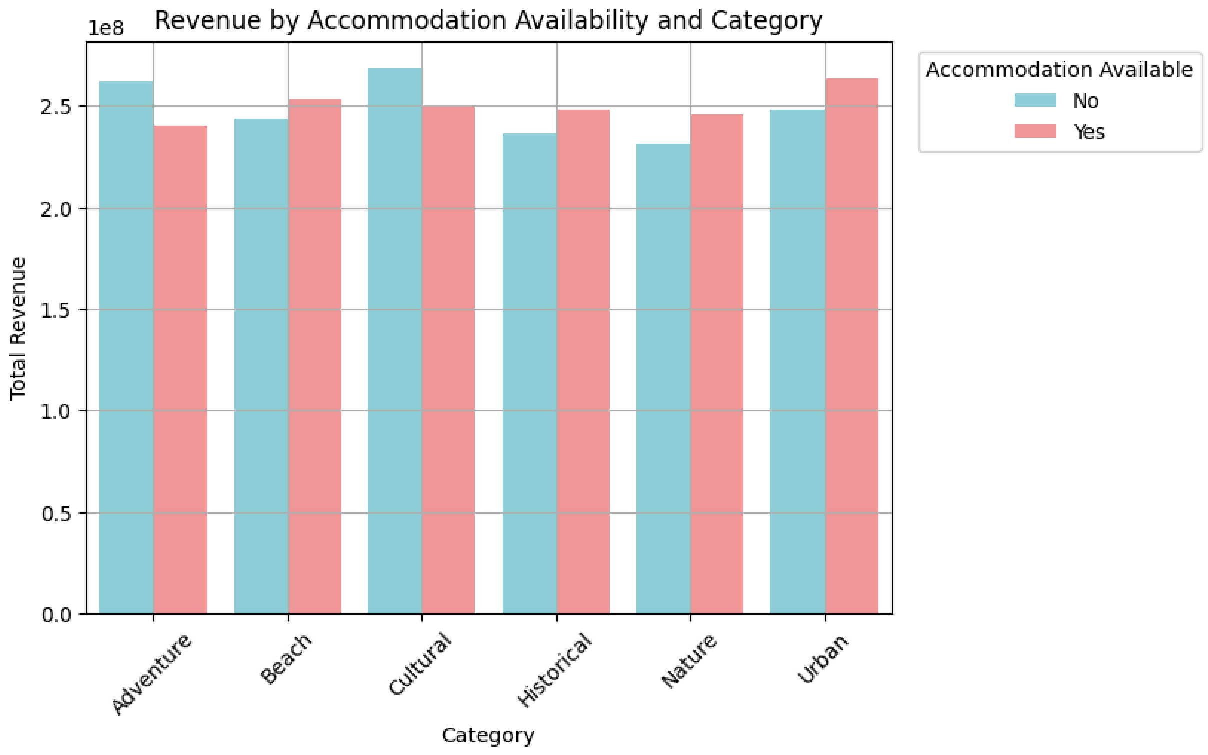
In [162…

```
revenue_by_accommodation_category = df.groupby(['Accommodation_Available', '

plt.figure(figsize=(7, 5))
sns.barplot(data=revenue_by_accommodation_category, x='Category', y='Revenue
plt.title('Revenue by Accommodation Availability and Category')
plt.xlabel('Category')
plt.ylabel('Total Revenue')
plt.xticks(rotation=45)
plt.legend(title = "Accommodation Available", bbox_to_anchor=(1.02, 1), loc=

plt.grid(True)
plt.show()
```

Loading [MathJax]/extensions/Safe.js



It is used to visualizes how revenue generated by different categories based on whether or not accommodations are available.

```
In [163…  df_Egypt =df.Location[(df.Country =="Egypt") & (df.Accommodation_Available =
          for location in df_Egypt:
              print(location,end="  ,   ")
```

