In [1]:

```
## Libraries needed for EDA.
import numpy as np
import pandas as pd

#For visualization
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import plotly.express as px  ##Used for plotting
```

```
C:\Users\Rajarshi\anaconda3\lib\site-packages\scipy\__init__.py:146: UserWarning: A NumPy version >=1.17.3 and <1.25.0 is required f
or this version of SciPy (detected version 1.26.0
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

Loading Dataset

In [2]:

```
df=pd.read_csv('Hotel Bookings.csv', encoding='unicode_escape')
```

In [3]:

```
df
```

Out[3]:

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_date_day_of_month	stays_in_weekend_nights	st
0	Resort Hotel	0	342	2015	July	27	1	0	
1	Resort Hotel	0	737	2015	July	27	1	0	
2	Resort Hotel	0	7	2015	July	27	1	0	
3	Resort Hotel	0	13	2015	July	27	1	0	
4	Resort Hotel	0	14	2015	July	27	1	0	
...	...	...	...	...	...	...	...	...	
119385	City Hotel	0	23	2017	August	35	30	2	
119386	City Hotel	0	102	2017	August	35	31	2	
119387	City Hotel	0	34	2017	August	35	31	2	
119388	City Hotel	0	109	2017	August	35	31	2	
119389	City Hotel	0	205	2017	August	35	29	2	

119390 rows × 32 columns

In [4]:

```
##Information of Dataset
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 119390 entries, 0 to 119389
Data columns (total 32 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 
dtypes: float64(4), int64(16), object(12)
memory usage: 29.1+ MB
```

In [5]:
```
df.shape
```

Out[5]:
```
(119390, 32)
```

119390 = Rows, 32 = Columns

In [6]:
```
##Looking at the first 5 rows of the dataset.
df.head()
```

Out[6]:

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_date_day_of_month	stays_in_weekend_nights	stays_in
0	Resort Hotel	0	342	2015	July	27	1	0	
1	Resort Hotel	0	737	2015	July	27	1	0	
2	Resort Hotel	0	7	2015	July	27	1	0	
3	Resort Hotel	0	13	2015	July	27	1	0	
4	Resort Hotel	0	14	2015	July	27	1	0	

5 rows × 32 columns

Dataset First view.

In [7]:
```
df
```

Out[7]:

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_date_day_of_month	stays_in_weekend_nights	st
0	Resort Hotel	0	342	2015	July	27	1	0	
1	Resort Hotel	0	737	2015	July	27	1	0	
2	Resort Hotel	0	7	2015	July	27	1	0	
3	Resort Hotel	0	13	2015	July	27	1	0	
4	Resort Hotel	0	14	2015	July	27	1	0	
...	...	...	...	...	...	...	...	...	
119385	City Hotel	0	23	2017	August	35	30	2	
119386	City Hotel	0	102	2017	August	35	31	2	
119387	City Hotel	0	34	2017	August	35	31	2	
119388	City Hotel	0	109	2017	August	35	31	2	
119389	City Hotel	0	205	2017	August	35	29	2	

119390 rows × 32 columns

In [8]:

```
##Looking at the last 5 rows of the dataset
df.tail()
```

Out[8]:

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_date_day_of_month	stays_in_weekend_nights	sta
119385	City Hotel	0	23	2017	August	35	30	2	
119386	City Hotel	0	102	2017	August	35	31	2	
119387	City Hotel	0	34	2017	August	35	31	2	
119388	City Hotel	0	109	2017	August	35	31	2	
119389	City Hotel	0	205	2017	August	35	29	2	

5 rows × 32 columns

Dataset Rows & Columns count

In [9]:

```
print(f'Number of rows : {len(df.axes[0])}')
print(f'Number of rows : {len(df.axes[1])}')
```

```
Number of rows : 119390
Number of rows : 32
```

Missing Values/Null Values Count

In [10]:

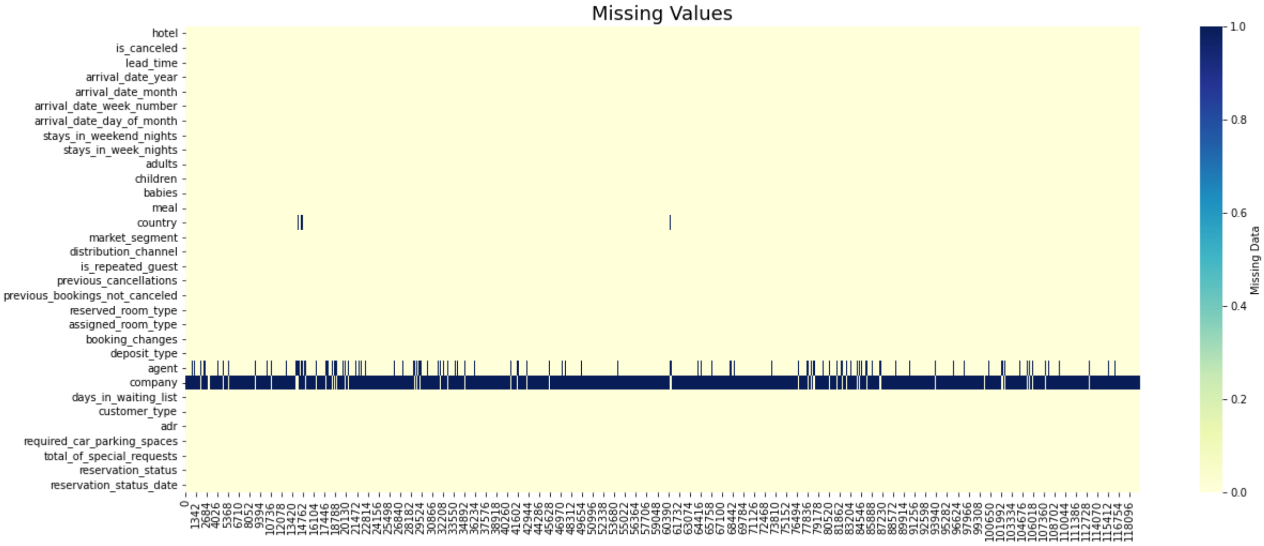
```
df.isnull().sum()
```

Out[10]:

```
hotel                                  0
is_canceled                            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
stays_in_week_nights                   0
adults                                 0
children                               4
babies                                 0
meal                                   0
country                              488
market_segment                         0
distribution_channel                   0
is_repeated_guest                      0
previous_cancellations                 0
previous_bookings_not_canceled         0
reserved_room_type                     0
assigned_room_type                     0
booking_changes                        0
deposit_type                           0
agent                              16340
company                           112593
days_in_waiting_list                   0
customer_type                          0
adr                                    0
required_car_parking_spaces            0
total_of_special_requests              0
reservation_status                     0
reservation_status_date                0
dtype: int64
```

In [11]:

```python
## Visualizing the missing values using Seaborn Heatmap.

plt.figure(figsize=(20,8))
sns.heatmap(df.isna().transpose(),
            cmap="YlGnBu",
            cbar_kws={'label' : 'Missing Data'})

plt.title('Missing Values', fontsize=18)
plt.show()
```



Insights about the dataset. We can see that there are 4 columns with missing/null values : company, agent, country, children. 1. In children column, I will replace null values with 0 assuming that customer did not have any children. 2. Column country has null values. I will replace null values in the column with 'Others' assuming customer's country was not mentioned while booking. 3. In company and agent column it might be a case when customers did not book hotel through them so these columns might have null values in it. As these 2 columns have numeric data in it, I will replace them with 0.

Understanding the variables

In [12]:

```python
## Dataset columns.
df.columns
```

Out[12]:

```
Index(['hotel', 'is_canceled', 'lead_time', 'arrival_date_year',
       'arrival_date_month', 'arrival_date_week_number',
       'arrival_date_day_of_month', 'stays_in_weekend_nights',
       'stays_in_week_nights', 'adults', 'children', 'babies', 'meal',
       'country', 'market_segment', 'distribution_channel',
       'is_repeated_guest', 'previous_cancellations',
       'previous_bookings_not_canceled', 'reserved_room_type',
       'assigned_room_type', 'booking_changes', 'deposit_type', 'agent',
       'company', 'days_in_waiting_list', 'customer_type', 'adr',
       'required_car_parking_spaces', 'total_of_special_requests',
       'reservation_status', 'reservation_status_date'],
      dtype='object')
```

In [13]:

```python
#Dataset describe
df.describe()
```

Out[13]:

	is_canceled	lead_time	arrival_date_year	arrival_date_week_number	arrival_date_day_of_month	stays_in_weekend_nights	stays_in_week_nights
count	119390.000000	119390.000000	119390.000000	119390.000000	119390.000000	119390.000000	119390.000000
mean	0.370416	104.011416	2016.156554	27.165173	15.798241	0.927599	2.500302
std	0.482918	106.863097	0.707476	13.605138	8.780829	0.998613	1.908286
min	0.000000	0.000000	2015.000000	1.000000	1.000000	0.000000	0.000000
25%	0.000000	18.000000	2016.000000	16.000000	8.000000	0.000000	1.000000
50%	0.000000	69.000000	2016.000000	28.000000	16.000000	1.000000	2.000000
75%	1.000000	160.000000	2017.000000	38.000000	23.000000	2.000000	3.000000
max	1.000000	737.000000	2017.000000	53.000000	31.000000	19.000000	50.000000

Variable description

Hotel: (Resort Hotel or City Hotel)

is_canceled: Value indicating if the booking was canceled (1) or not (0)

load_time: *Number of days that elapsed between the entering date of the booking into the PMS and the arrival date*

arrival_date_year: Year of arrival date

arrival_date_month : Month of arrival date

arrival_date_week_number : Week number of year for arrival date

arrival_date_day_of_month : Day of arrival date

stays_in_weekend_nights : Number of weekend nights (Saturday or Sunday) the guest stayed or booked to stay at the hotel

stays_in_week_nights : Number of week nights (Monday to Friday) the guest stayed or booked to stay at the hotel

adults : Number of adults

children : Number of children

babies : Number of babies

meal : Type of meal booked. Categories are presented in standard hospitality meal packages

country : Country of origin: market_segment : Market segment designation. In categories, the term -TA- means Travel Agents- and "TO" means Tour Operators- distribution_channel : Booking distribution channel. The term STA" means Travel Agents- and -TO" means Tour Operators

is_repeated_guest : Value indicating if the booking name was from a repeated guest (1) or not (O)

previous_cancellations : Number of previous bookings that were cancelled by the customer prior to the current booking

previous_bookings_not_canceled : Number of previous bookings not cancelled by the customer prior to the current booking

reserved_room_type : Code of room type reserved. Code is presented instead of designation for anonymity reasons.

assigned_room_type : Code for the type of room assigned to the booking.

booking_changes : Number of changes/amendments made to the booking from the moment the booking was entered on the PMS until the moment of check-in or cancellation

deposit_type : Indication on if the customer made a deposit to guarantee the booking.

agent : ID of the travel agency that made the booking

company : ID of the company/entity that made the booking or responsible for paying the booking.

davs_in_waiting_list : Number of davs the bookina was in the waitina list before it was confirmed to the customer.

customer_type : Type of booking, assuming one of four categories

adr : Average Daily Rate as defined by dividing the sum of all lodging transactions by the total number of staying nights

required_car_parking_spaces : Number of car parking spaces required by the customer

total_of_special_requests : Number of special requests made by the customer (e.g. twin bed or high floor)

reservation_status : Reservation last status, assuming one of three categories

Canceled — booking was canceled by the customer Check-Out — customer has checked in but already departed No-Show - customer did not check-in and did inform the hotel of the reason why reservation_status_date - Date at which the last status was set

Check Unique Values for each Variable

In [14]:

```
# Check Unique Values for each variable.
pd.Series({col:df[col].unique()for col in df})

#creating a series consisting every column name of the dataset and it's value
#used  for loop to iterate over every column in the dataset.
```

Out[14]:

```
hotel                                                    [Resort Hotel, City Hotel]
is_canceled                                                                  [0, 1]
lead_time                         [342, 737, 7, 13, 14, 0, 9, 85, 75, 23, 35, 68...
arrival_date_year                                                [2015, 2016, 2017]
arrival_date_month                [July, August, September, October, November, D...
arrival_date_week_number          [27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 3...
arrival_date_day_of_month         [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14...
stays_in_weekend_nights           [0, 1, 2, 4, 3, 6, 13, 8, 5, 7, 12, 9, 16, 18,...
stays_in_week_nights              [0, 1, 2, 3, 4, 5, 10, 11, 8, 6, 7, 15, 9, 12,...
adults                             [2, 1, 3, 4, 40, 26, 50, 27, 55, 0, 20, 6, 5, 10]
children                                            [0.0, 1.0, 2.0, 10.0, 3.0, nan]
babies                                                             [0, 1, 2, 10, 9]
meal                                                    [BB, FB, HB, SC, Undefined]
country                           [PRT, GBR, USA, ESP, IRL, FRA, nan, ROU, NOR, ...
market_segment                    [Direct, Corporate, Online TA, Offline TA/TO, ...
distribution_channel                     [Direct, Corporate, TA/TO, Undefined, GDS]
is_repeated_guest                                                            [0, 1]
previous_cancellations            [0, 1, 2, 3, 26, 25, 14, 4, 24, 19, 5, 21, 6, ...
previous_bookings_not_canceled    [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,...
reserved_room_type                                   [C, A, D, E, G, F, H, L, P, B]
assigned_room_type                             [C, A, D, E, G, F, I, B, H, P, L, K]
booking_changes                   [3, 4, 0, 1, 2, 5, 17, 6, 8, 7, 10, 16, 9, 13,...
deposit_type                                   [No Deposit, Refundable, Non Refund]
agent                             [nan, 304.0, 240.0, 303.0, 15.0, 241.0, 8.0, 2...
company                           [nan, 110.0, 113.0, 270.0, 178.0, 240.0, 154.0...
days_in_waiting_list              [0, 50, 47, 65, 122, 75, 101, 150, 125, 14, 60...
customer_type                         [Transient, Contract, Transient-Party, Group]
adr                               [0.0, 75.0, 98.0, 107.0, 103.0, 82.0, 105.5, 1...
required_car_parking_spaces                                         [0, 1, 2, 8, 3]
total_of_special_requests                                        [0, 1, 3, 2, 4, 5]
reservation_status                                   [Check-Out, Canceled, No-Show]
reservation_status_date           [2015-07-01, 2015-07-02, 2015-07-03, 2015-05-0...
dtype: object
```

Data Wrangling

Data Wrangling Code

In [15]:

```
#Creating a duplicate of the original dataset before making any changes in it.
df1 = df.copy()
```

In [16]:

```
df1.columns
```

Out[16]:

```
Index(['hotel', 'is_canceled', 'lead_time', 'arrival_date_year',
       'arrival_date_month', 'arrival_date_week_number',
       'arrival_date_day_of_month', 'stays_in_weekend_nights',
       'stays_in_week_nights', 'adults', 'children', 'babies', 'meal',
       'country', 'market_segment', 'distribution_channel',
       'is_repeated_guest', 'previous_cancellations',
       'previous_bookings_not_canceled', 'reserved_room_type',
       'assigned_room_type', 'booking_changes', 'deposit_type', 'agent',
       'company', 'days_in_waiting_list', 'customer_type', 'adr',
       'required_car_parking_spaces', 'total_of_special_requests',
       'reservation_status', 'reservation_status_date'],
      dtype='object')
```

In [17]:

```
# replacing null values in children column with 0 assuming that family had 0 childre
# replacing null values in company and agent columns with 0 assuming these rooms were booked without company/agent

df1['children'].fillna(0, inplace = True)
df1['company'].fillna(0, inplace = True)
df1['agent'].fillna(0, inplace = True)

# replacing null values in country column as 'Others'

df1['country'].fillna('Others', inplace = True)
```

In [18]:

```
# Checking for null values after replacing them
df1.isnull().sum()
```

Out[18]:

```
hotel                             0
is_canceled                       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
stays_in_week_nights              0
adults                            0
children                          0
babies                            0
meal                              0
country                           0
market_segment                    0
distribution_channel              0
is_repeated_guest                 0
previous_cancellations            0
previous_bookings_not_canceled    0
reserved_room_type                0
assigned_room_type                0
booking_changes                   0
deposit_type                      0
agent                             0
company                           0
days_in_waiting_list              0
customer_type                     0
adr                               0
required_car_parking_spaces       0
total_of_special_requests         0
reservation_status                0
reservation_status_date           0
dtype: int64
```

In [19]:

```
# dropping the 'company column as it contains a lot of null values in comparison to other columns
df1.drop(['company'], axis =1, inplace = True)  #dropping the values vertically at axis 1 (columns)
```

In [20]:

```
# dropping rows where no adult, children and babies are available because no bookings were made that day

no_guest = df1[df1['adults']+df1['babies']+df1['children']==0]
df1.drop(no_guest.index, inplace=True)
```

In [21]:

```
# adding some new columns to make our data analysis ready
df1['total_people'] = df1['adults'] + df1['babies'] + df1['children']  ## creating total people column by adding all the people in th

df1['total_stay'] = df1['stays_in_weekend_nights'] + df1['stays_in_week_nights']  ## creating a column to check total stay by people
```

In [22]:

```
# having a final look to check if our dataset is ready to analyse
df1.head()
```

Out[22]:

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_date_day_of_month	stays_in_weekend_nights	stays_in
0	Resort Hotel	0	342	2015	July	27	1	0	
1	Resort Hotel	0	737	2015	July	27	1	0	
2	Resort Hotel	0	7	2015	July	27	1	0	
3	Resort Hotel	0	13	2015	July	27	1	0	
4	Resort Hotel	0	14	2015	July	27	1	0	

5 rows × 33 columns

In [23]:

```
df1.tail()
```

Out[23]:

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_date_day_of_month	stays_in_weekend_nights	sta
119385	City Hotel	0	23	2017	August	35	30	2	
119386	City Hotel	0	102	2017	August	35	31	2	
119387	City Hotel	0	34	2017	August	35	31	2	
119388	City Hotel	0	109	2017	August	35	31	2	
119389	City Hotel	0	205	2017	August	35	29	2	

5 rows × 33 columns

In [24]:

```
## Checking the final shape of the dataset
## Added 2 more columns so it became 33 from 31.
```

```
print(f' final shape of the dataset is {df1.shape}')
```

```
final shape of the dataset is (119210, 33)
```

In [25]:
```
## Checking the unique values which is to be analysed.

