hotel-booking-analysis

June 30, 2024

[ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
from datetime import datetime
import seaborn as sns
import ast
```

[ ]:
```
hotel_booking_df=pd.read_csv('hotel_booking.csv')
```

[ ]:
```
hotel_booking_df.head()
```

[ ]:

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	\
0	Resort Hotel	0	342	2015	July	
1	Resort Hotel	0	737	2015	July	
2	Resort Hotel	0	7	2015	July	
3	Resort Hotel	0	13	2015	July	
4	Resort Hotel	0	14	2015	July	

	arrival_date_week_number	arrival_date_day_of_month	\
0	27	1	
1	27	1	
2	27	1	
3	27	1	
4	27	1	

	stays_in_weekend_nights	stays_in_week_nights	adults	…	customer_type	\
0	0	0	2	…	Transient	
1	0	0	2	…	Transient	
2	0	1	1	…	Transient	
3	0	1	1	…	Transient	
4	0	2	2	…	Transient	

	adr	required_car_parking_spaces	total_of_special_requests	\
0	0.0	0	0	
1	0.0	0	0	
2	75.0	0	0	

```
3  75.0                          0                       0
4  98.0                          0                       1

  reservation_status reservation_status_date           name  \
0          Check-Out              2015-07-01   Ernest Barnes
1          Check-Out              2015-07-01    Andrea Baker
2          Check-Out              2015-07-02  Rebecca Parker
3          Check-Out              2015-07-02    Laura Murray
4          Check-Out              2015-07-03     Linda Hines

                         email  phone-number       credit_card
0  Ernest.Barnes31@outlook.com  669-792-1661  ************4322
1       Andrea_Baker94@aol.com  858-637-6955  ************9157
2   Rebecca_Parker@comcast.net  652-885-2745  ************3734
3            Laura_M@gmail.com  364-656-8427  ************5677
4           LHines@verizon.com  713-226-5883  ************5498

[5 rows x 36 columns]
```

[ ]:
```
hotel_booking_df.tail()
```

[ ]:
```
            hotel  is_canceled  lead_time  arrival_date_year  \
119385  City Hotel            0         23               2017
119386  City Hotel            0        102               2017
119387  City Hotel            0         34               2017
119388  City Hotel            0        109               2017
119389  City Hotel            0        205               2017

       arrival_date_month  arrival_date_week_number  \
119385             August                        35
119386             August                        35
119387             August                        35
119388             August                        35
119389             August                        35

        arrival_date_day_of_month  stays_in_weekend_nights  \
119385                         30                        2
119386                         31                        2
119387                         31                        2
119388                         31                        2
119389                         29                        2

        stays_in_week_nights  adults  …  customer_type     adr  \
119385                     5       2  …      Transient   96.14
119386                     5       3  …      Transient  225.43
119387                     5       2  …      Transient  157.71
119388                     5       2  …      Transient  104.40
```

```
119389                     7       2  …      Transient  151.20

       required_car_parking_spaces total_of_special_requests  \
119385                            0                         0
119386                            0                         2
119387                            0                         4
119388                            0                         0
119389                            0                         2

       reservation_status reservation_status_date               name  \
119385          Check-Out              2017-09-06     Claudia Johnson
119386          Check-Out              2017-09-07      Wesley Aguilar
119387          Check-Out              2017-09-07        Mary Morales
119388          Check-Out              2017-09-07  Caroline Conley MD
119389          Check-Out              2017-09-07      Ariana Michael

                          email  phone-number       credit_card
119385       Claudia.J@yahoo.com  403-092-5582  ************8647
119386      WAguilar@xfinity.com  238-763-0612  ************4333
119387  Mary_Morales@hotmail.com  395-518-4100  ************1821
119388   MD_Caroline@comcast.net  531-528-1017  ************7860
119389      Ariana_M@xfinity.com  422-804-6403  ************4482

[5 rows x 36 columns]
```

[ ]:
```
hotel_booking_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 119390 entries, 0 to 119389
Data columns (total 36 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   hotel                           119390 non-null  object
 1   is_canceled                     119390 non-null  int64
 2   lead_time                       119390 non-null  int64
 3   arrival_date_year               119390 non-null  int64
 4   arrival_date_month              119390 non-null  object
 5   arrival_date_week_number        119390 non-null  int64
 6   arrival_date_day_of_month       119390 non-null  int64
 7   stays_in_weekend_nights         119390 non-null  int64
 8   stays_in_week_nights            119390 non-null  int64
 9   adults                          119390 non-null  int64
 10  children                        119386 non-null  float64
 11  babies                          119390 non-null  int64
 12  meal                            119390 non-null  object
 13  country                         118902 non-null  object
 14  market_segment                  119390 non-null  object
```

```
 15  distribution_channel            119390 non-null  object
 16  is_repeated_guest               119390 non-null  int64
 17  previous_cancellations          119390 non-null  int64
 18  previous_bookings_not_canceled  119390 non-null  int64
 19  reserved_room_type              119390 non-null  object
 20  assigned_room_type              119390 non-null  object
 21  booking_changes                 119390 non-null  int64
 22  deposit_type                    119390 non-null  object
 23  agent                           103050 non-null  float64
 24  company                         6797 non-null    float64
 25  days_in_waiting_list            119390 non-null  int64
 26  customer_type                   119390 non-null  object
 27  adr                             119390 non-null  float64
 28  required_car_parking_spaces     119390 non-null  int64
 29  total_of_special_requests       119390 non-null  int64
 30  reservation_status              119390 non-null  object
 31  reservation_status_date         119390 non-null  object
 32  name                            119390 non-null  object
 33  email                           119390 non-null  object
 34  phone-number                    119390 non-null  object
 35  credit_card                     119390 non-null  object
dtypes: float64(4), int64(16), object(16)
memory usage: 32.8+ MB
```

1 NUMBERS OF ROWS CONTAINS:

[ ]:
```
print("our dataset have",hotel_booking_df.shape[0],'rows.')
```

```
our dataset have 119390 rows.
```

2 NUMBERS OF COLUMN CONTAINS:

[ ]:
```
print("our dataset have",hotel_booking_df.shape[0],'columns.')
```

```
our dataset have 119390 columns.
```

3 CHECK FOR LOST DATA?

[ ]:
```
count_null=hotel_booking_df.isnull().sum()
for i in range(hotel_booking_df.shape[0]-2):
    if(count_null[i]>0):
        print('yes, we have at least one missing data.')
        break
```

```
yes, we have at least one missing data.
```

4 FROM WHICH COUNTRY MORE NUMBERS TRAVELLER COME AND

LISTING FIVE COUNTRIES WITH MOST PASSENGERS:

[ ]:
```
count_country=hotel_booking_df.groupby(by='country')['name'].count()
count_country=count_country.sort_values(ascending=False)
hotel_booking_df_count_country=pd.DataFrame(count_country)

print("The most travelers come from:\n ",count_country[:1])

print("\nList the 5 countries with the most passengers.\n",count_country[:5])
```

```
The most travelers come from:
  country
PRT    48590
Name: name, dtype: int64

List the 5 countries with the most passengers.
 country
PRT    48590
GBR    12129
FRA    10415
ESP     8568
DEU     7287
Name: name, dtype: int64
```

5 WHO HAS THE MOST ADR?

[ ]:
```
ADR=hotel_booking_df.groupby(by='name')['adr'].max()
ADR=ADR.sort_values(ascending=False)
ADR_df = pd.DataFrame(ADR)
print('The most ADR(Average Daily Rate) is:\n',ADR_df.iloc[0])
```

```
The most ADR(Average Daily Rate) is:
 adr    5400.0
Name: Daniel Walter, dtype: float64
```

6 CALCULATING THE MEAN OF THE TOTAL ADR?

[ ]:
```
num_adr=hotel_booking_df['adr'].count()
Avg_adr=hotel_booking_df['adr'].sum()/num_adr

print('Average of total Average Daily Rate is:', Avg_adr.round(decimals=2))
```

```
Average of total Average Daily Rate is: 101.83
```

7 CALCULATING THE AVERAGE NUMBER OF NIGHT SPENDITURE:

[ ]:
```
hotel_booking_df['all_night']= hotel_booking_df.apply(lambda x:␣
↪x['stays_in_weekend_nights'] + x['stays_in_week_nights'], axis=1)
m = hotel_booking_df['all_night'].mean()
np.round(m,2)
```

[ ]:
```
3.43
```

8 WHAT ARE THE MOST COMMON LAST NAME:

[ ]:
```
hotel_booking_df['last_name'] = hotel_booking_df['name'].apply(lambda x : x[x.
↪index(' ') + 1:])
hotel_booking_df['last_name'].value_counts()[:5]
```

[ ]:
```
last_name
Smith       2466
Johnson     1968
Williams    1590
Jones       1420
Brown       1403
Name: count, dtype: int64
```

9 FINDING WHO HAS BOOKED THE HOTEL WITH MOST CHILDRENS OR BABIES:

[ ]:
```
max_babies=hotel_booking_df['babies'].idxmax()
max_children=hotel_booking_df[ 'children'].idxmax()

print(hotel_booking_df.iloc[max_children]["name"],
      " has booked a hotel with the largest number of children.")

print(hotel_booking_df.iloc[max_babies]["name"],
      " has booked a hotel with the largest number of babies.")
```

```
Jamie Ramirez  has booked a hotel with the largest number of children.
Nicholas Parker  has booked a hotel with the largest number of babies.
```

10 SPECIFY THE PHONE CODE OF THE AREAS THA HAVE MOST HOTEL RESERVATIONS:

[ ]:
```
phone_code=hotel_booking_df['phone-number'].apply(lambda x : x[:3]).
 ↪value_counts()[:3]

print("3 first codes that have the most hotel reservations :\n", phone_code)
```

```
3 first codes that have the most hotel reservations :
 phone-number
799    168
185    167
541    166
Name: count, dtype: int64
```

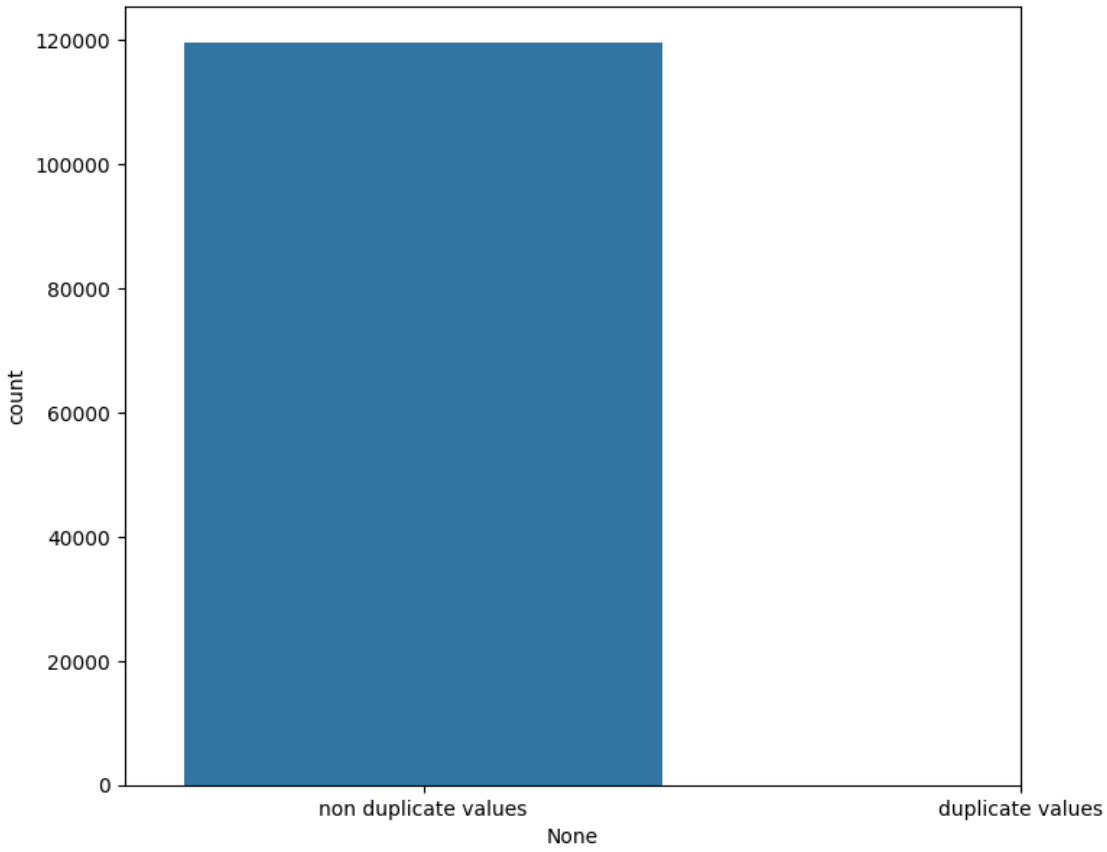
[ ]:
```
DUPLICATE VALUES:
```

[ ]:
```
len(hotel_booking_df[hotel_booking_df.duplicated()])
```

[ ]: 0

11 VISUALIZATION OF DUPLICATE VALUES:

[ ]:
```
plt.figure(figsize=(8,7))
sns.countplot(x=hotel_booking_df.duplicated())
plt.xticks([0,1],["non duplicate values","duplicate values"],fontsize=10)
plt.show()
```



12 FINDING MISSING VALUES:

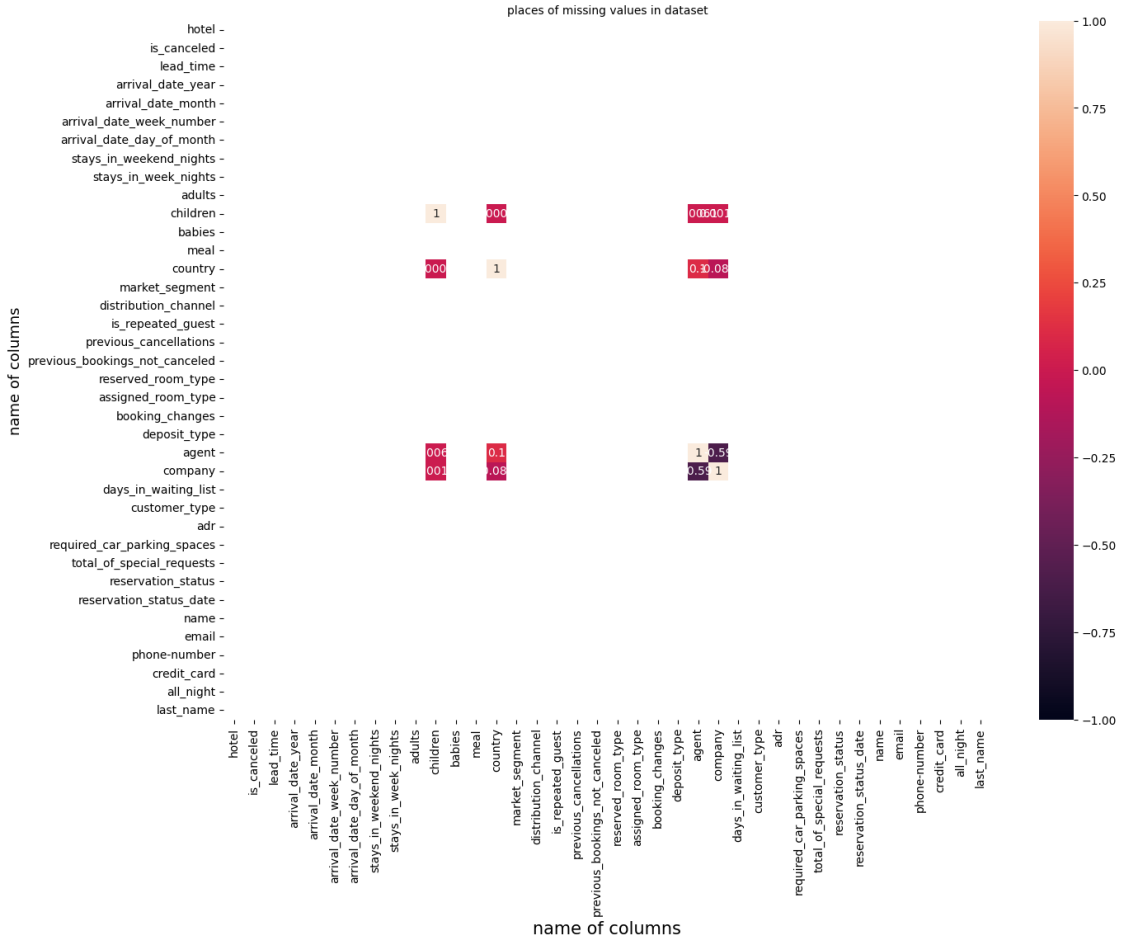
[ ]:
```
hotel_booking_df.isnull().sum().sort_values(ascending=False)
```

[ ]:
```
company                        112593
agent                           16340
country                           488
children                            4
assigned_room_type                  0
deposit_type                        0
days_in_waiting_list                0
customer_type                       0
adr                                 0
required_car_parking_spaces         0
hotel                               0
total_of_special_requests           0
reservation_status                  0
reservation_status_date             0
name                                0
```