Loading [MathJax]/extensions/Safe.js

IKdhVWFKRc , FOEgDMzluS , OWwgRpSeNj , asIaWBjFAf , DtRiOtXlOT , cebsVI
Qylz , zrvsAldpws , SmkajIHvrA , ZWKShywvrU , TlrDolHdpl , bFBoalAcRW ,
QzYTxhcxti , gdIkaBzpyJ , pnejoUGcUx , qbQUpweTcC , xolQKxHggX , UNnDYE
whXz , SJQkRVBekM , kEZAfaEXSH , LgKeRLsqpk , CbMQsNdMwN , ANWyZwqIlY ,
nfZZDrEVCs , MYeUrLkikD , XaEkfUeopT , sjqxIQHwzW , SxQrfqMYBC , kxGIJK
tCpU , RKCvHVkRIZ , QvoQwzIVft , gOGrsNHCRu , erFCFZEJfY , yviCCotwgL ,
lPQftFwLiM , smMjeQIMEa , fIuwlSPCud , fPrTVCEvVZ , mRamfeVrGK , UFJEfR
OFma , WiTQmCHwnT , oujDSeKQPa , quKByLfTrS , GFOIXDYcjx , hWVpZaWjPT ,
IqYCOGHdcU , UHOapEemot , RYKZaIvtid , gRIPgdkrUM , YmbbytNUcA , heBzro
WkIT , ExmGSRRWAZ , VtfIvHdtUJ , kstANZoqBx , pwAzGJgiWQ , MNumQhCyxc ,
NYQkFlvjcs , PjQZCBKVuw , NJsuerAfxv , qLVUimrewX , FWPfZENkiG , KAwoMg
cEMY , RvmkpbzcSJ , LBRtcRRQsX , GsvdqNpfcY , HYhRJTNRIm , tbhFgqjjYA ,
OQeeLpHKUV , geWKYgvgGz , kqzYmnkQuQ , pvEriZZhjv , ECXfuvTppO , uKdevC
FJWa , hGslUCuJsX , NBugGLLUmp , EsQoycESds , BQPyyzITBX , RyvTlgqKTK ,
tEeTRwlaDq , UwwnhCEfkM , IVLKfNvYOF , FaHlkgZdeE , BntekcoNcZ , peyoCB
vKwY , bQreneXgBM , XcbqrBGFhO , HewCHnNMdR , utRccybLan , lDXJnPvmkd ,
TSHsVIaOrI , bcGmeEqBiT , YiJBFKDAPt , ghyoxixhxC , nhseTNokoc , quYxpg
lyPd , sYxxNzOCzJ , zCCCkmjSQN , QKPziKeBfn , RIzzwhuckw , PrxculcMox ,
EGcpdaWoaJ , lGaGYEdaqx , fUhwQpBhno , UXpqxBnIcs , zqIEDzEVqh , QBRLzb
NXhp , pPNTjYhjQn , XwaLrYfvYi , elFLtBPwhT , UAqcRjomyO , OQcOqLKvFd ,
fAynrcpnzD , bVmhTKnYKz , jaIUBVvnyS , apwBvotTlA , qyVNqJWClj , gTQAFW
fihW , LfSRIYvGwH , asQMZAiyEp , lSOKXgXyCv , cGTWvpFDyQ , ERiNYvsnyi ,
opJDkZyFHZ , zJsUpymiLZ , lIbMmtfYcc , lDzwlzmOiG , EzGYYbuSBO , XRsGiR
bTkH , icVhzkjknc , dcrfLpJyFw , PFAArRNXIg , jUjYzzBFSp , LWHFlSHVrV ,
YjsoRIFXAx , qbwfDMncdS , MWwnwSzguT , jiagKMKRIX , gaRckCbMbL , MbdYQa
odRG , xGMYyxuzka , AOfhtePmHi , lTxEzyBcOi , eUVCdIDeEM , fKWFqEzryz ,
GUmwISwhvb , fAsaunFFjY , WObrdHNGmZ , IsuSUkLalP , ATbyeMJCYx , YyWJmp
kotX , XoTqSLaByQ , iMJeAjOgdC , bxrYlaoLzZ , yxoDTJYSMK , LNHxbYEAxy ,