pd.Series({col:df1[col].unique() for col in df1})
```

Out[25]:
```
hotel                                                   [Resort Hotel, City Hotel]
is_canceled                                                                 [0, 1]
lead_time                         [342, 737, 7, 13, 14, 0, 9, 85, 75, 23, 35, 68...
arrival_date_year                                               [2015, 2016, 2017]
arrival_date_month                [July, August, September, October, November, D...
arrival_date_week_number          [27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 3...
arrival_date_day_of_month         [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14...
stays_in_weekend_nights           [0, 1, 2, 4, 3, 6, 13, 8, 5, 7, 12, 9, 16, 18,...
stays_in_week_nights              [0, 1, 2, 3, 4, 5, 10, 11, 8, 6, 7, 15, 9, 12,...
adults                            [2, 1, 3, 4, 40, 26, 50, 27, 55, 20, 6, 5, 10, 0]
children                                                [0.0, 1.0, 2.0, 10.0, 3.0]
babies                                                            [0, 1, 2, 10, 9]
meal                                                   [BB, FB, HB, SC, Undefined]
country                           [PRT, GBR, USA, ESP, IRL, FRA, Others, ROU, NO...
market_segment                    [Direct, Corporate, Online TA, Offline TA/TO, ...
distribution_channel                     [Direct, Corporate, TA/TO, Undefined, GDS]
is_repeated_guest                                                           [0, 1]
previous_cancellations            [0, 1, 2, 3, 26, 25, 14, 4, 24, 19, 5, 21, 6, ...
previous_bookings_not_canceled    [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,...
reserved_room_type                                     [C, A, D, E, G, F, H, L, B]
assigned_room_type                               [C, A, D, E, G, F, I, B, H, L, K]
booking_changes                   [3, 4, 0, 1, 2, 5, 17, 6, 8, 7, 10, 16, 9, 13,...
deposit_type                                  [No Deposit, Refundable, Non Refund]
agent                             [0.0, 304.0, 240.0, 303.0, 15.0, 241.0, 8.0, 2...
days_in_waiting_list              [0, 50, 47, 65, 122, 75, 101, 150, 125, 14, 60...
customer_type                        [Transient, Contract, Transient-Party, Group]
adr                               [0.0, 75.0, 98.0, 107.0, 103.0, 82.0, 105.5, 1...
required_car_parking_spaces                                        [0, 1, 2, 8, 3]
total_of_special_requests                                       [0, 1, 3, 2, 4, 5]
reservation_status                                  [Check-Out, Canceled, No-Show]
reservation_status_date           [2015-07-01, 2015-07-02, 2015-07-03, 2015-05-0...
total_people                      [2.0, 1.0, 3.0, 4.0, 5.0, 12.0, 40.0, 26.0, 50...
total_stay                        [0, 1, 2, 3, 4, 5, 6, 7, 14, 15, 10, 11, 8, 9,...
dtype: object
```

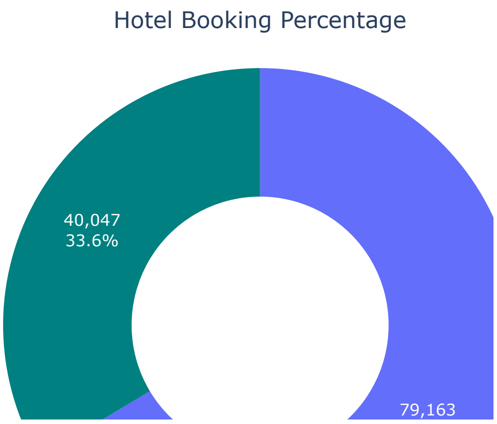
We can see that we have dealt with all the null values and added some new columns and now our dataset is ready to analysed.

What all manipulations have you done and insights you found? Created a copy of the dataset before doing any manipulation then filled missing values with O in children . company and agent columns as those columns had numerical values and in column country filled missing values with 'others'. after dealing with missing values I dropped the country column as this had 96% missing values and was of no use in our analysis. In next step I created 2 new columns named •total_people' and total_stay• for further analysis. In total people column I added all the babies. children and adults. similarly in second new column I added weekend stay and week stay column. After doing all the manipulation I checked new manipulated dataset to check if this is ready to be analyzed. After manipulating the dataset these were the insights I found: I. There are 2 types of hotel which guests could book so I can find which type of hotel was booked most. 2. There are different types of guests and they come from different countries. 3. Guests can choose different foods from the menu. 4. Guests can book hotel directly or through different channels that are available. 5. Guests can cancel their booking and there are repeated guests also. 6. Guests can choose rooms of their liking while booking. 7. There is column available in the dataset named 'add which could be used to analyze hotel's performance on the basis of revenue.

4. Data Vizualization, Storytelling & Experimenting with charts : Understand the relationships between variables

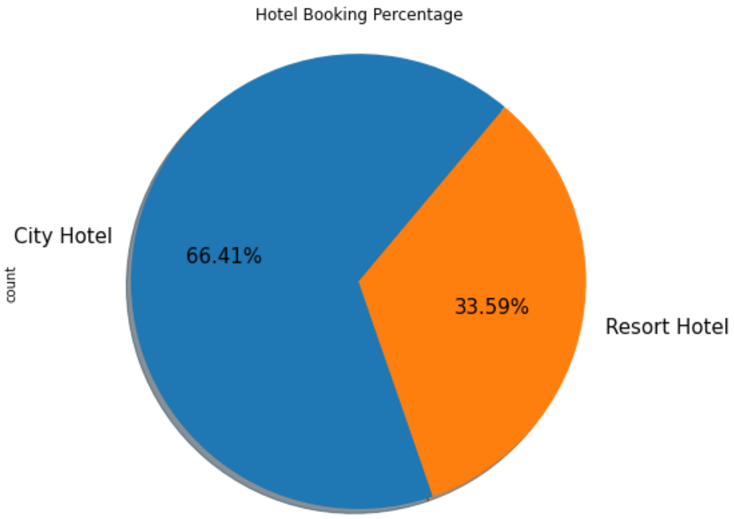
Chart- 1 Which type of hotel is most preffered by the guests?

In [26]:
```
# Chart - 1 visualization code
# Storing unique hotel names in a variable
hotel_name = df1['hotel'].unique()

# Checking the number of unique booking in each hotel type
unique_booking = df1.hotel.value_counts().sort_values(ascending=True)

# Creating a donut chart using plotly.express
fig1 = px.pie(names = hotel_name, values = unique_booking, hole = 0.5, color = hotel_name,
              color_discrete_map={
                  'Resort Hotel': 'teal' , 'City Hotel' : 'nude'})

# Giving it a title and updating the text info
fig1.update_traces(textinfo = 'percent + value')
fig1.update_layout(title_text = 'Hotel Booking Percentage', title_x = 0.5)

# Setting the legend at center
fig1.update_layout(legend=dict(
    orientation = 'h',
    yanchor = 'bottom',
    xanchor = 'center',
    x = 0.5
))

# Display the figure
fig1.show()
```



Creating a Pie chart also for the above problem statement as Donut chart is not exported to github.

In [27]:
```
# Count Hotel
hotel_count = df1.hotel.value_counts()

# Plotting Values in a simple pie chart
hotel_count.plot.pie(figsize=(9,7), autopct='%1.2f%%', shadow=True, fontsize=15,startangle=50)
# Setting the title
plt.title('Hotel Booking Percentage')
plt.axis('equal')
plt.show()
```

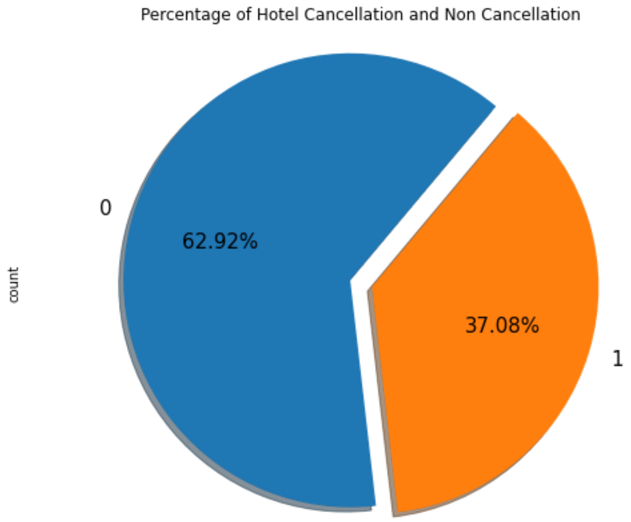


1. Why did you pick the specific chart? I used Donut chart here because it is used to show the proportions of categorical data, with the size of each piece representing the proportion of each category. 2. What is/are the insight(s) found from the chart? I found out that guests prefer Resort Hotel most over City Hotel. 3. Will the gained insights help creating a positive business impact? Are there any insights that lead to negative growth? Justify with specific reason. This insight is useful for the stakeholder to check which hotel is performing best and they can invest more capitals in that. There is no such negative growth but stakeholders can focus more on City Hotel to get more booking and icrease the overall revenue.

Chart - 2

What is perecentage of hotel booking cancellation?

In [28]:
```
# Chart - 2 visualization code
# Extracting and storing unique values of hotel cancelation
cancelled_hotel = df1.is_canceled.value_counts()

# Craeting a pie chart
cancelled_hotel.plot.pie(figsize=(9,7), explode=(0.05,0.05), autopct='%1.2f%%', shadow=True, fontsize=15,startangle=50)

# Giving our pie chart a title
plt.title('Percentage of Hotel Cancellation and Non Cancellation')
plt.axis('equal')
plt.show()
```

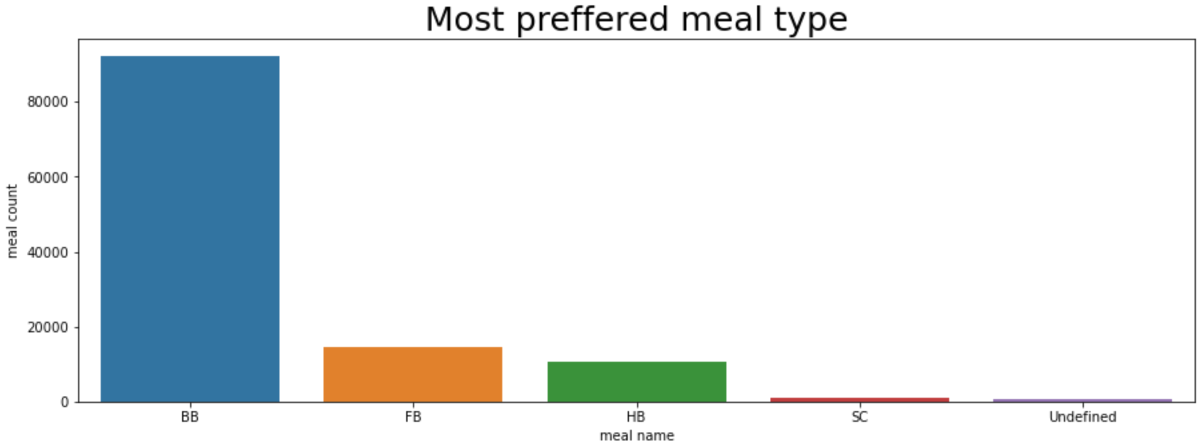


1. Why did you pick the specific chart? I had to show a part-to-a-whole relationship and percentage of both the values and here pie chart was a good option to show segmented values. 2. What is/are the insight(s) found from the chart? Here we can see that around 72.48% bookins are not canceled by guests but around 27.52% bookings are canceled by guests. 3. Will the gained insights help creating a positive business impact? Are there any insights that lead to negative growth? Justify with specific reason. This insight will help stakeholders in comparing the cancellation and non cancellation of bookings. With the help of this insight stakeholders can offer rescheduling the bookings instead of cancellation and set a flexible cancellation policy to reduce booking cancellation.

In [29]:

```
# Chart - 3 visualization code

# Counting each meal type
meal_count = df1.meal.value_counts() ##unique variable meal_count

# Extracting each meal type and storing in a variable
meal_name = df1['meal'].unique()

# Creating a dataset of each meal type and count
meal_df = pd.DataFrame(zip(meal_name,meal_count), columns = ['meal name', 'meal count'])

# Visualising the values on a bar chart
plt.figure(figsize=(15,5))
g = sns.barplot(data=meal_df, x='meal name', y ='meal count')
g.set_xticklabels(meal_df['meal name'])
plt.title('Most preffered meal type', fontsize=25)
plt.show()
```