```
email                                0
phone-number                         0
credit_card                          0
all_night                            0
booking_changes                      0
reserved_room_type                   0
is_canceled                          0
stays_in_week_nights                 0
lead_time                            0
arrival_date_year                    0
arrival_date_month                   0
arrival_date_week_number             0
arrival_date_day_of_month            0
stays_in_weekend_nights              0
adults                               0
previous_bookings_not_canceled       0
babies                               0
meal                                 0
market_segment                       0
distribution_channel                 0
is_repeated_guest                    0
previous_cancellations               0
last_name                            0
dtype: int64
```

13 VISUALIZATION OF MISSING VALUES:

[ ]:
```
plt.figure(figsize=(15,11))
sns.heatmap(hotel_booking_df.isnull().corr(),vmin=-1,annot=True)

plt.xlabel('name of columns',fontsize=15)
plt.ylabel('name of columns',fontsize=13)
plt.title('places of missing values in dataset',fontsize=10)

plt.show()
```



14 UNDERSTANDING THE VARIABLES:

[ ]:
```
hotel_booking_df.describe()
```

[ ]:
```
         is_canceled      lead_time  arrival_date_year  \
count  119390.000000  119390.000000      119390.000000
mean        0.370416     104.011416        2016.156554
std         0.482918     106.863097           0.707476
min         0.000000       0.000000        2015.000000
25%         0.000000      18.000000        2016.000000
50%         0.000000      69.000000        2016.000000
75%         1.000000     160.000000        2017.000000
max         1.000000     737.000000        2017.000000

       arrival_date_week_number  arrival_date_day_of_month  \
count             119390.000000              119390.000000
mean                  27.165173                  15.798241
```

```
std                   13.605138                   8.780829
min                    1.000000                   1.000000
25%                   16.000000                   8.000000
50%                   28.000000                  16.000000
75%                   38.000000                  23.000000
max                   53.000000                  31.000000

       stays_in_weekend_nights  stays_in_week_nights         adults  \
count            119390.000000         119390.000000  119390.000000
mean                  0.927599              2.500302       1.856403
std                   0.998613              1.908286       0.579261
min                   0.000000              0.000000       0.000000
25%                   0.000000              1.000000       2.000000
50%                   1.000000              2.000000       2.000000
75%                   2.000000              3.000000       2.000000
max                  19.000000             50.000000      55.000000

            children         babies  …  previous_cancellations  \
count  119386.000000  119390.000000  …          119390.000000
mean        0.103890       0.007949  …               0.087118
std         0.398561       0.097436  …               0.844336
min         0.000000       0.000000  …               0.000000
25%         0.000000       0.000000  …               0.000000
50%         0.000000       0.000000  …               0.000000
75%         0.000000       0.000000  …               0.000000
max        10.000000      10.000000  …              26.000000

       previous_bookings_not_canceled  booking_changes          agent  \
count                  119390.000000    119390.000000  103050.000000
mean                        0.137097         0.221124      86.693382
std                         1.497437         0.652306     110.774548
min                         0.000000         0.000000       1.000000
25%                         0.000000         0.000000       9.000000
50%                         0.000000         0.000000      14.000000
75%                         0.000000         0.000000     229.000000
max                        72.000000        21.000000     535.000000

           company  days_in_waiting_list            adr  \
count  6797.000000         119390.000000  119390.000000
mean    189.266735              2.321149     101.831122
std     131.655015             17.594721      50.535790
min       6.000000              0.000000      -6.380000
25%      62.000000              0.000000      69.290000
50%     179.000000              0.000000      94.575000
75%     270.000000              0.000000     126.000000
max     543.000000            391.000000    5400.000000
```

	required_car_parking_spaces	total_of_special_requests	all_night
count	119390.000000	119390.000000	119390.000000
mean	0.062518	0.571363	3.427900
std	0.245291	0.792798	2.557439
min	0.000000	0.000000	0.000000
25%	0.000000	0.000000	2.000000
50%	0.000000	0.000000	3.000000
75%	0.000000	1.000000	4.000000
max	8.000000	5.000000	69.000000

[8 rows x 21 columns]

15 CHECKING UNIQUE VALUES IN VARIABLES:

[ ]:
```
#CHECKING THE  NUMBERS OF UNIQUE VALUES FOR EACH VARIABLLES

for elem in hotel_booking_df.columns:
  print("numbers of unique values in",elem,"column is",hotel_booking_df[elem].
 ↪nunique())
```

```
numbers of unique values in hotel column is 2
numbers of unique values in is_canceled column is 2
numbers of unique values in lead_time column is 479
numbers of unique values in arrival_date_year column is 3
numbers of unique values in arrival_date_month column is 12
numbers of unique values in arrival_date_week_number column is 53
numbers of unique values in arrival_date_day_of_month column is 31
numbers of unique values in stays_in_weekend_nights column is 17
numbers of unique values in stays_in_week_nights column is 35
numbers of unique values in adults column is 14
numbers of unique values in children column is 5
numbers of unique values in babies column is 5
numbers of unique values in meal column is 5
numbers of unique values in country column is 177
numbers of unique values in market_segment column is 8
numbers of unique values in distribution_channel column is 5
numbers of unique values in is_repeated_guest column is 2
numbers of unique values in previous_cancellations column is 15
numbers of unique values in previous_bookings_not_canceled column is 73
numbers of unique values in reserved_room_type column is 10
numbers of unique values in assigned_room_type column is 12
numbers of unique values in booking_changes column is 21
numbers of unique values in deposit_type column is 3
numbers of unique values in agent column is 333
numbers of unique values in company column is 352
numbers of unique values in days_in_waiting_list column is 128
```

```
numbers of unique values in customer_type column is 4
numbers of unique values in adr column is 8879
numbers of unique values in required_car_parking_spaces column is 5
numbers of unique values in total_of_special_requests column is 6
numbers of unique values in reservation_status column is 3
numbers of unique values in reservation_status_date column is 926
numbers of unique values in name column is 81503
numbers of unique values in email column is 115889
numbers of unique values in phone-number column is 119390
numbers of unique values in credit_card column is 9000
numbers of unique values in all_night column is 45
numbers of unique values in last_name column is 4784
```

[ ]:
```
#CHECKING UNIQUE VALUES IN HOTEL COLUMN
hotel_booking_df['hotel'].unique()
```

[ ]:
```
array(['Resort Hotel', 'City Hotel'], dtype=object)
```

[ ]:
```
#CHECKING UNIQUE VALUES IN IS_CANCELED COLUMN
hotel_booking_df['is_canceled'].unique()
```

[ ]:
```
array([0, 1])
```

[ ]:
```
#CHECKING UNIQUE VALUES IN LEAD_TIME COLUN
hotel_booking_df['lead_time'].unique()
```

[ ]:
```
array([342, 737,   7,  13,  14,   0,   9,  85,  75,  23,  35,  68,  18,
        37,  12,  72, 127,  78,  48,  60,  77,  99, 118,  95,  96,  69,
        45,  40,  15,  36,  43,  70,  16, 107,  47, 113,  90,  50,  93,
        76,   3,   1,  10,   5,  17,  51,  71,  63,  62, 101,   2,  81,
       368, 364, 324,  79,  21, 109, 102,   4,  98,  92,  26,  73, 115,
        86,  52,  29,  30,  33,  32,   8, 100,  44,  80,  97,  64,  39,
        34,  27,  82,  94, 110, 111,  84,  66, 104,  28, 258, 112,  65,
        67,  55,  88,  54, 292,  83, 105, 280, 394,  24, 103, 366, 249,
        22,  91,  11, 108, 106,  31,  87,  41, 304, 117,  59,  53,  58,
       116,  42, 321,  38,  56,  49, 317,   6,  57,  19,  25, 315, 123,
        46,  89,  61, 312, 299, 130,  74, 298, 119,  20, 286, 136, 129,
       124, 327, 131, 460, 140, 114, 139, 122, 137, 126, 120, 128, 135,
       150, 143, 151, 132, 125, 157, 147, 138, 156, 164, 346, 159, 160,
       161, 333, 381, 149, 154, 297, 163, 314, 155, 323, 340, 356, 142,
       328, 144, 336, 248, 302, 175, 344, 382, 146, 170, 166, 338, 167,
       310, 148, 165, 172, 171, 145, 121, 178, 305, 173, 152, 354, 347,
       158, 185, 349, 183, 352, 177, 200, 192, 361, 207, 174, 330, 134,
       350, 334, 283, 153, 197, 133, 241, 193, 235, 194, 261, 260, 216,
       169, 209, 238, 215, 141, 189, 187, 223, 284, 214, 202, 211, 168,
       230, 203, 188, 232, 709, 219, 162, 196, 190, 259, 228, 176, 250,
       201, 186, 199, 180, 206, 205, 224, 222, 182, 210, 275, 212, 229,
```

```
       218, 208, 191, 181, 179, 246, 255, 226, 288, 253, 252, 262, 236,
       256, 234, 254, 468, 213, 237, 198, 195, 239, 263, 265, 274, 217,
       220, 307, 221, 233, 257, 227, 276, 225, 264, 311, 277, 204, 290,
       266, 270, 294, 319, 282, 251, 322, 291, 269, 240, 271, 184, 231,
       268, 247, 273, 300, 301, 267, 244, 306, 293, 309, 272, 242, 295,
       285, 243, 308, 398, 303, 245, 424, 279, 331, 281, 339, 434, 357,
       325, 329, 278, 332, 343, 345, 360, 348, 367, 353, 373, 374, 406,
       400, 326, 379, 399, 316, 341, 320, 385, 355, 363, 358, 296, 422,
       390, 335, 370, 376, 375, 397, 289, 542, 403, 383, 384, 359, 393,
       337, 362, 365, 435, 386, 378, 313, 351, 287, 471, 462, 411, 450,
       318, 372, 371, 454, 532, 445, 389, 388, 407, 443, 437, 451, 391,
       405, 412, 419, 420, 426, 433, 440, 429, 418, 447, 461, 605, 457,
       475, 464, 482, 626, 489, 496, 503, 510, 517, 524, 531, 538, 545,
       552, 559, 566, 573, 580, 587, 594, 601, 608, 615, 622, 629, 396,
       410, 395, 423, 408, 409, 448, 465, 387, 414, 476, 479, 467, 490,
       493, 478, 504, 507, 458, 518, 521, 377, 444, 380, 463])
```

[ ]:
```
#CHECKING UNIQUE VALUES IN ARRIVAL_DATE_YEAR COLUMN
hotel_booking_df['arrival_date_year'].unique()
```

[ ]:
```
array([2015, 2016, 2017])
```

[ ]:
```
#CHECKING UNIQUE VALUES IN ARRIVAL_DATE_MONTH COLUMN
hotel_booking_df['arrival_date_month'].unique()
```

[ ]:
```
array(['July', 'August', 'September', 'October', 'November', 'December',
       'January', 'February', 'March', 'April', 'May', 'June'],
      dtype=object)
```

[ ]:
```
#CHECKING UNIQUE VALUES IN ARRIVAL_DATE_WEEK_NUMBER COLUMN
hotel_booking_df['arrival_date_week_number'].unique()
```

[ ]:
```
array([27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43,
       44, 45, 46, 47, 48, 49, 50, 51, 52, 53,  1,  2,  3,  4,  5,  6,  7,
        8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24,
       25, 26])
```

[ ]:
```
#CHECKING UNIQUE  VALUES IN ARRIVAL_DATE_DAY_OF_MONTH COLUMN
hotel_booking_df['arrival_date_day_of_month'].unique()
```

[ ]:
```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17,
       18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31])
```

[ ]:
```
#CHECKING UNIQUE VALUES IN STAYS_IN_WEEKEND_NIGHTS COLUMN
hotel_booking_df['stays_in_weekend_nights'].unique()
```

[ ]:
```
array([ 0,  1,  2,  4,  3,  6, 13,  8,  5,  7, 12,  9, 16, 18, 19, 10, 14])
```

