Data Anlaysis for Hotel Booking

Importing Necessary Modules

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
#Load the DataSET
```

In [3]:
```
df=pd.read_csv('hotel_booking.csv')
df
```

Out[3]:

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

119390 rows × 36 columns

In [4]:
```
#Check Null Values
```

In [5]:
```
#DATA CLEANING
```

In [6]:
```
df.isnull()
```

Out[6]:

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_date_day_of_month
0	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...
119385	False	False	False	False	False	False	False
119386	False	False	False	False	False	False	False
119387	False	False	False	False	False	False	False
119388	False	False	False	False	False	False	False
119389	False	False	False	False	False	False	False

119390 rows × 36 columns

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
adr                                 0
required_car_parking_spaces         0
total_of_special_requests           0
reservation_status                  0
reservation_status_date             0
name                                0
email                               0
phone-number                        0
credit_card                         0
dtype: int64
```

In [19]:
```
# WE HAVE NULL VALUES AND WE NEED TO Remove them
# So first we need to find the datatype of each column in order to fill the null values
```

In [20]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 119390 entries, 0 to 119389
Data columns (total 36 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   hotel                           119390 non-null  object
 1   is_canceled                     119390 non-null  int64
 2   lead_time                       119390 non-null  int64
 3   arrival_date_year               119390 non-null  int64
 4   arrival_date_month              119390 non-null  object
 5   arrival_date_week_number        119390 non-null  int64
 6   arrival_date_day_of_month       119390 non-null  int64
 7   stays_in_weekend_nights         119390 non-null  int64
 8   stays_in_week_nights            119390 non-null  int64
 9   adults                          119390 non-null  int64
 10  children                        119386 non-null  float64
 11  babies                          119390 non-null  int64
 12  meal                            119390 non-null  object
 13  country                         118902 non-null  object
 14  market_segment                  119390 non-null  object
 15  distribution_channel            119390 non-null  object
 16  is_repeated_guest               119390 non-null  int64
 17  previous_cancellations          119390 non-null  int64
 18  previous_bookings_not_canceled  119390 non-null  int64
 19  reserved_room_type              119390 non-null  object
 20  assigned_room_type              119390 non-null  object
 21  booking_changes                 119390 non-null  int64
 22  deposit_type                    119390 non-null  object
 23  agent                           103050 non-null  float64
 24  company                         6797 non-null    float64
 25  days_in_waiting_list            119390 non-null  int64
 26  customer_type                   119390 non-null  object
 27  adr                             119390 non-null  float64
 28  required_car_parking_spaces     119390 non-null  int64
 29  total_of_special_requests       119390 non-null  int64
 30  reservation_status              119390 non-null  object
 31  reservation_status_date         119390 non-null  object
 32  name                            119390 non-null  object
 33  email                           119390 non-null  object
 34  phone-number                    119390 non-null  object
 35  credit_card                     119390 non-null  object
dtypes: float64(4), int64(16), object(16)
memory usage: 32.8+ MB
```

In [21]:
```
#determine each column data type having a null value
```

In [24]:
```
cnull = df.columns[df.isnull().any()]
for c in cnull:
    ncount = df[c].isnull().sum()
    print(f"Column: {c}, Null values: {ncount}")
```

```
Column: children, Null values: 4
Column: country, Null values: 488
Column: agent, Null values: 16340
Column: company, Null values: 112593
```

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 119390 entries, 0 to 119389
Data columns (total 36 columns):
 #   Column                          Non-Null Count   Dtype  
---  ------                          --------------   -----  
 0   hotel                           119390 non-null  object 
 1   is_canceled                     119390 non-null  int64  
 2   lead_time                       119390 non-null  int64  
 3   arrival_date_year               119390 non-null  int64  
 4   arrival_date_month              119390 non-null  object 
 5   arrival_date_week_number        119390 non-null  int64  
 6   arrival_date_day_of_month       119390 non-null  int64  
 7   stays_in_weekend_nights         119390 non-null  int64  
 8   stays_in_week_nights            119390 non-null  int64  
 9   adults                          119390 non-null  int64  
 10  children                        119386 non-null  float64
 11  babies                          119390 non-null  int64  
 12  meal                            119390 non-null  object 
 13  country                         118902 non-null  object 
 14  market_segment                  119390 non-null  object 
 15  distribution_channel            119390 non-null  object 
 16  is_repeated_guest               119390 non-null  int64  
 17  previous_cancellations          119390 non-null  int64  
 18  previous_bookings_not_canceled  119390 non-null  int64  
 19  reserved_room_type              119390 non-null  object 
 20  assigned_room_type              119390 non-null  object 
 21  booking_changes                 119390 non-null  int64  
 22  deposit_type                    119390 non-null  object 
 23  agent                           103050 non-null  float64
 24  company                         6797 non-null    float64
 25  days_in_waiting_list            119390 non-null  int64  
 26  customer_type                   119390 non-null  object 
 27  adr                             119390 non-null  float64
 28  required_car_parking_spaces     119390 non-null  int64  
 29  total_of_special_requests       119390 non-null  int64  
 30  reservation_status              119390 non-null  object 
 31  reservation_status_date         119390 non-null  object 
 32  name                            119390 non-null  object 
 33  email                           119390 non-null  object 
 34  phone-number                    119390 non-null  object 
 35  credit_card                     119390 non-null  object 
dtypes: float64(4), int64(16), object(16)
memory usage: 32.8+ MB
```

In [37]:
```
df['children'].fillna(df['children'].mean(), inplace=True)
df.isnull().sum()
```

Out[37]:
```
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
name                            0
email                           0
phone-number                    0
credit_card                     0
dtype: int64
```

In [32]:
```
#Now we need to remove the nul values for object data type
```

In [33]:
```
df = df.dropna(subset=['country'])
```

In [34]:
```
df.isnull().sum()
```

Out[34]:
```
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                       16006
company                    112279
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
name                            0
email                           0
phone-number                    0
credit_card                     0
dtype: int64
```

In [38]:
```
df['agent'].fillna(df['agent'].mean(), inplace=True)
```

```
df['company'].fillna(df['company'].mean(), inplace=True)
```

In [40]:
```
df.isnull().sum()
```

Out[40]:
```
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
name                            0
email                           0
phone-number                    0
credit_card                     0
dtype: int64
```

In [41]:
```
# OUr dataset has no null , missing values
```

In [42]:
```
cnull = df.columns[df.isnull().any()]
for c in cnull:
    ncount = df[c].isnull().sum()
    print(f"Column: {c}, Null values: {ncount}")
```

In [43]:
```
#NO output meaning no null or missing or noisy data in our dataset
```

In [44]:
```
 df['reservation_status_date'] = pd.to_datetime(df['reservation_status_date'])
```

In [46]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 118902 entries, 0 to 119389
Data columns (total 36 columns):
 #   Column                          Non-Null Count   Dtype         
---  ------                          --------------   -----         
 0   hotel                           118902 non-null  object        
 1   is_canceled                     118902 non-null  int64         
 2   lead_time                       118902 non-null  int64         
 3   arrival_date_year               118902 non-null  int64         
 4   arrival_date_month              118902 non-null  object        
 5   arrival_date_week_number        118902 non-null  int64         
 6   arrival_date_day_of_month       118902 non-null  int64         
 7   stays_in_weekend_nights         118902 non-null  int64         
 8   stays_in_week_nights            118902 non-null  int64         
 9   adults                          118902 non-null  int64         
 10  children                        118902 non-null  float64       
 11  babies                          118902 non-null  int64         
 12  meal                            118902 non-null  object        
 13  country                         118902 non-null  category      
 14  market_segment                  118902 non-null  object        
 15  distribution_channel            118902 non-null  object        
 16  is_repeated_guest               118902 non-null  int64         
 17  previous_cancellations          118902 non-null  int64         
 18  previous_bookings_not_canceled  118902 non-null  int64         
 19  reserved_room_type              118902 non-null  object        
 20  assigned_room_type              118902 non-null  object        
 21  booking_changes                 118902 non-null  int64         
 22  deposit_type                    118902 non-null  object        
 23  agent                           118902 non-null  float64       
 24  company                         118902 non-null  float64       
 25  days_in_waiting_list            118902 non-null  int64         
 26  customer_type                   118902 non-null  object        
 27  adr                             118902 non-null  float64       
 28  required_car_parking_spaces     118902 non-null  int64         
 29  total_of_special_requests       118902 non-null  int64         
 30  reservation_status              118902 non-null  object        
 31  reservation_status_date         118902 non-null  datetime64[ns]
 32  name                            118902 non-null  object        
 33  email                           118902 non-null  object        
 34  phone-number                    118902 non-null  object        
 35  credit_card                     118902 non-null  object        
dtypes: category(1), datetime64[ns](1), float64(4), int64(16), object(14)
memory usage: 32.9+ MB
```

In [ ]:
```
# to display the unique values for each categorical column in DataFrame.
#It can be useful for understanding the diversity and distribution of values within these columns,
#especially when dealing with categorical or textual data.
```

In [48]:
```
for col in df.describe(include = 'object').columns:
    print(col)
    print(df[col].unique())
    print('-'*50)
```