```
C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead
```



Meal type variable description:

BB - (Bed and Breakfast)

HB- (Half Board)

FB- (Full Board)

SC- (Self Catering)

1. Why did you pick the specific chart? There were 4 values to compare and Bar graphs are used to compare things between different groups that is why I used this chart.
2. What is/are the insight(s) found from the chart? After visualizing the above chart we can see that BB - (Bed and Breakfast) is the most preffered meal type by guests.
3. Will the gained insights help creating a positive business impact? Are there any insights that lead to negative growth? Justify with specific reason.

Yes, from the gained insight above now stakeholders know that BB(Bed and Breakfast) is most preferred meal type so they can arrange raw material for this meal in advance and deliver the meal without any delay.

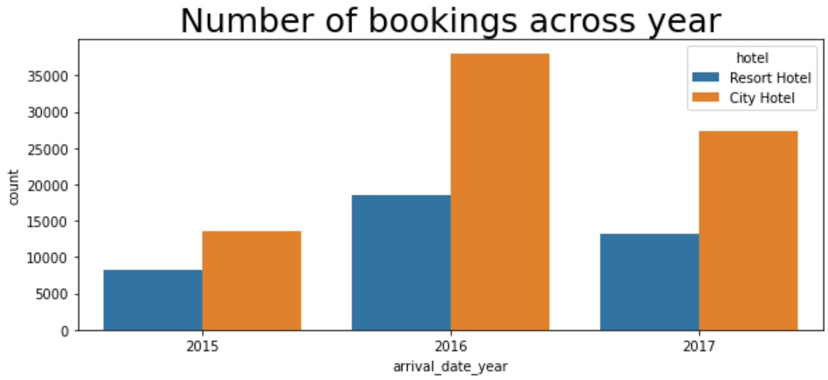
Chart - 4

Which year has the most bookings ?

In [30]:

```
# Chart - 4 visualization code
# Plotting with countplot
plt.figure(figsize=(10,4))
sns.countplot(x=df1['arrival_date_year'],hue=df1['hotel'])
plt.title("Number of bookings across year", fontsize = 25)
plt.show()
```

```
C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead
```



1. Why did you pick the specific chart? Bar graphs are used to compare things between different groups that is why I used this chart. 2. What is/are the insight(s) found from the chart? From above insight I found out that hotel was booked most times in year 2016. 3. Will the gained insights help creating a positive business impact? Are there any insights that lead to negative growth? Justify with specific reason. Above insight shows that number of booking was declined after year 2016. Stakeholders can now what went wrong after 2016 and fix that problem to increase the umber of bookings. One way to do this is ask for feedbacks from guests and have a meeting with old employees who else were serving int the year 2016.

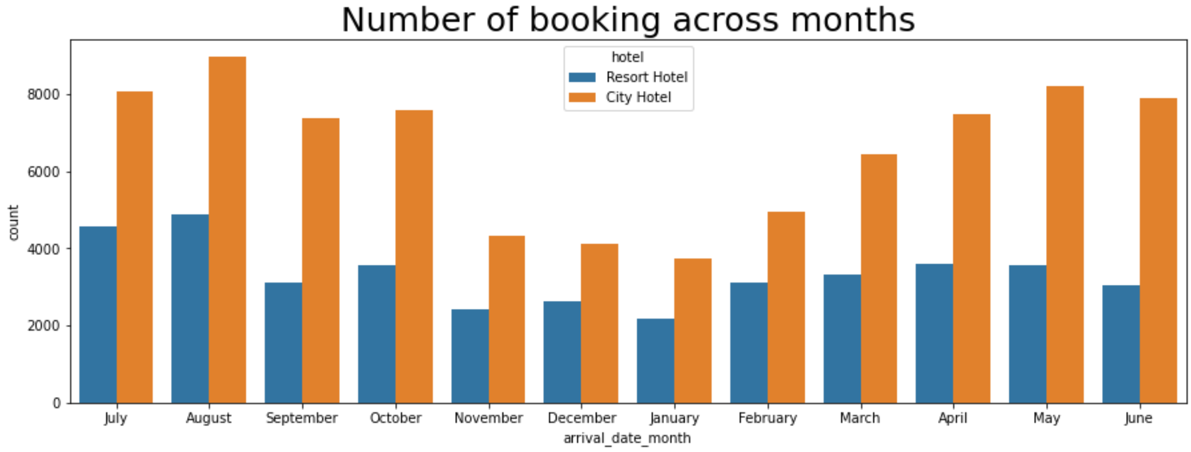
Chart - 5

Which month has the most bookings in each hotel type?

In [31]:

```
# Chart - 5 visualization code
plt.figure(figsize=(15,5))
sns.countplot(x=df1['arrival_date_month'],hue=df1['hotel'])
plt.title("Number of booking across months", fontsize = 25)
plt.show()
```

```
C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead
```



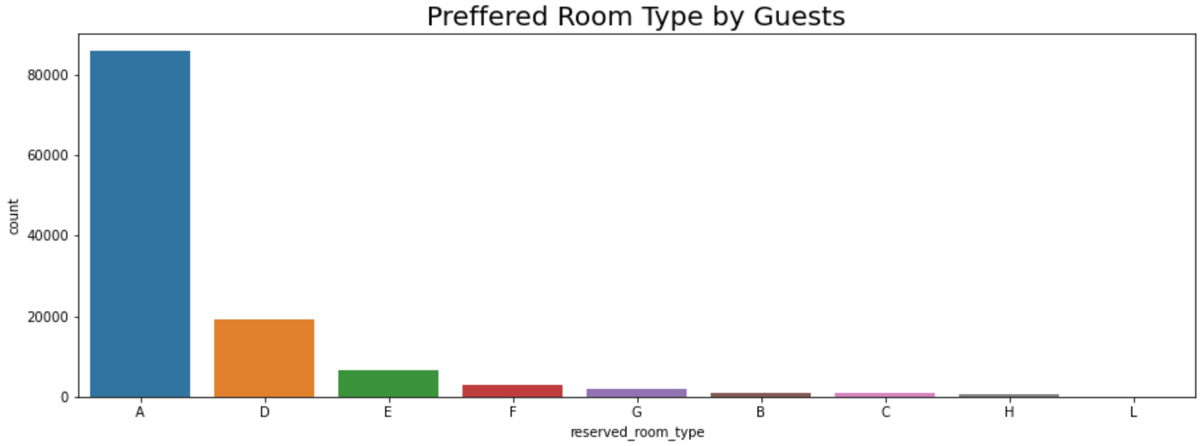
1. Why did you pick the specific chart? I had to compare values across the months and for that bar chart was one of the best choice. 2. What is/are the insight(s) found from the chart? Above insight shows that August and July ware 2 most busy months in compare to others. 3. Will the gained insights help creating a positive business impact? Are there any insights that lead to negative growth? Justify with specific reason. There is negative insight but hotel can use this insight to arrange everything in advance and welcome their guest in the best way possible and hotel can also run some promotional offer in these 2 months to attract more guests.

Chart - 6

Which room type is most preffered by guests?

In [35]:
```
# Chart - 8 visualization code
# Setting the figure size
plt.figure(figsize=(15,5))

# Plotting the values in chart
sns.countplot(x=df1['reserved_room_type'],order=df1['reserved_room_type'].value_counts().index)

# Setting the title
plt.title('Preffered Room Type by Guests', fontsize = 20)

# Show the chart
plt.show()
```

```
C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead
```



1. Why did you pick the specific chart? A bar plot shows catergorical data as rectangular bars with the height of bars proportional to the value they represent. It is often used to compare between values of different categories in the data. 2. What is/are the insight(s) found from the chart? By observing the above chart we can understand that the room type A most preffered (almost 55,000) by the guests while booking the hotel. 3. Will the gained insights help creating a positive business impact? Are

there any insights that lead to negative growth? Justify with specific reason. As it is clear that room type A is most used hotel should increase the number of A type room to maximize the revenue.

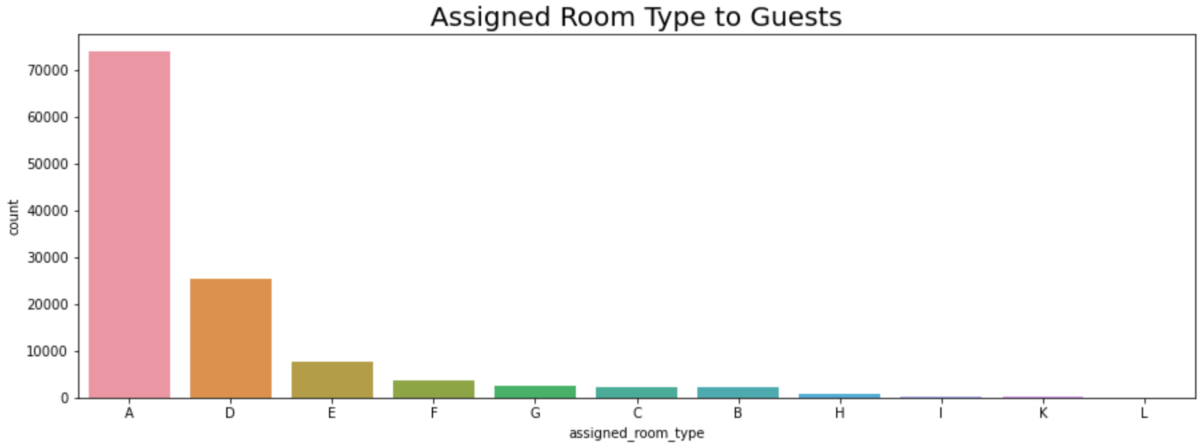
Chart - 7

Which room type is most assigned?

In [52]:

```
# Chart - 9 visualization code
# Setting the figure size
plt.figure(figsize=(15,5))