```
[ ]: #CHECKING UNIQUE VALUES IN STAYS_IN_WEEK_NIGHTS COLUMN
     hotel_booking_df['stays_in_week_nights'].unique()
```

```
[ ]: array([ 0,  1,  2,  3,  4,  5, 10, 11,  8,  6,  7, 15,  9, 12, 33, 20, 14,
            16, 21, 13, 30, 19, 24, 40, 22, 42, 50, 25, 17, 32, 26, 18, 34, 35,
            41])
```

```
[ ]: #CHECKING UNIQUE VALUES IN ADULTS COLUMN
     hotel_booking_df['adults'].unique()
```

```
[ ]: array([ 2,  1,  3,  4, 40, 26, 50, 27, 55,  0, 20,  6,  5, 10])
```

```
[ ]: #CHECKING UNIQUE VALUES IN CHILDREN COLUMN
     hotel_booking_df['children'].unique()
```

```
[ ]: array([ 0.,  1.,  2., 10.,  3., nan])
```

```
[ ]: #CHECKING UNIQUE VALUES IN BABIES COLUMN
     hotel_booking_df['babies'].unique()
```

```
[ ]: array([ 0,  1,  2, 10,  9])
```

```
[ ]: #CHECKING UNIQUE  VALUES IN MEAL COLUMN
     hotel_booking_df['meal'].unique()
```

```
[ ]: array(['BB', 'FB', 'HB', 'SC', 'Undefined'], dtype=object)
```

```
[ ]: #CHECKING UNIQUE VALUES IN COUNTRY COLUMN
     hotel_booking_df['country'].unique()
```

```
[ ]: array(['PRT', 'GBR', 'USA', 'ESP', 'IRL', 'FRA', nan, 'ROU', 'NOR', 'OMN',
            'ARG', 'POL', 'DEU', 'BEL', 'CHE', 'CN', 'GRC', 'ITA', 'NLD',
            'DNK', 'RUS', 'SWE', 'AUS', 'EST', 'CZE', 'BRA', 'FIN', 'MOZ',
            'BWA', 'LUX', 'SVN', 'ALB', 'IND', 'CHN', 'MEX', 'MAR', 'UKR',
            'SMR', 'LVA', 'PRI', 'SRB', 'CHL', 'AUT', 'BLR', 'LTU', 'TUR',
            'ZAF', 'AGO', 'ISR', 'CYM', 'ZMB', 'CPV', 'ZWE', 'DZA', 'KOR',
            'CRI', 'HUN', 'ARE', 'TUN', 'JAM', 'HRV', 'HKG', 'IRN', 'GEO',
            'AND', 'GIB', 'URY', 'JEY', 'CAF', 'CYP', 'COL', 'GGY', 'KWT',
            'NGA', 'MDV', 'VEN', 'SVK', 'FJI', 'KAZ', 'PAK', 'IDN', 'LBN',
            'PHL', 'SEN', 'SYC', 'AZE', 'BHR', 'NZL', 'THA', 'DOM', 'MKD',
            'MYS', 'ARM', 'JPN', 'LKA', 'CUB', 'CMR', 'BIH', 'MUS', 'COM',
            'SUR', 'UGA', 'BGR', 'CIV', 'JOR', 'SYR', 'SGP', 'BDI', 'SAU',
            'VNM', 'PLW', 'QAT', 'EGY', 'PER', 'MLT', 'MWI', 'ECU', 'MDG',
            'ISL', 'UZB', 'NPL', 'BHS', 'MAC', 'TGO', 'TWN', 'DJI', 'STP',
            'KNA', 'ETH', 'IRQ', 'HND', 'RWA', 'KHM', 'MCO', 'BGD', 'IMN',
            'TJK', 'NIC', 'BEN', 'VGB', 'TZA', 'GAB', 'GHA', 'TMP', 'GLP',
            'KEN', 'LIE', 'GNB', 'MNE', 'UMI', 'MYT', 'FRO', 'MMR', 'PAN',
```

```
       'BFA', 'LBY', 'MLI', 'NAM', 'BOL', 'PRY', 'BRB', 'ABW', 'AIA',
       'SLV', 'DMA', 'PYF', 'GUY', 'LCA', 'ATA', 'GTM', 'ASM', 'MRT',
       'NCL', 'KIR', 'SDN', 'ATF', 'SLE', 'LAO'], dtype=object)
```

[ ]:
```
#CHECKING UNIQUE VALUES IN MARKET_SEGMENT COLUMN
hotel_booking_df['market_segment'].unique()
```

[ ]:
```
array(['Direct', 'Corporate', 'Online TA', 'Offline TA/TO',
       'Complementary', 'Groups', 'Undefined', 'Aviation'], dtype=object)
```

[ ]:
```
#CHECKING UNIQUE VALUES IN DISTRIBUTION_CHANNEL COLUMN
hotel_booking_df['distribution_channel'].unique()
```

[ ]:
```
array(['Direct', 'Corporate', 'TA/TO', 'Undefined', 'GDS'], dtype=object)
```

[ ]:
```
#CHECKING UNIQUE VALUES IN IS_REPEATED_GUEST COLUMN
hotel_booking_df['is_repeated_guest'].unique()
```

[ ]:
```
array([0, 1])
```

[ ]:
```
#CHECKING UNIQUE VALUES IN RESERVED_ROOM_TYPE COLUMN
hotel_booking_df['reserved_room_type'].unique()
```

[ ]:
```
array(['C', 'A', 'D', 'E', 'G', 'F', 'H', 'L', 'P', 'B'], dtype=object)
```

[ ]:
```
#CHECKING UNIQUE VALUES IN ASSIGNED_ROOM_TYPE COLUMN
hotel_booking_df['assigned_room_type'].unique()
```

[ ]:
```
array(['C', 'A', 'D', 'E', 'G', 'F', 'I', 'B', 'H', 'P', 'L', 'K'],
      dtype=object)
```

[ ]:
```
#CHECKING UNIQUE VALUES IN BOOKING_CHANGES COLUMN
hotel_booking_df['booking_changes'].unique()
```

[ ]:
```
array([ 3,  4,  0,  1,  2,  5, 17,  6,  8,  7, 10, 16,  9, 13, 12, 20, 14,
       15, 11, 21, 18])
```

[ ]:
```
#CHECKING UNIQUE VALUES IN DEPOSIT_TYPE COLUMN
hotel_booking_df['deposit_type'].unique()
```

[ ]:
```
array(['No Deposit', 'Refundable', 'Non Refund'], dtype=object)
```

[ ]:
```
#CHECKING UNIQUE VALUES IN AGENT COLUMN
hotel_booking_df['agent'].unique()
```

[ ]:
```
array([ nan, 304., 240., 303.,  15., 241.,   8., 250., 115.,   5., 175.,
       134., 156., 243., 242.,   3., 105.,  40., 147., 306., 184.,  96.,
```

```
  2., 127.,  95., 146.,   9., 177.,   6., 143., 244., 149., 167.,
300., 171., 305.,  67., 196., 152., 142., 261., 104.,  36.,  26.,
 29., 258., 110.,  71., 181.,  88., 251., 275.,  69., 248., 208.,
256., 314., 126., 281., 273., 253., 185., 330., 334., 328., 326.,
321., 324., 313.,  38., 155.,  68., 335., 308., 332.,  94., 348.,
310., 339., 375.,  66., 327., 387., 298.,  91., 245., 385., 257.,
393., 168., 405., 249., 315.,  75., 128., 307.,  11., 436.,   1.,
201., 183., 223., 368., 336., 291., 464., 411., 481.,  10., 154.,
468., 410., 390., 440., 495., 492., 493., 434.,  57., 531., 420.,
483., 526., 472., 429.,  16., 446.,  34.,  78., 139., 252., 270.,
 47., 114., 301., 193., 182., 135., 350., 195., 352., 355., 159.,
363., 384., 360., 331., 367.,  64., 406., 163., 414., 333., 427.,
431., 430., 426., 438., 433., 418., 441., 282., 432.,  72., 450.,
180., 454., 455.,  59., 451., 254., 358., 469., 165., 467., 510.,
337., 476., 502., 527., 479., 508., 535., 302., 497., 187.,  13.,
  7.,  27.,  14.,  22.,  17.,  28.,  42.,  20.,  19.,  45.,  37.,
 61.,  39.,  21.,  24.,  41.,  50.,  30.,  54.,  52.,  12.,  44.,
 31.,  83.,  32.,  63.,  60.,  55.,  56.,  89.,  87., 118.,  86.,
 85., 210., 214., 129., 179., 138., 174., 170., 153.,  93., 151.,
119.,  35., 173.,  58.,  53., 133.,  79., 235., 192., 191., 236.,
162., 215., 157., 287., 132., 234.,  98.,  77., 103., 107., 262.,
220., 121., 205., 378.,  23., 296., 290., 229.,  33., 286., 276.,
425., 484., 323., 403., 219., 394., 509., 111., 423.,   4.,  70.,
 82.,  81.,  74.,  92.,  99.,  90., 112., 117., 106., 148., 158.,
144., 211., 213., 216., 232., 150., 267., 227., 247., 278., 280.,
285., 289., 269., 295., 265., 288., 122., 294., 325., 341., 344.,
346., 359., 283., 364., 370., 371.,  25., 141., 391., 397., 416.,
404., 299., 197.,  73., 354., 444., 408., 461., 388., 453., 459.,
474., 475., 480., 449.])
```

[ ]:
```
#CHECKING UNIQUE VALUES IN COMPANY COLUMN
hotel_booking_df['company'].unique()
```

[ ]:
```
array([ nan, 110., 113., 270., 178., 240., 154., 144., 307., 268.,  59.,
       204., 312., 318.,  94., 174., 274., 195., 223., 317., 281., 118.,
        53., 286.,  12.,  47., 324., 342., 373., 371., 383.,  86.,  82.,
       218.,  88.,  31., 397., 392., 405., 331., 367.,  20.,  83., 416.,
        51., 395., 102.,  34.,  84., 360., 394., 457., 382., 461., 478.,
       386., 112., 486., 421.,   9., 308., 135., 224., 504., 269., 356.,
       498., 390., 513., 203., 263., 477., 521., 169., 515., 445., 337.,
       251., 428., 292., 388., 130., 250., 355., 254., 543., 531., 528.,
        62., 120.,  42.,  81., 116., 530., 103.,  39.,  16.,  92.,  61.,
       501., 165., 291., 290.,  43., 325., 192., 108., 200., 465., 287.,
       297., 490., 482., 207., 282., 437., 225., 329., 272.,  28.,  77.,
       338.,  72., 246., 319., 146., 159., 380., 323., 511., 407., 278.,
        80., 403., 399.,  14., 137., 343., 346., 347., 349., 289., 351.,
       353.,  54.,  99., 358., 361., 362., 366., 372., 365., 277., 109.,
```