```
hotel
['Resort Hotel' 'City Hotel']
--------------------------------------------------
arrival_date_month
['July' 'August' 'September' 'October' 'November' 'December' 'January'
 'February' 'March' 'April' 'May' 'June']
--------------------------------------------------
meal
['BB' 'FB' 'HB' 'SC' 'Undefined']
--------------------------------------------------
market_segment
['Direct' 'Corporate' 'Online TA' 'Offline TA/TO' 'Complementary' 'Groups'
 'Undefined' 'Aviation']
--------------------------------------------------
distribution_channel
['Direct' 'Corporate' 'TA/TO' 'Undefined' 'GDS']
--------------------------------------------------
reserved_room_type
['C' 'A' 'D' 'E' 'G' 'F' 'H' 'L' 'B' 'P']
--------------------------------------------------
assigned_room_type
['C' 'A' 'D' 'E' 'G' 'F' 'I' 'B' 'H' 'L' 'K' 'P']
--------------------------------------------------
deposit_type
['No Deposit' 'Refundable' 'Non Refund']
--------------------------------------------------
customer_type
['Transient' 'Contract' 'Transient-Party' 'Group']
--------------------------------------------------
reservation_status
['Check-Out' 'Canceled' 'No-Show']
--------------------------------------------------
name
['Ernest Barnes' 'Andrea Baker' 'Rebecca Parker' ... 'Wesley Aguilar'
 'Caroline Conley MD' 'Ariana Michael']
--------------------------------------------------
email
['Ernest.Barnes31@outlook.com' 'Andrea_Baker94@aol.com'
 'Rebecca_Parker@comcast.net' ... 'Mary_Morales@hotmail.com'
 'MD_Caroline@comcast.net' 'Ariana_M@xfinity.com']
--------------------------------------------------
phone-number
['669-792-1661' '858-637-6955' '652-885-2745' ... '395-518-4100'
 '531-528-1017' '422-804-6403']
--------------------------------------------------
credit_card
['************4322' '************9157' '************3734' ...
 '************9170' '************6349' '************7959']
--------------------------------------------------
```

In [49]:
```
#Now let us summarize the dataset
```

In [52]:
```
df.describe()
```

Out[52]:

	is_canceled	lead_time	arrival_date_year	arrival_date_week_number	arrival_date_day_of_month	stays_in_weekend_ni
count	118902.000000	118902.000000	118902.000000	118902.000000	118902.000000	118902.00
mean	0.371373	104.308027	2016.157617	27.166726	15.800567	0.92
min	0.000000	0.000000	2015.000000	1.000000	1.000000	0.00
25%	0.000000	18.000000	2016.000000	16.000000	8.000000	0.00
50%	0.000000	69.000000	2016.000000	28.000000	16.000000	1.00
75%	1.000000	161.000000	2017.000000	38.000000	23.000000	2.00
max	1.000000	737.000000	2017.000000	53.000000	31.000000	16.00
std	0.483174	106.903127	0.707479	13.589774	8.780371	0.99

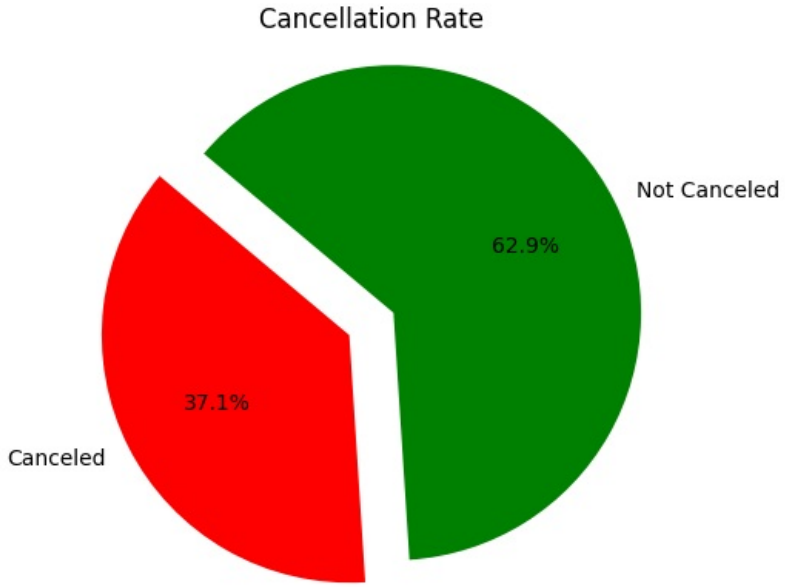
8 rows × 21 columns

Data Visualization

1. Cancellation Rate Visualization

In [63]:
```
cancelrate = df['is_canceled'].mean() * 100
```

```
labels = ['Canceled', 'Not Canceled']
sizes = [cancelrate, 100 - cancelrate]

# Step 3: Choose Colors and Explode
colors = ['red', 'green']
explode = (0.2, 0)  # Separation Between 2 slices (i.e., 'Canceled')

# Step 4: Plot the Pie Chart
plt.pie(sizes, explode=explode, labels=labels, colors=colors, autopct='%1.1f%%', startangle=140)
plt.axis('equal')