# Plotting the values
sns.countplot(x=df1['assigned_room_type'], order = df1['assigned_room_type'].value_counts().index)

# Setting the title
plt.title('Assigned Room Type to Guests', fontsize = 20)

# show the chart
plt.show()
```

```
C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead
```



1. Why did you pick the specific chart? A bar plot shows catergorical data as rectangular bars with the height of bars proportional to the value they represent. 2. What is/are the insight(s) found from the chart? From the above chart it is clear that room type A is most assigned to guests. 3. Will the gained insights help creating a positive business impact? Are there any insights that lead to negative growth? Justify with specific reason. In the 8th chart we saw that around 55,000 guests preffered room type A but 45,000 people were assigned A type room. This could be a reason to cancel the bookings. Hotel could increase A type room to decrease cancellation.

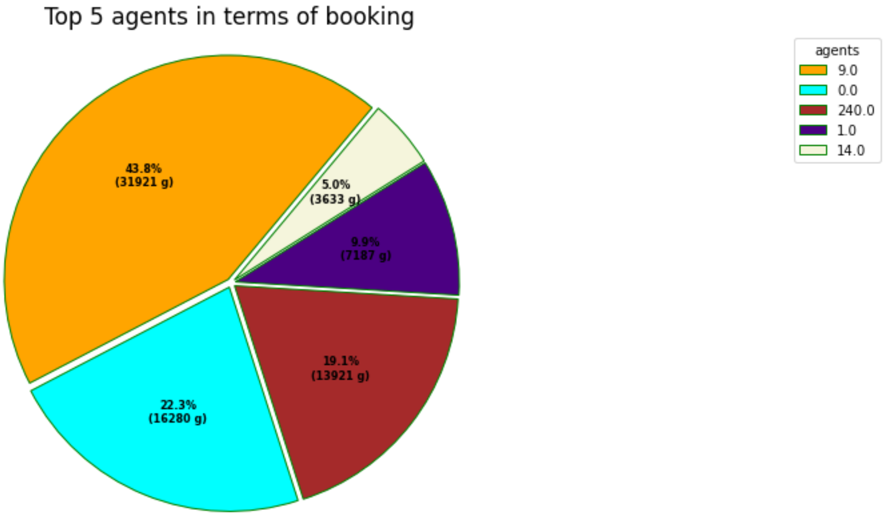
Chart - 8

Top 5 agents in terms of most bookings?

In [53]:

```
# Chart - 10 visualization code
# Creating a dataset by grouping by agent column and it's count
agents = df1.groupby(['agent'])['agent'].agg({'count'}).reset_index().rename(columns={'count':'Booking Count'}
                                                                            ).sort_values(by = 'Booking Count', ascending = False

# Extracting top 5 agents by booking count
top_5 = agents[:5]

# Explosion
explode = (0.02,0.02,0.02,0.02,0.02)

# Colors
colors = ( "orange", "cyan", "brown", "indigo", "beige")

# Wedge properties
wp = { 'linewidth' : 1, 'edgecolor' : "green" }

# Creating autocpt arguments
def func(pct, allvalues):
    absolute = int(pct / 100.*np.sum(allvalues))
    return "{:.1f}%\n({:d} g)".format(pct, absolute)

# Plotting the values
fig, ax = plt.subplots(figsize =(15, 7))
wedges, texts, autotexts = ax.pie(top_5['Booking Count'],
                                  autopct = lambda pct: func(pct, top_5['Booking Count']),
                                  explode = explode,
                                  shadow = False,
                                  colors = colors,
                                  startangle = 50,
                                  wedgeprops = wp)
```

```
# Adding legend
ax.legend(wedges, top_5['agent'],
          title ="agents",
          loc ="upper left",
          bbox_to_anchor =(1, 0, 0.5, 1))

plt.setp(autotexts, size = 8, weight ="bold")
ax.set_title("Top 5 agents in terms of booking", fontsize = 17)

# Show chart
plt.axis('equal')
plt.show()
```

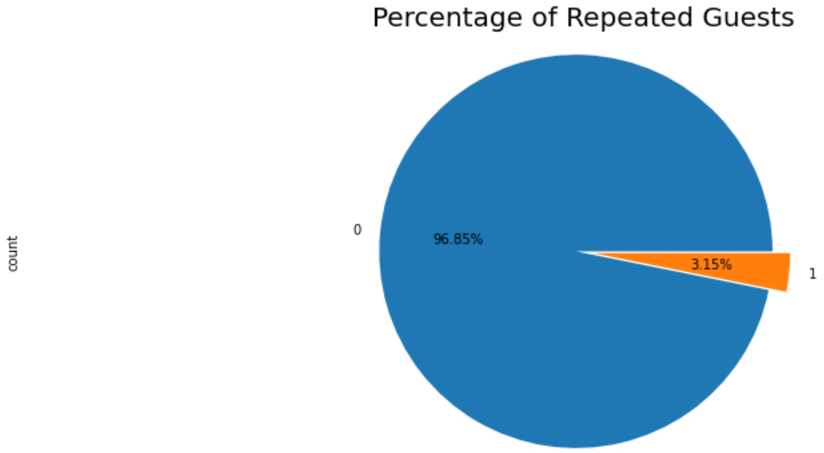


1. Why did you pick the specific chart? A pie chart helps organize and show data as a percentage of a whole 2. What is/are the insight(s) found from the chart? We can see that agent number 9 has made the most number of bookings followed by agent number 240, 0, 14 and 7. 3. Will the gained insights help creating a positive business impact? Are there any insights that lead to negative growth? Justify with specific reason. Hotel can offer them bonus for their incredible work and to motivate them. This will help to increase the revenue.

Chart - 9

What is the percentage of repeated guests?

In [54]:
```
# Chart - 11 visualization code
# Creating a variable containing guests with their repeated counts
rep_guests = df1['is_repeated_guest'].value_counts()

# Plotting the values in a pie chart
rep_guests.plot.pie(autopct='%1.2f%%', explode=(0.00,0.09), figsize=(15,6), shadow=False)

# Setting the title
plt.title('Percentage of Repeated Guests', fontsize=20)

# Setting the chart in centre
plt.axis('equal')

# Show the chart
plt.show()
```

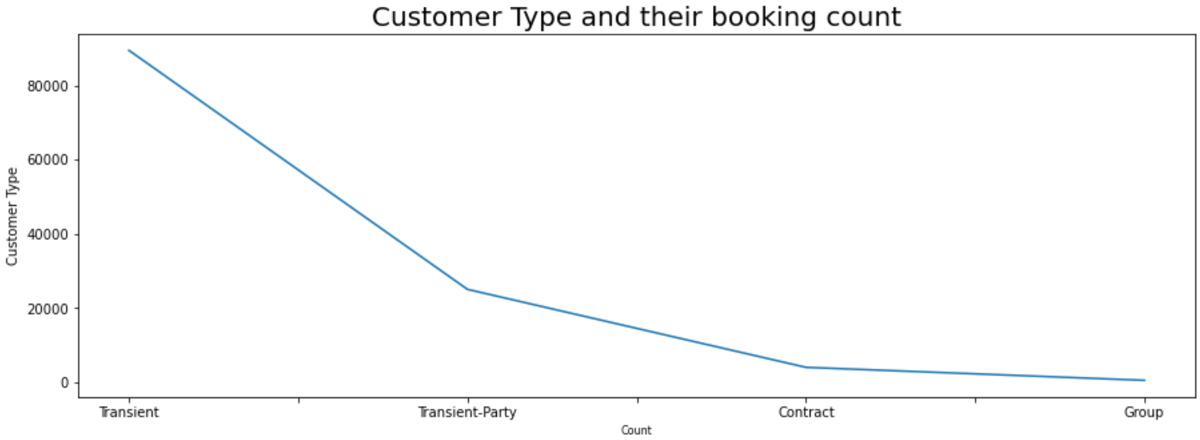


1. Why did you pick the specific chart? A pie chart helps organize and show data as a percentage of a whole 2. What is/are the insight(s) found from the chart? From the above insight we can see that 3.86% guests are repeated guests. 3. Will the gained insights help creating a positive business impact? Are there any insights that lead to negative growth? Justify with specific reason. We can see that number of repeated guests is very low and it shows negative growth of the hotel. Hotel can offer loyality discount to their guests to increase repeated guests.

Chart - 10

Which customer type has the most booking?

In [55]:
```
# Chart - 12 visualization code
cust_type = df1['customer_type'].value_counts()

# Plotting the values in a line chart
cust_type.plot(figsize=(15,5))

# Setting the x label , y label and title
plt.xlabel('Count', fontsize=8)
plt.ylabel('Customer Type', fontsize=10)
plt.title('Customer Type and their booking count', fontsize=20)

# Show the chart
plt.show()
```



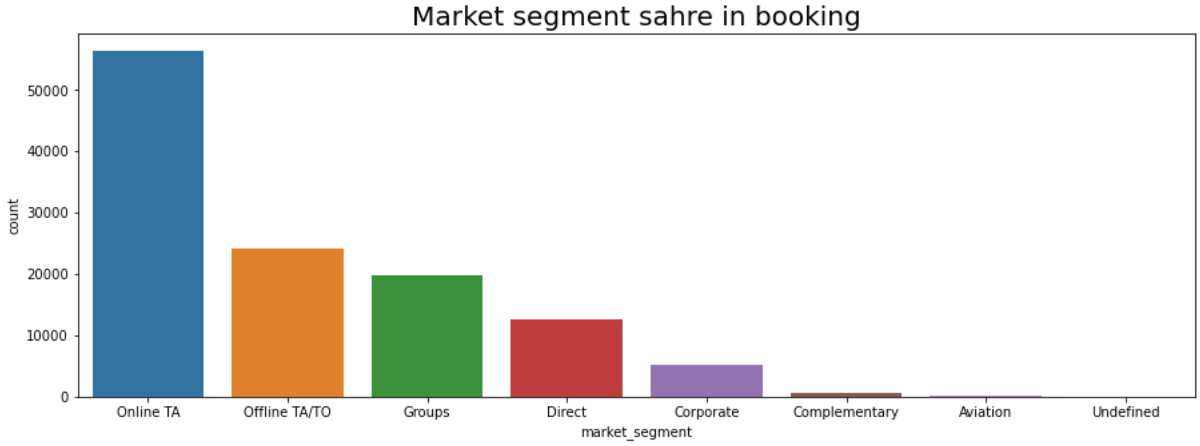
1. Why did you pick the specific chart? Line graphs are used to track changes over different categories. 2. What is/are the insight(s) found from the chart? We can see that Transient customer type has most number of bookings. 3. Will the gained insights help creating a positive business impact? Are there any insights that lead to negative growth? Justify with specific reason. Hotel can run promotional offers to increase the number of bookings over other categories. such as hotel could offer discounts for groups.

Chart - 11

Which Market Segment has the most booking?

In [40]:
```
# Chart - 13 visualization code
plt.figure(figsize=(15,5))
sns.countplot(x=df1['market_segment'], order = df1['market_segment'].value_counts().index)
plt.title('Market segment sahre in booking', fontsize=20)
plt.show()
```