```
       377., 379.,  22., 378., 330., 364., 401., 232., 255., 384., 167.,
       212., 514., 391., 400., 376., 402., 396., 302., 398.,   6., 370.,
       369., 409., 168., 104., 408., 413., 148.,  10., 333., 419., 415.,
       424., 425., 423., 422., 435., 439., 442., 448., 443., 454., 444.,
        52., 459., 458., 456., 460., 447., 470., 466., 484., 184., 485.,
        32., 487., 491., 494., 193., 516., 496., 499.,  29.,  78., 520.,
       507., 506., 512., 126.,  64., 242., 518., 523., 539., 534., 436.,
       525., 541.,  40., 455., 410.,  45.,  38.,  49.,  48.,  67.,  68.,
        65.,  91.,  37.,   8., 179., 209., 219., 221., 227., 153., 186.,
       253., 202., 216., 275., 233., 280., 309., 321.,  93., 316.,  85.,
       107., 350., 279., 334., 348., 150.,  73., 385., 418., 197., 450.,
       452., 115.,  46.,  76.,  96., 100., 105., 101., 122.,  11., 139.,
       142., 127., 143., 140., 149., 163., 160., 180., 238., 183., 222.,
       185., 217., 215., 213., 237., 230., 234.,  35., 245., 158., 258.,
       259., 260., 411., 257., 271.,  18., 106., 210., 273.,  71., 284.,
       301., 305., 293., 264., 311., 304., 313., 288., 320., 314., 332.,
       341., 352., 243., 368., 393., 132., 220., 412., 420., 426., 417.,
       429., 433., 446., 357., 479., 483., 489., 229., 481., 497., 451.,
       492.])
```

[ ]:
```
#CHECKING UNIQUE VALUES IN DAYS_IN_WAITING_LIST COLUMN
hotel_booking_df['days_in_waiting_list'].unique()
```

[ ]:
```
array([  0,  50,  47,  65, 122,  75, 101, 150, 125,  14,  60,  34, 100,
        22, 121,  61,  39,   5,   1,   8, 107,  43,  52,   2,  11, 142,
       116,  13,  44,  97,  83,   4, 113,  18,  20, 185,  93, 109,   6,
        37, 105, 154,  64,  99,  38,  48,  33,  77,  21,  80,  59,  40,
        58,  89,  53,  49,  69,  87,  91,  57, 111,  79,  98,  85,  63,
        15,   3,  41, 224,  31,  56, 187, 176,  71,  55,  96, 236, 259,
       207, 215, 160, 120,  30,  32,  27,  62,  24, 108, 147, 379,  70,
        35, 178, 330, 223, 174, 162, 391,  68, 193,  10,  76,  16,  28,
         9, 165,  17,  25,  46,   7,  84, 175, 183,  23, 117,  12,  54,
        26,  73,  45,  19,  42,  72,  81,  92,  74, 167,  36])
```

[ ]:
```
#CHECKING UNIQUE VALUES IN CUSTOMER_TYPE COLUMN
hotel_booking_df['customer_type'].unique()
```

[ ]:
```
array(['Transient', 'Contract', 'Transient-Party', 'Group'], dtype=object)
```

[ ]:
```
#CHECKING UNIQUE VALUES IN REQUIRED_CAR_PARKING_SPACES COLUMN
hotel_booking_df['required_car_parking_spaces'].unique()
```

[ ]:
```
array([0, 1, 2, 8, 3])
```

[ ]:
```
#CHECKING UNIQUE VALUES IN RESERVATION_STATUS COLUMN
hotel_booking_df['reservation_status'].unique()
```

[ ]:
```
array(['Check-Out', 'Canceled', 'No-Show'], dtype=object)
```

[ ]:
```
#CHECKING UNIQUE VALUES IN RESERVATION_STATUS_DATE COLUMN
hotel_booking_df['reservation_status_date'].unique()
```

[ ]:
```
array(['2015-07-01', '2015-07-02', '2015-07-03', '2015-05-06',
       '2015-04-22', '2015-06-23', '2015-07-05', '2015-07-06',
       '2015-07-07', '2015-07-08', '2015-05-11', '2015-07-15',
       '2015-07-16', '2015-05-29', '2015-05-19', '2015-06-19',
       '2015-05-23', '2015-05-18', '2015-07-09', '2015-06-02',
       '2015-07-13', '2015-07-04', '2015-06-29', '2015-06-16',
       '2015-06-18', '2015-06-12', '2015-06-09', '2015-05-26',
       '2015-07-11', '2015-07-12', '2015-07-17', '2015-04-15',
       '2015-05-13', '2015-07-10', '2015-05-20', '2015-05-12',
       '2015-07-14', '2015-06-17', '2015-05-01', '2015-03-30',
       '2015-07-19', '2015-06-03', '2015-06-26', '2015-05-14',
       '2015-07-20', '2015-05-07', '2015-05-28', '2015-04-13',
       '2015-03-25', '2015-07-21', '2015-06-27', '2015-07-18',
       '2015-07-23', '2015-06-08', '2015-06-22', '2015-06-24',
       '2015-03-05', '2015-06-01', '2015-04-24', '2015-07-22',
       '2015-05-27', '2015-04-06', '2015-04-11', '2015-07-25',
       '2015-07-28', '2015-07-29', '2015-06-25', '2015-07-24',
       '2015-06-05', '2015-06-30', '2015-06-13', '2015-06-11',
       '2015-07-30', '2015-07-27', '2015-04-29', '2015-06-04',
       '2015-07-26', '2015-08-01', '2015-08-02', '2015-06-15',
       '2015-04-23', '2015-07-31', '2015-05-25', '2015-08-03',
       '2015-04-17', '2015-08-04', '2015-08-06', '2015-05-15',
       '2015-05-09', '2015-03-17', '2015-05-22', '2015-08-07',
       '2015-04-04', '2015-08-05', '2015-08-08', '2015-08-10',
       '2015-05-04', '2015-06-06', '2015-08-09', '2015-08-15',
       '2015-08-11', '2015-03-28', '2015-08-14', '2015-08-12',
       '2015-08-16', '2015-05-16', '2015-08-21', '2015-08-13',
       '2015-08-17', '2015-04-20', '2015-08-18', '2015-08-23',
       '2015-08-22', '2015-08-19', '2015-08-20', '2015-08-29',
       '2015-03-31', '2015-05-30', '2015-08-25', '2015-04-14',
       '2015-08-24', '2015-03-24', '2015-05-21', '2015-08-28',
       '2015-08-26', '2015-08-27', '2015-08-30', '2015-08-31',
       '2015-09-06', '2015-09-03', '2015-09-04', '2015-09-02',
       '2015-09-01', '2015-09-05', '2015-06-20', '2015-09-07',
       '2015-09-10', '2015-09-11', '2015-09-08', '2015-09-09',
       '2015-09-13', '2015-09-15', '2015-04-10', '2015-01-02',
       '2014-11-18', '2015-09-12', '2015-09-17', '2015-09-14',
       '2015-04-07', '2015-09-19', '2015-09-16', '2015-09-20',
       '2015-01-18', '2015-10-23', '2015-01-22', '2015-01-01',
       '2015-09-22', '2015-09-24', '2015-09-18', '2015-09-21',
       '2015-09-30', '2015-09-25', '2015-09-27', '2015-09-28',
       '2015-10-12', '2015-09-29', '2015-09-23', '2015-10-01',
```

```
'2015-09-26', '2015-04-18', '2015-10-02', '2015-10-04',
'2015-10-08', '2015-10-03', '2015-10-07', '2015-10-09',
'2015-10-11', '2015-10-05', '2015-10-06', '2015-10-10',
'2015-10-14', '2015-10-15', '2015-10-18', '2015-10-13',
'2015-10-20', '2015-10-19', '2015-10-31', '2015-10-16',
'2015-10-21', '2015-10-22', '2015-10-17', '2015-10-24',
'2015-10-25', '2015-10-28', '2015-10-27', '2015-10-26',
'2015-10-30', '2015-11-05', '2015-10-29', '2015-11-03',
'2015-11-07', '2015-11-04', '2015-11-01', '2015-11-02',
'2015-11-17', '2015-11-06', '2015-11-10', '2015-11-08',
'2015-11-09', '2015-11-15', '2015-11-16', '2015-11-11',
'2015-11-12', '2015-11-14', '2015-11-13', '2015-11-18',
'2015-11-22', '2015-11-19', '2015-11-21', '2015-11-20',
'2015-11-24', '2015-11-25', '2015-11-23', '2015-11-28',
'2015-11-26', '2015-11-27', '2015-11-29', '2015-12-04',
'2015-12-01', '2015-12-06', '2015-12-08', '2015-12-02',
'2015-12-03', '2015-12-31', '2015-12-05', '2015-12-10',
'2015-12-17', '2015-11-30', '2015-12-12', '2015-12-07',
'2016-01-05', '2015-12-11', '2015-12-13', '2015-12-15',
'2015-12-16', '2015-12-19', '2015-12-18', '2015-12-26',
'2015-12-27', '2015-12-22', '2015-12-23', '2015-12-24',
'2015-12-29', '2015-12-28', '2015-12-20', '2015-12-30',
'2016-01-02', '2016-01-01', '2015-12-25', '2016-01-03',
'2016-01-04', '2016-01-11', '2016-01-07', '2015-12-21',
'2016-01-09', '2016-01-10', '2016-01-08', '2016-01-06',
'2016-01-12', '2016-01-13', '2016-01-23', '2016-02-09',
'2016-01-15', '2016-01-16', '2016-01-17', '2016-01-19',
'2016-01-18', '2016-01-21', '2016-01-24', '2016-01-22',
'2016-01-29', '2016-01-27', '2016-01-25', '2016-03-08',
'2016-01-26', '2016-01-20', '2016-01-30', '2016-02-01',
'2016-02-02', '2016-02-08', '2016-02-07', '2016-01-28',
'2016-02-05', '2016-02-03', '2016-02-13', '2016-02-10',
'2016-02-04', '2016-02-12', '2016-02-11', '2016-02-16',
'2016-02-14', '2016-02-15', '2016-02-20', '2016-02-06',
'2016-01-14', '2016-02-17', '2016-02-21', '2016-02-24',
'2016-02-25', '2016-02-19', '2016-02-18', '2016-02-26',
'2016-02-23', '2016-03-05', '2016-02-22', '2016-02-27',
'2016-03-03', '2016-03-24', '2016-03-04', '2016-02-29',
'2016-03-01', '2016-03-02', '2016-03-30', '2016-03-07',
'2016-03-14', '2016-03-21', '2016-03-09', '2016-03-12',
'2016-03-22', '2016-03-10', '2016-03-11', '2016-03-20',
'2016-03-15', '2016-03-17', '2016-03-16', '2016-03-19',
'2016-03-27', '2016-03-18', '2016-03-26', '2016-03-31',
'2016-03-28', '2016-03-29', '2016-04-01', '2016-03-23',
'2016-04-02', '2016-03-25', '2016-03-13', '2016-04-04',
'2016-04-03', '2016-04-05', '2016-04-08', '2016-04-06',
'2016-04-09', '2016-04-12', '2016-04-16', '2016-04-17',
```

```
'2016-04-27', '2016-04-14', '2016-04-18', '2016-04-21',
'2016-04-19', '2016-04-20', '2016-04-10', '2016-04-13',
'2016-04-11', '2016-04-07', '2016-04-15', '2016-04-22',
'2016-04-23', '2016-04-26', '2016-04-28', '2016-04-24',
'2016-04-25', '2016-04-29', '2016-04-30', '2016-05-01',
'2016-05-10', '2016-05-02', '2016-05-07', '2016-05-08',
'2016-05-12', '2016-05-04', '2016-05-06', '2016-05-03',
'2016-05-09', '2016-05-05', '2016-05-13', '2016-05-14',
'2016-05-18', '2016-05-19', '2016-05-15', '2016-05-16',
'2016-05-11', '2016-05-21', '2016-05-22', '2016-05-20',
'2016-05-24', '2016-05-25', '2016-05-26', '2016-05-23',
'2016-05-27', '2016-05-17', '2016-05-29', '2016-05-28',
'2016-05-30', '2016-05-31', '2016-06-01', '2016-06-03',
'2016-06-08', '2016-06-02', '2016-06-05', '2016-06-06',
'2016-06-13', '2016-06-07', '2016-06-10', '2016-06-11',
'2016-06-16', '2016-06-12', '2016-06-14', '2016-06-17',
'2016-06-04', '2016-06-18', '2016-06-21', '2016-06-09',
'2016-06-24', '2016-06-20', '2016-06-25', '2016-06-22',
'2016-06-26', '2016-06-23', '2016-07-01', '2016-06-15',
'2016-06-28', '2016-07-02', '2016-06-19', '2016-06-27',
'2016-07-04', '2016-06-30', '2016-07-05', '2016-07-08',
'2016-07-09', '2016-07-07', '2016-07-12', '2016-06-29',
'2016-07-10', '2016-07-15', '2016-07-03', '2016-07-16',
'2016-07-14', '2016-07-18', '2016-07-13', '2016-07-06',
'2016-07-20', '2016-07-21', '2016-07-23', '2016-07-19',
'2016-07-11', '2016-07-28', '2016-07-17', '2016-07-25',
'2016-07-22', '2016-07-29', '2016-08-03', '2016-08-02',
'2016-08-04', '2016-08-08', '2016-08-10', '2016-08-01',
'2016-08-06', '2016-03-06', '2016-08-05', '2016-07-26',
'2016-08-07', '2016-07-30', '2016-07-24', '2016-08-12',
'2016-07-27', '2016-08-13', '2016-08-18', '2016-08-16',
'2016-08-15', '2016-08-17', '2016-08-11', '2016-07-31',
'2016-08-19', '2016-09-01', '2016-08-23', '2016-08-26',
'2016-08-20', '2016-08-21', '2016-09-04', '2016-08-22',
'2016-08-27', '2016-08-25', '2016-08-09', '2016-09-05',
'2016-08-24', '2016-09-10', '2016-08-29', '2016-09-09',
'2016-08-30', '2016-09-13', '2016-08-31', '2016-09-14',
'2016-09-12', '2016-09-15', '2016-08-14', '2016-09-02',
'2016-09-08', '2016-09-19', '2016-09-16', '2016-09-07',
'2016-09-21', '2016-09-06', '2016-09-22', '2016-09-17',
'2016-09-20', '2016-09-03', '2016-09-26', '2016-09-23',
'2016-09-18', '2016-09-29', '2016-10-02', '2016-10-01',
'2016-09-27', '2016-09-25', '2016-10-05', '2016-09-11',
'2016-09-30', '2016-10-09', '2016-10-03', '2016-10-06',
'2016-10-11', '2016-09-24', '2016-10-13', '2016-09-28',
'2016-10-08', '2016-10-07', '2016-10-16', '2016-08-28',
'2016-10-17', '2016-10-18', '2016-10-10', '2016-10-04',
```

```
'2016-10-15', '2016-10-19', '2016-10-21', '2016-10-12',
'2016-10-24', '2016-10-26', '2016-10-23', '2016-10-20',
'2016-10-25', '2016-10-27', '2016-10-28', '2016-10-30',
'2016-10-29', '2016-11-01', '2016-11-04', '2016-10-14',
'2016-11-07', '2016-11-03', '2016-11-10', '2016-11-14',
'2016-11-02', '2016-10-31', '2016-11-11', '2016-11-08',
'2016-11-05', '2016-11-25', '2016-11-09', '2016-11-20',
'2016-11-21', '2016-10-22', '2016-11-22', '2016-11-16',
'2016-11-23', '2016-11-17', '2016-11-06', '2016-11-15',
'2016-11-13', '2016-11-12', '2016-11-27', '2016-11-19',
'2016-11-30', '2016-11-18', '2016-12-02', '2016-12-04',
'2016-11-29', '2016-12-07', '2016-11-28', '2016-12-03',
'2016-12-06', '2016-11-24', '2016-12-08', '2016-12-05',
'2016-12-10', '2016-12-13', '2016-12-14', '2016-12-16',
'2016-12-15', '2016-12-17', '2016-12-19', '2016-12-21',
'2016-12-20', '2016-12-22', '2016-12-23', '2016-12-24',
'2016-12-01', '2016-12-27', '2016-12-29', '2016-12-30',
'2016-12-12', '2017-01-02', '2016-12-11', '2017-01-03',
'2017-01-04', '2017-01-01', '2016-12-26', '2017-01-06',
'2016-12-28', '2016-12-18', '2017-01-10', '2017-01-11',
'2017-01-07', '2017-01-12', '2017-01-16', '2017-01-14',
'2017-01-13', '2017-01-05', '2017-01-17', '2017-01-20',
'2016-12-09', '2017-01-26', '2016-12-31', '2017-01-23',
'2017-01-27', '2017-01-28', '2017-01-19', '2017-01-25',
'2017-01-24', '2017-01-29', '2017-01-18', '2016-12-25',
'2017-01-15', '2017-01-21', '2017-02-01', '2017-02-02',
'2017-01-31', '2017-02-03', '2017-02-04', '2017-02-06',
'2017-02-07', '2017-02-08', '2017-01-30', '2017-02-09',
'2017-01-09', '2017-02-11', '2017-02-10', '2017-02-12',
'2017-02-13', '2017-02-14', '2017-02-16', '2017-02-17',
'2017-02-18', '2017-02-19', '2017-02-20', '2017-02-15',
'2017-02-21', '2017-02-22', '2017-02-26', '2017-02-23',
'2017-02-24', '2017-02-25', '2017-02-28', '2017-03-05',
'2017-02-27', '2017-03-03', '2017-03-06', '2017-03-02',
'2017-03-08', '2017-03-09', '2017-03-10', '2017-03-07',
'2017-03-12', '2017-03-13', '2017-03-14', '2017-03-01',
'2017-03-18', '2017-03-17', '2017-03-24', '2017-03-22',
'2017-03-26', '2017-03-27', '2017-03-11', '2017-03-28',
'2017-03-29', '2017-03-30', '2017-03-31', '2017-03-19',
'2017-01-22', '2017-04-02', '2017-03-20', '2017-04-03',
'2017-01-08', '2017-03-23', '2017-04-05', '2017-02-05',
'2017-04-04', '2017-03-15', '2017-04-07', '2017-03-25',
'2017-04-08', '2017-04-06', '2017-03-21', '2017-04-10',
'2017-04-01', '2017-04-11', '2017-04-13', '2017-04-15',
'2017-04-12', '2017-03-04', '2017-04-19', '2017-04-22',
'2017-04-20', '2017-05-02', '2017-04-09', '2017-04-23',
'2017-04-24', '2017-04-16', '2017-04-28', '2017-04-18',
```

```
'2017-04-26', '2017-04-25', '2017-04-17', '2017-04-21',
'2017-05-03', '2017-05-04', '2017-03-16', '2017-05-05',
'2017-04-29', '2017-04-14', '2017-05-08', '2017-04-27',
'2017-05-11', '2017-05-01', '2017-05-10', '2017-05-13',
'2017-05-06', '2017-05-14', '2017-05-16', '2017-04-30',
'2017-05-15', '2017-05-07', '2017-05-09', '2017-05-17',
'2017-05-21', '2017-05-12', '2017-05-22', '2017-05-24',
'2017-05-23', '2017-05-25', '2017-05-26', '2017-05-28',
'2017-05-27', '2017-05-29', '2017-05-19', '2017-05-31',
'2017-05-20', '2017-06-01', '2017-05-30', '2017-06-02',
'2016-11-26', '2017-06-04', '2017-06-05', '2017-06-06',
'2017-06-07', '2017-05-18', '2017-06-09', '2017-06-10',
'2017-06-11', '2017-06-12', '2017-06-14', '2017-06-08',
'2017-06-16', '2017-06-13', '2017-06-03', '2017-06-24',
'2017-06-20', '2017-06-19', '2017-06-21', '2017-06-26',
'2017-06-27', '2017-06-22', '2017-06-28', '2017-06-15',
'2017-06-29', '2017-06-30', '2017-06-18', '2017-07-04',
'2017-07-08', '2017-07-05', '2017-07-03', '2017-07-07',
'2017-07-01', '2017-07-06', '2017-07-11', '2017-07-12',
'2017-06-23', '2017-07-13', '2017-07-02', '2017-07-10',
'2017-07-14', '2017-07-15', '2017-07-16', '2017-07-18',
'2017-07-17', '2017-07-19', '2017-07-20', '2017-07-21',
'2017-06-25', '2017-06-17', '2017-07-24', '2017-07-26',
'2017-07-09', '2017-07-27', '2017-07-28', '2017-07-31',
'2017-07-29', '2017-07-22', '2017-08-02', '2017-08-01',
'2017-08-03', '2017-08-04', '2017-07-25', '2017-07-23',
'2017-08-09', '2017-08-10', '2017-07-30', '2017-08-07',
'2017-08-13', '2017-08-05', '2017-08-14', '2017-08-08',
'2017-08-16', '2017-08-17', '2017-08-15', '2017-08-18',
'2017-08-20', '2017-08-22', '2017-08-06', '2017-08-25',
'2017-08-26', '2017-08-23', '2017-08-11', '2017-08-27',
'2017-08-21', '2017-08-29', '2017-08-31', '2017-08-12',
'2017-08-19', '2016-01-31', '2017-09-01', '2017-08-28',
'2015-04-03', '2015-01-21', '2015-01-28', '2015-01-29',
'2015-01-30', '2015-02-02', '2015-02-05', '2015-02-06',
'2015-02-09', '2015-02-10', '2015-02-11', '2015-02-12',
'2015-02-19', '2015-02-20', '2015-02-23', '2015-02-24',
'2015-02-25', '2015-02-26', '2015-02-27', '2015-03-03',
'2015-03-04', '2015-03-06', '2015-03-09', '2015-03-11',
'2015-03-12', '2015-03-18', '2015-04-02', '2015-06-14',
'2015-04-08', '2015-04-16', '2015-04-25', '2015-04-28',
'2015-05-08', '2017-09-06', '2016-02-28', '2015-12-09',
'2015-12-14', '2017-09-09', '2017-09-02', '2017-08-24',
'2017-08-30', '2017-09-03', '2017-09-04', '2017-09-05',
'2017-09-07', '2017-09-08', '2017-09-10', '2017-09-12',
'2017-09-14', '2015-04-30', '2015-04-21', '2015-04-05',
'2015-03-13', '2015-05-05', '2015-03-29', '2015-06-10',
```

```
       '2015-04-27', '2014-10-17', '2015-01-20', '2015-02-17',
       '2015-03-10', '2015-03-23'], dtype=object)
```

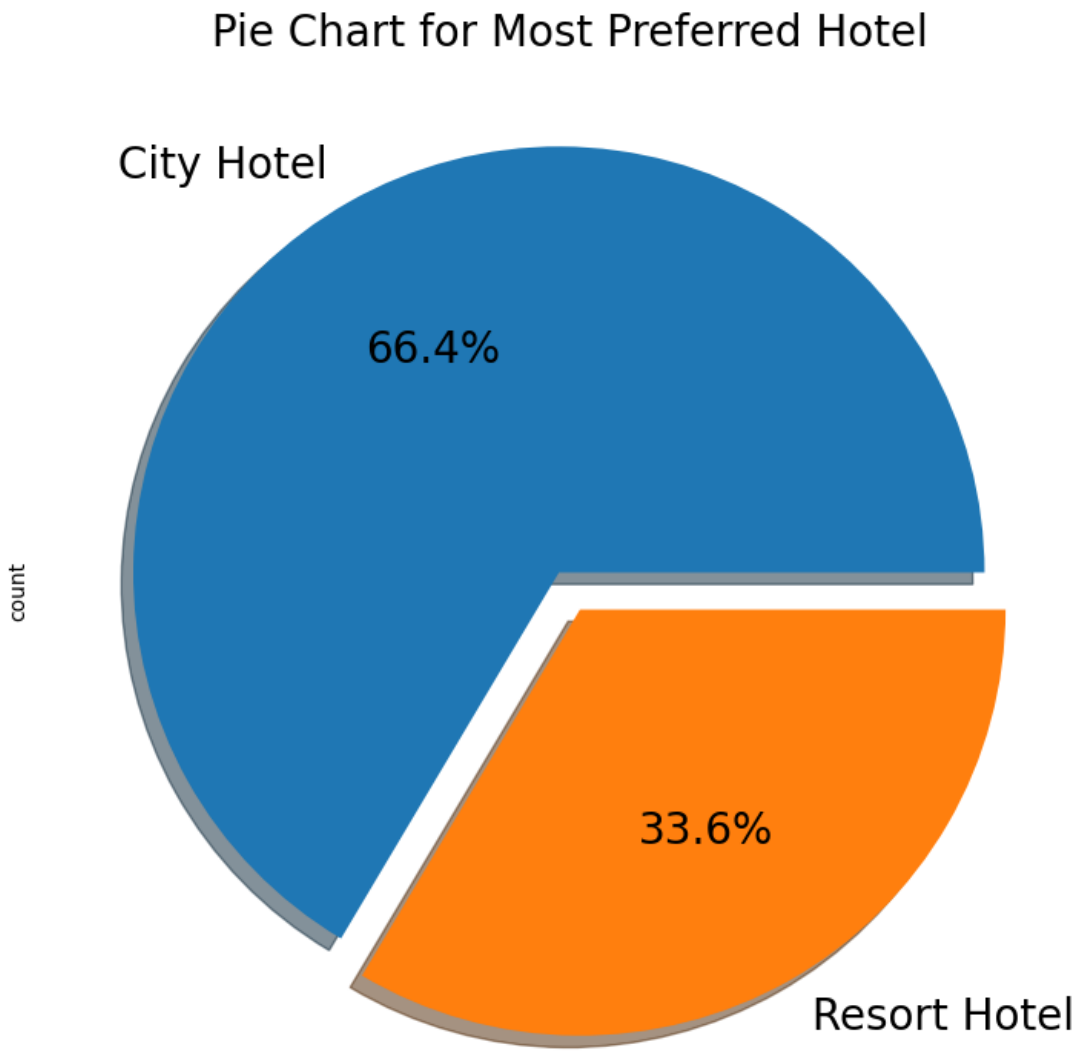
[ ]:
```
#CHECKING FINAL NUMBERS OF ROWWWWS ANND COLUMN
hotel_booking_df.shape
```

[ ]:
```
(119390, 38)
```

16 DATA VISUALISATION,STORYTELLING AND EXPERIMENTING WITH CHARTS

17 CHART1: PIE CHAT FOR MOST PREFERRED HOTEL

[ ]:
```
# Visualizing by pie chart
hotel_booking_df['hotel'].value_counts().plot.pie(explode=[0.05, 0.05], autopct␣
 ↪='%1.1f%%', shadow = True, figsize =(10,9), fontsize = 20)

# Set labels
plt.title('Pie Chart for Most Preferred Hotel', fontsize = 20)

# To show
plt.show()
```

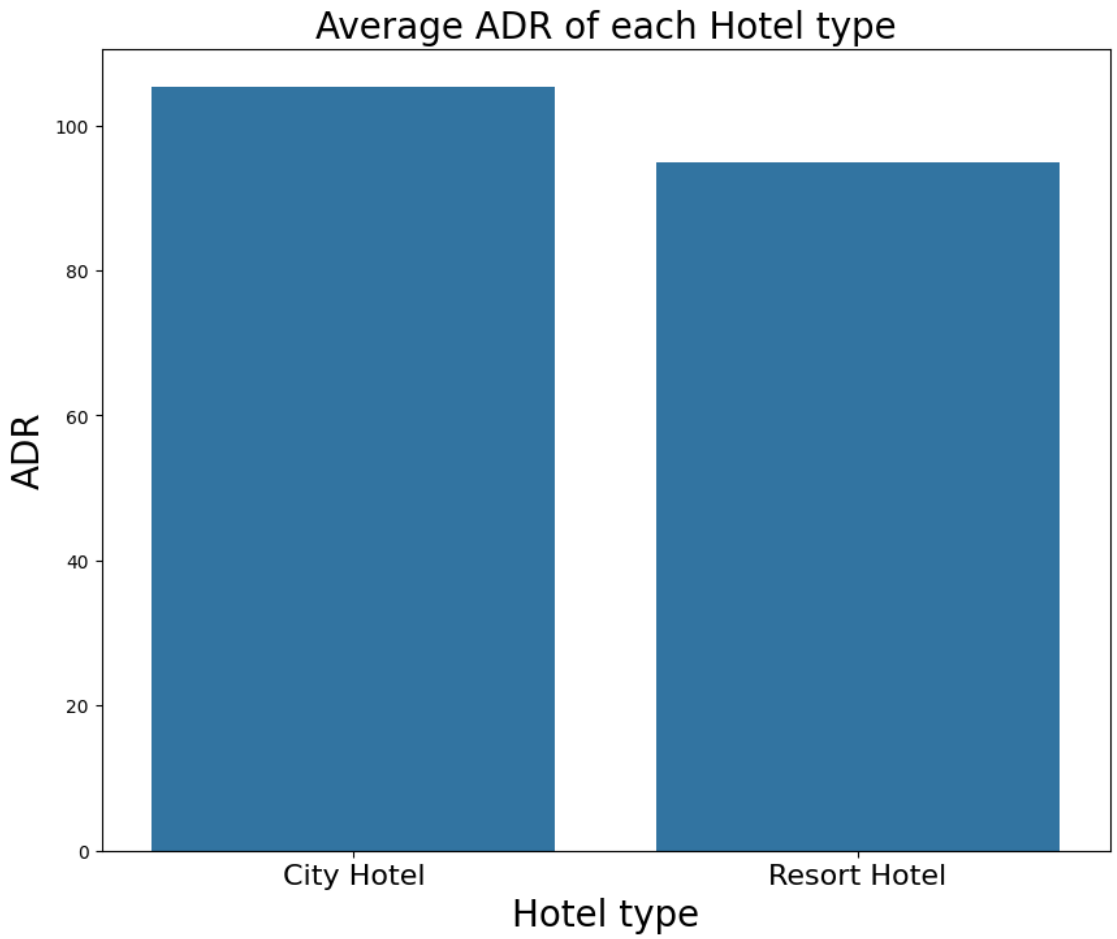


18 CHART-2: HOTEL TYPE WITH HIGHEST ADR(BIVARIATE WITH CATEGOTICAL-NUMERICAL)

[ ]:
```
# Group by Hotel
group_by_hotel = hotel_booking_df.groupby('hotel')