LtqbHnfdUw , GTjbjoYCNS , TVppvnIeLD , ROJJpcdCnu , lYEYqDWuHi , feJTop
FEaj , TBiupRukeP , SoUwxFUozn , FJpKtkAPNe , jVKSQbnhEL , GUrTVKmuiO ,
JYAzkWJpjC , zMCXXQKwWH , CSBmUjKbWs , NnrAsQZMCU , hSJpMJAcRF , ndCMFB
wTEG , IrkvjfuqDe , TBrkfKOVkt , VbOBMchoBN , SIqXxssmNW , rykKBNVqhH ,
YOdhtjqZhb , VvfuzhLXea , LsrAjYFWiy , AkWmSlQQdH , ojgXgSEzQi , uypyNy
zBUw , BSfnEmMLHL , CxozCjTsEs , PsfGnInqRE , tbAugNWmak , arOSnxwfTd ,
FKULWjKVoc , stdLUWnptc , maSrarypgz , eDelThPPmB , YCfMUFPrFG , TqIhTi
xDyt , svlyRYQFKP , HXFIJcYArb , ResxHmFybY , rPtlYdKeTu , fuSXkIsNoo ,
rrZdbgEIqB , BzvUmgYXnX , vipLRaojFT , qTyQYwsOAy , uXyEKeBZcQ , zhAxbr
ahog , DHGaEPazdg , FNIdUWmVME , FhJafDRJmu , zRpulPnkyg , IRrOsswFew ,
GxQSXdrVTl , qBiPDYjsuf , wrvztaDqpt , DXoYKxZKaB , kjhhHAreSr , UawWtS
okDk , AZgGWuQjMJ , IVSRhkqmku , LAeWbrtMAH , RsUoSEQvTS , bNnpsEBBcV ,
yCjpfqAnQx , XpKSvdGbKf , JXYrpNYIYg , ntfBuvznQn , NmhqqfYPJJ , wJgYiO
XDrU , oIfrqMZPBS , ccAaghdOJI , PQmMCRHBCG , CZzlGSZcmF , vIjrZwizkh ,
bomrExLfnj , nXwpzgUIIa , KrXjWZzrJT , MwutiYqhHn , UjQOEhvKqD , ZMeKGQ
GykZ , bDgggxaiwJ , QTsmBPjYgc , WLpmYlwoBQ , JCYSllcHkb , dUUplyXdZT ,
fXNnaILqaS , nFLCbNWwZj , mlTEnkLSMD , lecHUTeiEy , vSJxZyQmPF , jxDigt
LXWi , zYIBDXdiUm , QMUiNTGmIB , GtkotAlzCi , TBAGEiczso , CrTBHfLONr ,
pTTtFEtkqe , DvQXAuoqiI , MhkEhkuZfG , cPOiXYKpvC , VeBLHMjMnX , OvYknU
RmTT , PkPmfUCmBO , zCsgPmiqzZ , NIOQpeKxyc , gSWaiodInq , Rzsruwbszy ,
NbmeYBINFg , MCdAeIIoCq , fCcBMYhtHt , NLcOOjREZS , dGRxyiiDdp , clDoDL
LrCI , loiGbsDEEb , KSVZqwKlvR , iBYONdYmjC , ZHIbVnxXsl , yPrvtvaste ,
HYHYEIdOic , BcHdUCjcgS , qDgEbWkcWH , oCnpzeqRYJ , IebAsPyiRj , WwuBaR
kcjb , ffVzrqcfmP , pnTdPBGTjg , hxuTMkhTor , cDujYkPcYh , aUewxRJMGL ,
NUePruLMhh , myPXiqjcUI , aoqpNFJjNH , IBgnwdlztt , XpfmcNApWF , IJWoad
VIpq , HFDHexVYRs , zzqsjdBSdg , yGVefJDHye , UeBDkMQlEI , JYCwGLNvze ,
TtiaFqkTTW , KQjvmjVFDh , CukejytywZ , DkJVCEtPgo , yzWykGMCJD , CBXUme
gaYi , kCBnqwUZhq , NSTKrlrTrX , qSpEncZmHm , wsTyneRrxf , pzdqllCyop ,