```
C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead
```



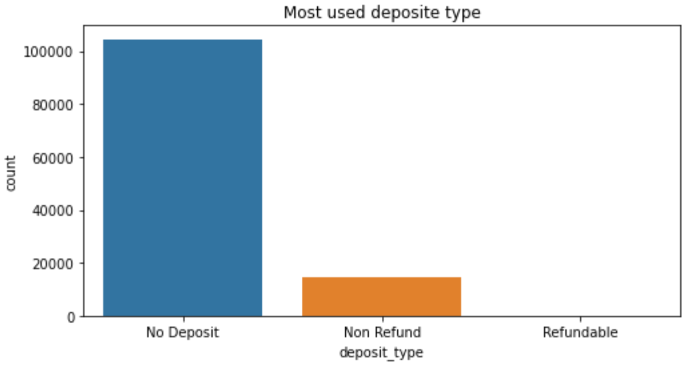
1. Why did you pick the specific chart? A bar plot shows catergorical data as rectangular bars with the height of bars proportional to the value they represent. 2. What is/are the insight(s) found from the chart? Above insight shows that Online TA (Travel Agent) has the most bookings. 3. Will the gained insights help creating a positive business impact? Are there any insights that lead to negative growth? Justify with specific reason. There is no negative growth. Hotel should come up with some great idea to increase sahre among other market segments to increase the revenue.

Chart -12

Which deposite type is most preffered?

In [41]:
```
# Visualization Code
# Counting each deposte type
deposite = df1['deposit_type'].value_counts().index

# Setting the chart size
plt.figure(figsize=(8,4))

# plotting the values
sns.countplot(x=df1['deposit_type'], order= deposite)
plt.title('Most used deposite type')
plt.show()
```

```
C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead
```



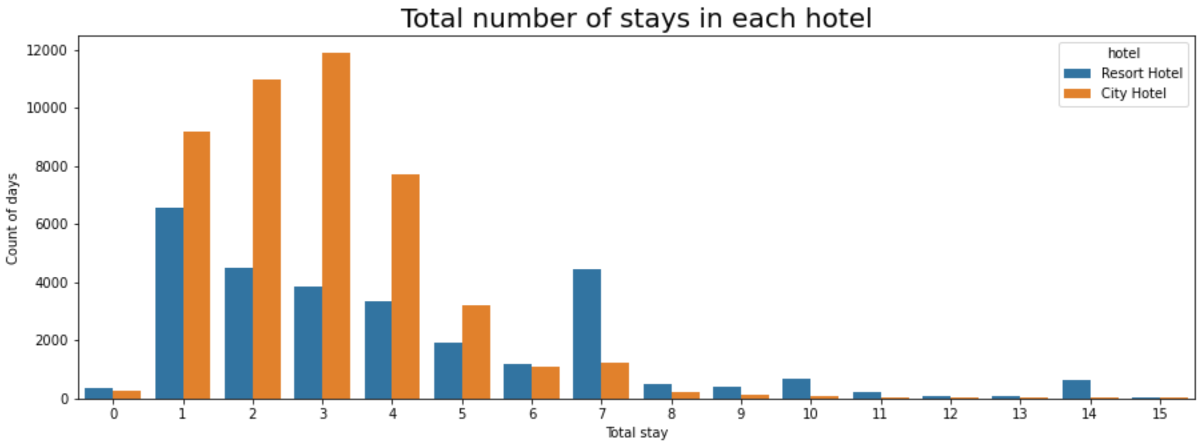
Bivariate and Multivariate Analysis

Chart - 13

How long people stay in the hotel?

In [42]:
```
# Chart - 11 visualization code
# Creating a not cancelled dataframe
not_cancelled_df = df1[df1['is_canceled'] == 0]
# Creating a hotel stay dataframe
hotel_stay = not_cancelled_df[not_cancelled_df['total_stay'] <= 15]  #Visualizing pattern till 15days stay


# Setting plot size and plotting barchart
plt.figure(figsize = (15,5))
sns.countplot(x = hotel_stay['total_stay'], hue = hotel_stay['hotel'])

# Adding the label of the chart
plt.title('Total number of stays in each hotel',fontsize = 20)
plt.xlabel('Total stay')
plt.ylabel("Count of days")
plt.show()
```

```
C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead
```



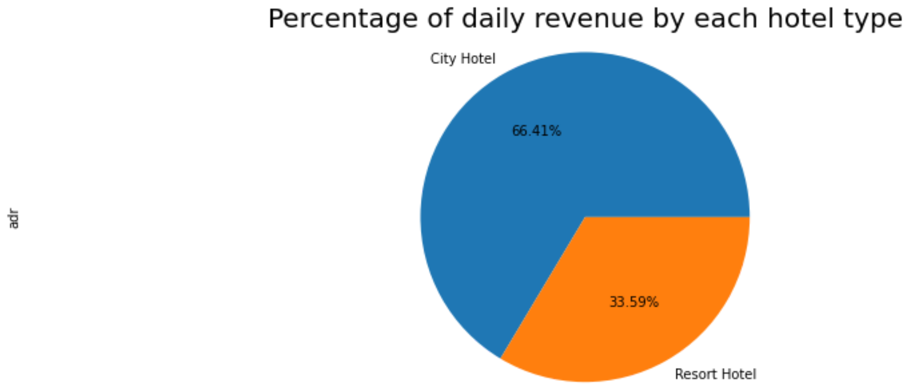
From the above chart we can see that in City hotel most people stay for 3 days and in Resort hotel most people stay for only 1 day. Hotel should work on to increase total stay in Resort hotel to increase revenue.

Chart-14

Which hotel makes most revenue?

In [43]:

```
# Counting the revnue for each hotel type using groupby function
most_rev = df1.groupby('hotel')['adr'].count()

# Plotting the values in a pie chart
most_rev.plot.pie(autopct='%1.2f%%', figsize=(15,5))

# Setting the title
plt.title('Percentage of daily revenue by each hotel type', fontsize=20)
plt.axis('equal')

# Show the chart
plt.show()
```



From the above insight it is clear that City hotel has more share in revenue generation over Resort Hotel. Stake holderscould improve the service of Resort hotel so that people stay more in resort hotel and increase the revenue.

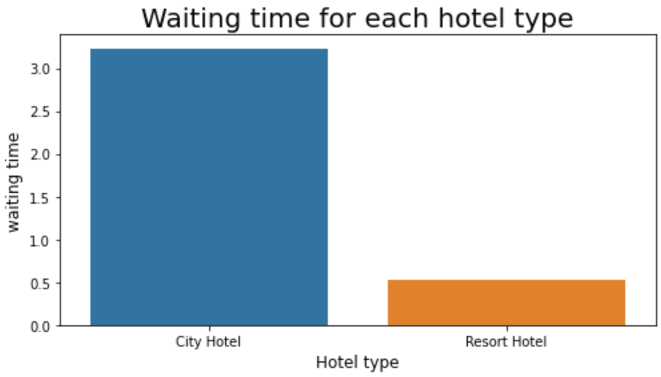
Chart - 15

Which hotel has the longer waiting time?

In [44]:

```
# Grouping by hotel and taking the mean of days in waiting list
waiting_time_df = df1.groupby('hotel')['days_in_waiting_list'].mean().reset_index()
# Waiting_time_df

# Setting the plot size
plt.figure(figsize=(8,4))

# Plotting the barchart
sns.barplot(x=waiting_time_df['hotel'],y=waiting_time_df['days_in_waiting_list'])

# Setting the labels
plt.xlabel('Hotel type',fontsize=12)
plt.ylabel('waiting time',fontsize=12)
plt.title("Waiting time for each hotel type",fontsize=20)

# Show chart
plt.show()
```

```
C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead
```



Above chart shows that City hotel has more waiting period. This could be because people stay more in City hotel as we saw in previous insight. Stakeholders should increase rooms in City hotel or convert some of rooms of Resort hotel into City Hotel to decrease the waiting time.

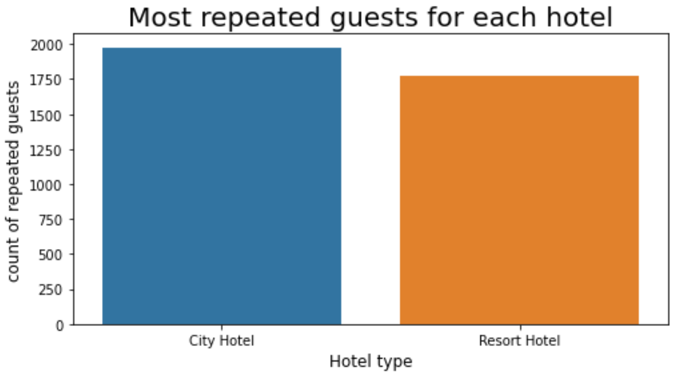
Chart - 16

Hotel with most repeated guests.

In [45]:

```
# Grouping hotel types on repeated guests
rep_guest = df1[df1['is_repeated_guest']==1].groupby('hotel').size().reset_index()

# Renaming the column
rep_guest = rep_guest.rename(columns={0:'number_of_repated_guests'})

# Setting the chart size
plt.figure(figsize=(8,4))

# Plotting the values in a bar chart
sns.barplot(x=rep_guest['hotel'],y=rep_guest['number_of_repated_guests'])

# Setting the labels and title
plt.xlabel('Hotel type', fontsize=12)
plt.ylabel('count of repeated guests', fontsize=12)
plt.title('Most repeated guests for each hotel', fontsize=20)

# Show Chart
plt.show()
```

```
C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead
```



We can see that Resort Hotel has slightly more repeated guests over City Hotel this could be because of less waiting time in Resort Hotel and better service there because of less rush.

Chart - 17

What is the adr across different months?

In [46]:

```python
# Grouping arrival_month and hotel on mean of adr
bookings_months=df1.groupby(['arrival_date_month','hotel'])['adr'].mean().reset_index()

# Creating a month list to order the months in ascending
months = ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October', 'November', 'December']

# Creating a dataset of months, hotel and their adr
bookings_months['arrival_date_month']=pd.Categorical(bookings_months['arrival_date_month'],categories=months,ordered=True)