# Grouping by Hotel adr
highest_adr = group_by_hotel['adr'].mean().reset_index()

# Set plot size
plt.figure(figsize = (10,8))
```

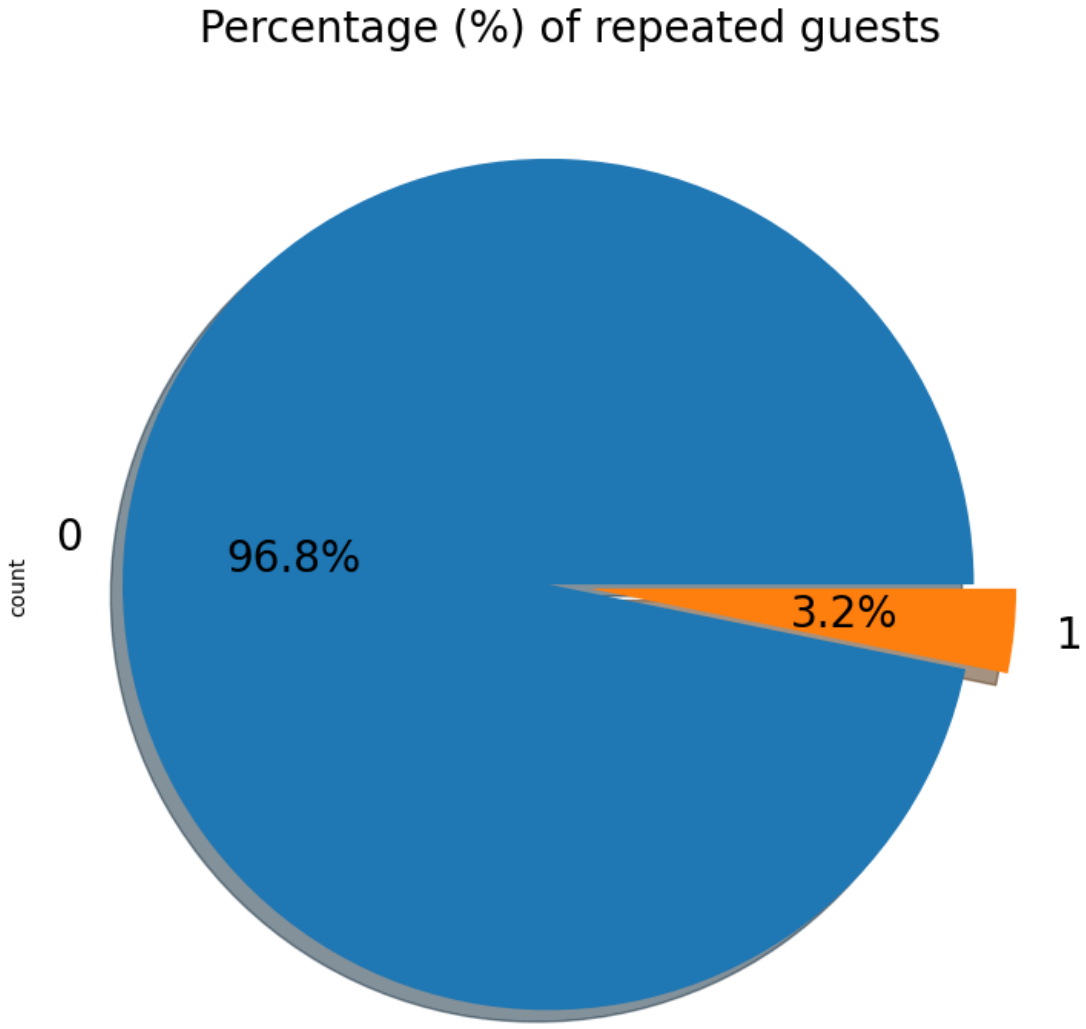
```
# Create the figure object
ax = sns.barplot(x= highest_adr['hotel'], y= highest_adr['adr'])

# Set labels
ax.set_xlabel("Hotel type", fontsize = 20)
ax.set_ylabel("ADR", fontsize = 20)
ax.set_xticklabels(['City Hotel', 'Resort Hotel'], fontsize = 16)
ax.set_title('Average ADR of each Hotel type', fontsize = 20)

# To show
plt.show(ax)
```

```
<ipython-input-84-96636897a9ff>:16: UserWarning: FixedFormatter should only be
used together with FixedLocator
  ax.set_xticklabels(['City Hotel', 'Resort Hotel'], fontsize = 16)
```



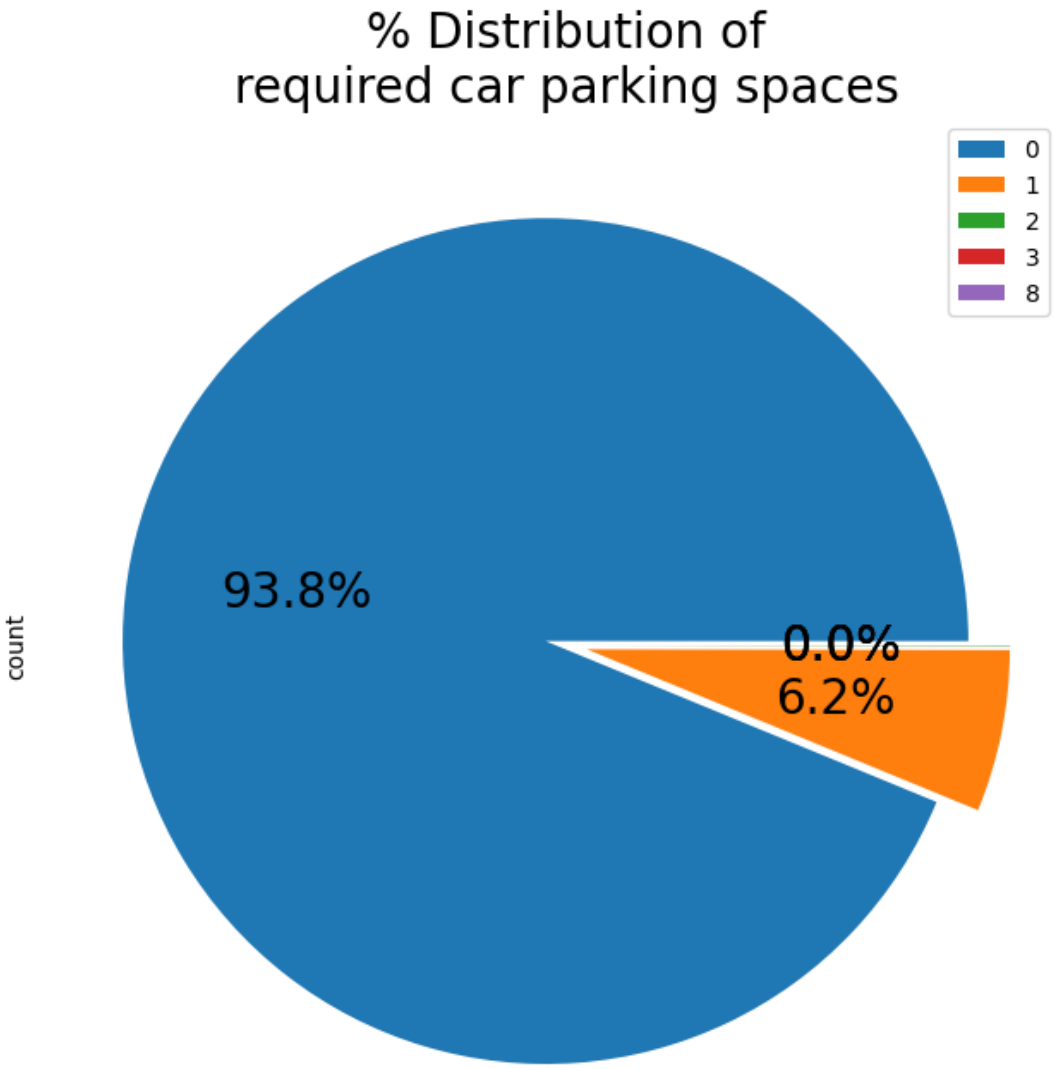
19 CHART-3: PERCENTAGE OF REPEATED GUESTS

[ ]:
```
# Visualizing by pie chart
hotel_booking_df['is_repeated_guest'].value_counts().plot.pie(explode=[0.05, 0.
↪05], autopct ='%1.1f%%', shadow = True, figsize =(10,9), fontsize = 20)

# Set labels
plt.title('Percentage (%) of repeated guests', fontsize = 20)

# To show
plt.show()
```



20 CHART-4: PERCENTAGE DISTRIBUTION OF REQUIRED CAR PARKING SPACES

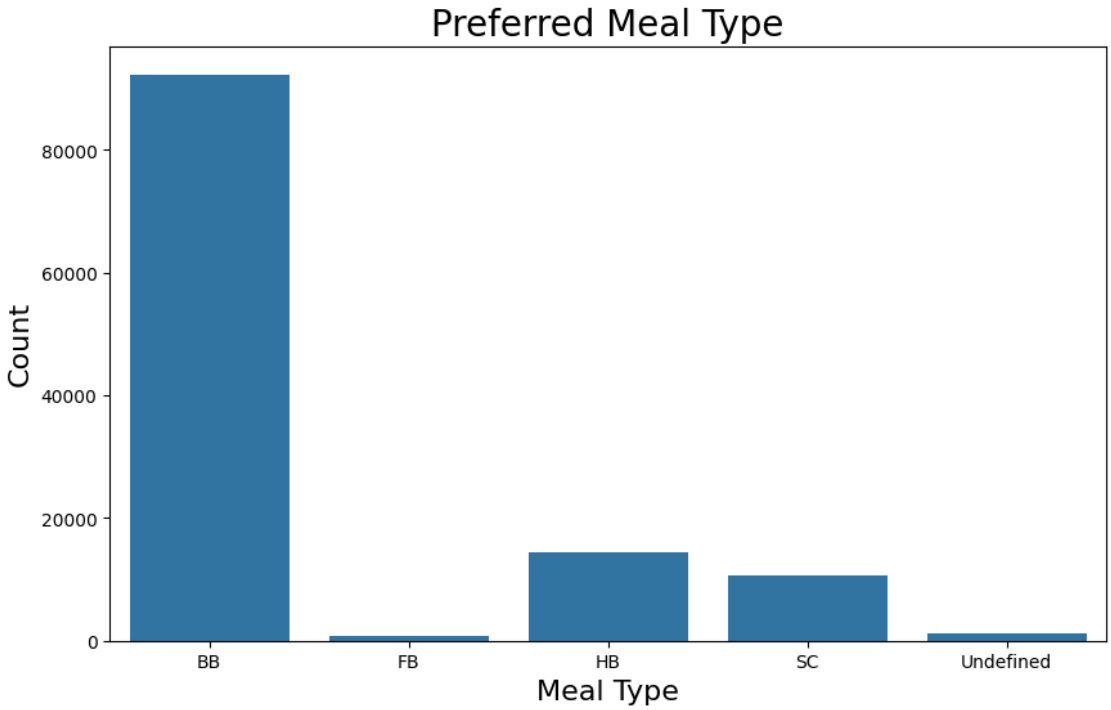
[ ]:
```
# Visualizing by pie chart
hotel_booking_df['required_car_parking_spaces'].value_counts().plot.
 ↪pie(explode=[0.05]*5, autopct ='%1.1f%%', shadow = False, figsize =(12,8),␣
 ↪fontsize = 20, labels = None)

# Create the figure object
labels = hotel_booking_df['required_car_parking_spaces'].value_counts().index

# Set labels
plt.title('% Distribution of\nrequired car parking spaces', fontsize = 20)
plt.legend(bbox_to_anchor = (0.85, 1), loc = 'upper left', labels = labels)

# To show
plt.show()
```



21 CHART-5: MEAL TYPE DISTRIBUTION

[ ]:
```
# Set plot size
plt.figure(figsize=(10,6))

# Create the figure object
sns.countplot(x = hotel_booking_df['meal'])

# Set labels
plt.xlabel('Meal Type', fontsize = 16)
```

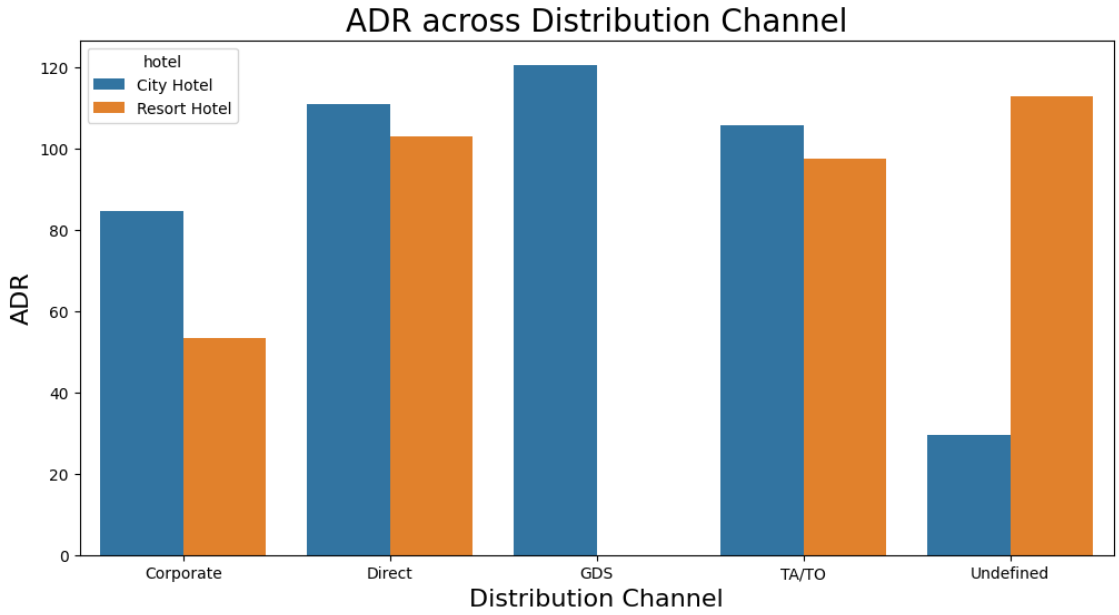
```
plt.ylabel('Count', fontsize = 16)
plt.title('Preferred Meal Type', fontsize = 20)

# To show
plt.show()
```



22 CHART-6: PIE CHART FOR MOSTLY USED DISTRIBUTION CHANNEL AND BAR PLOT FOR RELATIONSHIP OF DISTRIBUTION CHANNEL AND ADR

[ ]:
```
# Visualizing with the help of bar plot for checking relationship between ADR␣
 ↪and distribution channel
# Using group by on distribution channel and hotel
distribution_channel_df = hotel_booking_df.groupby(['distribution_channel',␣
 ↪'hotel'])['adr'].mean().reset_index()

# Set plot size
plt.figure(figsize = (12,6))

# Plotting the bar chart
sns.barplot(x = 'distribution_channel', y = 'adr', data =␣
 ↪distribution_channel_df, hue = 'hotel')
```

```
# Set labels
plt.xlabel("Distribution Channel", fontsize = 16)
plt.ylabel("ADR", fontsize = 16)
plt.title('ADR across Distribution Channel', fontsize = 20)

# To show
plt.show()
```



23 CHART-7: BOOKING BY MONTH AND OPTIMAL STAY LENGTH IN HOTELS

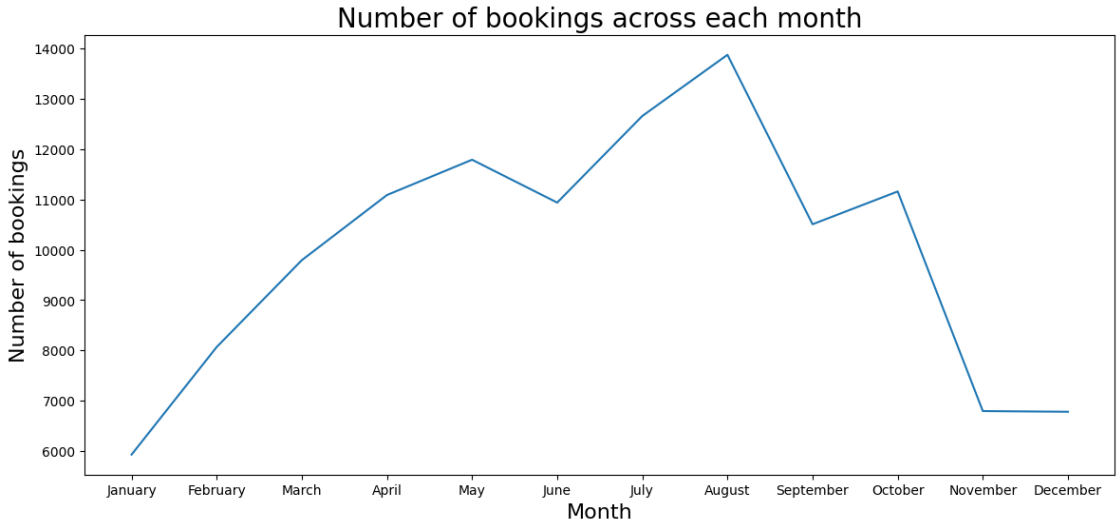
[ ]:
```
# Using groupby on arrival_date_month and taking the hotel count
bookings_by_months_df = hotel_booking_df.
 ↪groupby(['arrival_date_month'])['hotel'].count().reset_index().
 ↪rename(columns = {'hotel':'Counts'})

# Creating list of months in order
months = ['January', 'February', 'March', 'April', 'May', 'June', 'July',␣
 ↪'August', 'September', 'October', 'November', 'December']

# Creating dataframe which will map the order of above months list without␣
 ↪changing its values
```

```
bookings_by_months_df['arrival_date_month'] = pd.
 ↪Categorical(bookings_by_months_df['arrival_date_month'], categories =␣
 ↪months, ordered = True)

# Sorting by arrival_date_month
bookings_by_months_df = bookings_by_months_df.sort_values('arrival_date_month')

bookings_by_months_df
```

[ ]:

	arrival_date_month	Counts
4	January	5929
3	February	8068
7	March	9794
0	April	11089
8	May	11791
6	June	10939
5	July	12661
1	August	13877
11	September	10508
10	October	11160
9	November	6794
2	December	6780

[ ]:
```
# Set plot size
plt.figure(figsize = (14,6))

# Plotting lineplot on x- months & y- bookings counts
sns.lineplot(x = bookings_by_months_df['arrival_date_month'], y =␣
 ↪bookings_by_months_df['Counts'])

# Set title
plt.title('Number of bookings across each month', fontsize = 20)

# Set labels
plt.xlabel('Month', fontsize = 16)
plt.ylabel('Number of bookings', fontsize = 16)

# To show
plt.show()
```



24 CHART-8: PLOTTING HISTOGRAM

[ ]:
```
hotel_booking_df.hist(figsize = (23,18))

# To show
plt.show()
```

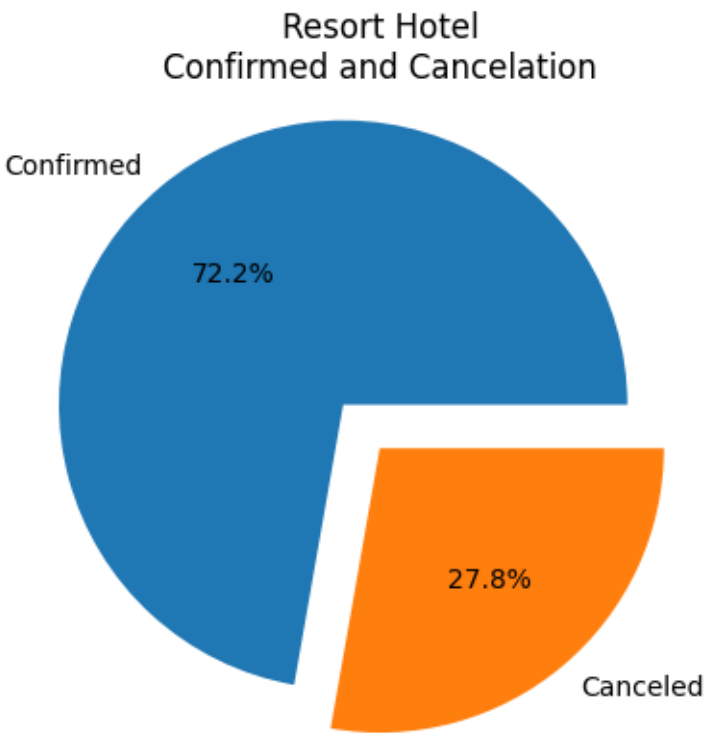
25 CHART-9: YEAR AND HOTEL WISE CONFIRMED BOOKING AND CANCELLETION DISTRIBUTION

[ ]:
```
# Set plot size
plt.figure(figsize = (10,6))

# Create the figure object
sns.countplot(x = 'hotel', hue = 'is_canceled', palette = 'Set2', data =␣
 ↪hotel_booking_df)

# Set legends
plt.legend(['Confirmed', 'Canceled'])

# Set labels
plt.title('Hotel wise confirmation and cancelation of the bookings', fontsize =␣
 ↪20)
plt.ylabel('Count of\nconfirmation and cancelation', fontsize = 16)
plt.xlabel('Hotel Type', fontsize = 16)
```

```
# To show
plt.show()
```

[ ]:
```
# Plotting a Pie chart using matplotlib for percentage of confirmed and␣
 ↪canceled bookings of Resort Hotel
resort_hotel = hotel_booking_df.loc[(hotel_booking_df['hotel'] == 'Resort␣
 ↪Hotel')]
resort_hotel_checking_cancel = resort_hotel['is_canceled'].value_counts()

# Set labels
mylabels = ['Confirmed', 'Canceled']

# Set figure size
myexplode = [0.2, 0]

# Create the figure object
resort_hotel_cancelation = plt.pie(resort_hotel_checking_cancel, labels =␣
 ↪mylabels, explode = myexplode, autopct = '%1.1f%%')

# Set title
plt.title('Resort Hotel\nConfirmed and Cancelation')

resort_hotel_checking_cancel
```

[ ]:
```
is_canceled
0    28938
1    11122
Name: count, dtype: int64
```



[ ]:
```
# Removing the canceled bookings from the data and creating a new dataframe
data_not_canceled = hotel_booking_df[hotel_booking_df['is_canceled'] == 0]

# Year wise Bookings of hotels
# Set style
sns.set_style(style = 'darkgrid')

# Set plot size
plt.figure(figsize = (12,6))

# Create the figure object
sns.countplot(x= 'arrival_date_year', hue= 'hotel', palette = 'tab10', data =␣
 ↪data_not_canceled)

# Set legends
plt.legend(['Resort Hotel', 'City Hotel'])
```

```
# Set labels
plt.title('Year wise bookings of hotels', fontsize = 20)
plt.ylabel('Number of bookings', fontsize = 16)
plt.xlabel('Year', fontsize = 16)

# To show
plt.show()
```

26 CHART-10: ADR ACROSS DIFFERENT MONTHS

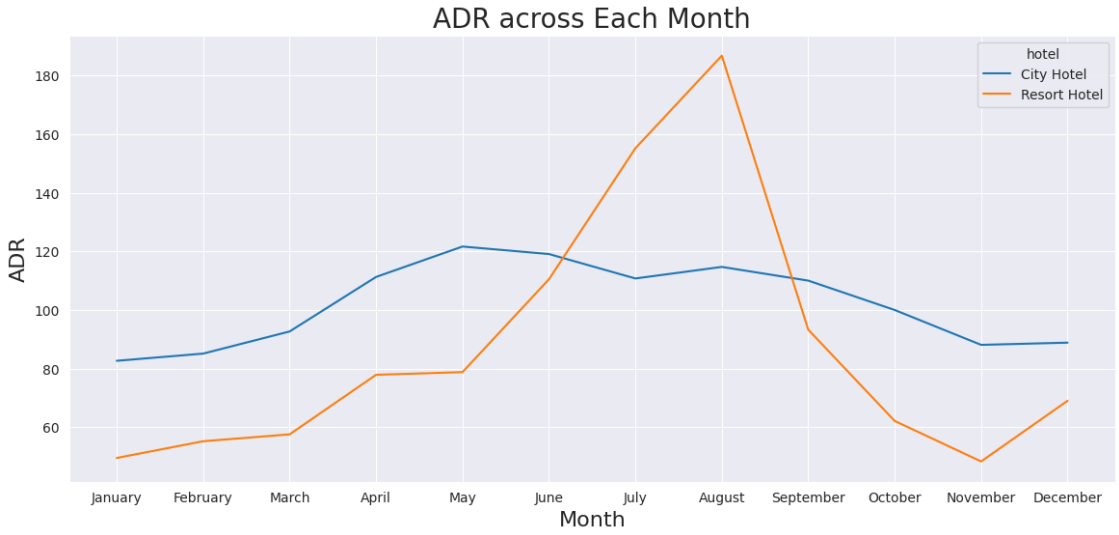
[ ]:
```
# Using groupby funtion
bookings_by_months_df = hotel_booking_df.groupby(['arrival_date_month',␣
 ↪'hotel'])['adr'].mean().reset_index()

# Create month list
months = ['January', 'February', 'March', 'April', 'May', 'June', 'July',␣
 ↪'August', 'September', 'October', 'November', 'December']

# It will take the order of the month list in the dataframe along with values
bookings_by_months_df['arrival_date_month'] = pd.
 ↪Categorical(bookings_by_months_df['arrival_date_month'], categories =␣
 ↪months, ordered = True)

# Sorting values
bookings_by_months_df = bookings_by_months_df.sort_values('arrival_date_month')
```

```
bookings_by_months_df
```

[ ]:

	arrival_date_month	hotel	adr
8	January	City Hotel	82.628986
9	January	Resort Hotel	49.461883
6	February	City Hotel	85.088278
7	February	Resort Hotel	55.171930
15	March	Resort Hotel	57.520147
14	March	City Hotel	92.643116
0	April	City Hotel	111.251838
1	April	Resort Hotel	77.849496
17	May	Resort Hotel	78.758134
16	May	City Hotel	121.638560
13	June	Resort Hotel	110.444749
12	June	City Hotel	119.074341
11	July	Resort Hotel	155.181299
10	July	City Hotel	110.734292
3	August	Resort Hotel	186.790574
2	August	City Hotel	114.680455
22	September	City Hotel	110.004661
23	September	Resort Hotel	93.252030
20	October	City Hotel	99.974498
21	October	Resort Hotel	62.097617
18	November	City Hotel	88.069601
19	November	Resort Hotel	48.273993
5	December	Resort Hotel	68.984230
4	December	City Hotel	88.826307

[ ]:
```
# Visualizing with the help of line plot

# Set plot size
plt.figure(figsize = (14,6))

# Create the figure object and plotting the line
sns.lineplot(x = bookings_by_months_df['arrival_date_month'], y =␣
 ↪bookings_by_months_df['adr'], hue = bookings_by_months_df['hotel'])

# Set labels
plt.title('ADR across Each Month', fontsize = 20)
plt.xlabel('Month', fontsize = 16)
plt.ylabel('ADR', fontsize = 16)

# To show
plt.show()
```



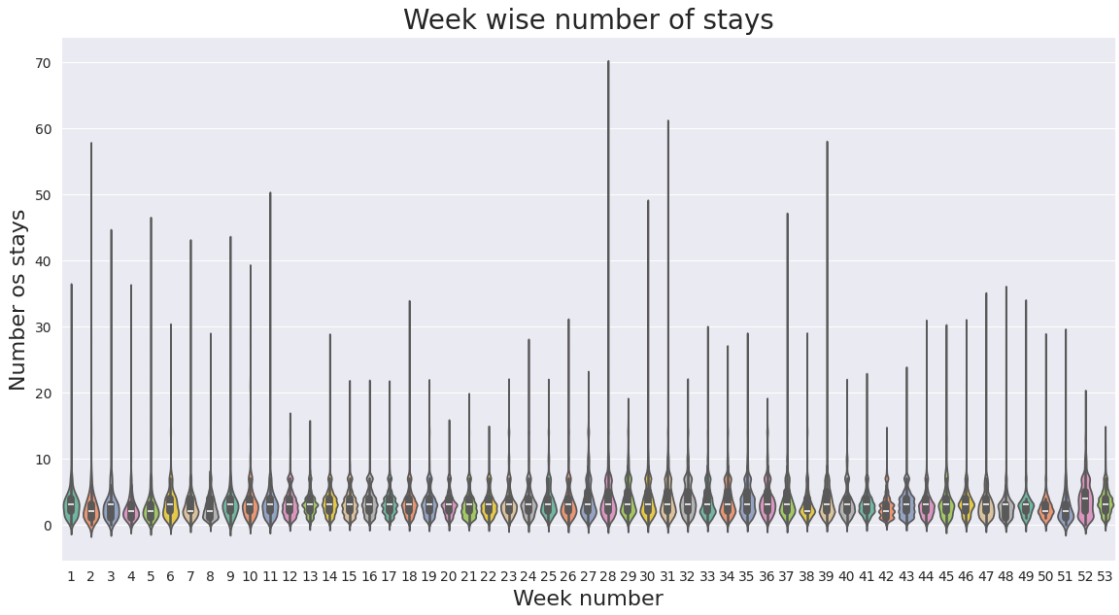
27 CHART-11: WEEKLY STAY DISTRIBUTION AND CALCULATION OF CANCELLETION AND NON-CANCELLATION

[ ]:
```
# Chart - 12 visualization code
# As i have already created a column 'total_stay' above i.e.
# Adding total staying days in hotels
hotel_booking_df['total_stay'] = hotel_booking_df['stays_in_weekend_nights'] +␣
 ↪hotel_booking_df['stays_in_week_nights']

# Set the plot size
plt.figure(figsize=(14,7))

# Using a violin plot to know in which weeks, visitors stays the most
sns.violinplot(x = 'arrival_date_week_number', y = 'total_stay', palette =␣
 ↪'Set2', data = hotel_booking_df)

# Set labels
plt.title('Week wise number of stays', fontsize = 20)
plt.ylabel('Number os stays', fontsize = 16)
plt.xlabel('Week number', fontsize = 16)

# To show
plt.show()
```

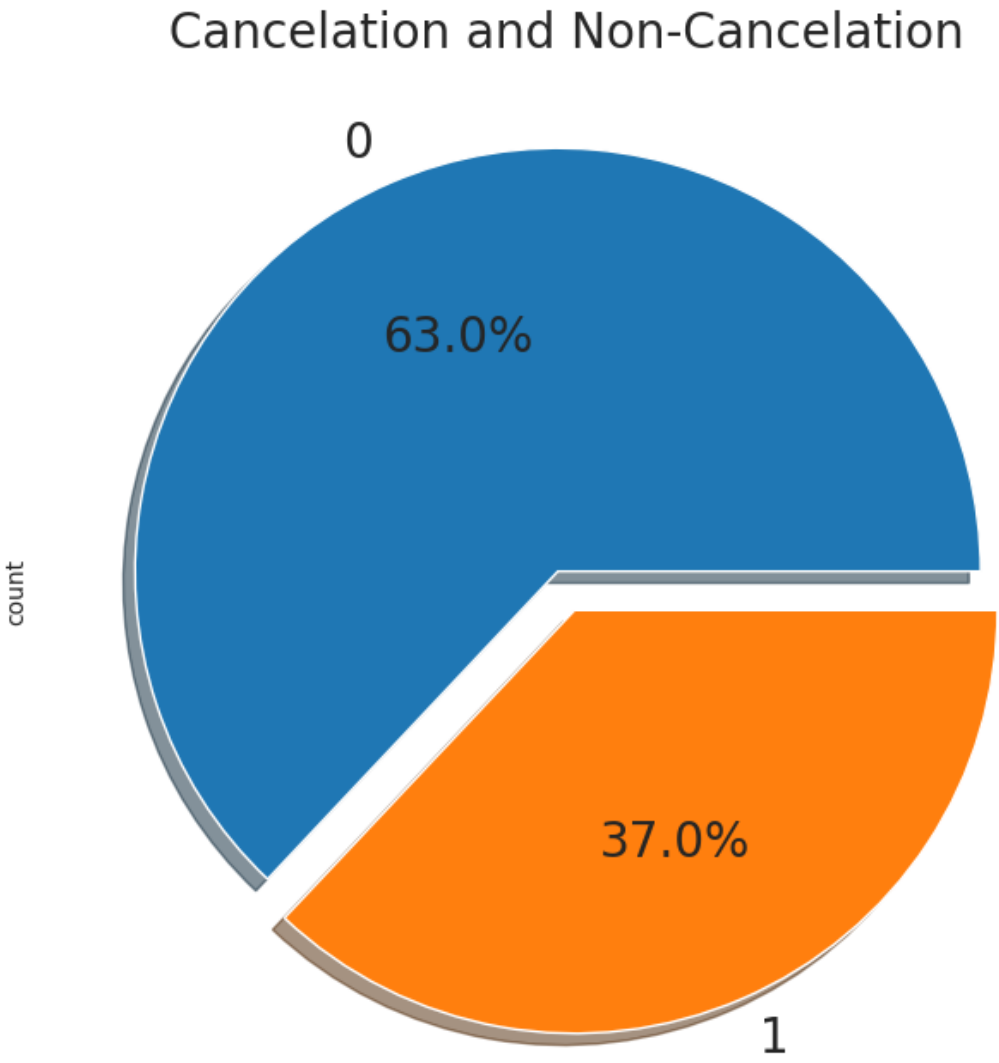
```
<ipython-input-107-714aeee34269>:10: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
```

v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

```
  sns.violinplot(x = 'arrival_date_week_number', y = 'total_stay', palette =
'Set2', data = hotel_booking_df)
```



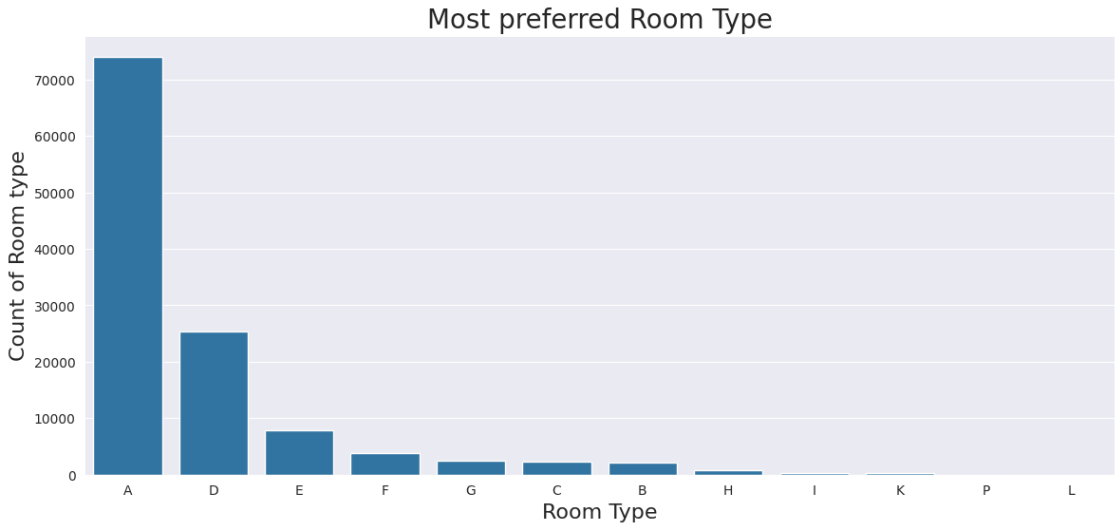
[ ]:
```
# Visualizing with the help of pie plot
hotel_booking_df['is_canceled'].value_counts().plot.pie(explode = [0.05,0.05],␣
 ↪autopct = '%1.1f%%', shadow = True, figsize = (10,8), fontsize = 20)

# Set title
plt.title('Cancelation and Non-Cancelation', fontsize = 20)

# To show
plt.show()
```



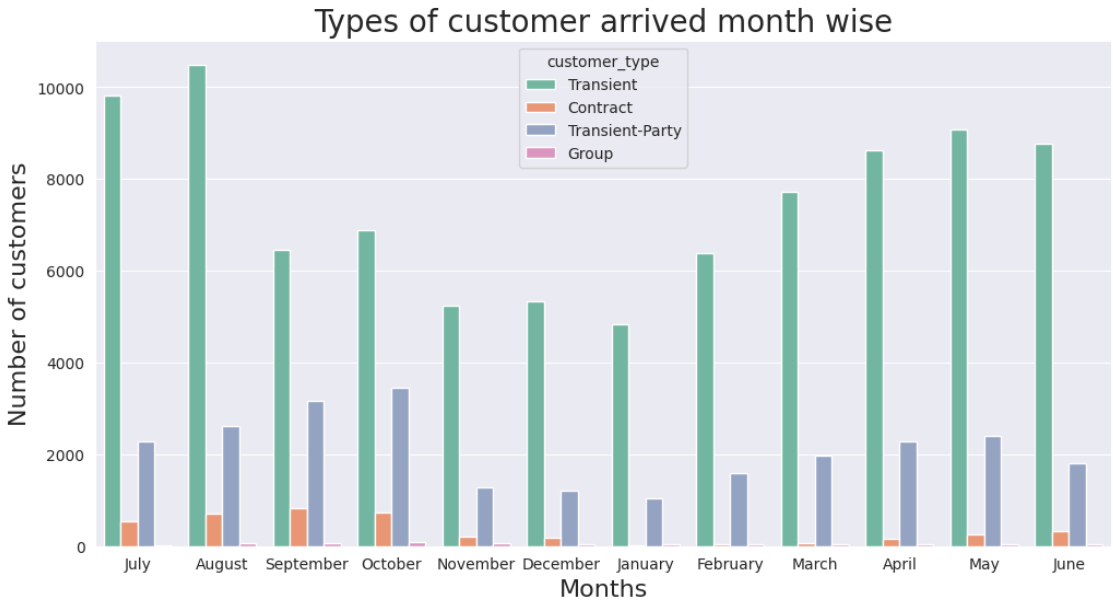
28 CHART-12: ROOM TYPE PREFFERENCE AND CUSTOMER TYPE

[ ]:
```
# Set the plot size
plt.figure(figsize = (14,6))

# Create the figure object
sns.countplot(x = hotel_booking_df['assigned_room_type'], order =␣
↪hotel_booking_df['assigned_room_type'].value_counts().index)

# Set labels
```

```
plt.xlabel('Room Type', fontsize = 16)
plt.ylabel('Count of Room type', fontsize = 16)
plt.title('Most preferred Room Type', fontsize = 20)

# To show
plt.show()
```



[ ]:
```
# Using seeborn to plot a count plot chart to demonstrate the types of customer␣
 ↪visit the most
# Set the plot size
plt.figure(figsize = (12,6))

# Create the figure object
sns.countplot(x = 'arrival_date_month', hue = 'customer_type', palette =␣
 ↪'Set2', data = hotel_booking_df)

# Set labels
plt.xlabel('Months', fontsize = 16)
plt.ylabel('Number of customers', fontsize = 16)
plt.title('Types of customer arrived month wise', fontsize = 20)

# To show
plt.show()
```


Types of customer arrived month wise
customer_type
Transient
Contract
Transient-Party
Group
Number of customers
0
2000
4000
6000
8000
10000
July
August
September
October
November
December
January
February
March
April
May
June
Months