# Step 5: Display the Pie Chart
plt.title('Cancellation Rate')
plt.show()
```



In [ ]:
```
#explode parameter is used to emphasize a particular slice by "exploding" it away from the center of the pie cha
#It is a tuple of values where each value represents the fraction of the radius with which to offset each wedge
```

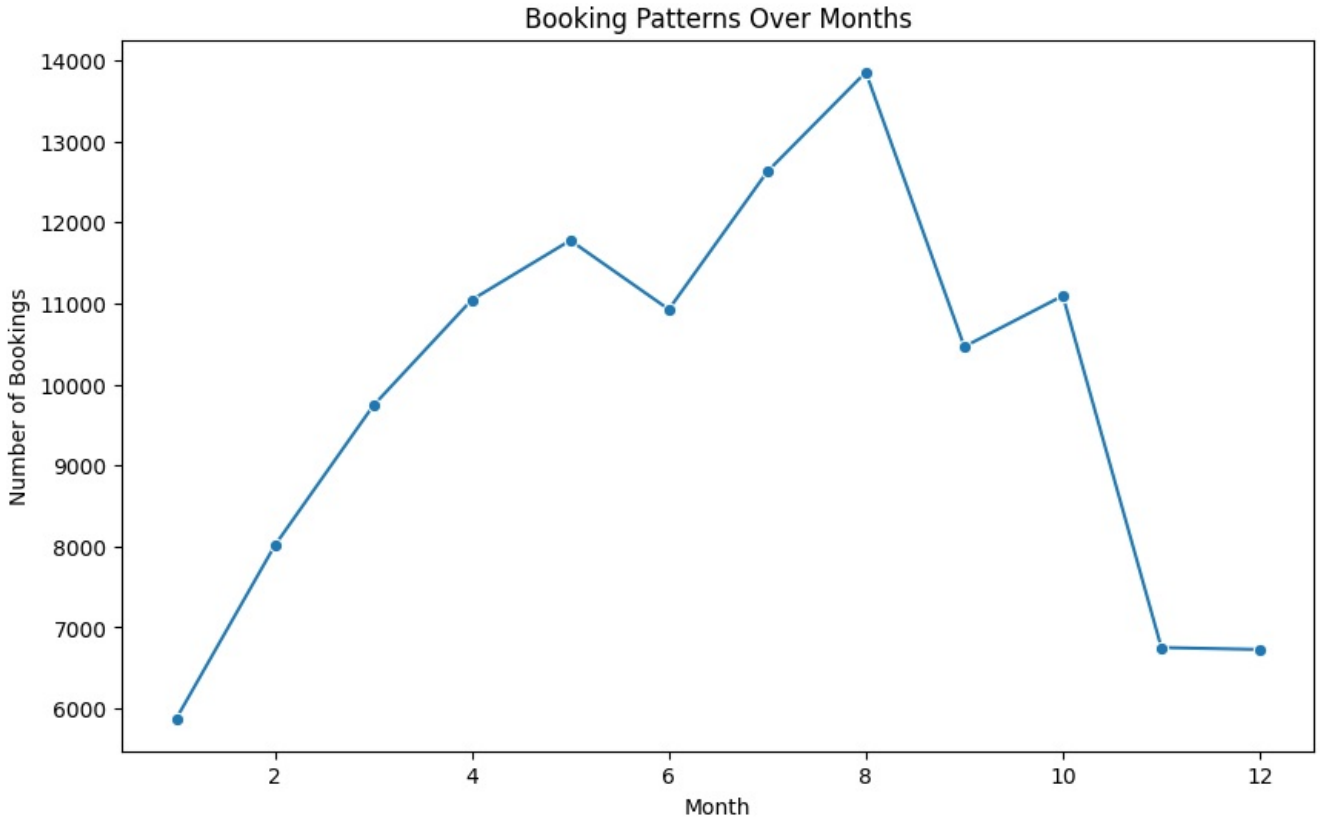
1.1 Indetail Check for City Hotel and Resort Hotel

In [79]:
```
plt.figure(figsize=(8,4))
ax1=sns.countplot(x='hotel',hue='is_canceled',data=df,palette='Blues')
legends_labels,_=ax1.get_legend_handles_labels()
ax1.legend(bbox_to_anchor=(1,1))
plt.title('Reservation status in different hotels', size=18)
plt.xlabel('Hotel')
plt.ylabel('Number of Reservations',)
plt.legend(['Not Cancelled','Cancelled'])
plt.tight_layout()
plt.show()
```

2.Booking patterns over Months

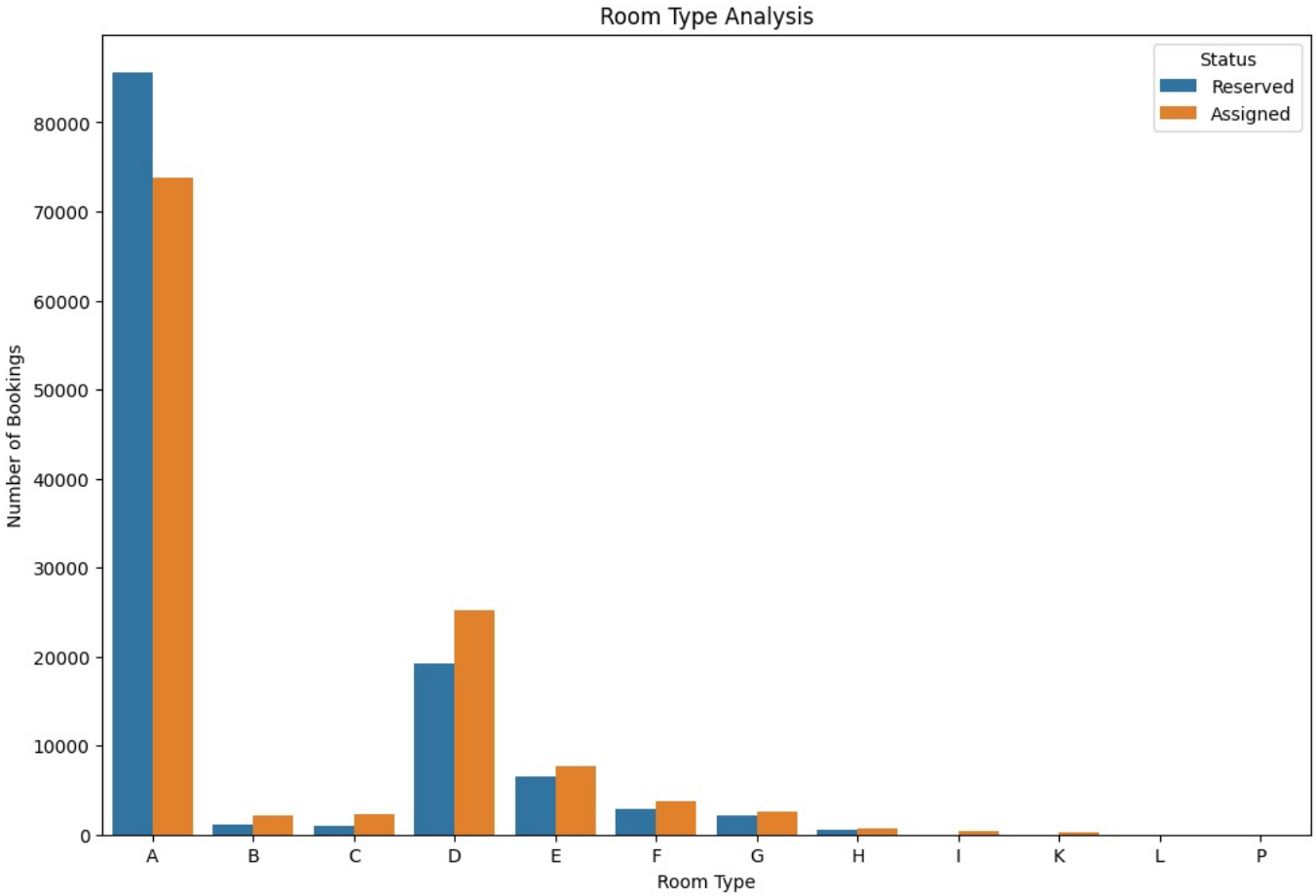
Analyzing the temporal dynamics of hotel bookings provides invaluable insights into the ebb and flow of reservation patterns. Our investigation delves into monthly and annual trends, shedding light on when guests are most inclined to secure accommodations. From the seasonal peaks that mark heightened tourist activity to the nuanced variations in lead times, understanding these temporal patterns empowers hotel management to optimize strategies, allocate resources efficiently, and enhance the overall guest experience.

In [65]:
```
df['arrival_month'] = pd.to_datetime(df['arrival_date_month'], format='%B').dt.month
monthlybook = df.groupby('arrival_month')['hotel'].count()
plt.figure(figsize=(10, 6))
sns.lineplot(x=monthly_bookings.index, y=monthlybook.values, marker='o')

# Set labels and title
plt.xlabel('Month')
plt.ylabel('Number of Bookings')
plt.title('Booking Patterns Over Months')

# Display the plot
plt.show()
```



3. Room Type Analysis Visualization

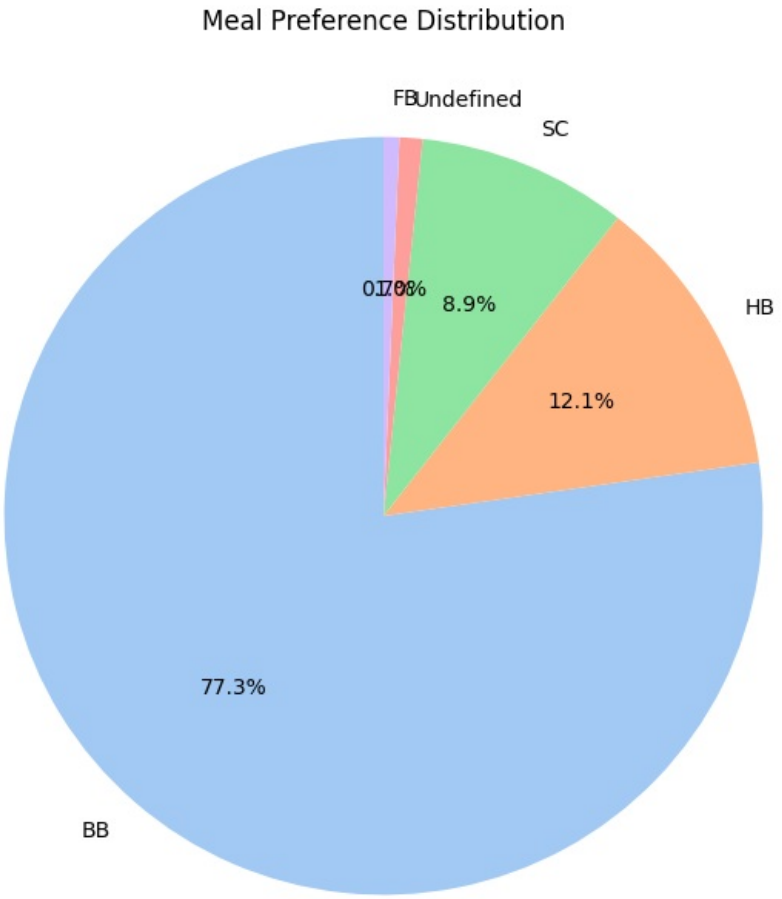
In [97]:
```
reserved_counts = df['reserved_room_type'].value_counts()
assigned_counts = df['assigned_room_type'].value_counts()

# Create a DataFrame for plotting
room_type_df = pd.DataFrame({'Reserved': reserved_counts, 'Assigned': assigned_counts}).reset_index()

# Melt the DataFrame for better visualization
room_type_df_melted = pd.melt(room_type_df, id_vars='index', var_name='Status', value_name='Count')

# Plotting the grouped bar chart using seaborn
plt.figure(figsize=(12, 8))
sns.barplot(x='index', y='Count', hue='Status', data=room_type_df_melted)

# Set labels and title
plt.xlabel('Room Type')
plt.ylabel('Number of Bookings')
plt.title('Room Type Analysis')

# Display the plot
plt.show()
```



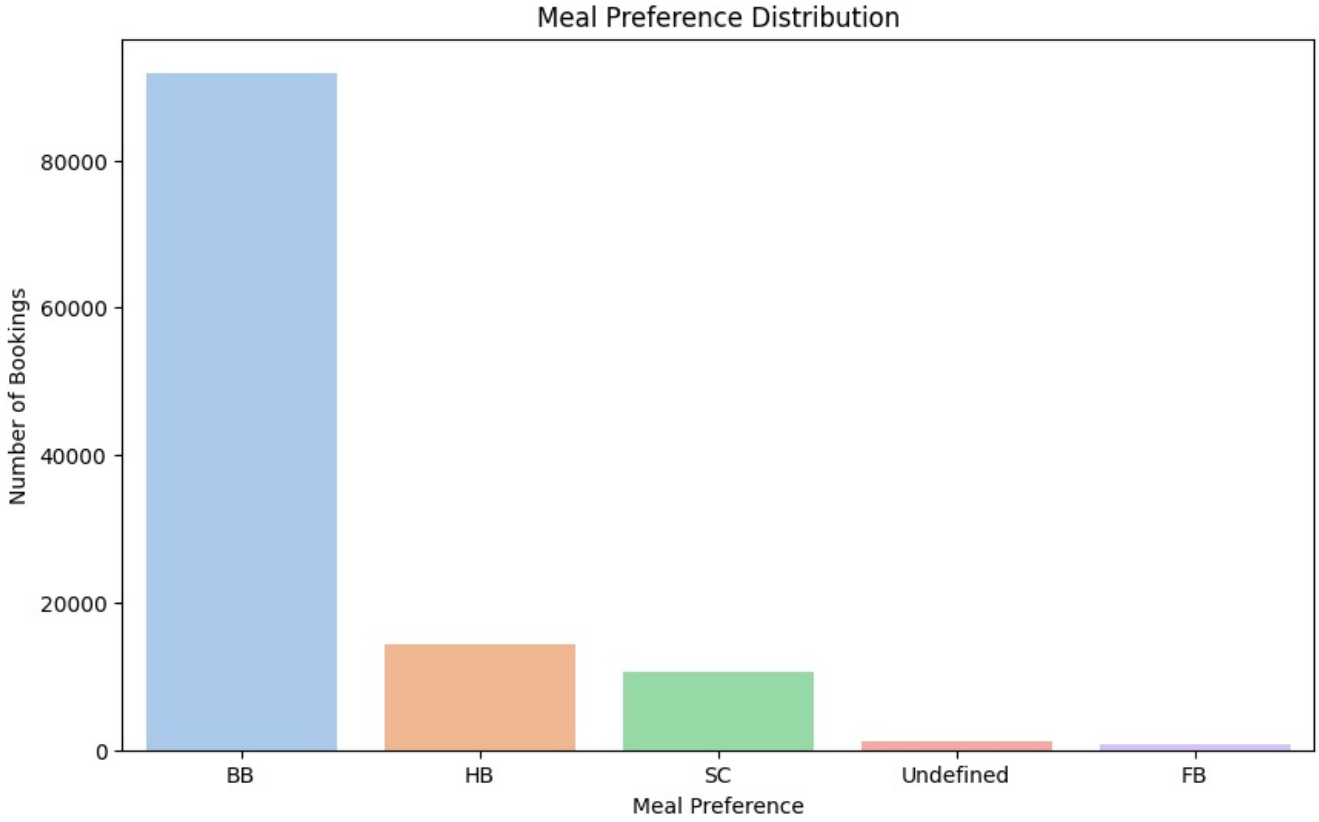
4.Meal Type Visualization

In [108…

```
meal_counts = df['meal'].value_counts()
plt.figure(figsize=(8, 8))
plt.pie(meal_counts, labels=meal_counts.index, autopct='%1.1f%%', startangle=90, colors=sns.color_palette("past
plt.title('Meal Preference Distribution')
plt.show()
```



In [99]:
```
plt.figure(figsize=(10, 6))
sns.barplot(x=meal_counts.index, y=meal_counts, palette="pastel")
plt.xlabel('Meal Preference')
plt.ylabel('Number of Bookings')
plt.title('Meal Preference Distribution')
plt.show()
```



5.Insights for Room Cancellation by Room Type

Analysis of room cancellations by room type reveals varying cancellation rates across different room categories. Identifying room types with higher cancellations can guide strategies for optimization, while understanding the distribution of room type preferences informs operational decisions and marketing efforts to mitigate cancellations. These insights enable targeted approaches to enhance guest satisfaction and overall hotel performance.

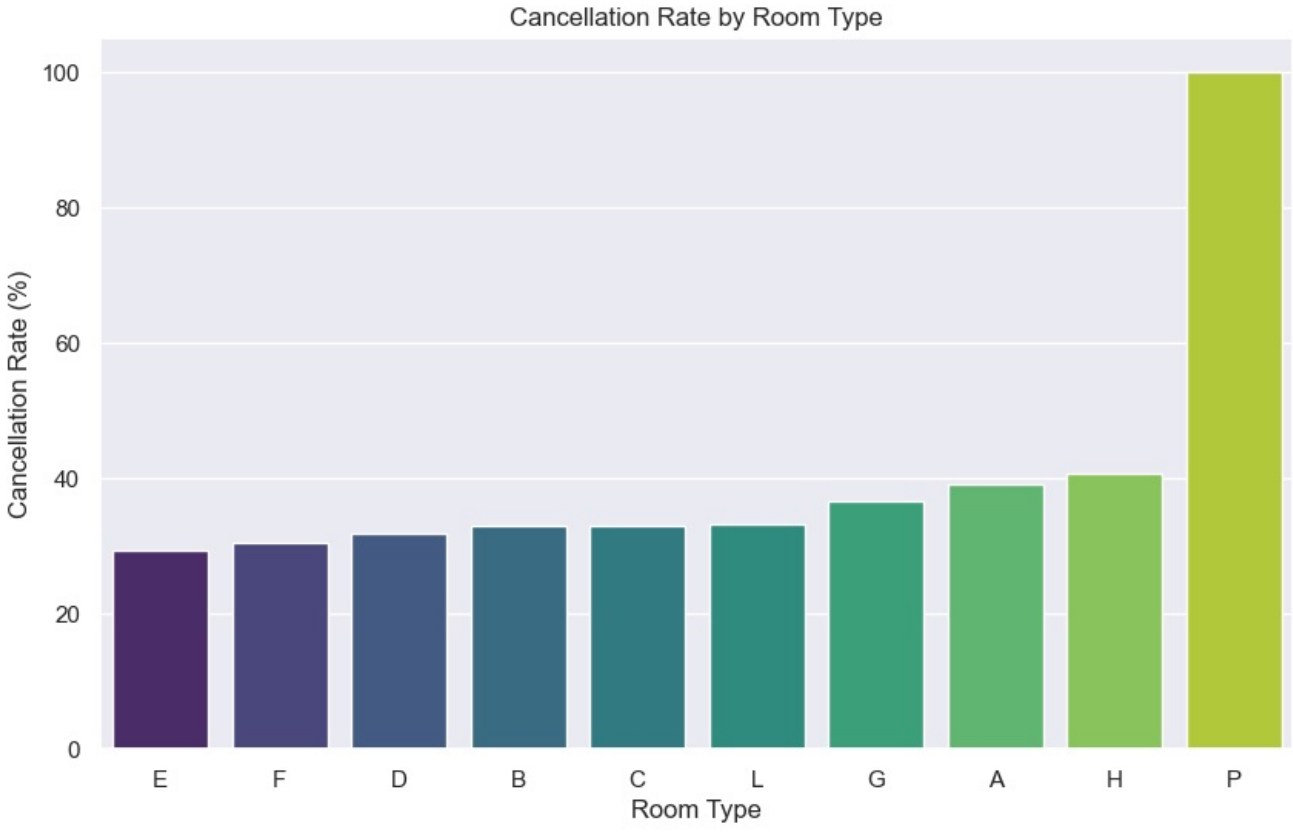
Step 1.Calculate Cancellation Rates by Room Type:

In [46]:
```
room_cancellation = df.groupby('reserved_room_type')['is_canceled'].mean().sort_values()
print("Cancellation Rates by Room Type:")
print(room_cancellation)
```