# Sorting the months
bookings_months=bookings_months.sort_values('arrival_date_month')
bookings_months
```

Out[46]:

	arrival_date_month	hotel	adr
8	January	City Hotel	82.754477
9	January	Resort Hotel	49.507033
6	February	City Hotel	85.327519
7	February	Resort Hotel	55.189716
15	March	Resort Hotel	57.554652
14	March	City Hotel	92.973339
0	April	City Hotel	111.397415
1	April	Resort Hotel	77.849496
17	May	Resort Hotel	78.758134
16	May	City Hotel	121.764614
13	June	Resort Hotel	110.481032
12	June	City Hotel	119.186056
11	July	Resort Hotel	155.181299
10	July	City Hotel	110.945950
3	August	Resort Hotel	186.790574
2	August	City Hotel	114.857330
22	September	City Hotel	110.120296
23	September	Resort Hotel	93.252030
20	October	City Hotel	100.119313
21	October	Resort Hotel	62.132572
18	November	City Hotel	88.372486
19	November	Resort Hotel	48.313643
5	December	Resort Hotel	69.051887
4	December	City Hotel	89.209560

In [47]:

```python
# Setting the chart size
plt.figure(figsize=(15,5))

# Plotting the values in a line chart
sns.lineplot(x=bookings_months['arrival_date_month'],y=bookings_months['adr'],hue=bookings_months['hotel'])

# Setting the labels and title
plt.title('ADR across each month', fontsize=20)
plt.xlabel('Month Name', fontsize=12)
plt.ylabel('ADR', fontsize=12)

# Show chart
plt.show()
```

```
C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning:

use_inf_as_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning:

use_inf_as_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Input In [47], in <cell line: 5>()
      2 plt.figure(figsize=(15,5))
      4 # Plotting the values in a line chart
----> 5 sns.lineplot(x=bookings_months['arrival_date_month'],y=bookings_months['adr'],hue=bookings_months['hotel'])
      7 # Setting the labels and title
      8 plt.title('ADR across each month', fontsize=20)

File ~\anaconda3\lib\site-packages\seaborn\relational.py:645, in lineplot(data, x, y, hue, size, style, units, palette, hue_order, h
ue_norm, sizes, size_order, size_norm, dashes, markers, style_order, estimator, errorbar, n_boot, seed, orient, sort, err_style, err
_kws, legend, ci, ax, **kwargs)
    642 color = kwargs.pop("color", kwargs.pop("c", None))
    643 kwargs["color"] = _default_color(ax.plot, hue, color, kwargs)
--> 645 p.plot(ax, kwargs)
    646 return ax

File ~\anaconda3\lib\site-packages\seaborn\relational.py:459, in _LinePlotter.plot(self, ax, kws)
    457         lines.extend(ax.plot(unit_data["x"], unit_data["y"], **kws))
    458 else:
--> 459     lines = ax.plot(sub_data["x"], sub_data["y"], **kws)
    461 for line in lines:
    463     if "hue" in sub_vars:

File ~\anaconda3\lib\site-packages\matplotlib\axes\_axes.py:1632, in Axes.plot(self, scalex, scaley, data, *args, **kwargs)
   1390 """
   1391 Plot y versus x as lines and/or markers.
   1392
   (...)
   1629 (``'green'``) or hex strings (``'#008000'``).
   1630 """
   1631 kwargs = cbook.normalize_kwargs(kwargs, mlines.Line2D)
-> 1632 lines = [*self._get_lines(*args, data=data, **kwargs)]
   1633 for line in lines:
   1634     self.add_line(line)

File ~\anaconda3\lib\site-packages\matplotlib\axes\_base.py:312, in _process_plot_var_args.__call__(self, data, *args, **kwargs)
    310     this += args[0],
    311     args = args[1:]
--> 312 yield from self._plot_args(this, kwargs)

File ~\anaconda3\lib\site-packages\matplotlib\axes\_base.py:487, in _process_plot_var_args._plot_args(self, tup, kwargs, return_kwar
gs)
    484         kw[prop_name] = val
    486 if len(xy) == 2:
--> 487     x = _check_1d(xy[0])
    488     y = _check_1d(xy[1])
    489 else:

File ~\anaconda3\lib\site-packages\matplotlib\cbook\__init__.py:1327, in _check_1d(x)
   1321 with warnings.catch_warnings(record=True) as w:
   1322     warnings.filterwarnings(
   1323         "always",
   1324         category=Warning,
   1325         message='Support for multi-dimensional indexing')
-> 1327     ndim = x[:, None].ndim
   1328     # we have definitely hit a pandas index or series object
   1329     # cast to a numpy array.
   1330     if len(w) > 0:

File ~\anaconda3\lib\site-packages\pandas\core\series.py:1072, in Series.__getitem__(self, key)
   1069     key = np.asarray(key, dtype=bool)
   1070     return self._get_rows_with_mask(key)
-> 1072 return self._get_with(key)

File ~\anaconda3\lib\site-packages\pandas\core\series.py:1082, in Series._get_with(self, key)
   1077     raise TypeError(
   1078         "Indexing a Series with DataFrame is not "
   1079         "supported, use the appropriate DataFrame column"
   1080     )
   1081 elif isinstance(key, tuple):
-> 1082     return self._get_values_tuple(key)
   1084 elif not is_list_like(key):
   1085     # e.g. scalars that aren't recognized by lib.is_scalar, GH#32684
   1086     return self.loc[key]

File ~\anaconda3\lib\site-packages\pandas\core\series.py:1122, in Series._get_values_tuple(self, key)
   1117 if com.any_none(*key):
   1118     # mpl compat if we look up e.g. ser[:, np.newaxis];
   1119     #  see tests.series.timeseries.test_mpl_compat_hack
   1120     # the asarray is needed to avoid returning a 2D DatetimeArray
   1121     result = np.asarray(self._values[key])
-> 1122     disallow_ndim_indexing(result)
   1123     return result
   1125 if not isinstance(self.index, MultiIndex):

File ~\anaconda3\lib\site-packages\pandas\core\indexers\utils.py:341, in disallow_ndim_indexing(result)
    333 """
    334 Helper function to disallow multi-dimensional indexing on 1D Series/Index.
    335
   (...)
    338 in GH#30588.
    339 """
    340 if np.ndim(result) > 1:
--> 341     raise ValueError(
    342         "Multi-dimensional indexing (e.g. `obj[:, None]`) is no longer "
    343         "supported. Convert to a numpy array before indexing instead."
    344     )

ValueError: Multi-dimensional indexing (e.g. `obj[:, None]`) is no longer supported. Convert to a numpy array before indexing instea
d.
```

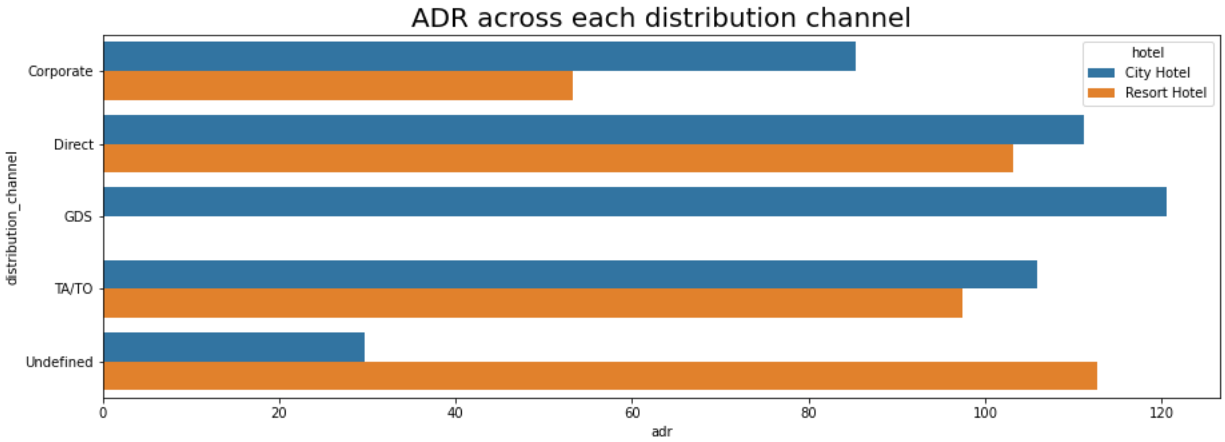
City Hotel : It is clear that City Hotel generates more revenue in May months in comparison to other months. Resort Hotel : Resort Hotel generates more revenue in between July and August months. Stakeholders could prepare in advance for these 2 months as these 2 months generate more revenue.

Chart - 18

Which distribution channel has highest adr?

In [48]:
```
# Grouping dist_channel and hotels on their adr
dist_channel_adr = df1.groupby(['distribution_channel','hotel'])['adr'].mean().reset_index()

# Setting the figure size
plt.figure(figsize=(15,5))

# Creating a horizontal bar chart
sns.barplot(x='adr', y='distribution_channel', data=dist_channel_adr, hue='hotel')

# Setting the title
plt.title('ADR across each distribution channel', fontsize=20)

# Show chart
plt.show()
```

```
C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead

C:\Users\Rajarshi\anaconda3\lib\site-packages\seaborn\_oldcore.py:1498: FutureWarning:

is_categorical_dtype is deprecated and will be removed in a future version. Use isinstance(dtype, CategoricalDtype) instead
```



GDS has contributed more in generating the ADR. GDS is a worldwide conduit between travel bookers and suppliers, such as hotels and other accommodation providers. It communicates live product, price and availability data to travel agents and online booking engines, and allows for automated transactions. Direct- means that bookings are directly made with the respective hotels TA/TO- means that booings are made through travel agents or travel operators. Undefined- Bookings are undefined. may be customers made their bookings on arrival.

Chart - 21 - Correlation Heatmap

In [49]:
```
pip install --upgrade pandas
```