Loading [MathJax]/extensions/Safe.js

```
VCTInimxdv ,  qjgGjqvhaN ,  nRhwDFJIdw ,  eZUvRKkEdw ,  avBHVOpQwN ,  Kiuema
YBhQ ,  zQkiAoJOzF ,  FsfWbpThQH ,  PkbOHVpelA ,  nHfXXujUtQ ,  YKoJigKObC ,
YspVRTPULE ,  wzNSvadTwY ,  zgXZbLMTPz ,  YYaZEWhxLc ,  qPrPXofFxL ,  jxsMUP
BudN ,  xIrwvSXseu ,  HSWwYRhLGq ,  wlmgtOpIQW ,  WTLzPScmmM ,  HYxMicgkQN ,
NeFFsRpvPU ,  lhXcnMgwir ,  liMGBjAJfj ,  nDyDiFVkkt ,  BzTCAWCnXC ,  VPwrOe
zknm ,  HKXzdfaIdp ,  kewRHZuMZl ,  bpHIKalkFi ,  EWUryWKnVr ,  tliEokPgaM ,
NDEustwEvj ,  OqWOMMobzd ,  KDepMFSNgg ,  qfKXfQgHHt ,  NqbmfTbfFM ,  wrQghk
WnbX ,  wzKjDhsMwu ,  ygkliomfhH ,  JRyNsHvSLV ,  ctgMEEDyXK ,  aOpExaXRlM ,
JQmMbGCqwg ,  wEkzKoTvDK ,  gYNqYFakOc ,  nDbQNGkPLl ,  glMDxFYrSA ,  XYHssz
dxSC ,  QAWWykGjNV ,  RhxemNFzhO ,  lOzUZtZLaI ,  YqKzrKDmvm ,  obUzFMCUQt ,
LRNXZxRgPY ,  iSgzkJLBmP ,  sproIkqwCT ,  TlfgJVHGsV ,  kOUaGrhozW ,  jaxArp
ueaw ,  mEjYhaltjj ,  ocGzJcvOkS ,  yTZIQRHCXK ,  SGRTOKpiHB ,  LOlFNmkBsU ,
DGvLjXcJdR ,  XfcARdNxeY ,  NAymlklfmt ,  AmsxENePwv ,  wXidvNDFiY ,  zXYULm
JAwO ,  nTxWuoZChp ,  cEGvYRQJJB ,  GaaxBjZsJb ,  sYZhBCgRAA ,  XvJNVUbgXH ,
iuaSNpXXIU ,  rSBeruDYBt ,  uoUrIpjPSz ,  gcMzdEmvlY ,  YbeFAXOQWn ,  IcuJXZ
RJjo ,  wDslfVKypQ ,  JiqoYJpUdh ,  JwFDbvVqPH ,  VmynvzoIlh ,  mGhSrrehkX ,
dCYySYDNxt ,  ZgnkWOOjMT ,  aKtdvCZXPY ,  xAEfLlGLxL ,  hIYAFRZMyB ,  dlHWpv
gNTK ,  PdkwudRyLk ,  UrlWeipaXB ,  BtrqpXPvLB ,  SrDxQRSAAp ,  rfchMzBxHl ,
glQlbLrMuv ,  pRsaflkkWN ,  MWOyLGYFMS ,  kpwWXLrKkO ,  aDMllVKIhs ,  zYeqze
VxpG ,  PZyuAVduyj ,  PlkaafiBZB ,  fZCFdGfryC ,  roclPXVqHq ,  oLxvggHLXZ ,
MGkAnYJUwK ,  eCWAOfNabb ,  atCYUHnpet ,  ZXaTDAsVZS ,  OUnGhpGcWL ,  VKKAnT
RfjG ,  WJJFVOkPhm ,  DEEuZbHzpd ,  LSJsvesmWn ,  MWaTLaarVK ,  NriqxUJfsj ,
JDuULWkceh ,  SJfTGKGubd ,  XuFrOhHDPZ ,  REujBLZBTT ,  FmjSTSqOYr ,  OKNVKF
NIjS ,  zNRIhzwMBo ,  DybboXUaSZ ,  gvnmZKgrDH ,  uBOCKQJolY ,  SmCZsVAKxz ,
gPRLYnBFoZ ,  ASrLFXmorH ,  XAZSaPsQMW ,  KXiAeiTgDn ,  fbZMYbYbhi ,  VhAcaV
hkcn ,  erYhkmErwN ,  EgQXlHhHGU ,  aziUliiYpI ,  vWndAMSIIt ,  lZjgvpdyQB ,
KJXkCYuMeI ,  ZltwnozDDK ,  AEuJwTINDL ,  kkWIucpBnu ,
```

It is used to extracting and displaying the specific locations in Egypt that have accommodations are available.

Conclusion

The dataset shows that popular tourist locations with more accommodations tend to attract higher numbers of visitors and generate more revenue. Categories like'Nature','Historical',appear to be key drivers of tourism. Countries like India and the USA show significant tourism activity

Happy Learning

Thank You

```
In [ ]:
```

Loading [MathJax]/extensions/Safe.js