```
Cancellation Rates by Room Type:
reserved_room_type
E    0.293828
F    0.304498
D    0.318208
B    0.329159
C    0.330827
L    0.333333
G    0.366299
A    0.392273
H    0.407654
P    1.000000
Name: is_canceled, dtype: float64
```

Step 2.Visualize Cancellation Rates by Room Type:

In [47]:
```
sns.set(style="darkgrid")
plt.figure(figsize=(10, 6))
sns.barplot(x=room_cancellation.index, y=room_cancellation.values * 100, palette='viridis')
plt.title('Cancellation Rate by Room Type')
plt.xlabel('Room Type')
plt.ylabel('Cancellation Rate (%)')
plt.show()
```



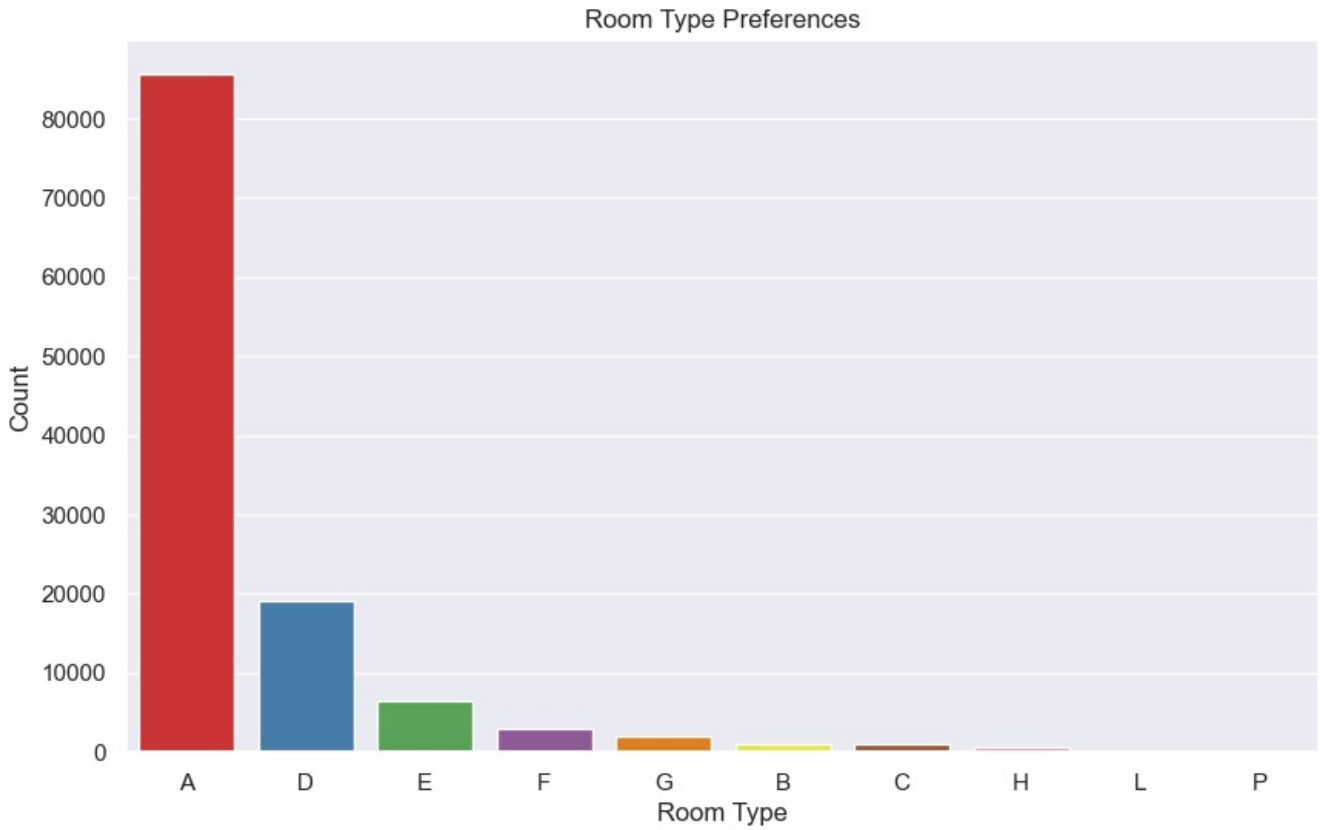
Step 3. Explore Room Type Preferences:

In [48]:
```
# Count the occurrences of each reserved room type
room_type_counts = df['reserved_room_type'].value_counts()
print("Room Type Preferences:")
print(room_type_counts)
```

```
Room Type Preferences:
reserved_room_type
A    85601
D    19173
E     6497
F     2890
G     2083
B     1118
C      931
H      601
L        6
P        2
Name: count, dtype: int64
```

Step 4. Visualize Room Type Preferences:

In [49]:
```
# Create a bar chart for room type preferences
plt.figure(figsize=(10, 6))
sns.countplot(x='reserved_room_type', data=df, order=room_type_counts.index, palette='Set1')
plt.title('Room Type Preferences')
plt.xlabel('Room Type')
plt.ylabel('Count')
plt.show()
```



6. Insights of Booking Trends

6.1 Temporal Patterns: Booking distribution over months

6.2 Lead Time Analysis: Distribution of lead time

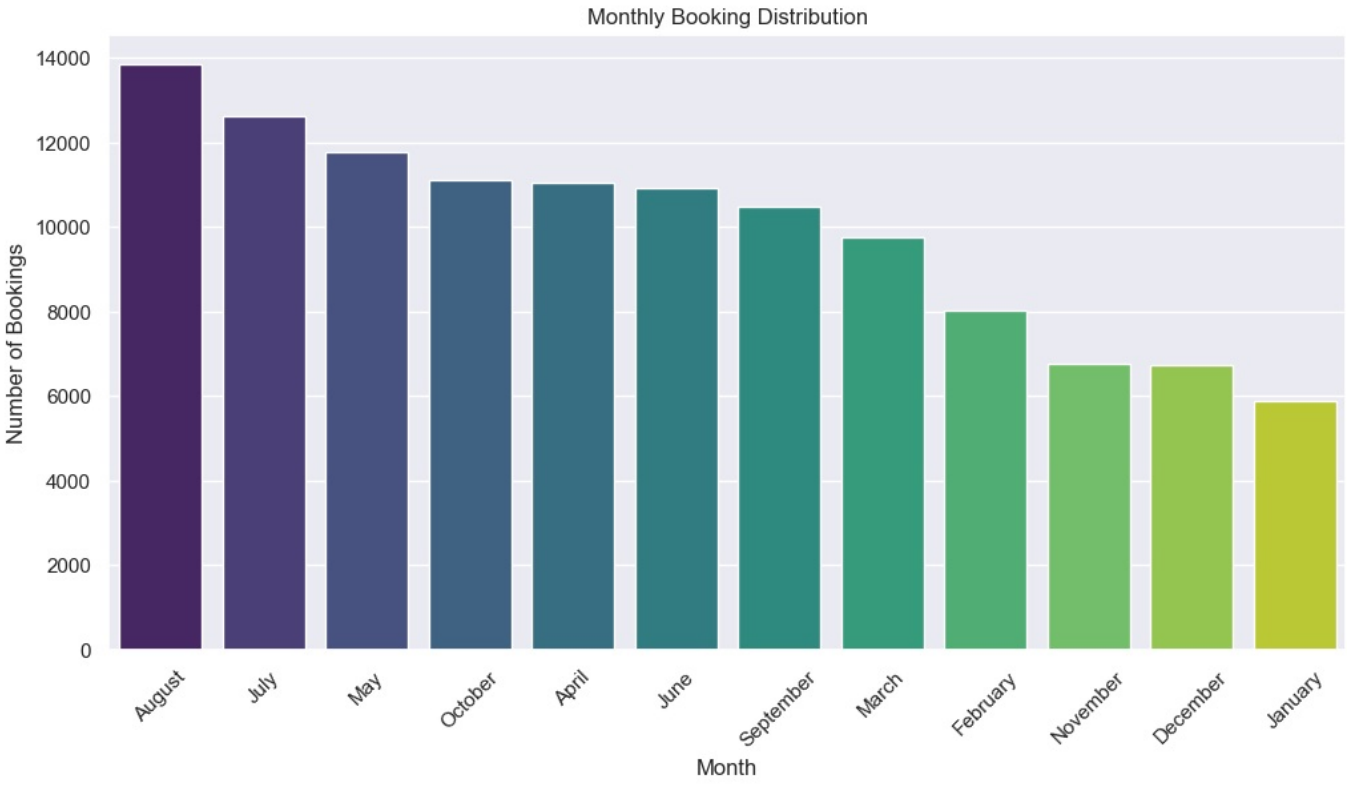
6.3 Booking Channel Trends: Distribution of booking channels

6.1 Temporal Patterns:

Seasonal peaks in bookings suggest high demand during certain months, allowing hotels to optimize pricing and resource allocation for peak periods.

In [53]:
```
# Temporal Patterns: Booking distribution over months
plt.figure(figsize=(12, 6))
sns.countplot(x='arrival_date_month', data=df, order=df['arrival_date_month'].value_counts().index, palette='vi
plt.title('Monthly Booking Distribution')
```

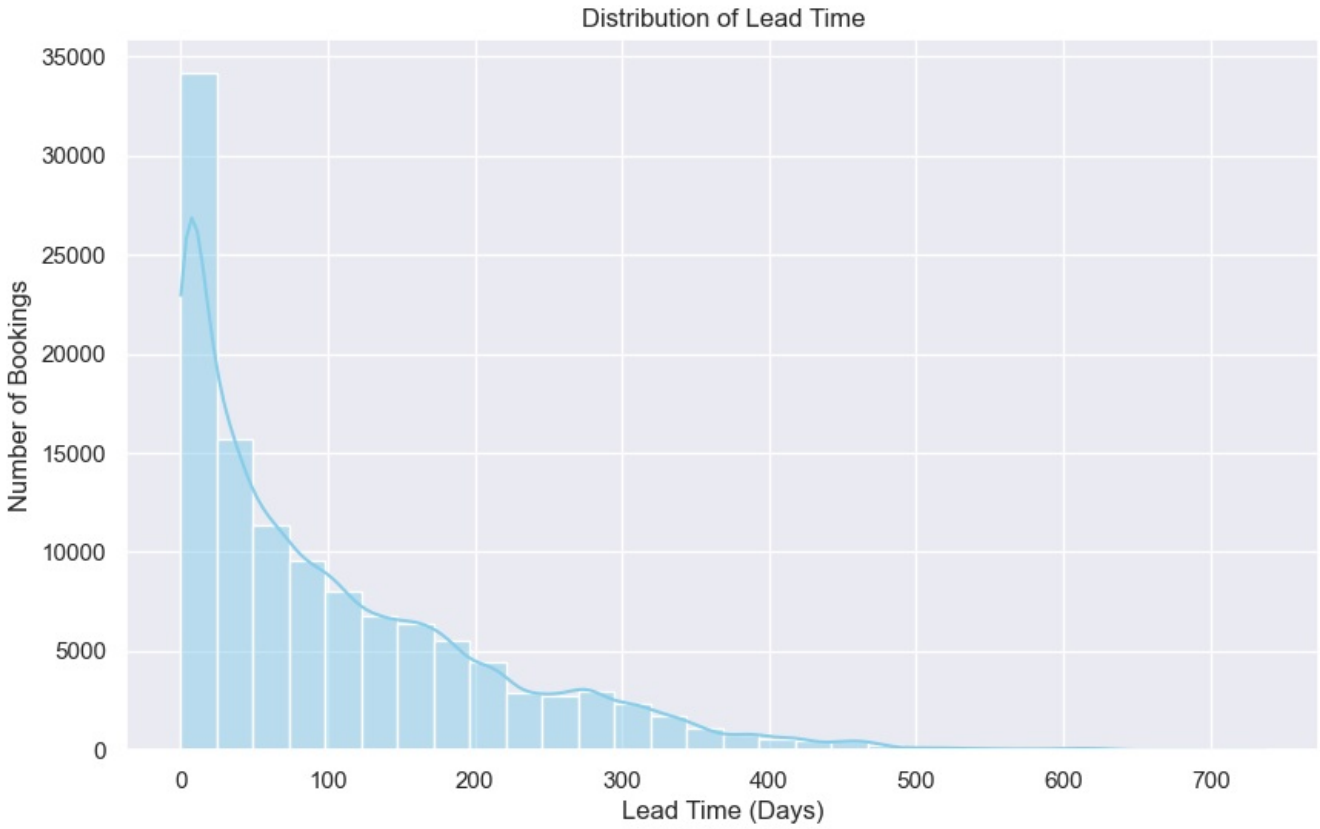
```
plt.xlabel('Month')
plt.ylabel('Number of Bookings')
plt.xticks(rotation=45)
plt.show()
```



6.2 Lead Time Analysis:

Understanding lead time patterns helps in managing reservations effectively, adjusting staff levels, and tailoring marketing strategies for early or last-minute bookings

In [54]:
```
# Lead Time Analysis: Distribution of lead time
plt.figure(figsize=(10, 6))
sns.histplot(df['lead_time'], bins=30, kde=True, color='skyblue')
plt.title('Distribution of Lead Time')
plt.xlabel('Lead Time (Days)')
plt.ylabel('Number of Bookings')
plt.show()
```



6.3 Booking Channel Trends:

Identifying popular booking channels informs marketing focus, ensuring resources are allocated to the most effective channels for attracting guests.

In [55]:
```
# Booking Channel Trends: Distribution of booking channels
plt.figure(figsize=(8, 6))
sns.countplot(x='distribution_channel', data=df, palette='Set2')
plt.title('Booking Channel Distribution')
plt.xlabel('Distribution Channel')
plt.ylabel('Number of Bookings')
plt.show()
```

7.Customer Segmented Insights

7.1 Customer Segment Distribution

7.2 Booking Channel Preferences

7.3 Cancellation Rates by Segment

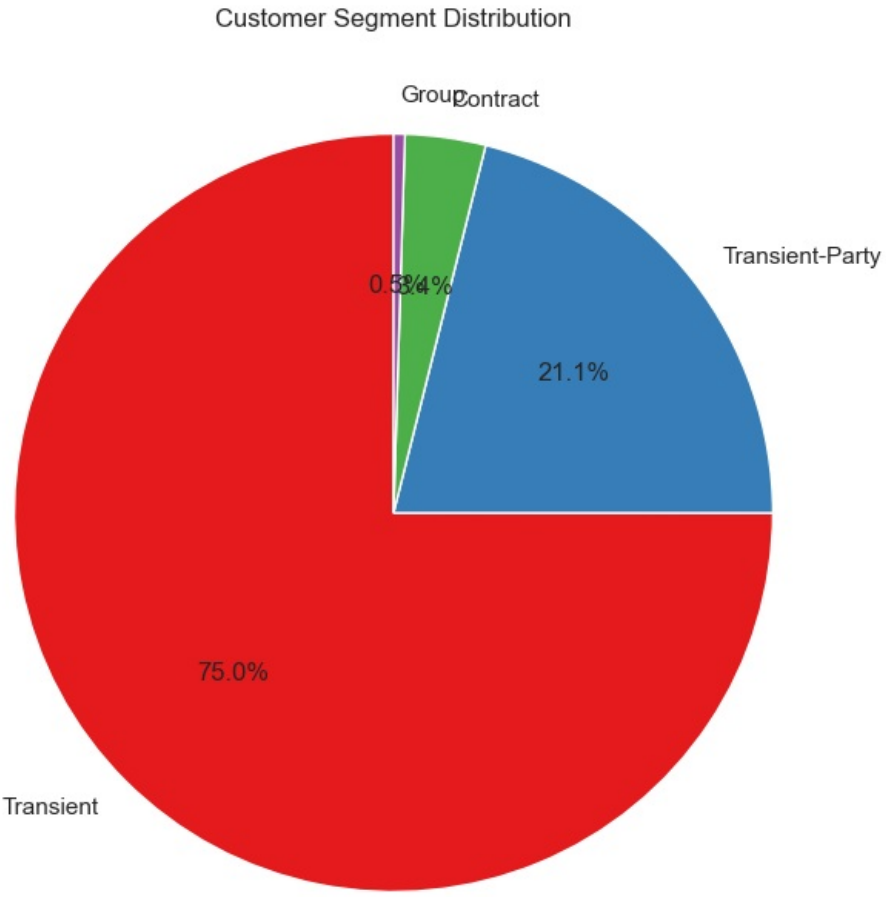
7.4 Lead Time Preferences

7.1 Customer Segment Distribution

Insight: Visualizing the proportion of bookings from diverse customer segments.

Use Case: Tailoring services and marketing strategies based on an understanding of the overall customer segment composition.

In [74]:
```
# Customer Segment Distribution: Pie chart
plt.figure(figsize=(8, 8))
ctypecount = df['customer_type'].value_counts()
plt.pie(ctypecount, labels=customer_type_counts.index, autopct='%1.1f%%', colors=sns.color_palette('Set1'), sta
plt.title('Customer Segment Distribution')
plt.show()
```



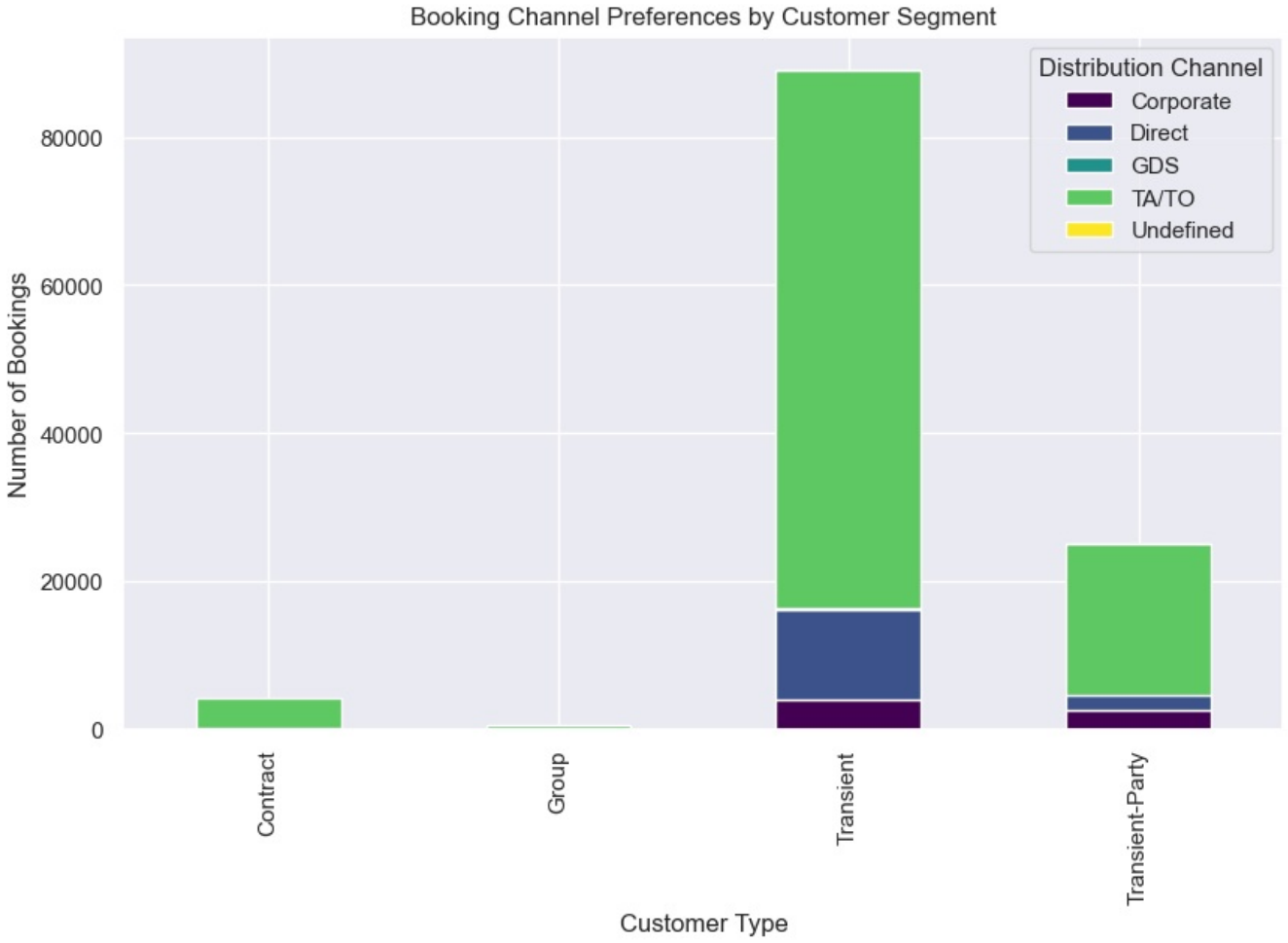
7.2 Booking Channel Preferences

Insight: Displaying booking channel preferences for each customer segment.

Use Case: Informed marketing decisions, allowing targeted promotions aligned with the preferred booking channels of different customer segments.

In [73]:
```
# Booking Channel Preferences: Stacked bar chart
```

```
channel_segment_preferences = df.groupby(['customer_type', 'distribution_channel']).size().unstack()
channel_segment_preferences.plot(kind='bar', stacked=True, colormap='viridis', figsize=(10, 6))
plt.title('Booking Channel Preferences by Customer Segment')
plt.xlabel('Customer Type')
plt.ylabel('Number of Bookings')
plt.legend(title='Distribution Channel', loc='upper right')
plt.show()
```



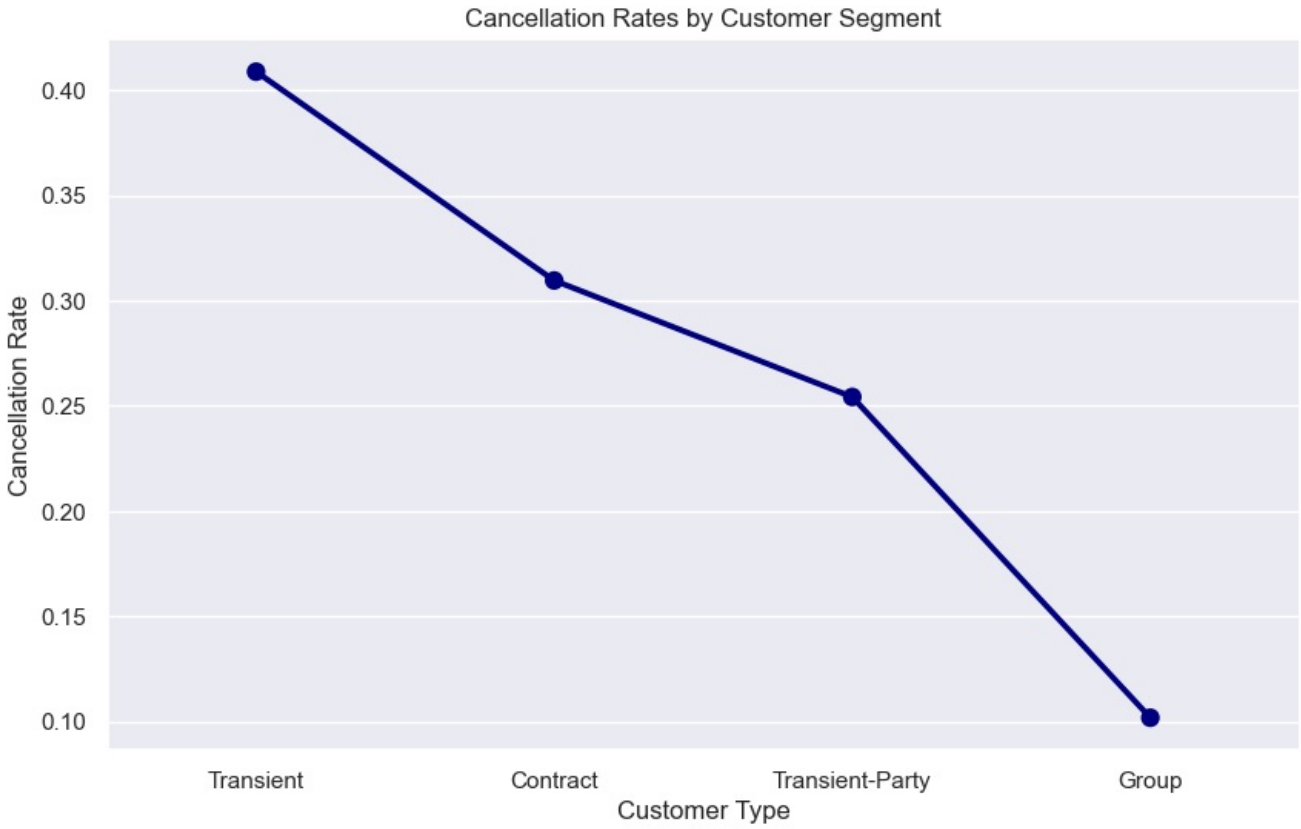
7.3 Cancellation Rates by Segment

Insight: Comparing cancellation rates across various customer segments.

Use Case: Identifying segments with differing cancellation behaviors to implement tailored policies and improve overall booking reliability

In [68]:
```
# Cancellation Rates by Segment: Point plot
plt.figure(figsize=(10, 6))
sns.pointplot(x='customer_type', y='is_canceled', data=df, ci=None, color='navy')
plt.title('Cancellation Rates by Customer Segment')
plt.xlabel('Customer Type')
plt.ylabel('Cancellation Rate')
plt.show()
```

```
C:\Users\ganes\AppData\Local\Temp\ipykernel_1452\4252886775.py:3: FutureWarning:

The `ci` parameter is deprecated. Use `errorbar=None` for the same effect.

  sns.pointplot(x='customer_type', y='is_canceled', data=df, ci=None, color='navy')
```

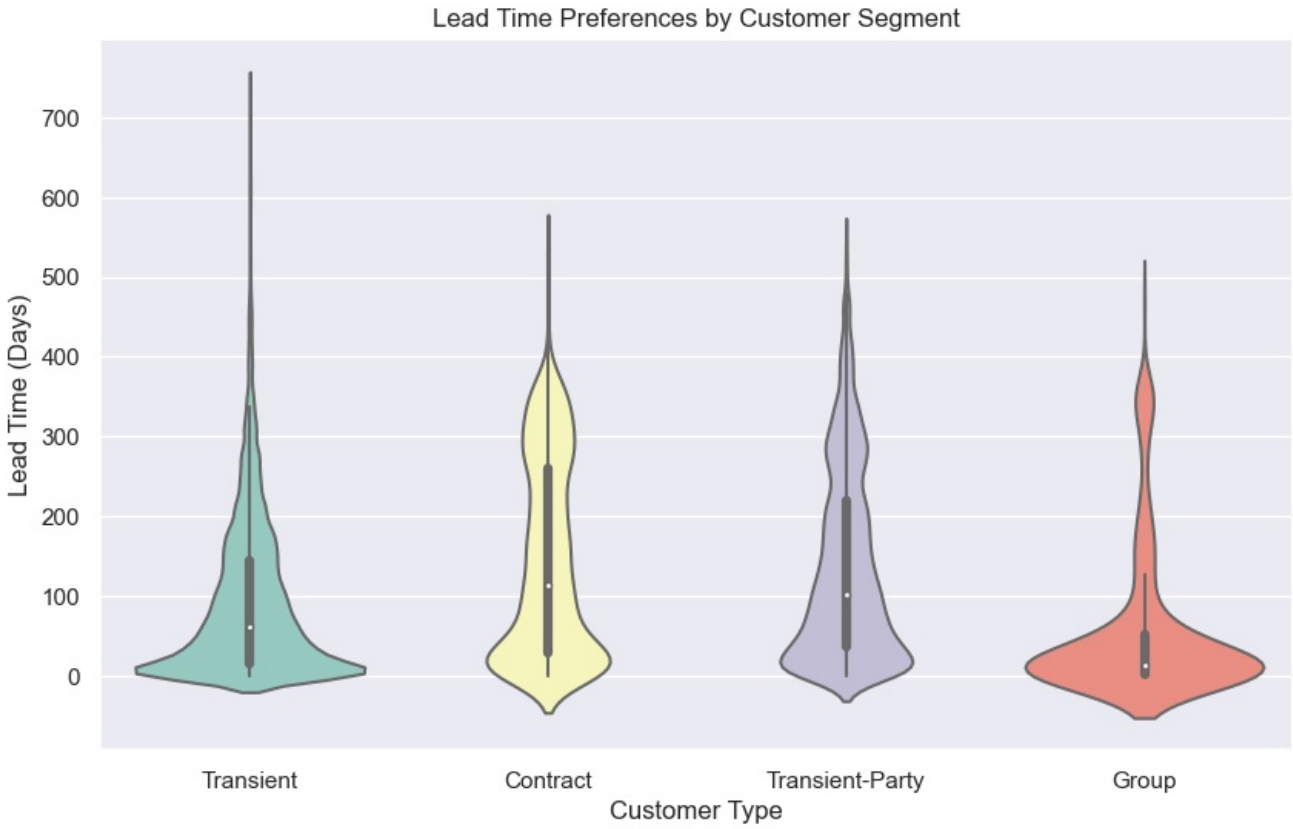


7.4 Lead Time Preferences

Insight: Illustrating the distribution of lead times for bookings in distinct customer segments.

Use Case: Informing strategies by understanding how different customer segments plan and book in advance or closer to their arrival dates.

In [70]:
```
# Lead Time Preferences: Violin plot
plt.figure(figsize=(10, 6))
sns.violinplot(x='customer_type', y='lead_time', data=df, palette='Set3')
plt.title('Lead Time Preferences by Customer Segment')
plt.xlabel('Customer Type')
plt.ylabel('Lead Time (Days)')
plt.show()
```



THANK YOU