```
Requirement already satisfied: pandas in c:\users\rajarshi\anaconda3\lib\site-packages (2.1.1)
Requirement already satisfied: numpy>=1.22.4 in c:\users\rajarshi\anaconda3\lib\site-packages (from pandas) (1.26.0)
Requirement already satisfied: tzdata>=2022.1 in c:\users\rajarshi\anaconda3\lib\site-packages (from pandas) (2023.3)
Requirement already satisfied: pytz>=2020.1 in c:\users\rajarshi\anaconda3\lib\site-packages (from pandas) (2021.3)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\rajarshi\anaconda3\lib\site-packages (from pandas) (2.8.2)
Requirement already satisfied: six>=1.5 in c:\users\rajarshi\anaconda3\lib\site-packages (from python-dateutil>=2.8.2->pandas) (1.1
6.0)
Note: you may need to restart the kernel to use updated packages.
```

In [50]:

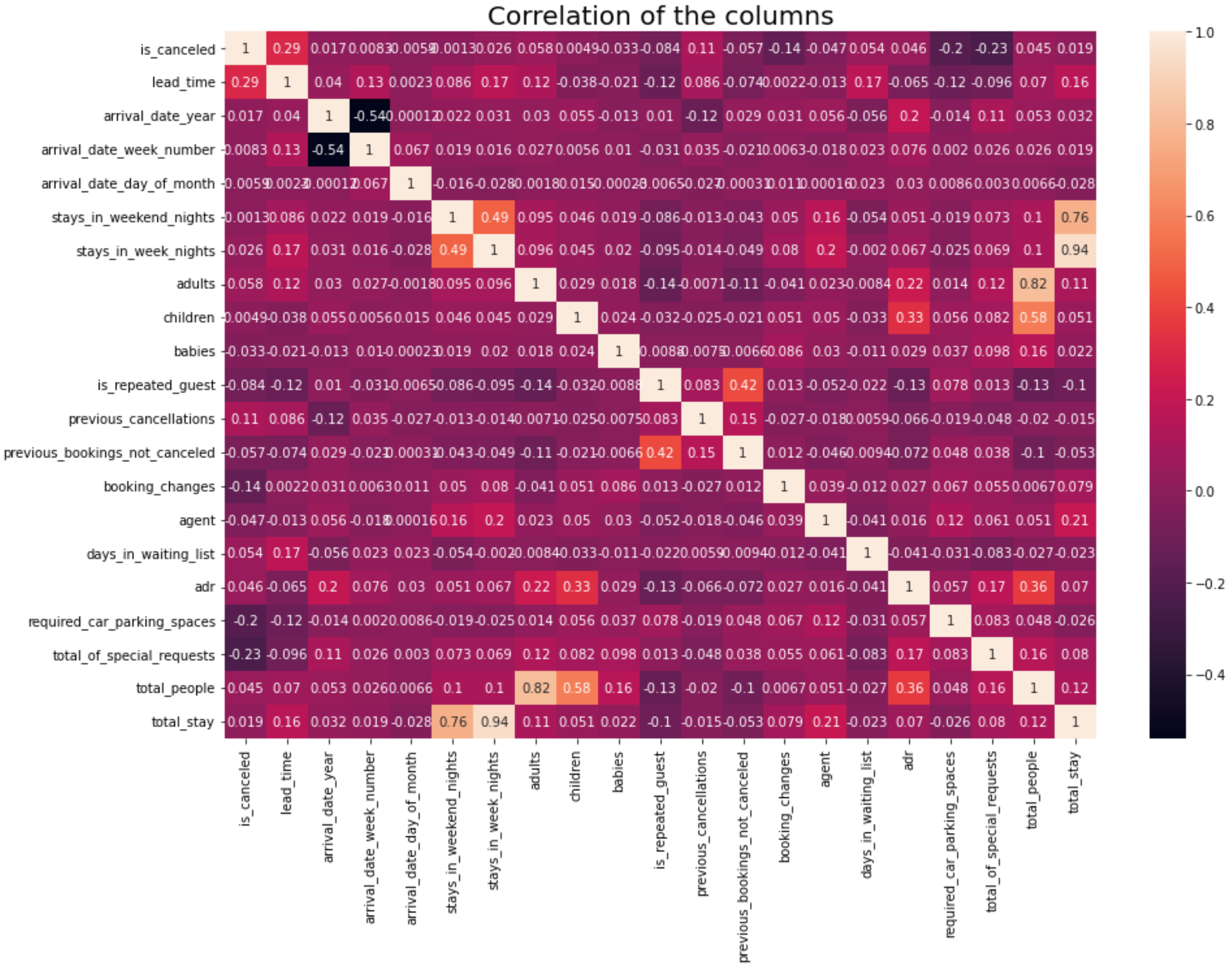
```
# Correlation Heatmap visualization code
# Setting the chart size
plt.figure(figsize=(15,10))
```

Out[50]:

```
<Figure size 1080x720 with 0 Axes>

<Figure size 1080x720 with 0 Axes>
```

In [51]:

```
# Correlation Heatmap visualization code
# Setting the chart size
plt.figure(figsize=(15,10))

# Creating heatmap to see correlation of each columns
sns.heatmap(df1.corr(numeric_only=True),annot=True)          # Setting the numeric only colun to True to avoid warning

# Setting the title
plt.title('Correlation of the columns', fontsize=20)

# Show heatmap
plt.show()
```



In [56]:

```
pip install nbconvert
```

```
Requirement already satisfied: nbconvert in c:\users\rajarshi\anaconda3\lib\site-packages (6.4.4)
Requirement already satisfied: pygments>=2.4.1 in c:\users\rajarshi\anaconda3\lib\site-packages (from nbconvert) (2.11.2)
Requirement already satisfied: entrypoints>=0.2.2 in c:\users\rajarshi\anaconda3\lib\site-packages (from nbconvert) (0.4)
Requirement already satisfied: pandocfilters>=1.4.1 in c:\users\rajarshi\anaconda3\lib\site-packages (from nbconvert) (1.5.0)
Requirement already satisfied: beautifulsoup4 in c:\users\rajarshi\anaconda3\lib\site-packages (from nbconvert) (4.11.1)
Requirement already satisfied: traitlets>=5.0 in c:\users\rajarshi\anaconda3\lib\site-packages (from nbconvert) (5.1.1)
Requirement already satisfied: mistune<2,>=0.8.1 in c:\users\rajarshi\anaconda3\lib\site-packages (from nbconvert) (0.8.4)
Requirement already satisfied: nbclient<0.6.0,>=0.5.0 in c:\users\rajarshi\anaconda3\lib\site-packages (from nbconvert) (0.5.13)
Requirement already satisfied: bleach in c:\users\rajarshi\anaconda3\lib\site-packages (from nbconvert) (4.1.0)
Requirement already satisfied: jinja2>=2.4 in c:\users\rajarshi\anaconda3\lib\site-packages (from nbconvert) (3.0.3)
Requirement already satisfied: testpath in c:\users\rajarshi\anaconda3\lib\site-packages (from nbconvert) (0.5.0)
Requirement already satisfied: jupyter-core in c:\users\rajarshi\anaconda3\lib\site-packages (from nbconvert) (4.9.2)
Requirement already satisfied: jupyterlab-pygments in c:\users\rajarshi\anaconda3\lib\site-packages (from nbconvert) (0.1.2)
Requirement already satisfied: nbformat>=4.4 in c:\users\rajarshi\anaconda3\lib\site-packages (from nbconvert) (5.3.0)
Requirement already satisfied: defusedxml in c:\users\rajarshi\anaconda3\lib\site-packages (from nbconvert) (0.7.1)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\rajarshi\anaconda3\lib\site-packages (from jinja2>=2.4->nbconvert) (2.1.
2)
Requirement already satisfied: jupyter-client>=6.1.5 in c:\users\rajarshi\anaconda3\lib\site-packages (from nbclient<0.6.0,>=0.5.0->
nbconvert) (6.1.12)
Requirement already satisfied: nest-asyncio in c:\users\rajarshi\anaconda3\lib\site-packages (from nbclient<0.6.0,>=0.5.0->nbconver
t) (1.5.5)
Requirement already satisfied: tornado>=4.1 in c:\users\rajarshi\anaconda3\lib\site-packages (from jupyter-client>=6.1.5->nbclient<
0.6.0,>=0.5.0->nbconvert) (6.1)
Requirement already satisfied: python-dateutil>=2.1 in c:\users\rajarshi\anaconda3\lib\site-packages (from jupyter-client>=6.1.5->nb
client<0.6.0,>=0.5.0->nbconvert) (2.8.2)
Requirement already satisfied: pyzmq>=13 in c:\users\rajarshi\anaconda3\lib\site-packages (from jupyter-client>=6.1.5->nbclient<0.6.
0,>=0.5.0->nbconvert) (22.3.0)
Requirement already satisfied: pywin32>=1.0 in c:\users\rajarshi\anaconda3\lib\site-packages (from jupyter-core->nbconvert) (302)
Requirement already satisfied: jsonschema>=2.6 in c:\users\rajarshi\anaconda3\lib\site-packages (from nbformat>=4.4->nbconvert) (4.
4.0)
Requirement already satisfied: fastjsonschema in c:\users\rajarshi\anaconda3\lib\site-packages (from nbformat>=4.4->nbconvert) (2.1
5.1)
Requirement already satisfied: pyrsistent!=0.17.0,!=0.17.1,!=0.17.2,>=0.14.0 in c:\users\rajarshi\anaconda3\lib\site-packages (from
jsonschema>=2.6->nbformat>=4.4->nbconvert) (0.18.0)
Requirement already satisfied: attrs>=17.4.0 in c:\users\rajarshi\anaconda3\lib\site-packages (from jsonschema>=2.6->nbformat>=4.4->
nbconvert) (21.4.0)
Requirement already satisfied: six>=1.5 in c:\users\rajarshi\anaconda3\lib\site-packages (from python-dateutil>=2.1->jupyter-client>
=6.1.5->nbclient<0.6.0,>=0.5.0->nbconvert) (1.16.0)
Requirement already satisfied: soupsieve>1.2 in c:\users\rajarshi\anaconda3\lib\site-packages (from beautifulsoup4->nbconvert) (2.3.
1)
Requirement already satisfied: webencodings in c:\users\rajarshi\anaconda3\lib\site-packages (from bleach->nbconvert) (0.5.1)
Requirement already satisfied: packaging in c:\users\rajarshi\anaconda3\lib\site-packages (from bleach->nbconvert) (21.3)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in c:\users\rajarshi\anaconda3\lib\site-packages (from packaging->bleach->nb
convert) (3.0.4)
Note: you may need to restart the kernel to use updated packages.
```

In [57]:

```
conda install pandoc nbconvert
```

In [ ]: