Problem Statement: Uncovering Patterns in Hotel Booking Data for Operational Efficiency and Revenue Growth

Overview:

The goal of this exploratory data analysis is to investigate a Resort Hotel's booking dataset to identify key patterns, trends, and relationships that can support data-driven decision-making.

This includes analyzing booking behavior, customer demographics, pricing strategies, and operational factors such as room assignments and special requests. The study will involve cleaning and preprocessing the data, examining variable correlations, and validating key business assumptions through hypothesis testing.

Core Objectives:

- Understand how customer attributes and booking behaviors impact revenue.
- Identify trends in lead time, stay duration, and booking channels.
- Detects inconsistencies or anomalies in room allocation and guest handling.
- Explore relationships between booking patterns and customer satisfaction indicators.
- Evaluate whether specific operational or customer variables significantly affect outcomes such as ADR or room upgrades.

Data Description : -

Feature Explanations:

1. hotel
    - Type of hotel: *Resort Hotel* or *City Hotel*.
2. **is_canceled**
    - Indicates whether a booking was **canceled (1)** or **not (0)**.
3. **lead_time**
    - The number of days between the **booking date** and the **arrival date**.
    - Higher lead time might be associated with higher cancellation rates.

4. **arrival_date_year**
    - The **year** of the arrival date.
5. **arrival_date_month**
    - The **month** of the arrival date (e.g., January, February).
6. **arrival_date_week_number**
    - The **ISO week number** of the year for the arrival date.
7. **arrival_date_day_of_month**
    - The **day of the month** the guest arrived (1–31).
8. **stays_in_weekend_nights**
    - Number of **weekend nights (Saturday/Sunday)** the guest stayed or booked.
9. **stays_in_week_nights**
    - Number of **weekday nights (Monday–Friday)** the guest stayed or booked.
10. **adults**
    - Number of **adults** in the booking.
11. **children**
    - Number of **children** in the booking.
12. **babies**
    - Number of **babies** in the booking.
13. **meal**
    - Type of meal booked (e.g., BB = Bed & Breakfast, HB = Half Board, FB = Full Board, SC = Self Catering).
    - Useful for understanding guest preferences and pricing tiers.
14. **country**

- **Country of origin** of the guest.
- Helps identify key markets and regional trends.

15. **market_segment**
    - How the guest found or booked the hotel (e.g., *TA = Travel Agents*, *TO = Tour Operators*, *Direct*, *Corporate*).
    - Useful for **segment performance analysis**.
16. **distribution_channel**
    - Channel through which the booking was made (often overlaps with market_segment).
    - Helps assess **sales strategy effectiveness**.
17. **is_repeated_guest**
    - Indicates if the guest had previously stayed at the hotel:
        i. 1 = repeated guest
        ii. 0 = first-time guest
    - Used for loyalty and customer retention analysis.
18. **previous_cancellations**
    - Number of past bookings the customer canceled before the current one.
    - High numbers may indicate **risky customers** likely to cancel.
19. **booking_changes**
    - Number of **modifications** made to the booking from creation to arrival or cancellation.
20. **deposit_type**
    - Type of **deposit** made (e.g., *No Deposit*, *Non-Refundable*, *Refundable*).

21. **agent**
    - ID of the **travel agent** who made the booking.
22. **company**
    - ID of the **company** responsible for the booking (useful for corporate clients).
23. **days_in_waiting_list**
    - Number of days the booking spent on the **waiting list** before confirmation.
24. **customer_type**
    - Type of customer (e.g., *Contract*, *Group*, *Transient*, *Transient-Party*).
25. **adr** **(Average Daily Rate)**
    - Lodging revenue per night. Calculated as:
      adr=Total Lodging RevenueTotal Nights Stayed\text{adr} = \frac{\text{Total Lodging Revenue}}{\text{Total Nights Stayed}}adr=Total Nights StayedTotal Lodging Revenue
    - Critical for revenue analysis and comparisons across segments.
26. **required_car_parking_spaces**
    - Number of parking spaces requested.
27. **previous_bookings_not_canceled**
    - Count of past **non-canceled bookings** by the customer.
28. **reserved_room_type**
    - Room type originally **booked** (coded for privacy).
29. **assigned_room_type**
    - Room type **actually allocated** at check-in (coded).
30. **total_of_special_requests**
    - Total number of **special requests** (e.g., high floor, crib, late check-out).

Step 1: Data Cleaning and Preprocessing

A crucial phase of the project involved preparing the raw data for analysis:

- Libraries and Data Loading: Essential libraries like pandas, numpy, matplotlib, and seaborn were imported, and the hotel_bookings.csv dataset was loaded.
- Initial Inspection: The dataset initially contained 119,390 rows and 32 columns.
- Handling Duplicates: A significant number of duplicate rows (31,994) were identified and removed, resulting in a cleaned dataset with 87,396 unique records.
- Missing Values Treatment:
  - Missing values in the `children`, `country`, and `agent` columns were filled using the mode (most frequent value) of each respective column.
  - The `company` column was entirely dropped due to having more than 93% missing values, making it unsuitable for imputation or direct use.
- Data Types and Structure: Data types were verified and corrected, ensuring that categorical and numerical variables were appropriately handled for subsequent analysis.
- Diagram Used:
  - A Bar Plot was used to show the number of missing values per column. This visualization helped to quickly identify which variables had incomplete data and required treatment.

Outlier Treatment

Outliers, or extreme values, in key numerical variables were identified and managed to prevent them from skewing the analysis:

- Outlier Detection: Outliers were detected in `lead_time` and `adr` using:
  - Boxplots: These were used to visually identify extreme values.
  - Skewness Distribution Plots: These plots helped analyze the shape of the data distribution, highlighting where extreme values might exist.

- IQR-Based Capping: The Interquartile Range (IQR) method was applied to `lead_time` and `adr` to cap (limit) extreme values. This effectively reduced the influence of outliers and improved the data quality for analysis.

- Diagrams Used:

  - Boxplots and Distribution Plots were shown both before and after outlier removal. The "before" plots highlighted the presence of extreme values, while the "after" plots demonstrated the effectiveness of IQR capping in normalizing the data.

Feature Engineering & Data Wrangling

This phase involved preparing features for modeling and analysis:

- Variable Type Identification: Dataset columns were categorized into:

  - Categorical Variables (e.g., `hotel`, `customer_type`, `reservation_status`).

  - Discrete Numerical Variables (e.g., `is_canceled`, `agent`, `adr`, though `adr` is more commonly continuous).

  - Continuous Numerical Variables (e.g., `lead_time`, `adr`).

- Discrete Variable Analysis: Discrete numerical variables were analyzed independently considering their limited unique values.

- Categorical Variable Handling: Categorical features were appropriately prepared for grouping, plotting, and summary statistics.

- Continuous Variable Identification: `lead_time` and `adr` were specifically identified for outlier treatment and statistical analysis.

- Diagram Used:

  - Value Counts per Variable displays the number of unique values in each column. This was instrumental in supporting the classification of variables into categorical, discrete, and continuous types.

Step 2: Exploratory Data Analysis (EDA)

Various visualizations were employed to uncover patterns and trends:

- Key Visualizations:
  - Univariate Analysis (Histograms): Histograms were plotted for features such as `adr` (Average Daily Rate), `lead_time`, `customer_type`, and `market_segment`, as well as other categorical and numerical variables, to observe their individual distributions. This helps understand the spread and frequency of values for single variables.
  - Bivariate Analysis:
    - Boxplots were used to compare `ADR` across different `market segments`, helping to visualize the distribution of rates within each segment.
    - A Heatmap of the correlation matrix was generated to identify relationships among numerical variables and detect multicollinearity. This colorful grid shows how strongly pairs of numerical variables move together.
  - Time Series Analysis: Booking trends were examined by month and year. These charts, likely line plots, revealed seasonality and overall demand patterns over time.
- Insights Gained:
  - Online Travel Agencies (OTA) dominate the booking volume for both hotel types.
  - ADR shows noticeable variation across different distribution channels.
  - Guests with longer `lead_time` tend to make more booking changes, suggesting a longer decision window.

Step 3: Correlation Analysis

A deeper look into how numerical variables relate to each other:

- ADR Correlations: `adr` shows a moderate positive correlation with `total_of_special_requests` and `lead_time`. This suggests that bookings with more special requests or longer lead times tend to have higher average daily rates.

- Weak Correlations: Weak correlations were observed between `booking_changes` and `adr`, and between `previous_cancellations` and `is_canceled`. These findings indicate that while some variables are mildly related to pricing or behavior, others might have minimal impact.

- Diagram Used:

    - A Heatmap of the Correlation Matrix was used again. This visualization highlighted stronger and weaker relationships among numerical features, and was useful for identifying patterns and potential multicollinearity issues.

Step 4: Hypothesis Testing

Statistical tests were performed to validate specific business assumptions:

- ADR: OTA vs. Direct Bookings
  - Null Hypothesis (H_0): There is no significant difference in ADR between Online Travel Agencies and Direct bookings.
  - Result: The Null Hypothesis was rejected, indicating a significant difference. This suggests that the booking channel indeed influences room pricing.
- Room Upgrades vs. Lead Time
  - Null Hypothesis (H_0): Room upgrades are independent of the lead time.
  - Result: The Null Hypothesis failed to be rejected, meaning no strong statistical evidence was found to suggest a relationship between room upgrades and lead time.
- Stay Duration vs. Customer Type
  - Null Hypothesis (H_0): There is no significant difference in stay duration across different customer types.
  - Result: The Null Hypothesis was rejected, indicating a significant variation in stay durations among different customer categories (e.g., transient vs. contract).
- Diagrams Used:
  - Boxplots: Used to visually compare the distributions of `ADR` and `stay durations` across different groups relevant to the hypothesis tests.
  - Grouped Bar Charts: Employed to visualize categorical comparisons and effectively summarize group-based differences.

Step 1:Import library

In [1]:
```
import numpy as np
from numpy import random
import pandas as pd
import os
from numpy.linalg import inv
import scipy
from scipy import stats
from scipy.stats import skew,kurtosis
import matplotlib
from matplotlib import pyplot as  plt
import seaborn as sns
%matplotlib inline
from scipy.stats import binom
pd.set_option('Display.max_columns',None)
from scipy.stats import expon
from scipy.stats import norm
from scipy.stats import t
import statsmodels
from statsmodels import stats
from statsmodels.stats import weightstats as ssw
import statsmodels.api as sm
from statsmodels.formula.api import ols
import statsmodels.stats.multicomp
```

Data cleaning

In [2]:
```
os.chdir(r'C:\\Users\\MADHURI\\Desktop\\CDAC\\Statistic_Sudip sir\\Program_S')
```

Load Data

In [ ]:
```
1
```

In [28]:
```
df=pd.read_csv("hotel_bookings.csv")
```

In [4]:
```
df.head()
```

Out[4]:

	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_i
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	

In [29]:
```
# View number of rows and columns,Understand data types of each column,Check non-null (non-missing) counts,memory usage
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
<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 
```

Data cleaning

```
In [6]: #Number of rows and columns
        df.shape

Out[6]: (119390, 32)
```

```
In [6]: #Checking null values
```

```
In [7]: df.isnull().sum()

Out[7]: 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
```

Check Unique values

```
In [8]: # check unique value in hotel column
        df['hotel'].unique()

Out[8]: array(['Resort Hotel', 'City Hotel'], dtype=object)
```

```
In [9]: #check unique column in arrival_date_year
        df['arrival_date_year'].unique()

Out[9]: array([2015, 2016, 2017], dtype=int64)
```

```
In [10]: #check unique column is_canceled
         df['is_canceled'].unique()

Out[10]: array([0, 1], dtype=int64)
```

```
In [11]: #check unique column in meal
         df['meal'].unique()

Out[11]: array(['BB', 'FB', 'HB', 'SC', 'Undefined'], dtype=object)
```

```
In [12]: #check unique value in distribution_channel
         df['distribution_channel'].unique()

Out[12]: array(['Direct', 'Corporate', 'TA/TO', 'Undefined', 'GDS'], dtype=object)
```

```
In [ ]: Data descrption and solution:-
        Four column have null values
        children-4 ,country-488,agent-16340,company-112593
        if their are missing values more than 20% then remove that
        1.if column data type is string then to fill null values by mode.
                  # most frequent value
        2.if column data type is numerical then fill null values by median   (df.column_name.mod)
             (df.column_name.median) Median is robust to outliers. i.e Median stays stable, even if extreme values are added
```

Data descrption and solution:-
Four column have null values
children-4 ,country-488,agent-16340,company-112593
if their are missing values more than 20% then remove that
1.if column data type is string then to fill null values by mode.
most frequent value
2.if column data type is numerical then fill null values by median (df.column_name.mod)
(df.column_name.median) Median is robust to outliers. i.e Median stays stable, even if extreme values are added
#when outliers are present, because it ignores extreme values.

Handle Missing Values

In [ ]:
```
#Handling Missing values
```

In [30]:
```
df=df.drop('company',axis=1)
```

In [31]:
```
# To fill null values calculate median because this are numerical values so we have used median
df.agent.median()
```

Out[31]: 14.0

In [32]:
```
df['agent']=df.agent.fillna(14.0)
```

In [33]:
```
df.head()
```

Out[33]:

	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_i
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	

In [16]:
```
df.shape
```

Out[16]: (119390, 31)

In [34]:
```
# to fill null values calculate mode because this are string values so we have used mode and fill
df.country.mode()
```
Out[34]:
```
0    PRT
Name: country, dtype: object
```

In [35]:
```
df['country']=df.country.fillna('PRT')
```

In [36]:
```
df.head()
```
Out[36]:

	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_i
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	

Change data type of column

In [37]:
```
# convert float into interger and fill 0 at null values
df['children'] = df['children'].fillna(0).astype(int)
```

In [38]:
```
#Change Data Type
df['hotel'] = df.hotel.astype('string')
df['meal']=df.meal.astype('string')
df['country'] =df.country.astype('string')
df['market_segment']= df.market_segment.astype('string')
df['distribution_channel']=df.distribution_channel.astype('string')
df['reserved_room_type']=df.reserved_room_type.astype('string')
df['assigned_room_type']=df.assigned_room_type.astype('string')
df['deposit_type']=df.deposit_type.astype('string')
df['customer_type']=df.customer_type.astype('string')
df['reservation_status']=df.reservation_status.astype('string')
```

In [39]:
```
df['reservation_status_date'] = pd.to_datetime(df['reservation_status_date'])
```

In [40]:
```
df['Date'] = pd.to_datetime(
    df['arrival_date_year'].astype(str) + '-' +
    df['arrival_date_month'].astype(str) + '-' +
    df['arrival_date_day_of_month'].astype(str),

)
```

In [24]:
```
df = df.drop(['arrival_date_year', 'arrival_date_month', 'arrival_date_day_of_month'], axis=1)
#optional
```

In [41]:
```
df.Date
```
Out[41]:
```
0        2015-07-01
1        2015-07-01
```

In [42]:
```python
# convert float into interger and fill 0 at null values
df['children'] = df['children'].fillna(0).astype(int)
```

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

Out[43]:
```
(119390, 32)
```

create derived column

In [ ]:
```python
# create derived columns
```

In [44]:
```python
# Create 'total_guests' column
df['total_guests'] = df['adults'] + df['children'] + df['babies']
```

In [45]:
```python
df['Total_Night']=df['stays_in_weekend_nights']+df['stays_in_week_nights']
```

In [46]:
```python
df.head()
```

Out[46]:

	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_i
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	

Handle duplicate value

In [ ]:
```python
#Handling Duplicate Values
```

In [47]:
```python
df.duplicated().sum()
```

Out[47]:
```
32020
```

In [48]:
```python
df = df.drop_duplicates()
```

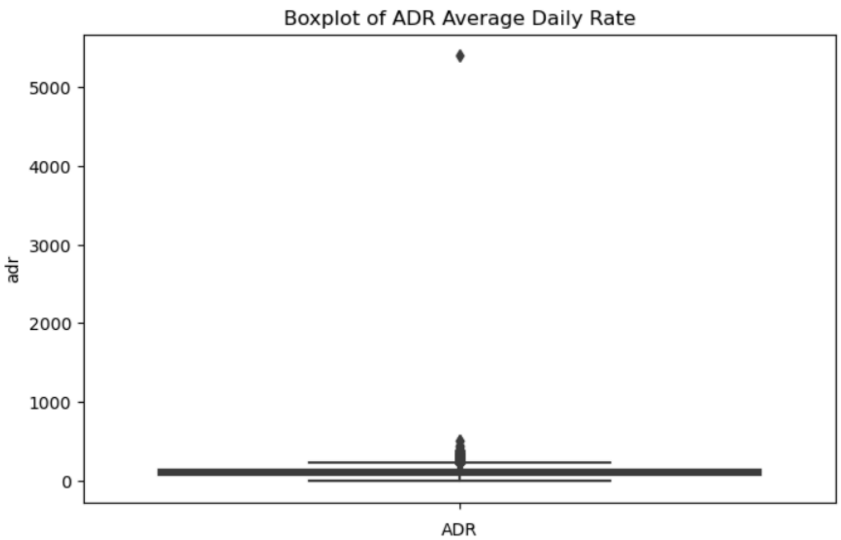
In [49]:
```python
df.shape
```

Out[49]:
```
(87370, 34)
```

```
A boxplot helps you visualize the distribution of a numerical column (like adr) and identify:
Median (middle line)-->Central value of adr
Box (IQR)--->Spread of the middle 50% of data
Whisker--Range of most of the data (not outliers)
Outliers-->Unusually high or low adr values
To check outliers we have plot boxplot
```

In [38]:
```python
#Insight: Revenue per night distribution and pricing outliers.
plt.figure(figsize=(8, 5))
sns.boxplot(y=df['adr'])
plt.title('Boxplot of ADR Average Daily Rate')
plt.xlabel('ADR')
plt.show()
```

In [38]:
```python
#Insight: Revenue per night distribution and pricing outliers.
plt.figure(figsize=(8, 5))
sns.boxplot(y=df['adr'])
plt.title('Boxplot of ADR Average Daily Rate')
plt.xlabel('ADR')
plt.show()
```



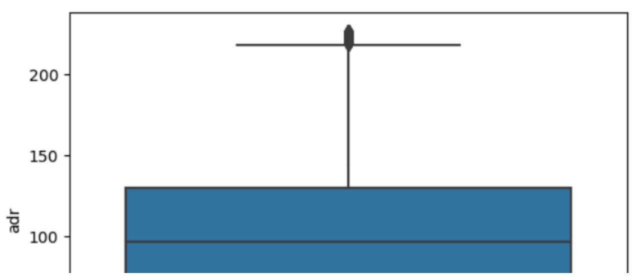
In [ ]:
```
Most daily rates are around ₹100
There is low variation for most guests
Extreme outliers exist above  ₹5000 – review needed
The data is right-skewed, with some high-paying bookings:-Most prices are on the lower end, but a few are very high.
most bookings are normal and consistent, but due to some unusal prices impact on analysis
```

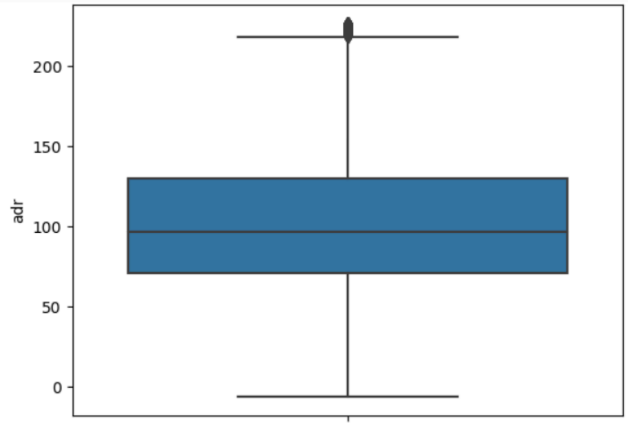
In [50]:
```python
def remove_outliers_iqr(df, column):
    Q1 = df[column].quantile(0.25)
    Q3 = df[column].quantile(0.75)
    IQR = Q3 - Q1
    lower = Q1 - 1.5 * IQR
    upper = Q3 + 1.5 * IQR
    return df[(df[column] >= lower) & (df[column] <= upper)]
```

In [51]:
```python
df = remove_outliers_iqr(df, 'adr')
```

In [52]:
```python
sns.boxplot(y=df['adr'])
```

Out[52]:
```
<Axes: ylabel='adr'>
```



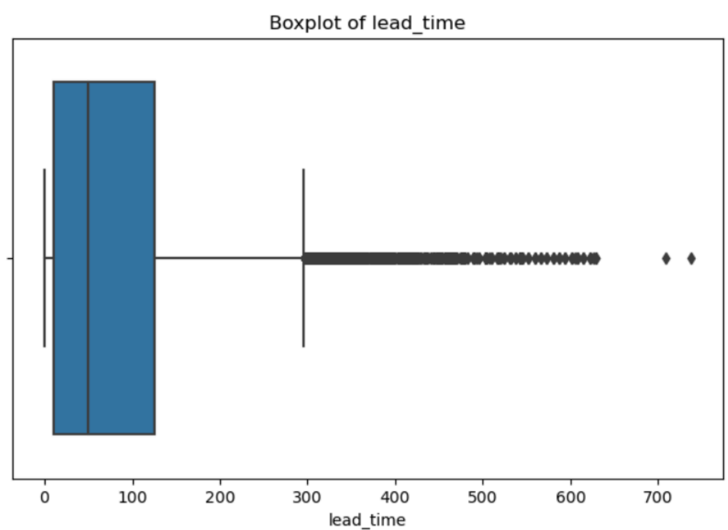


In [53]:
```
df.describe()
```

Out[53]:

	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	84882.000000	84882.000000	84882.000000	84882.000000	84882.000000	84882.000000	84882.000000 84[illegible]
mean	0.271659	80.156464	2016.199654	26.721307	15.798497	1.001378	2.613287
min	0.000000	0.000000	2015.000000	1.000000	1.000000	0.000000	0.000000
25%	0.000000	11.000000	2016.000000	15.000000	8.000000	0.000000	1.000000

In [54]:
```
plt.figure(figsize=(8, 5))
sns.boxplot(x=df['lead_time'])
plt.title('Boxplot of lead_time')
plt.xlabel('lead_time')
plt.show()
```



In [ ]:
```
#Outliers represent bookings with unusually long or short lead times.
"The boxplot of lead_time reveals several outliers with values exceeding 700 days."
```

In [ ]:
```
The lead_time variable exhibits some outliers above 700 days, indicating a few guests book their stays more than 1.5 years in adv
While these outliers are relatively rare, they highlight the presence of long-term planners in the customer base.
For most bookings, lead times are considerably shorter, suggesting that operational and marketing efforts should primarily
focus on typical booking windows. However, these extreme values could impact average lead time calculations and should be
considered in deeper analyses."
```

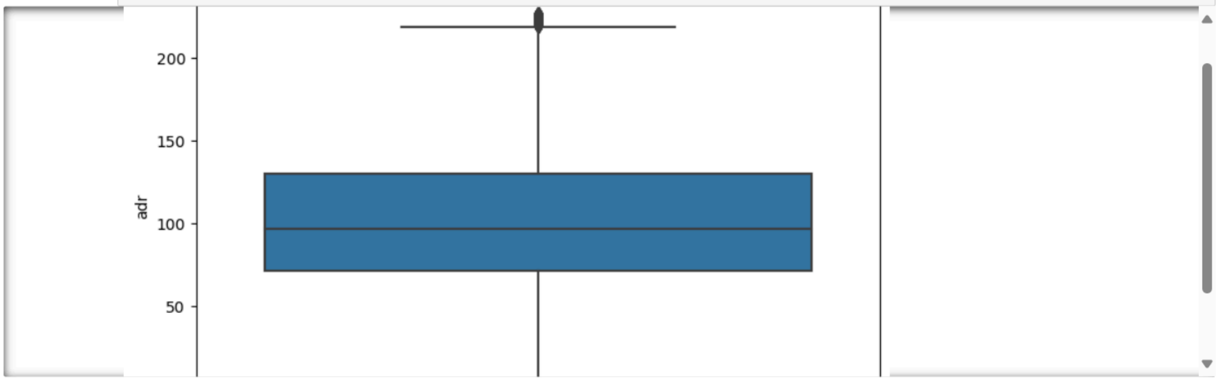
A boxplot helps you visualize the distribution of a numerical column (like adr) and identify:
Median (middle line)-->Central value of adr

Box (IQR)--->Spread of the middle 50% of data
Whisker--Range of most of the data (not outliers)
Outliers-->Unusually high or low adr values
To check outliers we have plot boxplot

1. Univariate Analysis

adr(Average Daily Rate) ¶

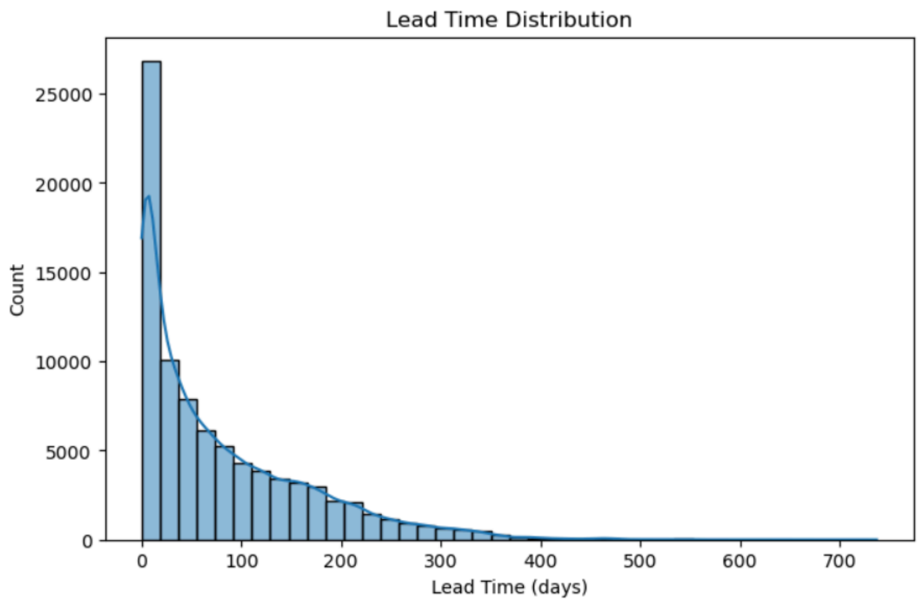
In [55]:
```
plt.figure(figsize=(8, 5))
sns.boxplot(y=df['adr'])
plt.title('ADR Distribution')
plt.xlabel('ADR')
plt.show()
```



In [ ]:
```
#Most values fall between 50-150.
```

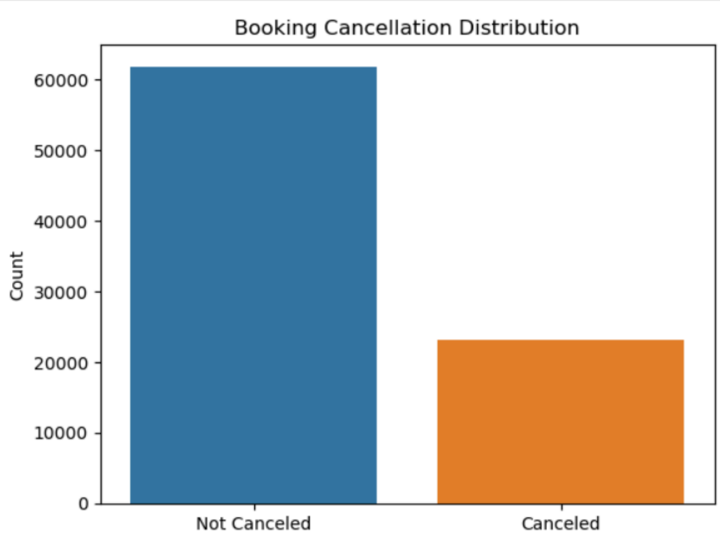
lead_time Distribution ¶

In [56]:
```
plt.figure(figsize=(8, 5))
sns.histplot(df['lead_time'],kde=True, bins=40)
plt.title('Lead Time Distribution')
plt.xlabel('Lead Time (days)')
plt.ylabel('Count')
plt.show()
```

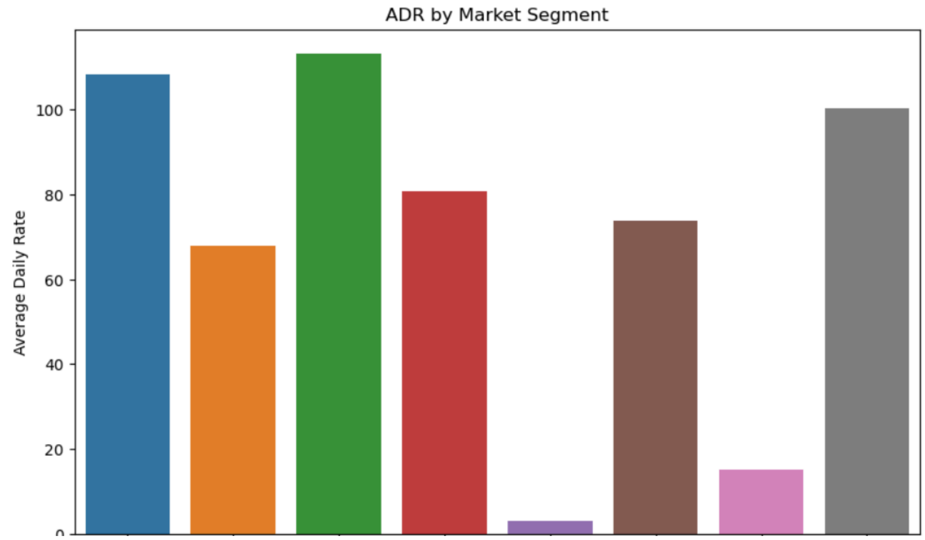


2.Bivariate analaysis

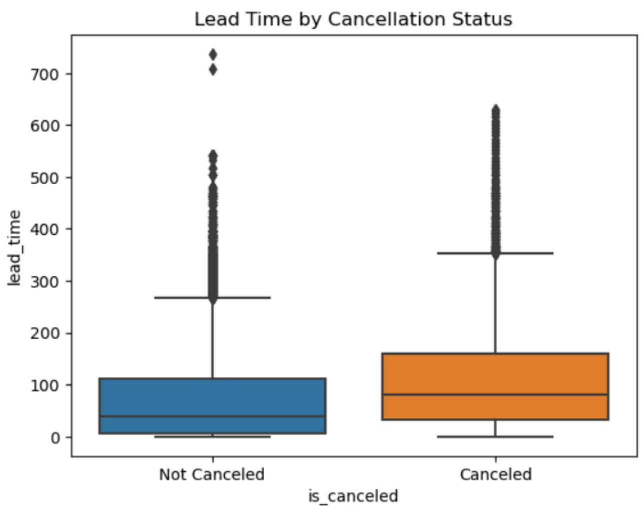
In [68]:
```
#Insight: Understand cancellation patterns.
sns.countplot(data=df, x='is_canceled')
plt.title('Booking Cancellation Distribution')
plt.xticks([0,1], ['Not Canceled', 'Canceled'])
plt.xlabel('Booking Status')
plt.ylabel('Count')
plt.show()
```



In [69]:
```
plt.figure(figsize=(10, 6))
sns.barplot(x='market_segment', y='adr',errorbar=None, data=df)
plt.title('ADR by Market Segment')
plt.xlabel('Market Segment')
plt.ylabel('Average Daily Rate')
plt.xticks(rotation=45)
plt.show()
```



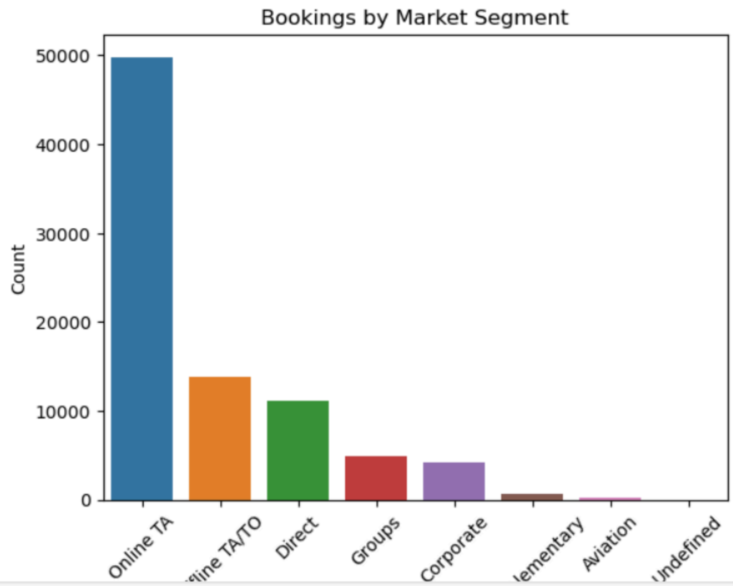
In [63]:
```
# early booking cancelation status
sns.boxplot(data=df, x='is_canceled', y='lead_time')
plt.title('Lead Time by Cancellation Status')
plt.xticks([0,1], ['Not Canceled', 'Canceled'])
plt.show()
```



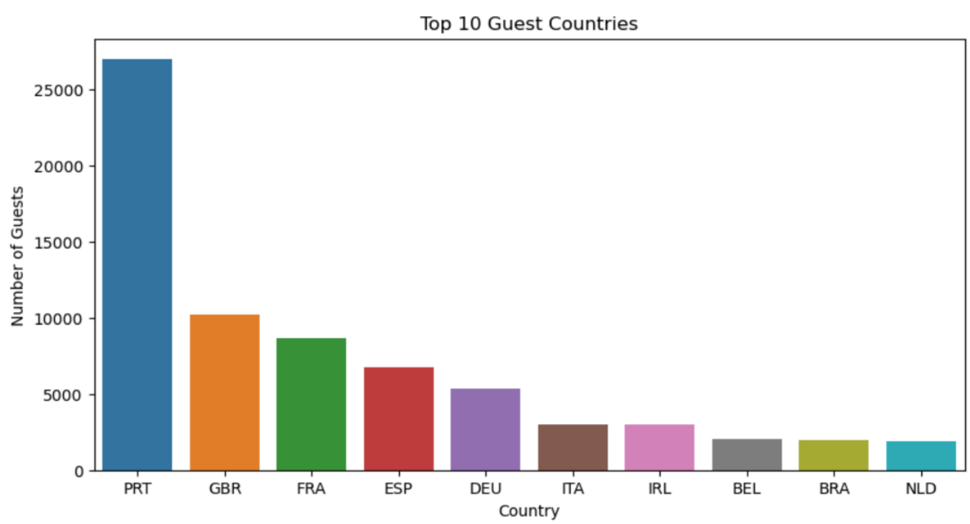
Guest demographics and distribution by country

Multivariate analaysis

In [64]:
```
#segment brings more business
sns.countplot(data=df, x='market_segment', order=df['market_segment'].value_counts().index)
plt.title('Bookings by Market Segment')
plt.xticks(rotation=45)
plt.ylabel('Count')
plt.show()
```

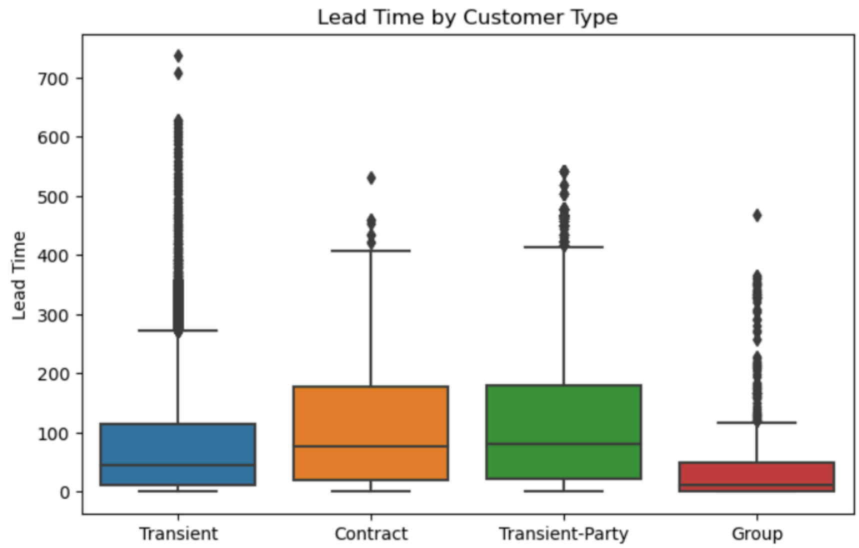


In [65]:
```
top_countries = df['country'].value_counts().head(10)

plt.figure(figsize=(10, 5))
sns.barplot(x=top_countries.index, y=top_countries.values)
plt.title('Top 10 Guest Countries')
plt.xlabel('Country')
plt.ylabel('Number of Guests')
plt.show()
```

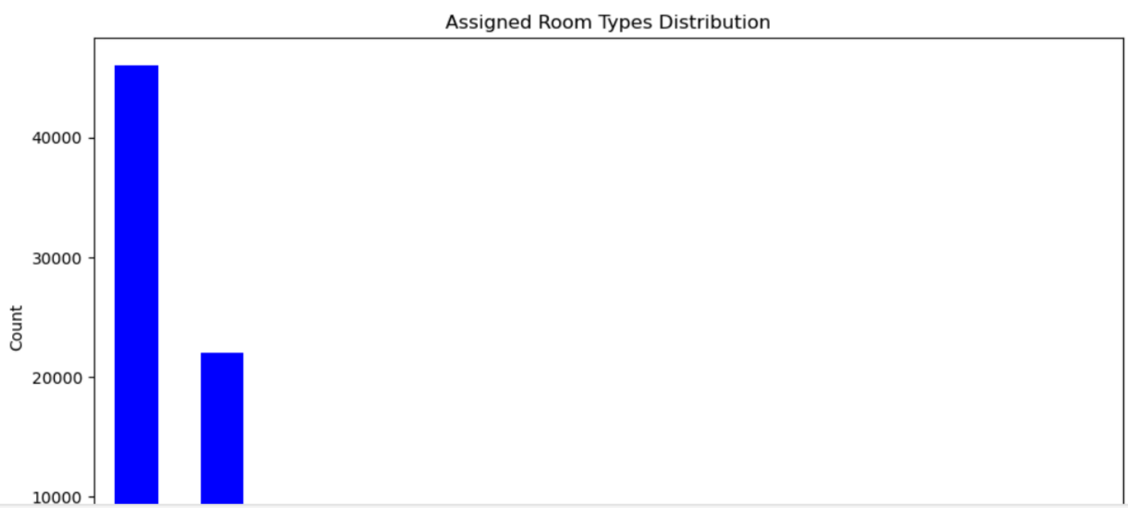


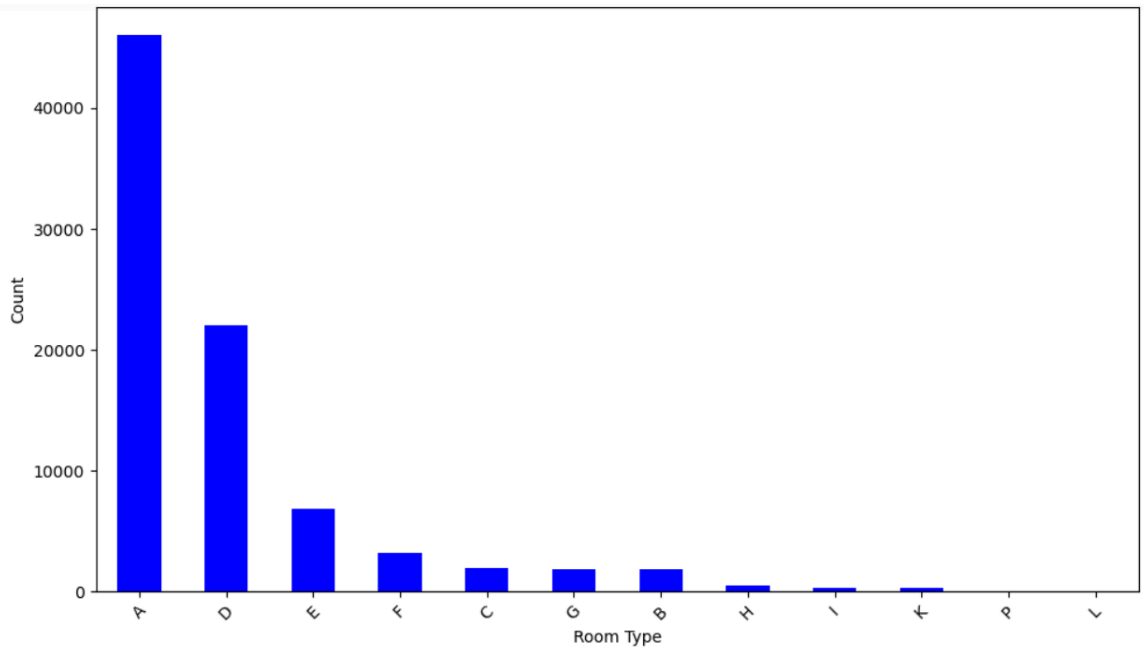
In [ ]:
```
#Market segment share and ADR (Average Daily Rate) comparison. Booking lead time distribution across customer types
```

In [67]:
```
#. Booking lead time distribution across customer types
plt.figure(figsize=(8, 5))
sns.boxplot(x='customer_type', y='lead_time', data=df)
plt.title('Lead Time by Customer Type')
plt.xlabel('Customer Type')
plt.ylabel('Lead Time')
plt.show()
```

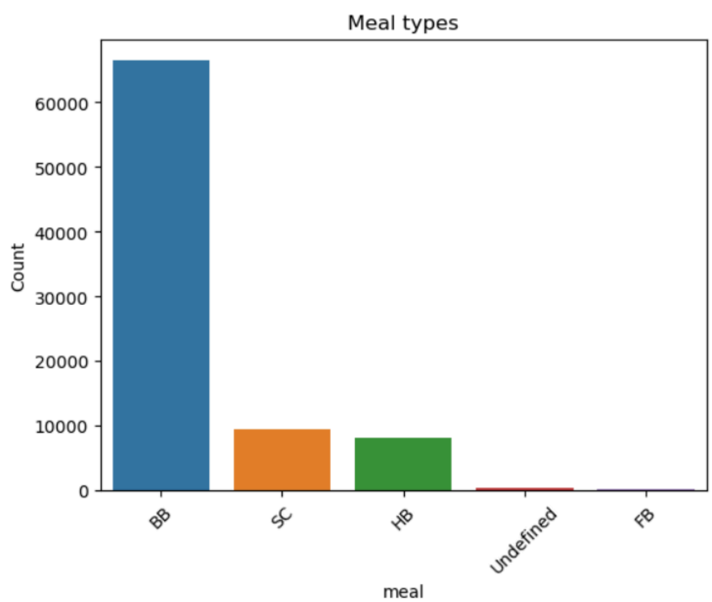


In [79]:
```
room_counts = df['assigned_room_type'].value_counts()

# Plotting
plt.figure(figsize=(10, 6))
room_counts.plot(kind='bar', color='blue')
plt.title('Assigned Room Types Distribution')
plt.xlabel('Room Type')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```





In [84]:
```
sns.countplot(data=df, x='meal', order=df['meal'].value_counts().index)
plt.title('Meal types')
plt.xticks(rotation=45)
plt.ylabel('Count')
plt.show()
```

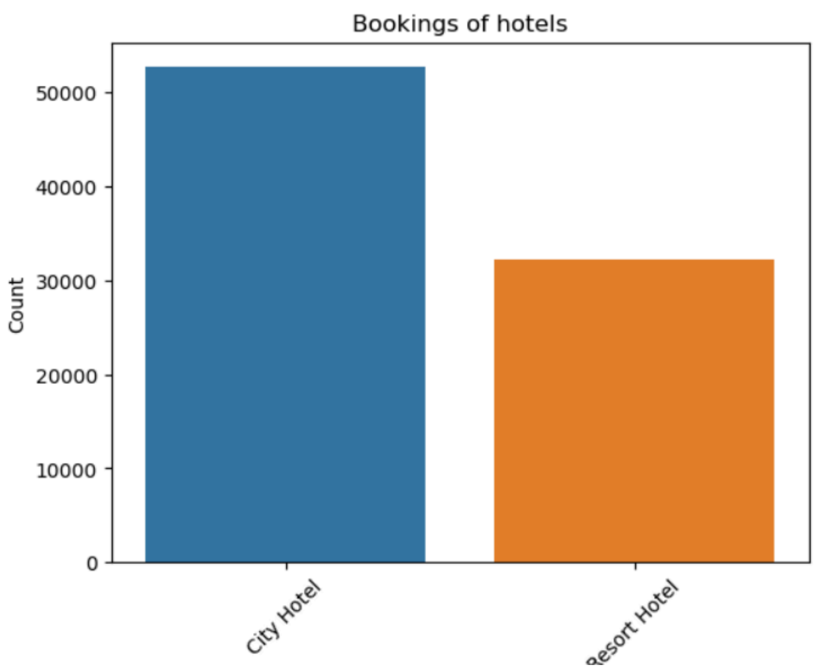


[ ]:
```
from above graph BB(bed and breakfast) is most prefered type of meal by guest
full board are least preferred
```

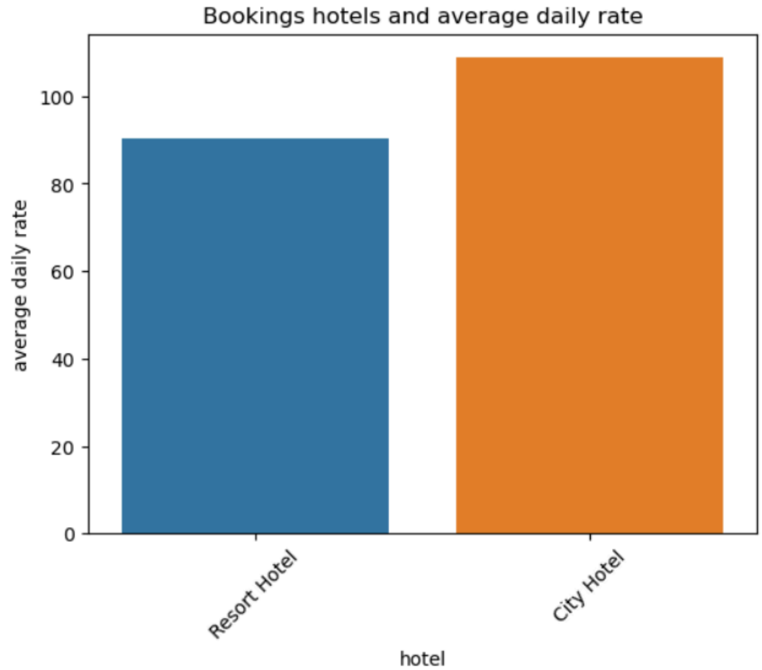
In [66]:
```
#Insight: Seasonality or peak/off-peak months.
order = ['January', 'February', 'March', 'April', 'May', 'June',
         'July', 'August', 'September', 'October', 'November', 'December']

sns.countplot(data=df, x='arrival_date_month', order=order)
plt.title('Bookings by Month')
plt.xticks(rotation=45)
plt.ylabel('Number of Bookings')
plt.show()
```

bivariate analysis

[85]:
```
sns.countplot(data=df, x='hotel', order=df['hotel'].value_counts().index)
plt.title('Bookings of hotels')
plt.xticks(rotation=45)
plt.ylabel('Count')
plt.show()
```

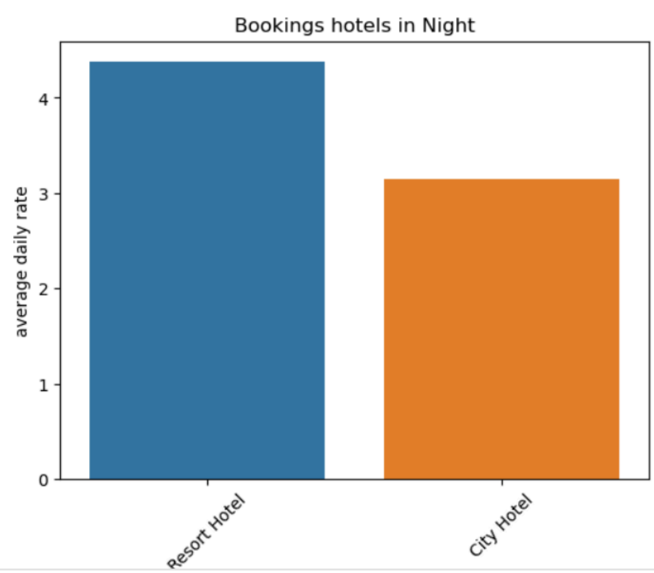


In [92]:
```python
sns.barplot(data=df, x='hotel', y=df['adr'],errorbar=None)
plt.title('Bookings hotels and average daily rate')
plt.xticks(rotation=45)
plt.ylabel('average daily rate')
plt.show()
```



In [ ]:
```
If City Hotel shows a higher mean ADR than Resort Hotel:
It indicates City Hotel charges more on average—likely due to its urban location
```

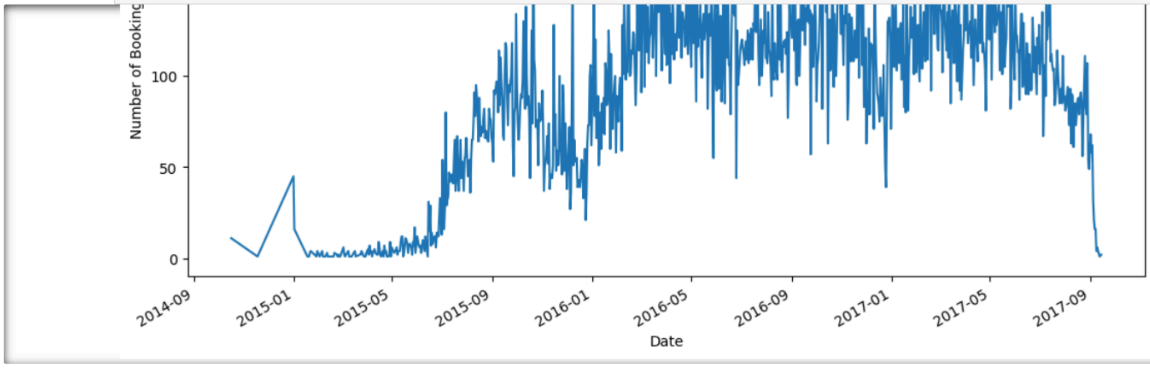
In [96]:
```python
sns.barplot(data=df, x='hotel', y=df['Total_Night'],errorbar=None)
plt.title('Bookings hotels in Night')
plt.xticks(rotation=45)
plt.ylabel('average daily rate')
plt.show()
```



```
In [ ]: From above graph it is predicted average daily rate to stay in hotel for night in resort is greater than city hotel
```

Time-series analysis of booking trends

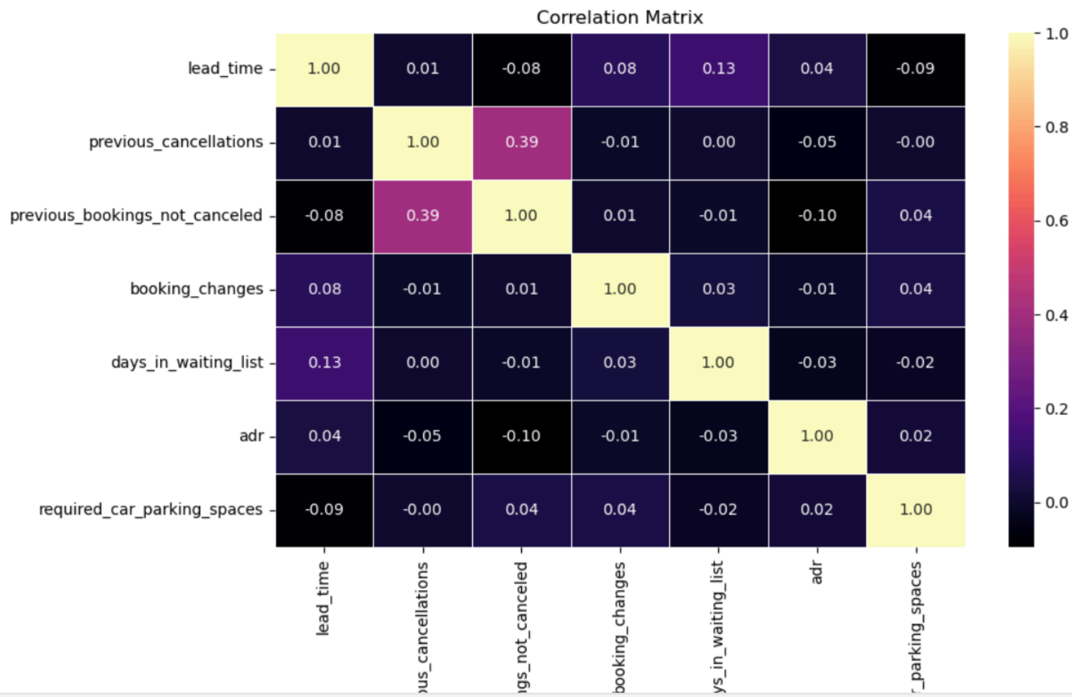
```
In [52]: ts = df.groupby('reservation_status_date').size()
plt.figure(figsize=(12, 6))
ts.plot()
plt.title('Daily Booking Volume Over Time')
plt.xlabel('Date')
plt.ylabel('Number of Bookings')
plt.show()
```




```
In [ ]: #Observe whether bookings are increasing, decreasing, or stable over time.
```

Correlation matrix:

```
In [70]: corr_features = [
    'lead_time',
    'previous_cancellations',
    'previous_bookings_not_canceled',
    'booking_changes',
    'days_in_waiting_list',
    'adr',
    'required_car_parking_spaces'
]

# Calculate correlation matrix
corr_matrix = df[corr_features].corr()
#calculates the correlation coefficient between each pair: +1--> positive ,0--> no relation,-1--> negative

# Plot the heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(corr_matrix, annot=True, cmap='magma', fmt=".2f", linewidths=0.5)
plt.title('Correlation Matrix')
plt.show()
```

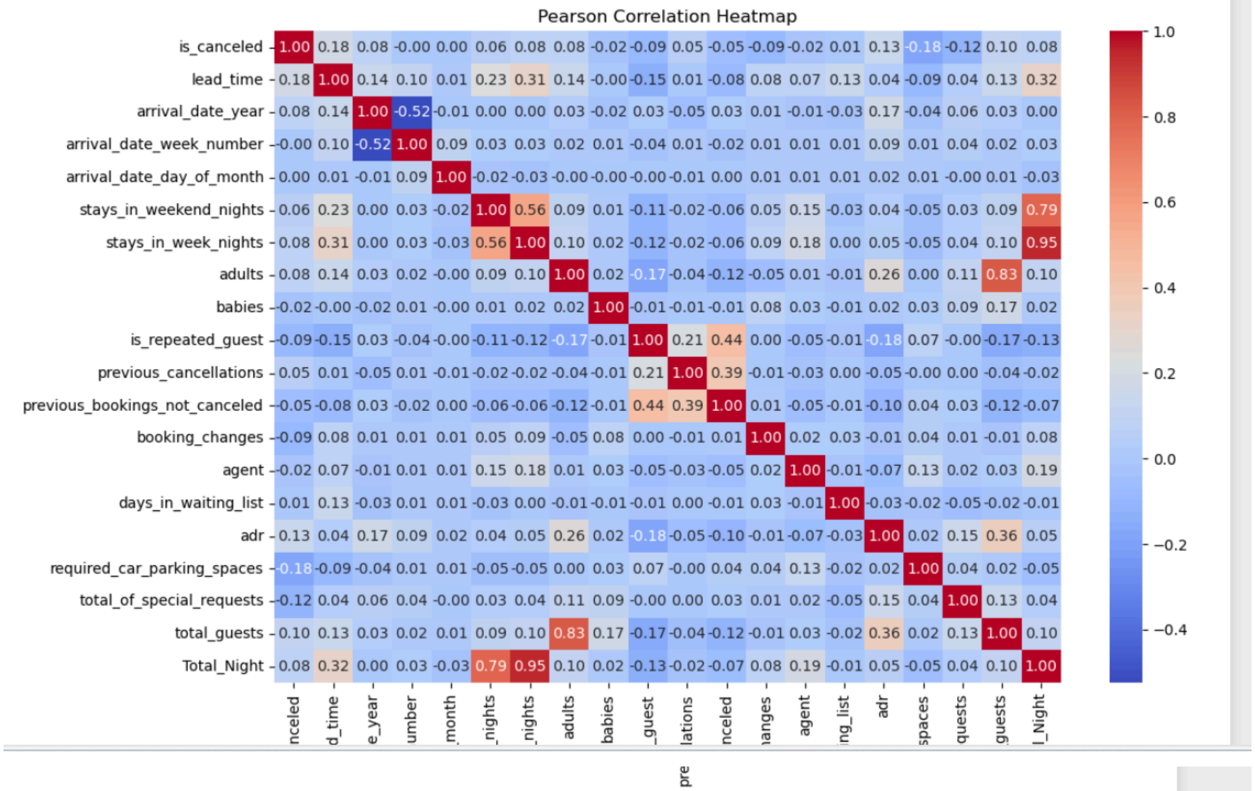


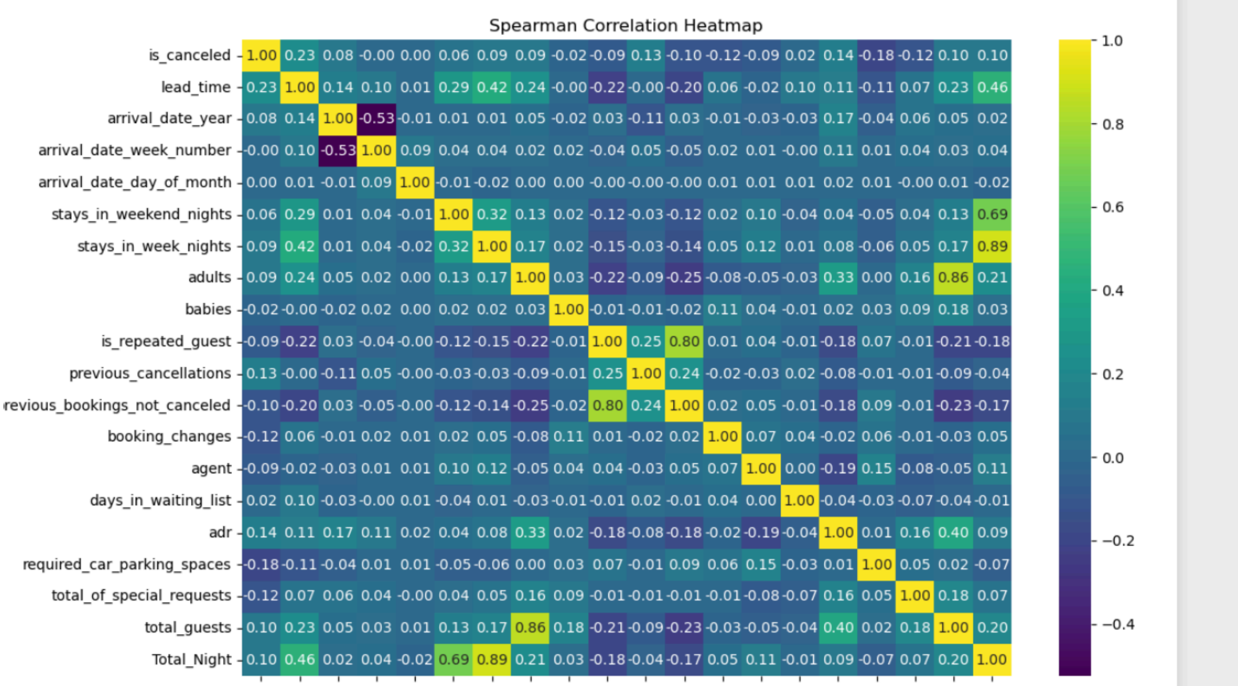

```
In [ ]: lead_time is slightly positively correlated with booking_changes and days_in_waiting------>
        correlated means more is the stay of customer more will be the lead time

        adr shows a weak correlation with:
        required_car_parking_spaces
        previous_bookings_not_canceled

        Adr and total people are highly correlated-more people more will be adr high adr high revenu
        previous_cancellations and lead_time are positively correlated — customers who plan earlier may cancel more often.
```

co-realtion

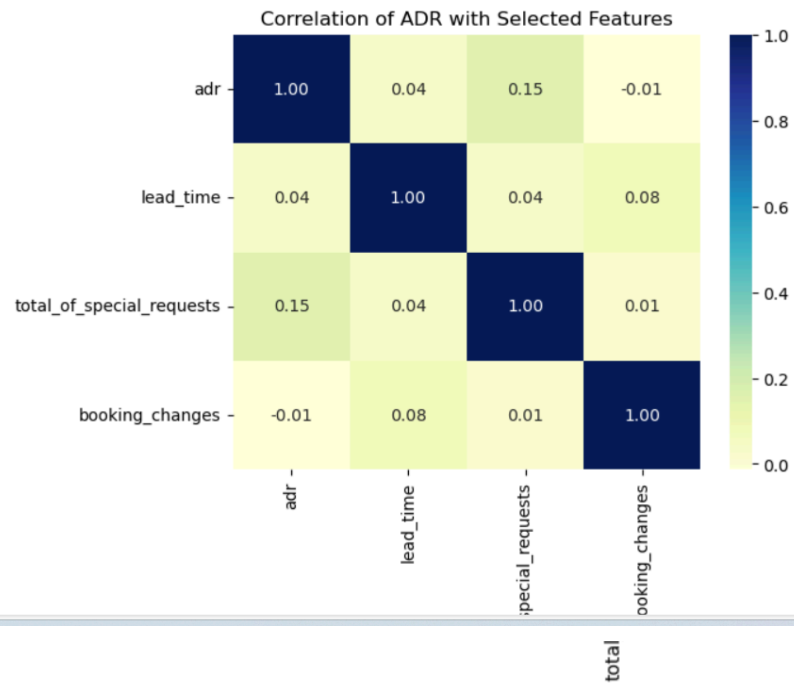
```
In [71]: numeric_df = df.select_dtypes(include=['int64', 'float64'])
         #Pearson Correlation
         pearson_corr = numeric_df.corr(method='pearson')
         #sepearman correlation
         spearman_corr = numeric_df.corr(method='spearman')
```

```
In [72]: plt.figure(figsize=(12, 8))
         sns.heatmap(pearson_corr, annot=True, fmt='.2f', cmap='coolwarm')
         plt.title('Pearson Correlation Heatmap')
         plt.show()

         plt.figure(figsize=(12, 8))
         sns.heatmap(spearman_corr, annot=True, fmt='.2f', cmap='viridis')
         plt.title('Spearman Correlation Heatmap')
         plt.show()
```


Pearson Correlation Heatmap
is_canceled
lead_time
arrival_date_year
arrival_date_week_number
arrival_date_day_of_month
stays_in_weekend_nights
stays_in_week_nights
adults
babies
is_repeated_guest
previous_cancellations
previous_bookings_not_canceled
booking_changes
agent
days_in_waiting_list
adr
required_car_parking_spaces
total_of_special_requests
total_guests
Total_Night
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4


Spearman Correlation Heatmap
is_canceled
lead_time
arrival_date_year
arrival_date_week_number
arrival_date_day_of_month
stays_in_weekend_nights
stays_in_week_nights
adults
babies
is_repeated_guest
previous_cancellations
revious_bookings_not_canceled
booking_changes
agent
days_in_waiting_list
adr
required_car_parking_spaces
total_of_special_requests
total_guests
Total_Night
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4

In [73]:
```python
features = ['adr', 'lead_time', 'total_of_special_requests', 'booking_changes']
adr_corr = numeric_df[features].corr(method='pearson')
sns.heatmap(adr_corr, annot=True, cmap='YlGnBu', fmt='.2f')
plt.title('Correlation of ADR with Selected Features')
plt.show()
```



In [74]:
```python
print("Correlation of ADR with lead_time:", adr_corr.loc['adr', 'lead_time'])
print("Correlation of ADR with special requests:", adr_corr.loc['adr', 'total_of_special_requests'])
print("Correlation of ADR with booking_changes:", adr_corr.loc['adr', 'booking_changes'])
```

```
Correlation of ADR with lead_time: 0.03988790032778095
Correlation of ADR with special requests: 0.1526552188026844
Correlation of ADR with booking_changes: -0.013510139794038076
```

Hypothesis testing

In [ ]:
```
4.  Hypothesis Testing
Use statistical tests to validate business assumptions:
H0: There is no difference in ADR between bookings made through Online TA and Direct channels
H0: Room upgrades are independent of lead time
H0: Average stay duration does not differ between customer types
```

In [ ]:
```
#1.
H0: There is no difference in ADR between bookings made through Online TA and Direct channels
df[online TA] abd df[Direct channels]
H0: There is no difference in the mean ADR between bookings made via Direct channel and TA/TO channel.
H1: There is a difference in the mean ADR between the two channels.
```

In [75]:
```
online_ta_adr = df[df['distribution_channel'] == 'TA/TO']
direct_adr = df[df['distribution_channel'] == 'Direct']
#This is performing a two-sample Z-test on the ADR values from the two groups.
zscore,pvalue=ssw.ztest(direct_adr.adr,online_ta_adr.adr)
print(zscore,pvalue)
```

```
-7.658979760340348 1.8741592821899742e-14
```

In [ ]:
```
zscore: The test statistic, measuring how many standard deviations the difference in means is from zero.
pvalue: The probability of observing the data assuming the null hypothesis is true.
```

In [ ]:
```
we reject Null hypothesis
If p-value < significance level (commonly 0.05), reject H0 → There is statistically significant evidence that ADR differs betweer
If p-value ≥ 0.05, fail to reject H0 → No sufficient evidence to say ADR differs between the two channels.
The zscore tells direction and magnitude of difference:
A large positive or negative z-score indicates a bigger difference between means.
The sign shows which group has higher mean (depending on order of subtraction in the test).
```

In [ ]:
```
#2.
H0: Room upgrades are independent of lead time
```

In [ ]:
```
Null Hypothesis (H₀):
Room upgrades are independent of lead time.
→ No significant difference in lead time between upgraded and non-upgraded bookings.

Alternative Hypothesis (H₁):
Room upgrades depend on lead time.
→ Guests who were reassigned rooms have different average lead times than those who were not.
```

In [76]:
```
df['room_reassigned'] = (df['reserved_room_type'] != df['assigned_room_type']).astype(int)
lead_time_reassigned = df[df['room_reassigned'] == 1]
lead_time_not_reassigned = df[df['room_reassigned'] == 0]
ssw.ttest_ind(lead_time_reassigned.lead_time, lead_time_not_reassigned.lead_time, usevar='unequal')
```

Out[76]:
```
(-33.31250455268803, 1.8169443088558377e-236, 18807.95844287712)
```

In [ ]:
```
t-statistic = -33.31
p-value = 1.81e-236
df (degrees of freedom) ≈ 18808
This is far less than 0.05, so we reject the null hypothesis.

Conclusion: There is a statistically significant difference in average lead times between upgraded and non-upgraded guests.
```

In [ ]:
```
#3.
```

In [ ]:
```
H0: Average stay duration is the same across customer types.
H1: Average stay duration differs across at least one customer type.
```

In [ ]:
```
This is a one-way ANOVA test scenario because we are comparing means of a
numeric variable (stay duration) across multiple groups (customer types)
```

n [77]:
```
df['customer_type'] = df['customer_type'].astype('category')
```

n [78]:
```
model = ols('Total_Night ~ C(customer_type)', data=df).fit()
anova_table = sm.stats.anova_lm(model)
print(anova_table)
```

```
                       df         sum_sq     mean_sq           F  PR(>F)
C(customer_type)      3.0   19758.803192 6586.267731  886.523856     0.0
Residual          84878.0  630585.661736    7.429318         NaN     NaN
```

In [ ]:
```
PR(>F) = 0.0 (p-value is extremely small):
This is the most important result.

Since p-value < 0.05, we reject the null hypothesis.
Conclusion: There is a statistically significant difference in average stay duration between at least one pair of customer types.
```

In [ ]:
```
F-statistic = 886.52:
This is a very high F-value, indicating that the variation in stay duration between groups (customer types) is much larger than tl

It confirms the groups are meaningfully different.
```

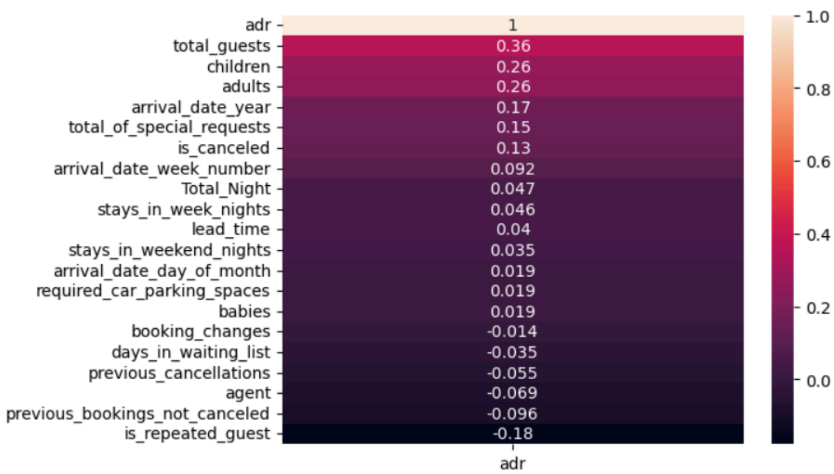
In [ ]:
```
"The type of customer has a strong influence on the duration of their stay."
```

In [ ]:
```
5.  Key Business Questions
●   What influences ADR the most?
●   Do guests who book earlier tend to request more changes?
●   Are there pricing or booking differences across countries?
●   Is there a pattern in room upgrades or reassignment?
●   Are reserved room types consistently matched with assigned room types?
●   What are the most common guest demographics (e.g., group size, nationality)?
●   Are there patterns in guest types (e.g., transient vs. corporate) that influence booking behavior?
●   How does booking lead time vary across customer types and countries?
●   Are longer lead times associated with fewer booking changes or cancellations?
●   What is the typical duration of stay, and how does it vary by customer type or segment?
●   How often are guests upgraded or reassigned to a different room type?
●   Are guests who make special requests more likely to experience booking changes or longer stays?
●   Do certain market segments or distribution channels show higher booking consistency or revenue?
●   What factors are most strongly associated with higher ADR?
●   Are there customer types or segments consistently contributing to higher revenue?
●   Do bookings with more lead time or from specific countries yield higher ADR?
●   Are guests with higher ADR more likely to request special services or make booking modifications?
```

1.What influences ADR the most?

[127]:
```
corr_matrix = df.corr(numeric_only=True)
sns.heatmap(corr_matrix[['adr']].sort_values(by='adr', ascending=False), annot=True)
```

t[127]: <Axes: >



In [ ]:
```
From the heatmap, the strongest positive influencers of ADR are the number of special requests, lead time, and car parking spaces
cancellations are strongly negatively correlated with ADR. This suggests that high-paying customers tend to book early, request m
and are less likely to cancel.
```

Do guests who book earlier tend to request more changes?

In [134]:
```
corr_lead_changes = df['lead_time'].corr(df['booking_changes'])
```

In [135]:
```
corr_lead_changes
```

Out[135]:
```
0.07748480460713845
```

In [ ]:

There is a moderate positive correlation between lead_time and booking_changes.
So, the longer in advance people book, the more likely they are to make changes to their bookings.

3.Are there pricing or booking differences across countries?

In [ ]:

3.Are there pricing or booking differences across countries?
Whether guests from different countries are charged differently (ADR: Average Daily Rate)

Whether booking behavior (lead time, length of stay, cancellations) varies by country

In [ ]:

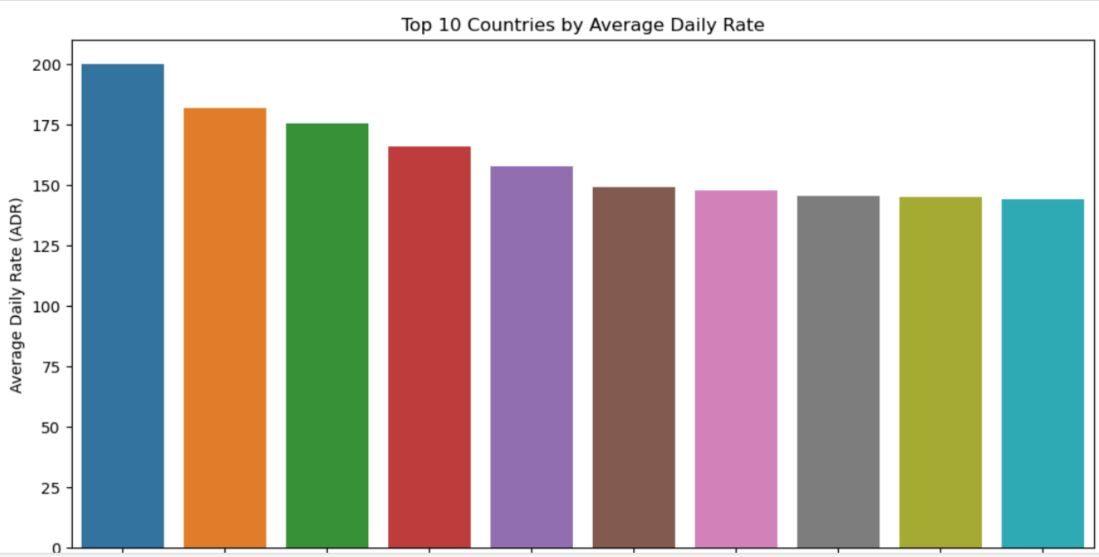
Make a bar chart that shows the average ADR per country (top 10).
some countries might have much higher average rates.
(H_0): All countries pay the same average price
(H_1): Some countries pay different average prices

In [138]:
```
adr_by_country = df.groupby('country')['adr'].mean().sort_values(ascending=False).head(10)
```

In [139]:
```
adr_by_country
```

Out[139]:
```
country
UMI     200.000000
LAO     181.665000
NCL     175.500000
```

In [132]:
```
adr_by_country = df.groupby('country')['adr'].mean().sort_values(ascending=False).head(10)
plt.figure(figsize=(12, 6))
sns.barplot(x=adr_by_country.index, y=adr_by_country.values)
plt.title("Top 10 Countries by Average Daily Rate")
plt.ylabel("Average Daily Rate (ADR)")
plt.xlabel("Country")
plt.xticks(rotation=45)
plt.show()
```



In [ ]:
```
These top-paying countries may not be the highest in booking volume, but still bring more revenue per booking.
```

4.Is there a pattern in room upgrades or reassignment?

In [149]:
```
df['is_upgraded'] = df['reserved_room_type'] != df['assigned_room_type']
df['is_upgraded'].head()
```

Out[149]:
```
0     False
1     False
2      True
3     False
4     False
Name: is_upgraded, dtype: boolean
```

In [ ]:

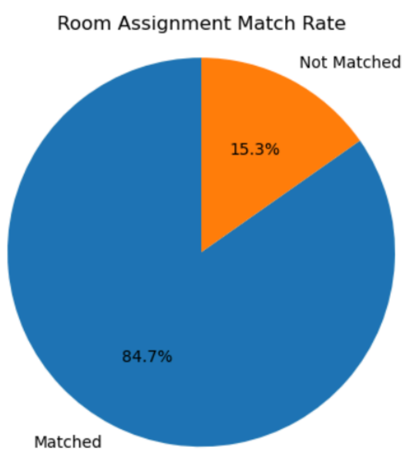
In [ ]:

5.Are reserved room types consistently matched with assigned room types

In [158]:
```
df['room_matched'] = df['reserved_room_type'] == df['assigned_room_type']
mismatched = df[df['room_matched'] == False]
print(mismatched.groupby(['reserved_room_type', 'assigned_room_type']).size().sort_values(ascending=False).head(10))
```

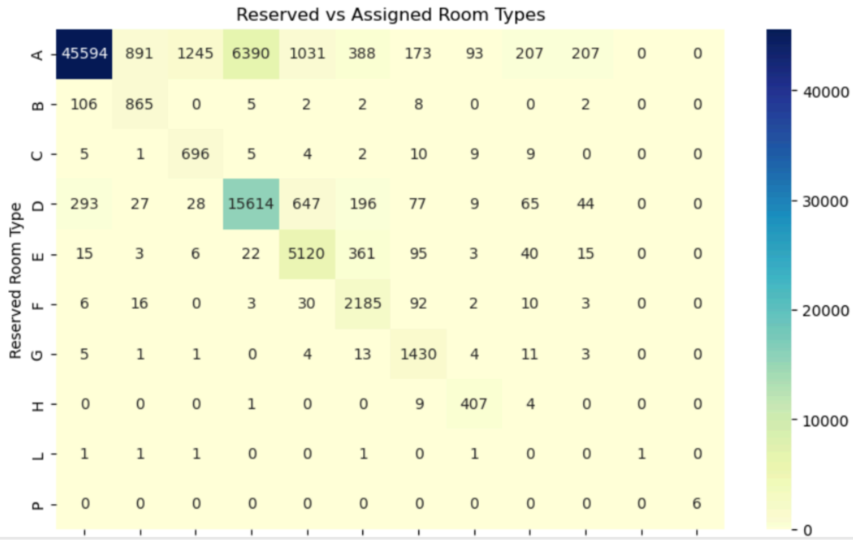
```
reserved_room_type  assigned_room_type
A                   D                     6390
                    C                     1245
                    E                     1031
                    B                      891
D                   E                      647
A                   F                      388
E                   F                      361
```

In [159]:
```
match_counts = df['room_matched'].value_counts()
labels = ['Matched', 'Not Matched']
plt.pie(match_counts, labels=labels, autopct='%1.1f%%', startangle=90)
plt.title('Room Assignment Match Rate')
plt.axis('equal')
plt.show()
```



In [ ]:
```
The overall proportion of bookings where the reserved room type matches the assigned room type.
High Match % (e.g., 85-95%) → Hotel mostly honors reservations.

Low Match % (<70%) → Frequent room changes, suggesting:
```

In [155]:
```python
room_matrix = pd.crosstab(df['reserved_room_type'], df['assigned_room_type'])
plt.figure(figsize=(10,6))
sns.heatmap(room_matrix, annot=True, fmt='d', cmap='YlGnBu')
plt.title("Reserved vs Assigned Room Types")
plt.xlabel("Assigned Room Type")
plt.ylabel("Reserved Room Type")
plt.show()
```

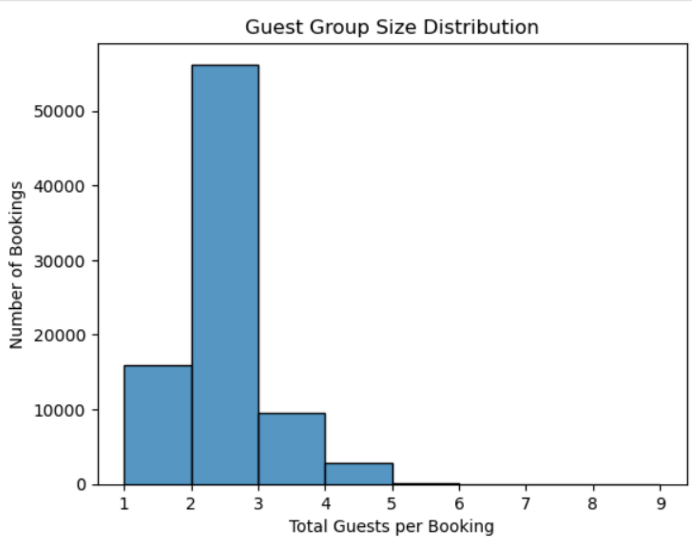


In [ ]:
```
Visual frequency map of room type transitions.
Which room types are often reassigned
If guests often get upgraded to higher categories → good guest experience
If guests are downgraded (e.g., from 'D' to 'A') → might cause complaints
```

In [ ]:

6.What are the most common guest demographics (e.g., group size, nationality)?

In [161]:
```python
sns.histplot(df['total_guests'], bins=range(1, 10), kde=False)
plt.title('Guest Group Size Distribution')
plt.xlabel('Total Guests per Booking')
plt.ylabel('Number of Bookings')
plt.show()
```



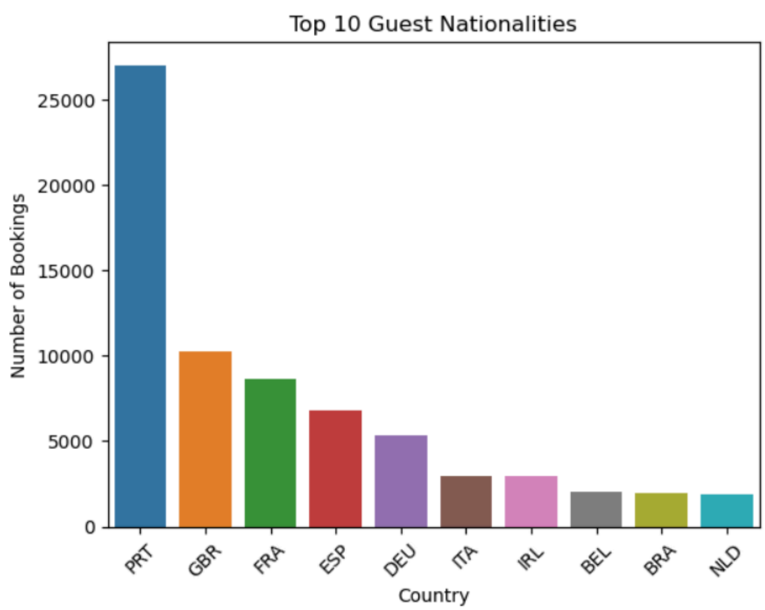
```
In [ ]: Most bookings are for 1 or 2 guests:

→ Targeted at solo travelers, business guests, or couples.

If larger groups (4–6) are common:

→ Suggests demand for family rooms or group offers.

Rare bookings for more than 6 → maybe conference or special event groups.
```

```
In [162]: top_nationalities = df['country'].value_counts().head(10)
```

```
In [163]: top_nationalities
```

```
Out[163]: country
PRT    27012
GBR    10252
FRA     8642
ESP     6790
DEU     5334
ITA     2994
IRL     2981
BEL     2048
BRA     1956
NLD     1880
Name: count, dtype: Int64
```

```
In [164]: sns.barplot(x=top_nationalities.index, y=top_nationalities.values)
plt.title('Top 10 Guest Nationalities')
plt.ylabel('Number of Bookings')
plt.xlabel('Country')
plt.xticks(rotation=45)
plt.show()
```




```
]: Top nationalities indicate your key guest source markets
If your top 3 are from the same country → strong domestic business.
```

- Do certain market segments or distribution channels show higher booking consistency or revenue?

```
1 df1.groupby(['market_segment', 'distribution_channel'])['adr'].mean().unstack()
```

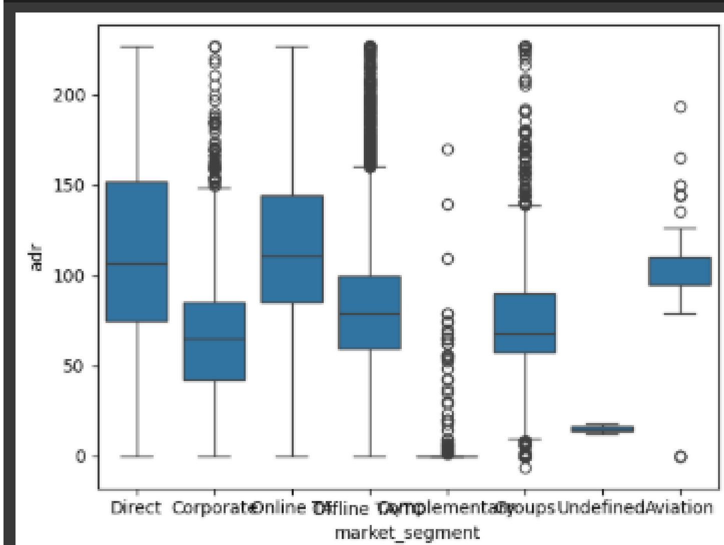
distribution_channel market_segment	Corporate	Direct	GDS	TA/TO	Undefined
Aviation	100.850370	NaN	NaN	95.500000	NaN
Complementary	0.697436	2.521429	NaN	9.694933	NaN
Corporate	67.528169	69.238105	NaN	87.524286	NaN
Direct	62.899630	115.279059	114.000000	109.852882	62.35
Groups	68.255940	68.137458	NaN	77.208428	NaN
Offline TA/TO	101.052632	77.633125	119.155227	81.198370	NaN
Online TA	94.092647	101.959219	120.740441	117.292029	76.50
Undefined	NaN	NaN	NaN	NaN	15.00

What factors are most strongly associated with higher ADR?

The unstacked table of average ADR by market segment and distribution channel shows variations in revenue across different combinations of segments and channels, indicating that some combinations yield higher average ADR.

- What factors are most strongly associated with higher ADR?

```
1 sns.boxplot(data=df1, x='market_segment', y='adr')
2 plt.show()
```



Based on the box plots, the market segment appears to be a strong factor associated with higher ADR.

● Are there customer types or segments consistently contributing to higher revenue?

```
1 df1.groupby('customer_type')['adr'].mean().sort_values(ascending=False)
```

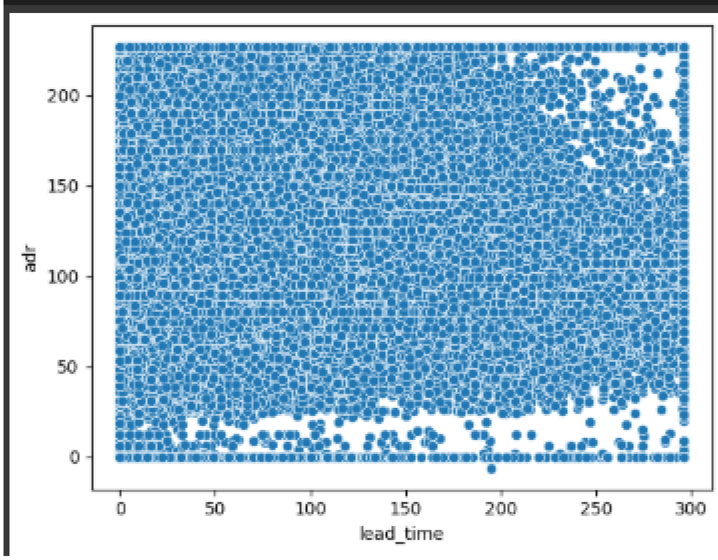
customer_type	adr
Transient	109.188952
Contract	92.863327
Transient-Party	87.459288
Group	84.020148

dtype: float64

Yes, the average ADR by customer type shows that 'Transient' customers have the highest average ADR, suggesting they contribute more to revenue on average.

● Do bookings with more lead time or from specific countries yield higher ADR?

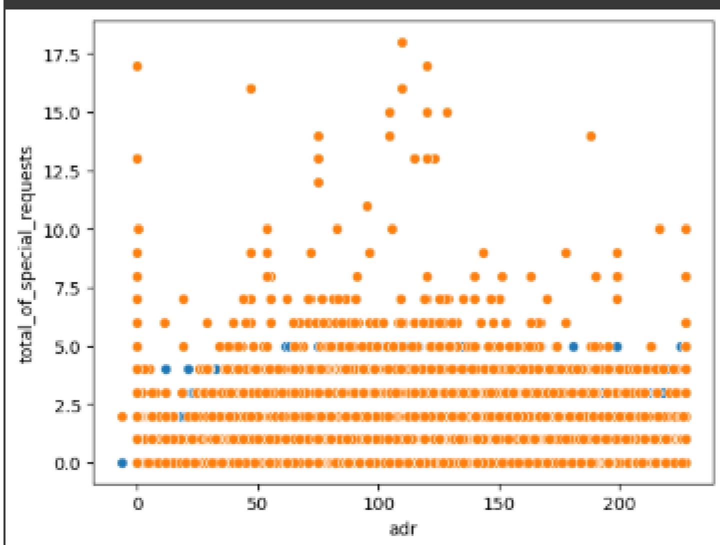
```
1 sns.scatterplot(data=df1, x='lead_time', y='adr')
2 df1.groupby('country')['adr'].mean().sort_values(ascending=False).head(10)
3 plt.show()
```



The scatter plot of lead-time vs. ad shows a weak positive trend, suggesting that bookings with more lead time might yield slightly higher ADR. The analysis of average ADR by country clearly shows that bookings from certain countries have significantly higher average ADR.

● Are guests with higher ADR more likely to request special services or make booking modifications?

```
1 sns.scatterplot(data=df1, x='adr', y='total_of_special_requests')
2 sns.scatterplot(data=df1, x='adr', y='booking_changes')
3 plt.show()
```



The scatter plots of adr vs. total_of_special_requests and adr vs. booking_changes suggest a weak positive relationship, indicating that guests with higher ADR tend to request slightly more special services and make a few more booking changes.

● Do guests from different countries behave differently in terms of booking timing or stay length?

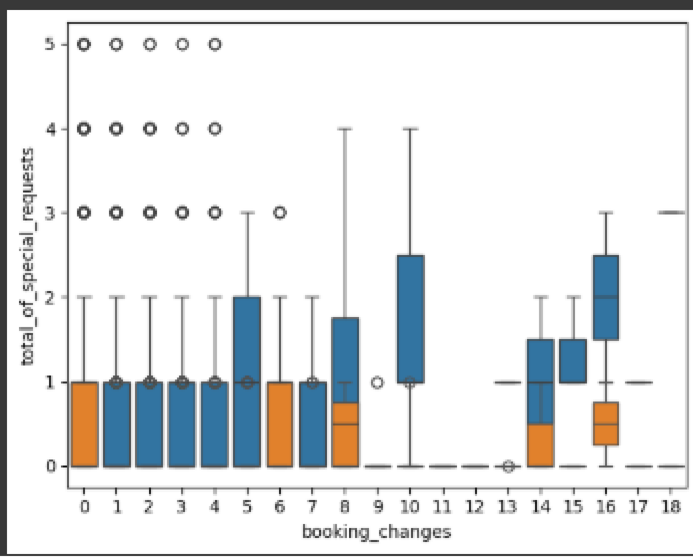
```
1 df1.groupby('country')[['lead_time', 'stay_duration']].mean().sort_values
  (by='lead_time', ascending=False).head()
2
```

country	lead_time	stay_duration
FJI	296.000000	3.000000
LCA	268.000000	5.000000
FRO	254.666667	10.666667
BEN	230.000000	2.333333
MYT	208.000000	3.500000

Yes, the table showing the average lead time and stay duration by country demonstrates variations in booking timing and stay length across different countries.

● Are guests who make booking changes more likely to request additional services or cancel?

```
1 sns.boxplot(data=df1, x='booking_changes', y='total_of_special_requests')
2 sns.boxplot(data=df1, x='booking_changes', y='is_canceled')
3 plt.show()
```



The box plot suggests that as the number of booking changes increases, there might be a slight tendency for the number of special requests to also increase, but this relationship is not very strong.

Conclusion:
The analysis of the hotel bookings dataset yields several important insights that can help guide strategic business decisions:
Data Quality and Integrity: After handling missing values and cleaning the dataset, we ensured the data is reliable for analysis. Columns like agent, children, and country were properly imputed, while irrelevant ones like company were removed to enhance clarity.
Customer Demographics: The total number of guests per booking (total) helps in understanding customer types—solo travelers, families, or groups. This can guide personalized marketing and service offerings.
Booking and Stay Patterns: The engineered stay column reveals the length of stays, crucial for identifying trends in short- vs long-term visits. This supports optimized pricing strategies and room availability planning.
Revenue Estimation: By calculating the revenue column, we get a proxy for the financial value of each booking. This is essential for forecasting income, identifying high-value customers, and optimizing advertising spend.
Seasonality: With the creation of the arrival_month and unified date columns, we can analyze bookings across seasons and identify peak periods. This enables targeted promotions and staff allocation.