```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

Cleaning Dataset

```
# Load the dataset
df = pd.read_csv('/kaggle/input/hotel-bookings/hotel_bookings.csv')

# 1. Create the Resort Hotel DataFrame
resort_hotel = df[df['hotel'] == 'Resort Hotel'].copy()

# 2. Create the City Hotel DataFrame
city_hotel = df[df['hotel'] == 'City Hotel'].copy()

# 3. Verify the split
print(f"Resort Hotel Shape: {resort_hotel.shape}")
print(f"City Hotel Shape:   {city_hotel.shape}")
```

```
Resort Hotel Shape: (40060, 32)
City Hotel Shape:   (79330, 32)
```

```
# 4. Create the 'is_family' feature
df['children'] = df['children'].fillna(0)
df['is_family'] = ((df['children'] > 0) | (df['babies'] >
0)).astype(int)

# 4. Create 'date_of_arrival'
# We combine Year (int), Month (str), and Day (int) into a single
datetime object
df['date_of_arrival'] = pd.to_datetime(
    df['arrival_date_year'].astype(str) + '-' +
    df['arrival_date_month'] + '-' +
    df['arrival_date_day_of_month'].astype(str)
)

# C. Create 'duration_of_stay'
# Summing weekend and week nights
df['duration_of_stay'] = df['stays_in_weekend_nights'] +
df['stays_in_week_nights']

# Define the mapping dictionary for Northern Hemisphere seasons
season_mapping = {
    'December': 'Winter', 'January': 'Winter', 'February': 'Winter',
    'March': 'Spring', 'April': 'Spring', 'May': 'Spring',
    'June': 'Summer', 'July': 'Summer', 'August': 'Summer',
    'September': 'Autumn', 'October': 'Autumn', 'November': 'Autumn'
}
```

```python
# Map the month column to the new season column
df['season_of_booking'] = df['arrival_date_month'].map(season_mapping)

# 1. Get the counts of each season
season_counts = df['season_of_booking'].value_counts()

print("Counts per season:")
print(season_counts)

# 2. (Optional) Calculate the percentage of bookings per season
season_percentages =
df['season_of_booking'].value_counts(normalize=True) * 100

print("\nPercentage per season:")
print(season_percentages)
```

```
Counts per season:
season_of_booking
Summer    37477
Spring    32674
Autumn    28462
Winter    20777
Name: count, dtype: int64

Percentage per season:
season_of_booking
Summer    31.390401
Spring    27.367451
Autumn    23.839518
Winter    17.402630
Name: proportion, dtype: float64
```

```python
columns_to_remove = [
    'company',
    'days_in_waiting_list',
    'required_car_parking_spaces',
    'total_of_special_requests',
    'arrival_date_week_number',
    'children',
    'babies',
    'stays_in_weekend_nights',
    'stays_in_week_nights',
    'arrival_date_year',
    'arrival_date_month',
    'arrival_date_day_of_month',
]

# Dropping the columns
df.drop(columns=columns_to_remove, inplace=True)
```

Handling Missing Data

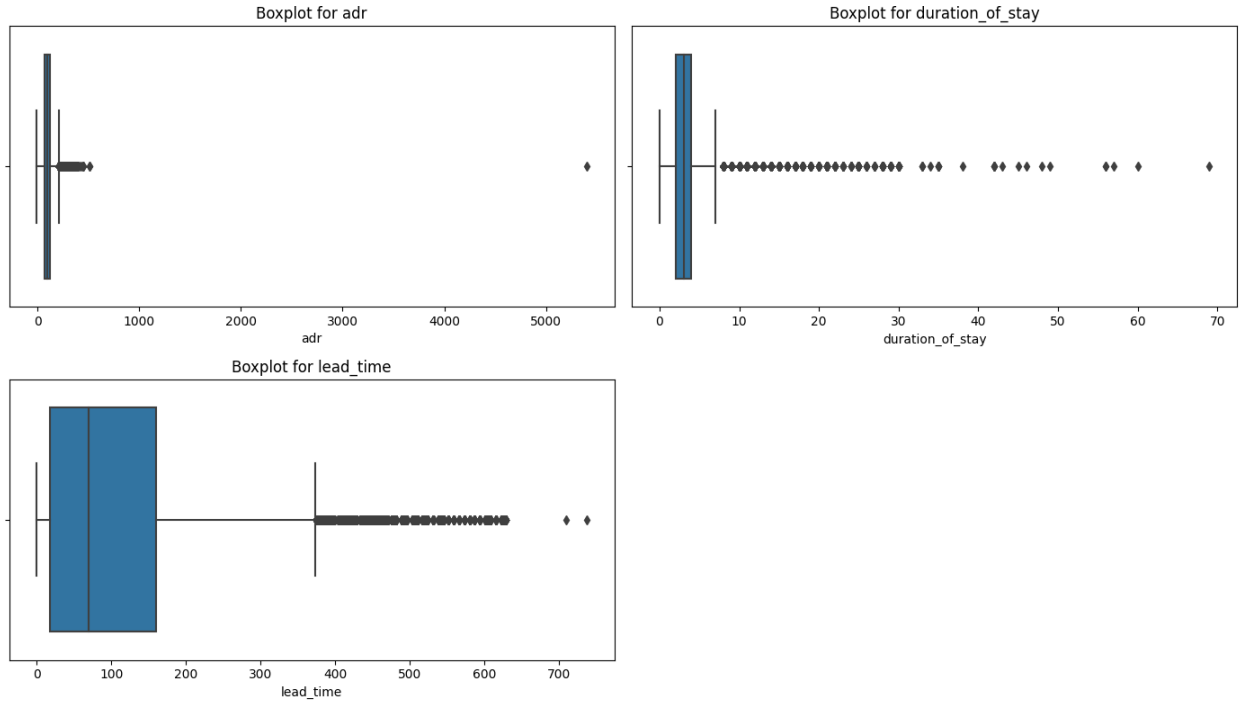
```
missing_data = df.isnull().sum()
print("Missing data per column:")
print(missing_data[missing_data > 0])
```

```
Missing data per column:
country      488
agent      16340
dtype: int64
```

```
# Convert columns to string type before filling missing values
df['country'] = df['country'].astype(str)
df['agent'] = df['agent'].astype(str)

# Fill missing values with 'Unknown'
df['country'] = df['country'].fillna('Unknown')
df['agent'] = df['agent'].fillna('Unknown')
```

Handling Outliers

```
import matplotlib.pyplot as plt
import seaborn as sns

# Columns we will check for outliers
numerical_columns = ['adr', 'duration_of_stay', 'lead_time']

# Plotting boxplots to visualize outliers
plt.figure(figsize=(14, 8))

for i, column in enumerate(numerical_columns, 1):
    plt.subplot(2, 2, i)
    sns.boxplot(x=df[column])
    plt.title(f'Boxplot for {column}')

plt.tight_layout()
plt.show()
```




```
# Define a function to remove outliers using IQR
def remove_outliers(df, column):
    Q1 = df[column].quantile(0.25)
    Q3 = df[column].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    return df[(df[column] >= lower_bound) & (df[column] <=
upper_bound)]

# Apply the function to the relevant columns
df = remove_outliers(df, 'adr')
df = remove_outliers(df, 'duration_of_stay')
df = remove_outliers(df, 'lead_time')
```

Exploratory Data Analysis and Data Visualization

```
# Basic numeric summary
numeric_summary = df.describe().T
print(numeric_summary)
```

```
                                  count
mean  \
is_canceled                    107631.0
```

```
0.362321
lead_time                         107631.0
93.900512
adults                            107631.0
1.842406
is_repeated_guest                 107631.0
0.034172
previous_cancellations            107631.0
0.093356
previous_bookings_not_canceled    107631.0
0.148637
booking_changes                   107631.0
0.210896
adr                               107631.0
97.522635
is_family                         107631.0
0.066728
date_of_arrival                     107631  2016-08-20
04:29:04.761267712
duration_of_stay                  107631.0
3.059472

                                                 min
25%  \
is_canceled                                      0.0
0.0
lead_time                                        0.0
16.0
adults                                           0.0
2.0
is_repeated_guest                                0.0
0.0
previous_cancellations                           0.0
0.0
previous_bookings_not_canceled                   0.0
0.0
booking_changes                                  0.0
0.0
adr                                              0.0
69.5
is_family                                        0.0
0.0
date_of_arrival                  2015-07-01 00:00:00  2016-03-04
00:00:00
duration_of_stay                                 0.0
2.0

                                                 50%
75%  \
```

```
is_canceled                                    0.0
1.0
lead_time                                     64.0
150.0
adults                                         2.0
2.0
is_repeated_guest                              0.0
0.0
previous_cancellations                         0.0
0.0
previous_bookings_not_canceled                 0.0
0.0
booking_changes                                0.0
0.0
adr                                           93.6
122.0
is_family                                      0.0
0.0
date_of_arrival                2016-08-28 00:00:00  2017-03-08
00:00:00
duration_of_stay                               3.0
4.0

                                               max         std
is_canceled                                    1.0    0.480673
lead_time                                    372.0   92.505491
adults                                        55.0    0.590164
is_repeated_guest                              1.0    0.181672
previous_cancellations                        26.0    0.886937
previous_bookings_not_canceled                72.0    1.560178
booking_changes                               18.0    0.622265
adr                                         211.03   40.653684
is_family                                      1.0    0.249551
date_of_arrival                2017-08-31 00:00:00         NaN
duration_of_stay                               7.0    1.726012
```

```
# Categorical summary
categorical_summary = df.describe(include='object').T
print(categorical_summary)
```

```
                         count unique         top   freq
hotel                   107631      2  City Hotel  74231
meal                    107631      5          BB  84122
country                 107631    176         PRT  43805
market_segment          107631      8   Online TA  51526
distribution_channel    107631      5       TA/TO  87999
reserved_room_type      107631     10           A  79960
assigned_room_type      107631     12           A  68447
deposit_type            107631      3  No Deposit  94777
agent                   107631    324         9.0  30511
```

```
customer_type             107631      4   Transient  80741
reservation_status        107631      3   Check-Out  68634
reservation_status_date   107631    915  2015-07-06    803
season_of_booking         107631      4      Summer  31107
```

```
# Cancellation baseline
cancellation_rate = df['is_canceled'].value_counts(normalize=True) *
100
print(cancellation_rate)
```

```
is_canceled
0    63.767874
1    36.232126
Name: proportion, dtype: float64
```

```
# Skewness of numeric features
skewness = df.skew(numeric_only=True)
print(skewness)
```

```
is_canceled                      0.572869
lead_time                        1.019403
adults                          19.249470
is_repeated_guest                5.128311
previous_cancellations          23.377568
previous_bookings_not_canceled  22.509617
booking_changes                  5.516513
adr                              0.329938
is_family                        3.472470
duration_of_stay                 0.773545
dtype: float64
```

```
# Show unique values for each categorical column for better
understanding
print("Unique values in each categorical column:")
for col in df.select_dtypes(include='object').columns:
    print(f"\n{col}:")
    print(df[col].unique())
    print('-'*50)
```

```
Unique values in each categorical column:

hotel:
['Resort Hotel' 'City Hotel']
--------------------------------------------------

meal:
['BB' 'FB' 'HB' 'SC' 'Undefined']
--------------------------------------------------

country:
['PRT' 'GBR' 'USA' 'ESP' 'IRL' 'FRA' 'ROU' 'NOR' 'ARG' 'POL' 'DEU'
```

```
'BEL'
 'CHE' 'CN' 'GRC' 'NLD' 'RUS' 'SWE' 'AUS' 'EST' 'CZE' 'BRA' 'ITA'
'FIN'
 'DNK' 'MOZ' 'BWA' 'LUX' 'SVN' 'ALB' 'CHN' 'MEX' 'MAR' 'SMR' 'LVA'
'PRI'
 'SRB' 'IND' 'CHL' 'AUT' 'LTU' 'OMN' 'TUR' 'ZAF' 'AGO' 'ISR' 'CYM'
'ZMB'
 'CPV' 'ZWE' 'DZA' 'KOR' 'CRI' 'HUN' 'nan' 'ARE' 'TUN' 'JAM' 'HRV'
'HKG'
 'IRN' 'AND' 'GIB' 'URY' 'BLR' 'JEY' 'CAF' 'CYP' 'COL' 'GGY' 'KWT'
'NGA'
 'MDV' 'VEN' 'FJI' 'SVK' 'LBN' 'PHL' 'SYC' 'BHR' 'NZL' 'KAZ' 'THA'
'DOM'
 'MYS' 'UKR' 'ARM' 'JPN' 'LKA' 'CUB' 'CMR' 'BIH' 'MUS' 'COM' 'SUR'
'UGA'
 'BGR' 'CIV' 'JOR' 'SYR' 'SGP' 'BDI' 'SAU' 'VNM' 'AZE' 'PLW' 'QAT'
'EGY'
 'MLT' 'MWI' 'ECU' 'MDG' 'IDN' 'ISL' 'UZB' 'NPL' 'BHS' 'PAK' 'MAC'
'TWN'
 'STP' 'SEN' 'PER' 'KNA' 'ETH' 'IRQ' 'HND' 'GEO' 'KHM' 'MCO' 'BGD'
'IMN'
 'TJK' 'NIC' 'BEN' 'VGB' 'TZA' 'GAB' 'MKD' 'TMP' 'GLP' 'LIE' 'GNB'
'KEN'
 'MNE' 'UMI' 'MYT' 'MMR' 'PAN' 'BFA' 'LBY' 'MLI' 'NAM' 'BOL' 'PRY'
'BRB'
 'ABW' 'SLV' 'DMA' 'PYF' 'GUY' 'LCA' 'ATA' 'RWA' 'GTM' 'GHA' 'ASM'
'TGO'
 'MRT' 'NCL' 'KIR' 'SDN' 'ATF' 'SLE' 'LAO' 'FRO']
--------------------------------------------------

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' 'P' 'B']
--------------------------------------------------

assigned_room_type:
['C' 'A' 'D' 'E' 'G' 'F' 'I' 'B' 'H' 'P' 'L' 'K']
--------------------------------------------------

deposit_type:
['No Deposit' 'Refundable' 'Non Refund']
```

```
------------------------------------------------

agent:
['nan' '304.0' '240.0' '303.0' '15.0' '241.0' '8.0' '250.0' '115.0'
'5.0'
 '175.0' '134.0' '156.0' '243.0' '242.0' '3.0' '105.0' '147.0' '306.0'
 '184.0' '96.0' '2.0' '127.0' '95.0' '146.0' '9.0' '40.0' '177.0'
'6.0'
 '143.0' '244.0' '149.0' '167.0' '300.0' '171.0' '305.0' '196.0'
'152.0'
 '142.0' '261.0' '104.0' '36.0' '67.0' '26.0' '258.0' '110.0' '71.0'
 '181.0' '88.0' '251.0' '275.0' '248.0' '208.0' '69.0' '256.0' '314.0'
 '126.0' '281.0' '273.0' '253.0' '330.0' '334.0' '321.0' '185.0'
'326.0'
 '324.0' '313.0' '38.0' '155.0' '68.0' '335.0' '308.0' '94.0' '332.0'
 '348.0' '310.0' '339.0' '375.0' '66.0' '327.0' '387.0' '298.0' '91.0'
 '245.0' '385.0' '257.0' '393.0' '168.0' '405.0' '249.0' '315.0'
'128.0'
 '307.0' '11.0' '75.0' '436.0' '1.0' '201.0' '183.0' '223.0' '368.0'
 '336.0' '291.0' '464.0' '411.0' '481.0' '154.0' '10.0' '468.0'
'410.0'
 '390.0' '495.0' '492.0' '493.0' '434.0' '420.0' '483.0' '531.0'
'526.0'
 '472.0' '429.0' '16.0' '446.0' '34.0' '78.0' '139.0' '252.0' '270.0'
 '47.0' '114.0' '29.0' '301.0' '193.0' '182.0' '328.0' '135.0' '350.0'
 '195.0' '352.0' '355.0' '159.0' '384.0' '360.0' '331.0' '367.0'
'64.0'
 '406.0' '414.0' '333.0' '427.0' '431.0' '430.0' '438.0' '418.0'
'441.0'
 '282.0' '432.0' '72.0' '450.0' '180.0' '454.0' '455.0' '59.0' '451.0'
 '254.0' '57.0' '358.0' '469.0' '165.0' '467.0' '510.0' '337.0'
'476.0'
 '502.0' '527.0' '479.0' '508.0' '535.0' '302.0' '187.0' '13.0' '7.0'
 '27.0' '14.0' '22.0' '17.0' '28.0' '42.0' '20.0' '19.0' '45.0' '37.0'
 '61.0' '39.0' '21.0' '24.0' '41.0' '50.0' '30.0' '54.0' '52.0' '12.0'
 '44.0' '31.0' '83.0' '32.0' '63.0' '60.0' '55.0' '56.0' '89.0' '87.0'
 '118.0' '86.0' '85.0' '210.0' '214.0' '129.0' '179.0' '138.0' '174.0'
 '170.0' '153.0' '151.0' '119.0' '35.0' '173.0' '58.0' '53.0' '133.0'
 '79.0' '235.0' '192.0' '191.0' '236.0' '162.0' '215.0' '157.0'
'287.0'
 '234.0' '132.0' '98.0' '77.0' '103.0' '107.0' '262.0' '220.0' '121.0'
 '205.0' '378.0' '23.0' '296.0' '290.0' '229.0' '33.0' '286.0' '276.0'
 '425.0' '484.0' '323.0' '403.0' '219.0' '509.0' '111.0' '423.0'
'394.0'
 '4.0' '82.0' '81.0' '74.0' '92.0' '99.0' '90.0' '112.0' '117.0'
'106.0'
 '148.0' '158.0' '144.0' '211.0' '213.0' '216.0' '232.0' '150.0'
'267.0'
 '227.0' '247.0' '278.0' '280.0' '285.0' '289.0' '269.0' '295.0'
```

```
'265.0'
 '288.0' '122.0' '325.0' '341.0' '344.0' '346.0' '359.0' '283.0'
'364.0'
 '370.0' '371.0' '25.0' '141.0' '391.0' '397.0' '416.0' '299.0'
'197.0'
 '73.0' '354.0' '444.0' '408.0' '461.0' '388.0' '453.0' '459.0'
'474.0'
 '475.0' '480.0' '449.0']
--------------------------------------------------

customer_type:
['Transient' 'Contract' 'Transient-Party' 'Group']
--------------------------------------------------

reservation_status:
['Check-Out' 'Canceled' 'No-Show']
--------------------------------------------------

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

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

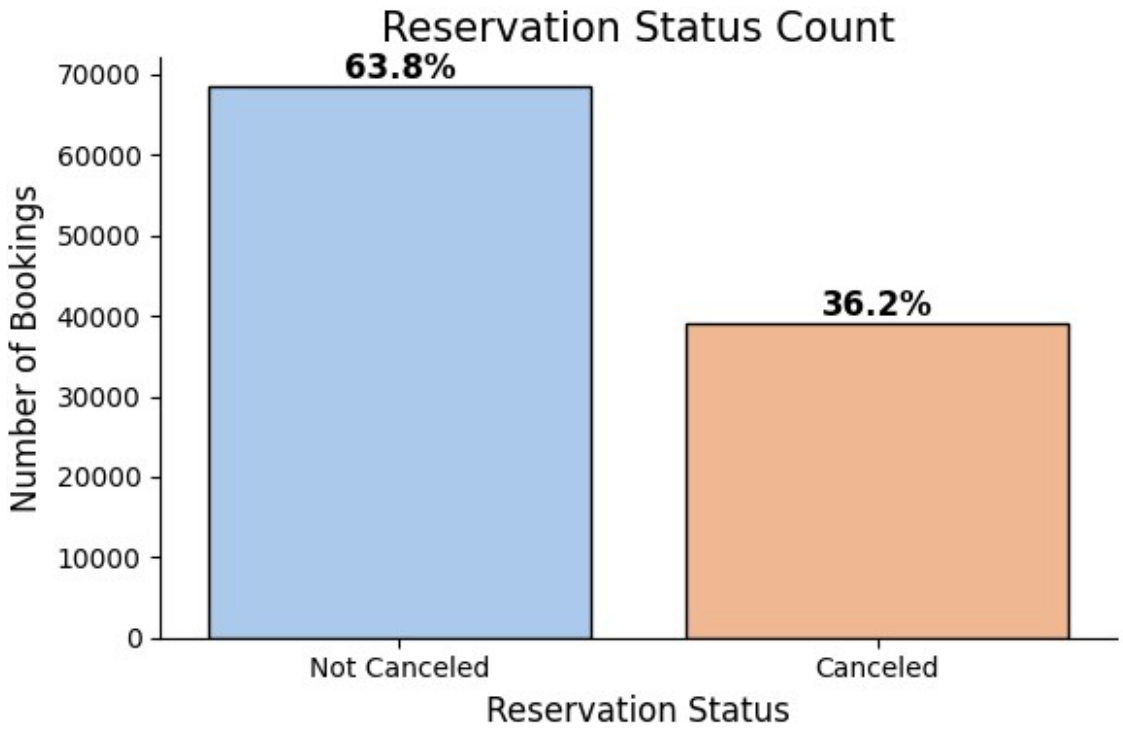
```
'2016-06-02' '2016-06-03' '2016-06-05' '2016-06-06' '2016-06-07'
'2016-06-01' '2016-06-08' '2016-06-10' '2016-06-04' '2016-06-12'
'2016-06-14' '2016-06-16' '2016-06-13' '2016-06-09' '2016-06-18'
'2016-06-17' '2016-06-20' '2016-06-22' '2016-06-23' '2016-06-21'
'2016-06-24' '2016-06-15' '2016-06-25'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

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

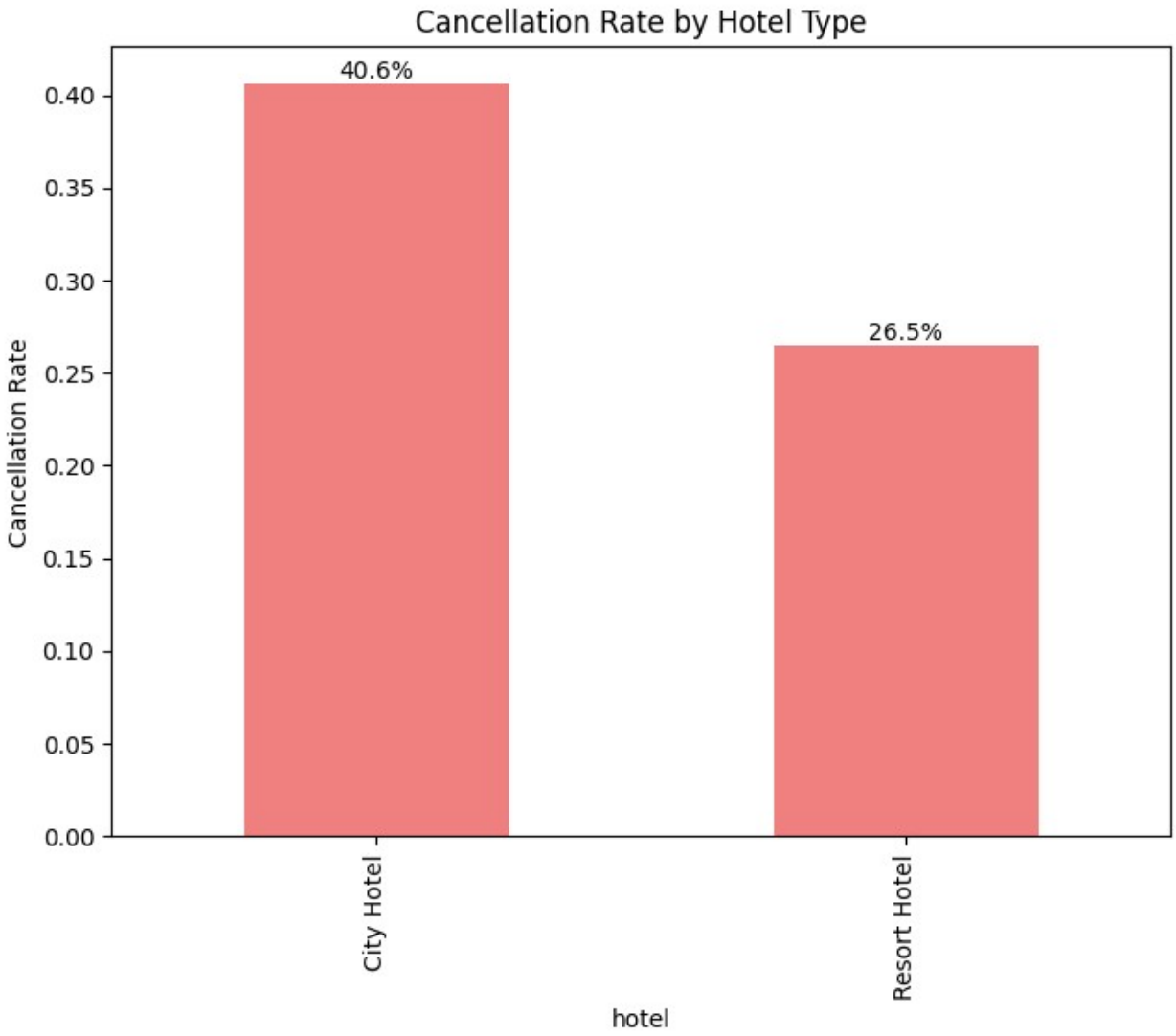
```
 '2017-08-27' '2017-09-02' '2017-08-24' '2017-06-25' '2017-07-09'
 '2017-08-12' '2017-08-26' '2017-08-30' '2017-09-03' '2017-09-04'
 '2017-09-05' '2017-09-06' '2017-09-07' '2015-04-30' '2015-04-21'
 '2015-04-05' '2015-03-13' '2015-05-05' '2015-03-29' '2015-06-10'
 '2014-10-17' '2015-01-20' '2015-02-17' '2015-03-10' '2015-03-23']
--------------------------------------------------

season_of_booking:
['Summer' 'Autumn' 'Winter' 'Spring']
--------------------------------------------------
```

```python
# Calculate the percentage of canceled and not canceled reservations
cancelled_perc = df['is_canceled'].value_counts(normalize=True) * 100
print("Reservation Status Percentage:\n", cancelled_perc)

# Enhanced: Plot the reservation status count with Seaborn for
aesthetics
plt.figure(figsize=(6, 4))
sns.set_palette('pastel')
ax = sns.countplot(
    x=df['is_canceled'].map({0: 'Not Canceled', 1: 'Canceled'}),
    order=['Not Canceled', 'Canceled'],
    edgecolor='black'
)

# Add percentage annotations on bars
for p in ax.patches:
    height = p.get_height()
    percentage = height / df.shape[0] * 100
    ax.annotate(f'{percentage:.1f}%', (p.get_x() + p.get_width() / 2.,
height),
                ha='center', va='bottom', fontsize=12, color='black',
weight='bold')

plt.title('Reservation Status Count', fontsize=15)
plt.xlabel('Reservation Status', fontsize=12)
plt.ylabel('Number of Bookings', fontsize=12)
sns.despine()
plt.tight_layout()
plt.show()
```

```
Reservation Status Percentage:
 is_canceled
0    63.767874
1    36.232126
Name: proportion, dtype: float64
```

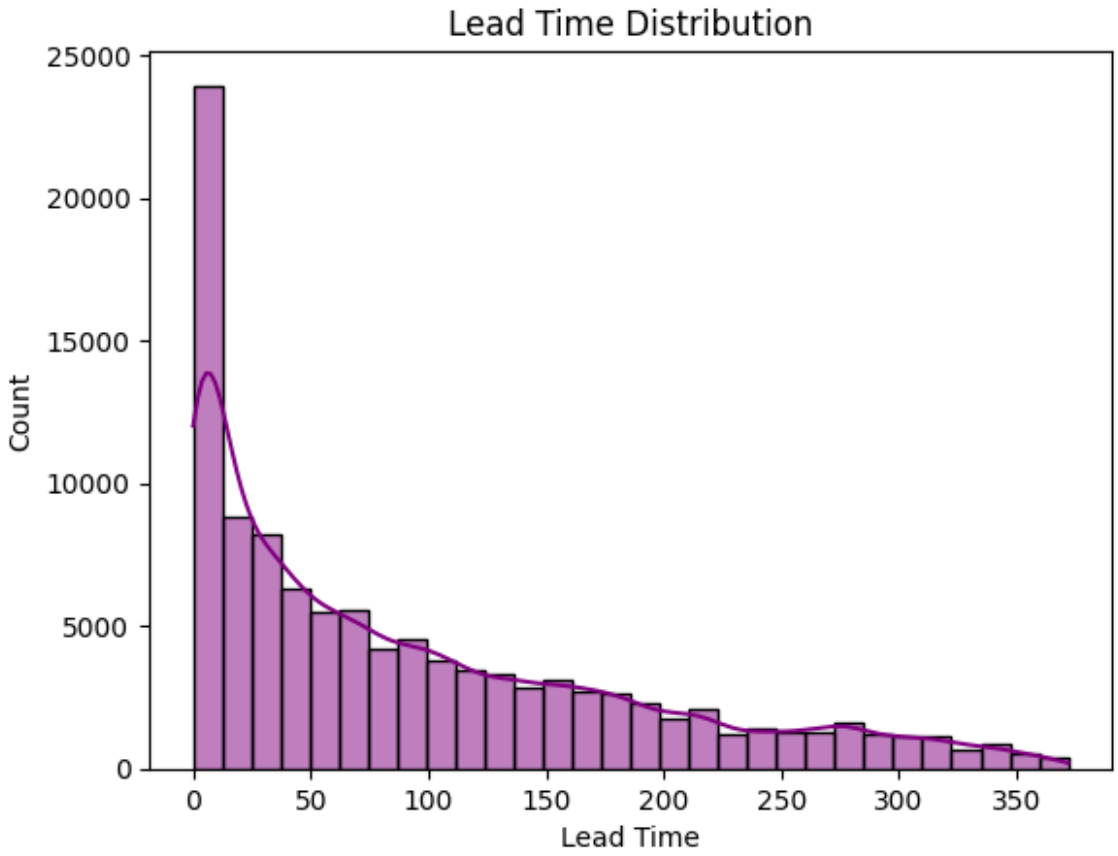


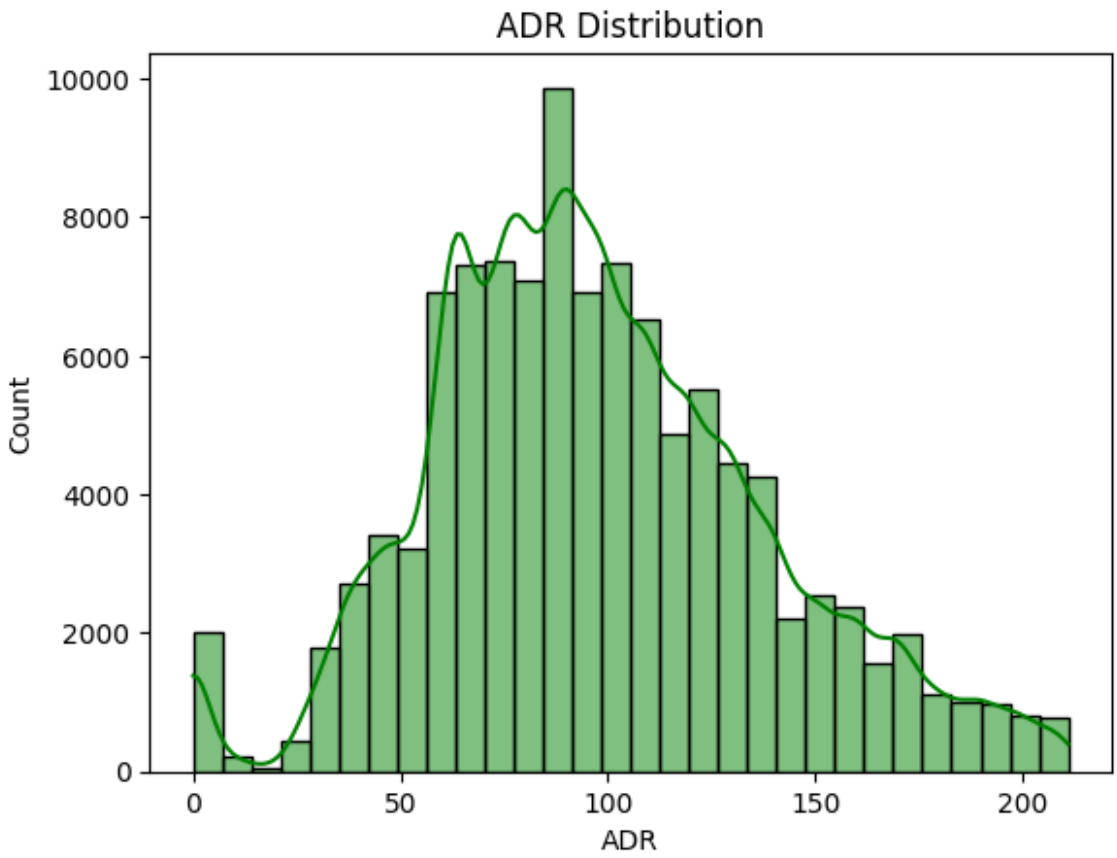

```
import matplotlib.pyplot as plt

# Your existing calculation
cancellation_rate_by_hotel = df.groupby('hotel')['is_canceled'].mean()

# Create the plot and assign it to a variable 'ax' so we can modify it
ax = cancellation_rate_by_hotel.plot(kind='bar', title='Cancellation
Rate by Hotel Type', color='lightcoral', figsize=(8, 6))

plt.ylabel('Cancellation Rate')

# Loop through each bar to add the label
for p in ax.patches:
    # Calculate the percentage value
    value = p.get_height()
    percentage = f'{value * 100:.1f}%'  # Convert to percentage string
(e.g., "27.5%")

    # Get X and Y coordinates for the text
    x = p.get_x() + p.get_width() / 2
    y = p.get_height()

    # Add the text annotation
    ax.annotate(percentage, (x, y), ha='center', va='bottom')

plt.show()
```

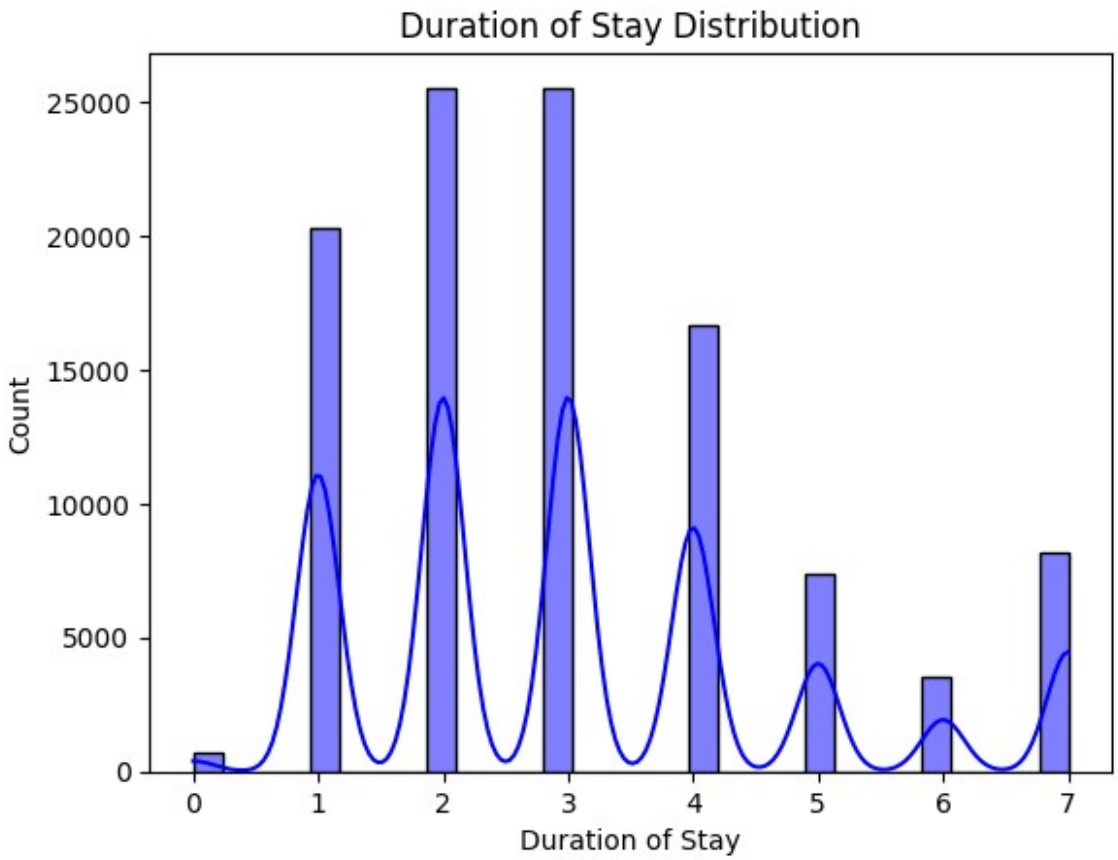



```
import warnings
# Ignore all FutureWarnings
warnings.simplefilter(action='ignore', category=FutureWarning)

# Lead time distribution
sns.histplot(df['lead_time'], kde=True, color='purple', bins=30)
plt.title('Lead Time Distribution')
plt.xlabel('Lead Time')
plt.show()

# ADR distribution
sns.histplot(df['adr'], kde=True, color='green', bins=30)
plt.title('ADR Distribution')
plt.xlabel('ADR')
plt.show()

# Duration of stay distribution
```

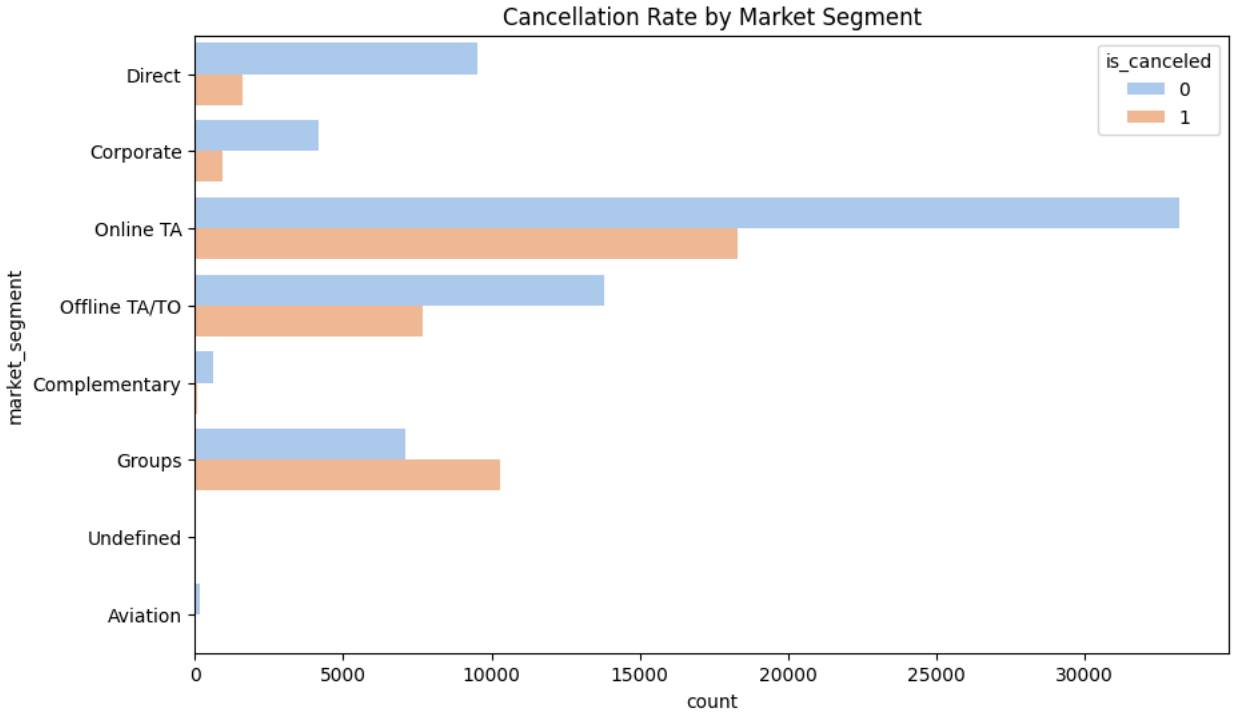
```
sns.histplot(df['duration_of_stay'], kde=True, color='blue', bins=30)
plt.title('Duration of Stay Distribution')
plt.xlabel('Duration of Stay')
plt.show()
```

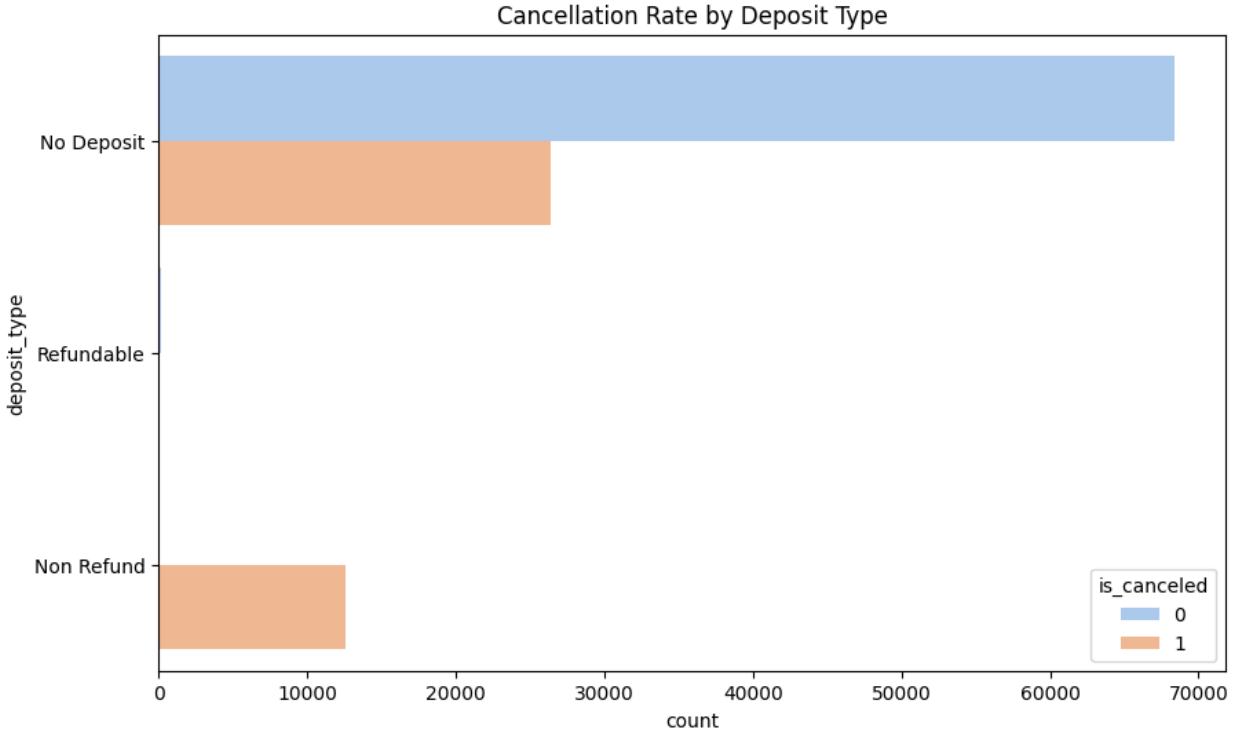


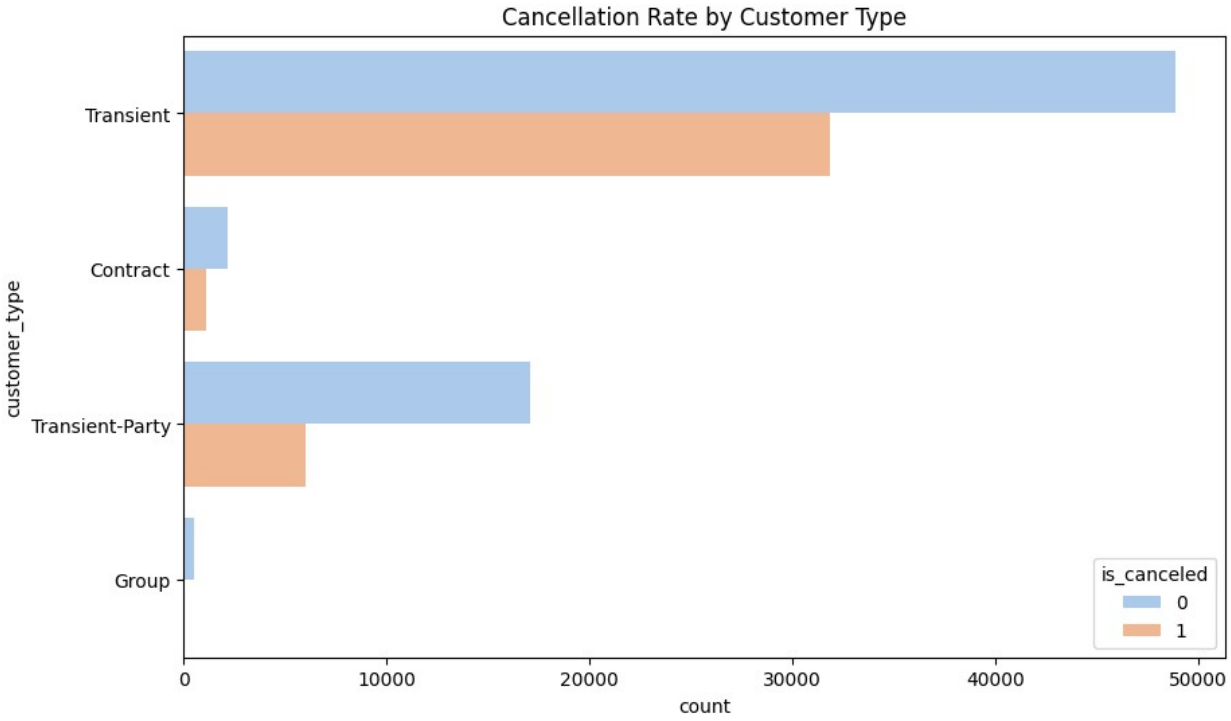

ADR Distribution
Count
ADR
0
2000
4000
6000
8000
10000
0
50
100
150
200

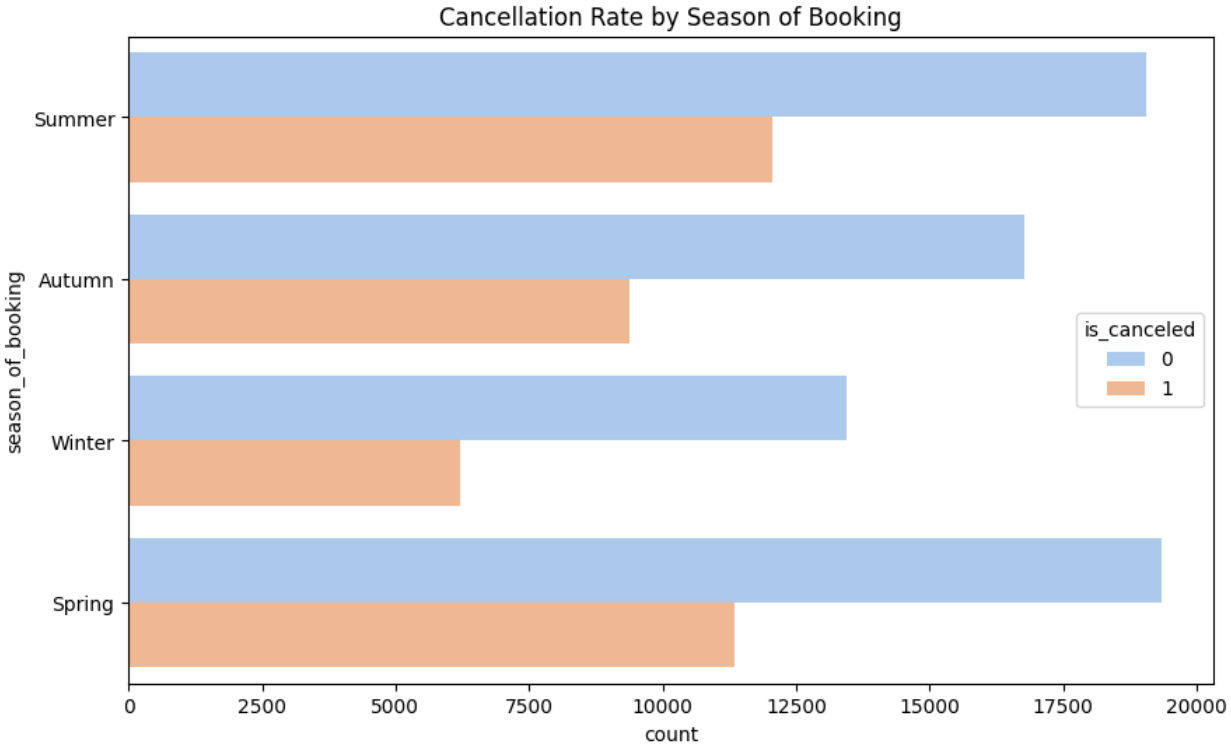



```
# Set a slightly larger figure size so the horizontal bars have room
plt.rcParams['figure.figsize'] = (10, 6)

# 1. Market segment (Horizontal)
sns.countplot(y='market_segment', hue='is_canceled', data=df)
plt.title('Cancellation Rate by Market Segment')
plt.show()

# 2. Deposit type (Horizontal)
sns.countplot(y='deposit_type', hue='is_canceled', data=df)
plt.title('Cancellation Rate by Deposit Type')
plt.show()

# 3. Customer type (Horizontal)
sns.countplot(y='customer_type', hue='is_canceled', data=df)
plt.title('Cancellation Rate by Customer Type')
plt.show()

# 4. Season of booking (Horizontal)
sns.countplot(y='season_of_booking', hue='is_canceled', data=df)
plt.title('Cancellation Rate by Season of Booking')
plt.show()
```


Cancellation Rate by Market Segment
is_canceled
0
1
Direct
Corporate
Online TA
Offline TA/TO
Complementary
Groups
Undefined
Aviation
market_segment
0
5000
10000
15000
20000
25000
30000
count

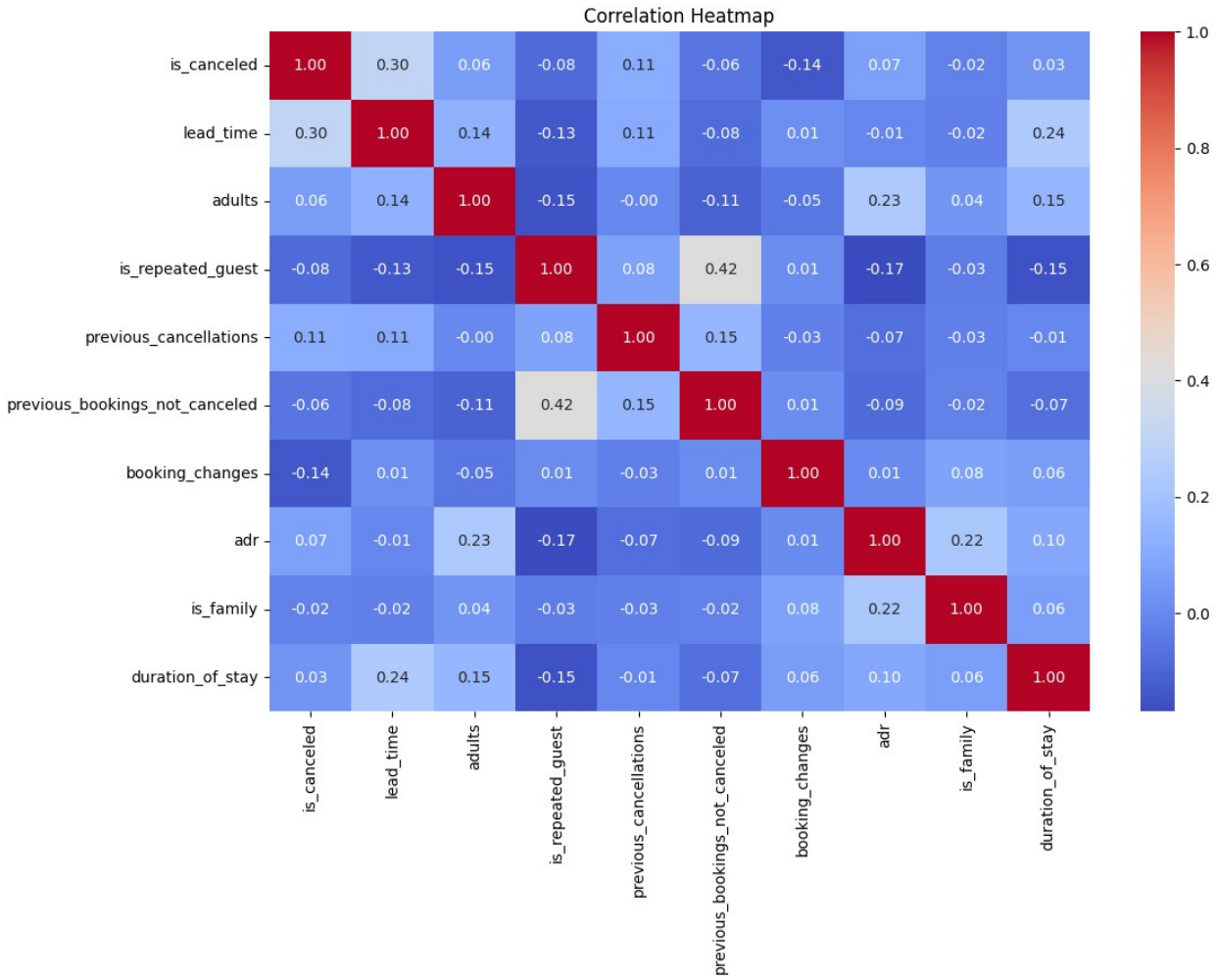

Cancellation Rate by Deposit Type
No Deposit
Refundable
Non Refund
deposit_type
is_canceled
0
1
0
10000
20000
30000
40000
50000
60000
70000
count



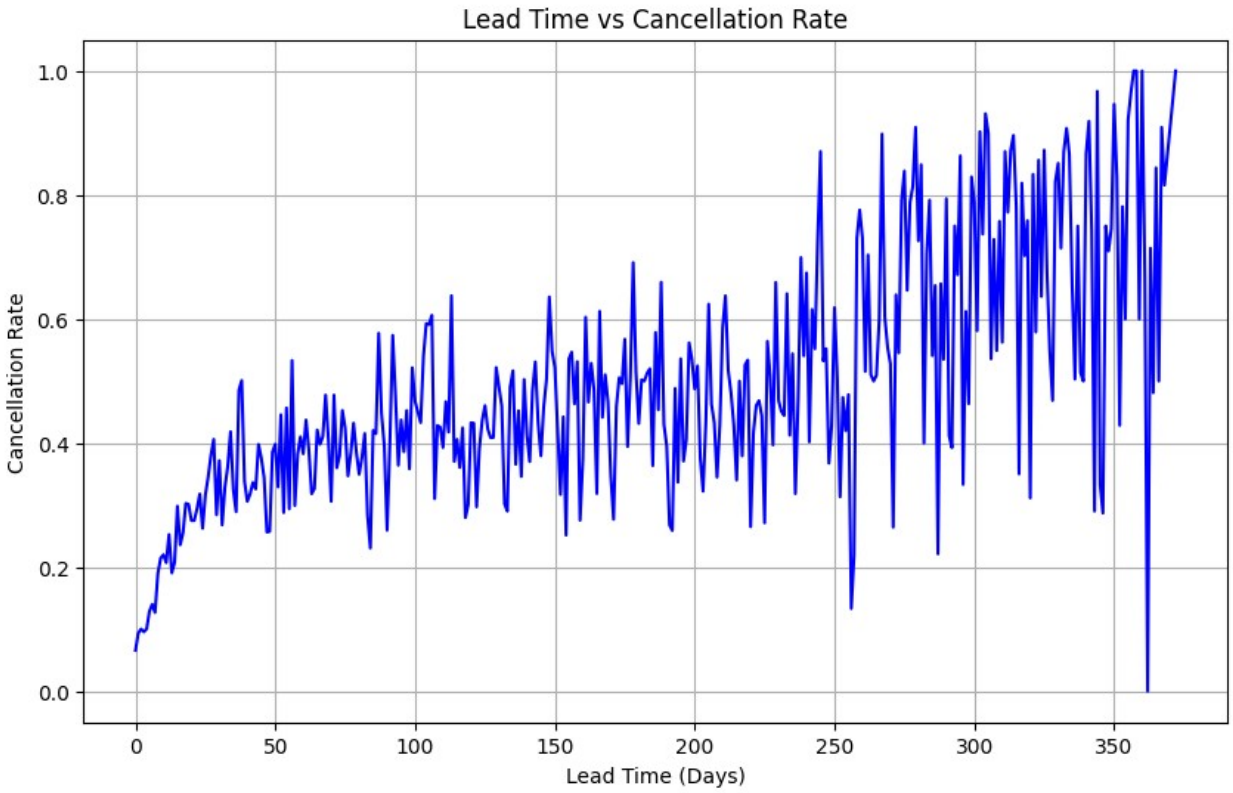



```
# Correlation heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(df.corr(numeric_only=True), annot=True, cmap='coolwarm',
fmt='.2f')
```

```
plt.title('Correlation Heatmap')
plt.show()
```

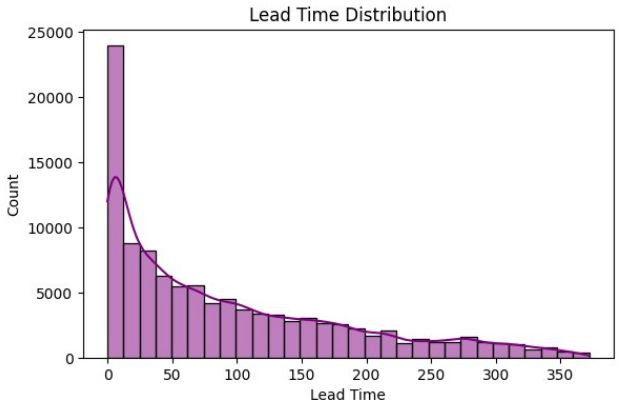


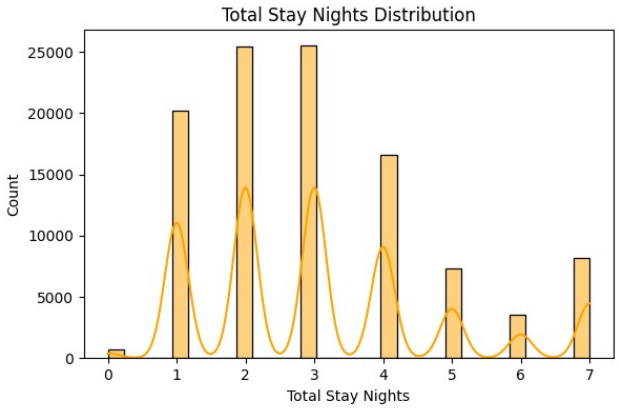

```
# Group by lead_time and calculate the cancellation rate for each
lead_time
lead_time_cancellation = df.groupby('lead_time')['is_canceled'].mean()

# Plot the line chart
plt.figure(figsize=(10, 6))
sns.lineplot(x=lead_time_cancellation.index,
y=lead_time_cancellation.values, color='b')
plt.title('Lead Time vs Cancellation Rate')
plt.xlabel('Lead Time (Days)')
plt.ylabel('Cancellation Rate')
plt.grid(True)
plt.show()
```

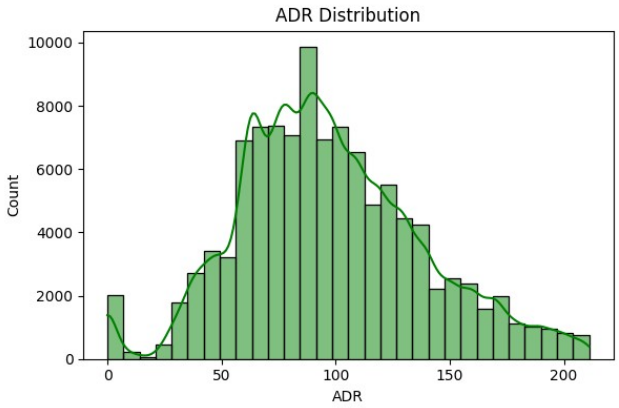


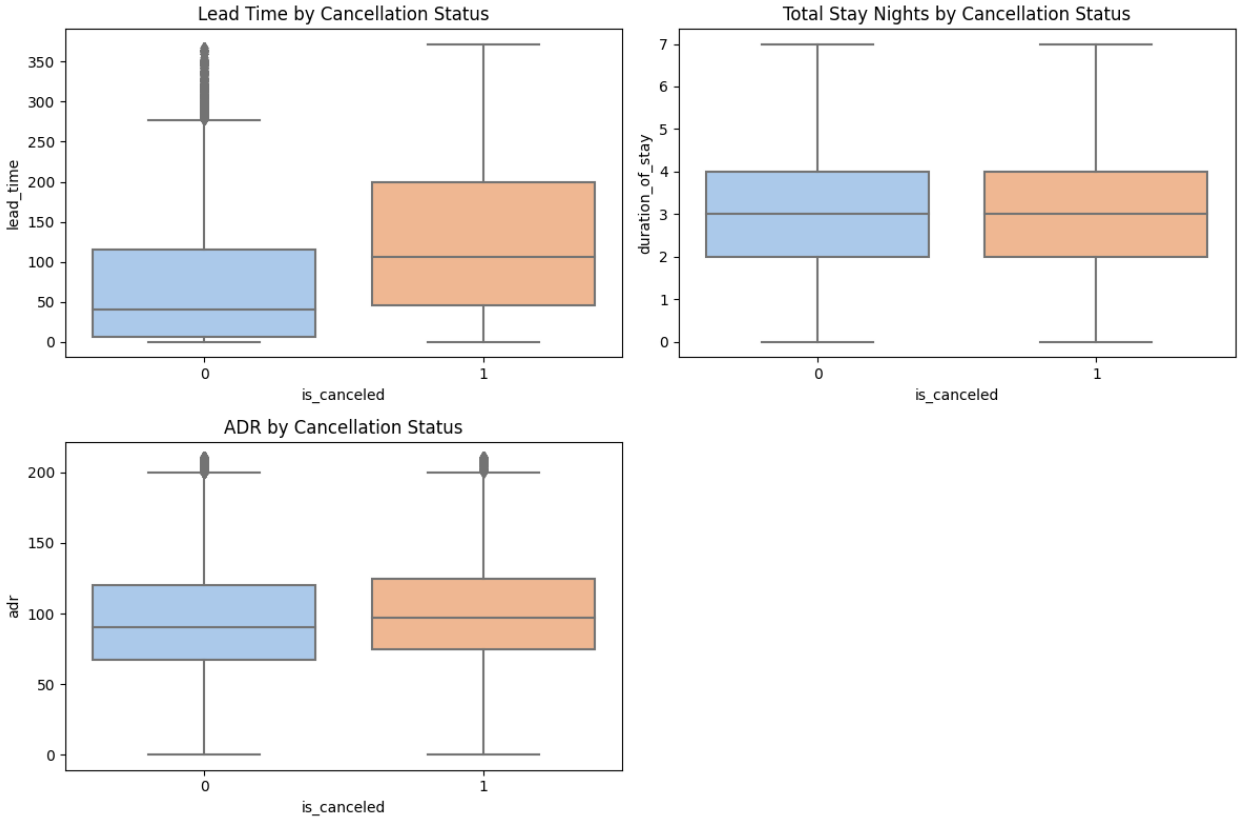

```
# Plot Histograms for numeric variables
plt.figure(figsize=(12, 8))

# Lead time histogram
plt.subplot(2, 2, 1)
sns.histplot(df['lead_time'], kde=True, color='purple', bins=30)
plt.title('Lead Time Distribution')
plt.xlabel('Lead Time')

# Total stay nights histogram
plt.subplot(2, 2, 2)
sns.histplot(df['duration_of_stay'], kde=True, color='orange',
bins=30)
plt.title('Total Stay Nights Distribution')
plt.xlabel('Total Stay Nights')

# ADR histogram
plt.subplot(2, 2, 3)
sns.histplot(df['adr'], kde=True, color='green', bins=30)
plt.title('ADR Distribution')
plt.xlabel('ADR')


plt.tight_layout()
plt.show()
```

```
# Plot Boxplots for numeric variables by is_canceled
plt.figure(figsize=(12, 8))

# Lead time boxplot
plt.subplot(2, 2, 1)
sns.boxplot(x='is_canceled', y='lead_time', data=df)
plt.title('Lead Time by Cancellation Status')

# Total stay nights boxplot
plt.subplot(2, 2, 2)
sns.boxplot(x='is_canceled', y='duration_of_stay', data=df)
plt.title('Total Stay Nights by Cancellation Status')

# ADR boxplot
plt.subplot(2, 2, 3)
sns.boxplot(x='is_canceled', y='adr', data=df)
plt.title('ADR by Cancellation Status')

plt.tight_layout()
plt.show()
```



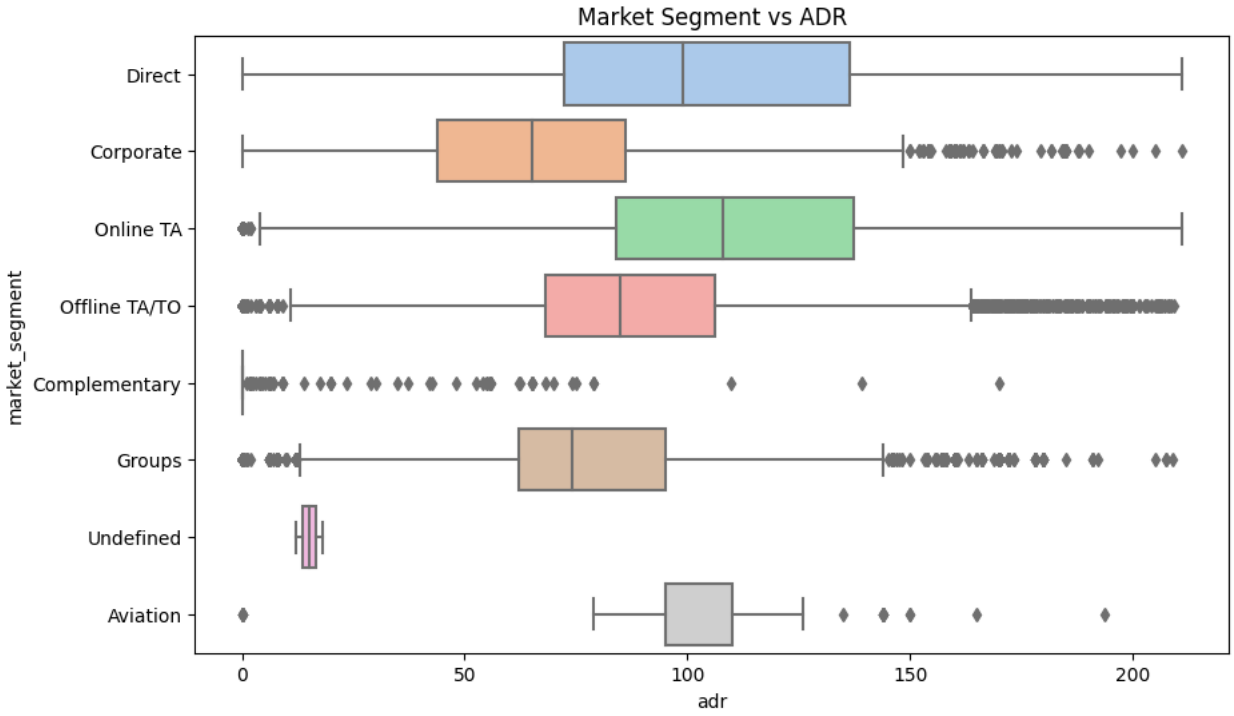


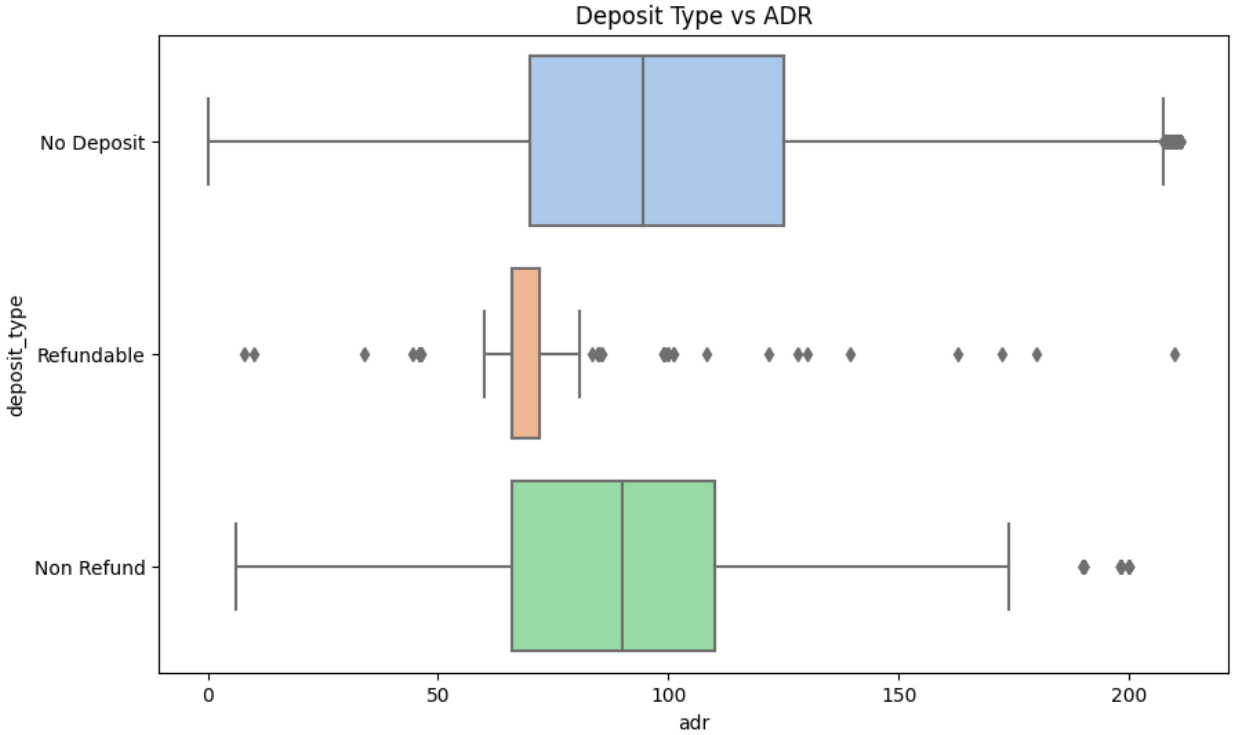


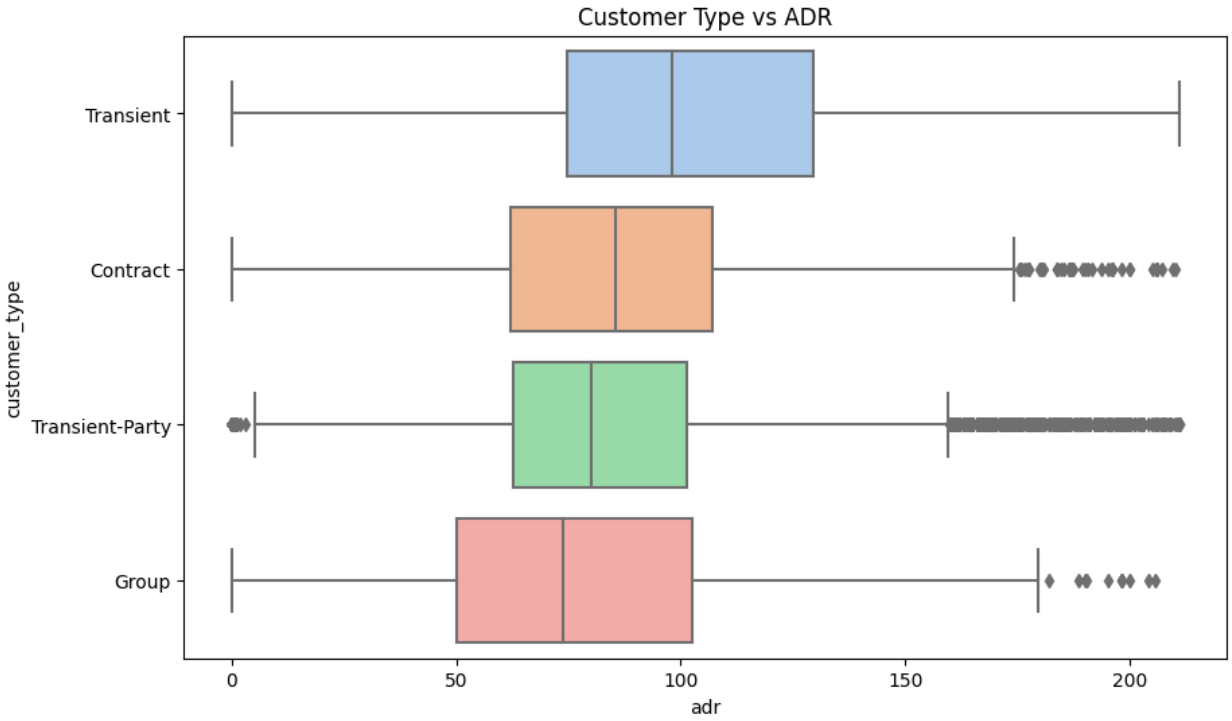


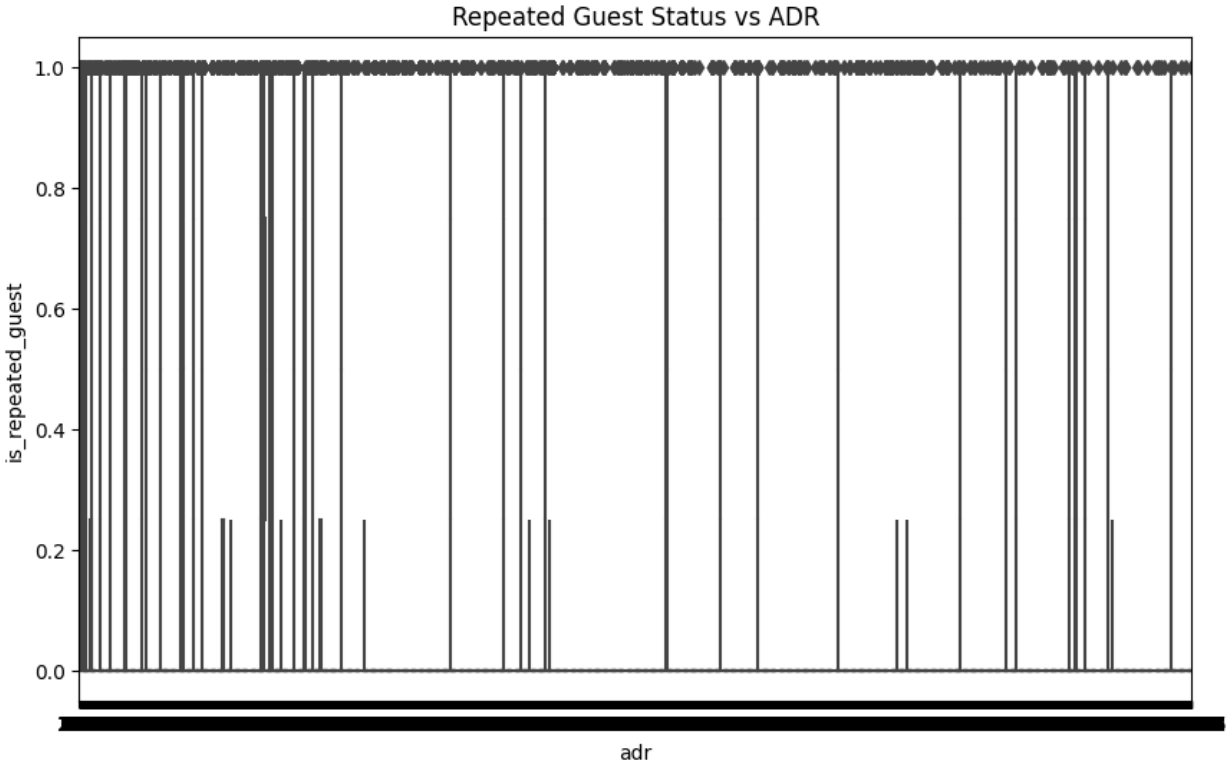

```
import matplotlib.pyplot as plt
import seaborn as sns

# Set a consistent figure size
plt.rcParams['figure.figsize'] = (10, 6)

# 1. Market segment vs ADR (Horizontal)
# Note: x is now ADR (numeric), y is Market Segment (categorical)
sns.boxplot(x='adr', y='market_segment', data=df)
plt.title('Market Segment vs ADR')
plt.show()

# 2. Deposit type vs ADR (Horizontal)
sns.boxplot(x='adr', y='deposit_type', data=df)
plt.title('Deposit Type vs ADR')
plt.show()

# 3. Customer type vs ADR (Horizontal)
sns.boxplot(x='adr', y='customer_type', data=df)
plt.title('Customer Type vs ADR')
plt.show()


# 5. Is repeated guest vs ADR (Horizontal)
sns.boxplot(x='adr', y='is_repeated_guest', data=df)
```

```
plt.title('Repeated Guest Status vs ADR')
plt.show()
```







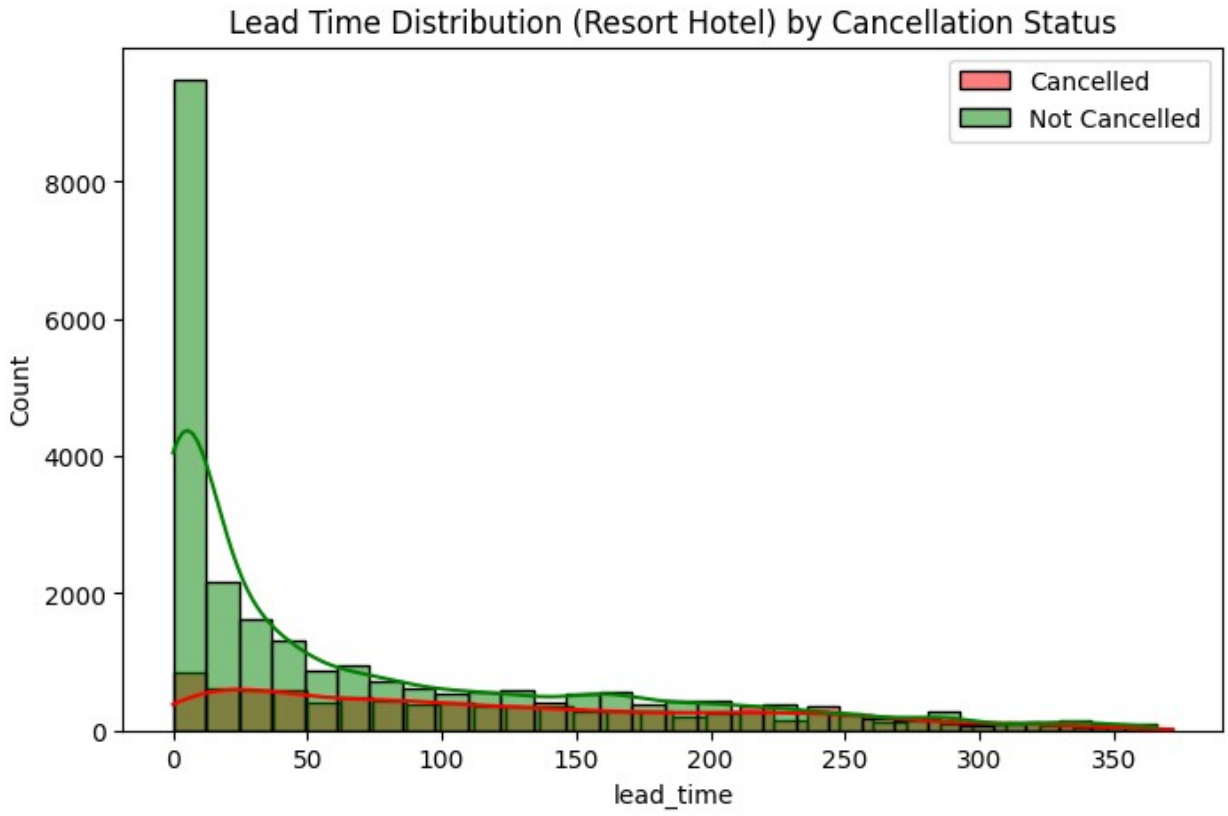



```
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# Resort Hotel Subset
resort_hotel = df[df['hotel'] == 'Resort Hotel']

# 1. Cancellation rate for Resort Hotel
resort_cancellation_rate =
resort_hotel['is_canceled'].value_counts(normalize=True) * 100
print("Resort Hotel Cancellation Rate:")
print(resort_cancellation_rate)

# 2. Average Lead Time and its distribution
resort_avg_lead_time = resort_hotel.groupby('is_canceled')
['lead_time'].mean()
print("\nResort Hotel Average Lead Time (by cancellation status):")
print(resort_avg_lead_time)

# Plotting distribution of lead_time
plt.figure(figsize=(8, 5))
sns.histplot(resort_hotel[resort_hotel['is_canceled'] == 1]
['lead_time'], kde=True, color='red', label='Cancelled', bins=30)
sns.histplot(resort_hotel[resort_hotel['is_canceled'] == 0]
['lead_time'], kde=True, color='green', label='Not Cancelled',
bins=30)
plt.title('Lead Time Distribution (Resort Hotel) by Cancellation
Status')
plt.legend()
plt.show()

# 3. Average ADR (canceled vs not canceled)
resort_avg_adr = resort_hotel.groupby('is_canceled')['adr'].mean()
print("\nResort Hotel Average ADR (by cancellation status):")
print(resort_avg_adr)

plt.figure(figsize=(8, 5))
sns.boxplot(x='is_canceled', y='adr', data=resort_hotel)
plt.title('ADR Distribution (Resort Hotel) by Cancellation Status')
plt.show()

# 4. Cancellation Rate by Key Categorical Variables

# --- SPECIAL CASE: Market Segment (Horizontal / Swapped Axes) ---
plt.figure(figsize=(10, 6))
sns.countplot(y='market_segment', hue='is_canceled',
data=resort_hotel)
plt.title('Cancellation Rate by Market Segment (Resort Hotel)')
plt.show()

# --- OTHER VARIABLES (Vertical / Standard) ---
categorical_vars = ['distribution_channel', 'deposit_type',
'is_repeated_guest']
```

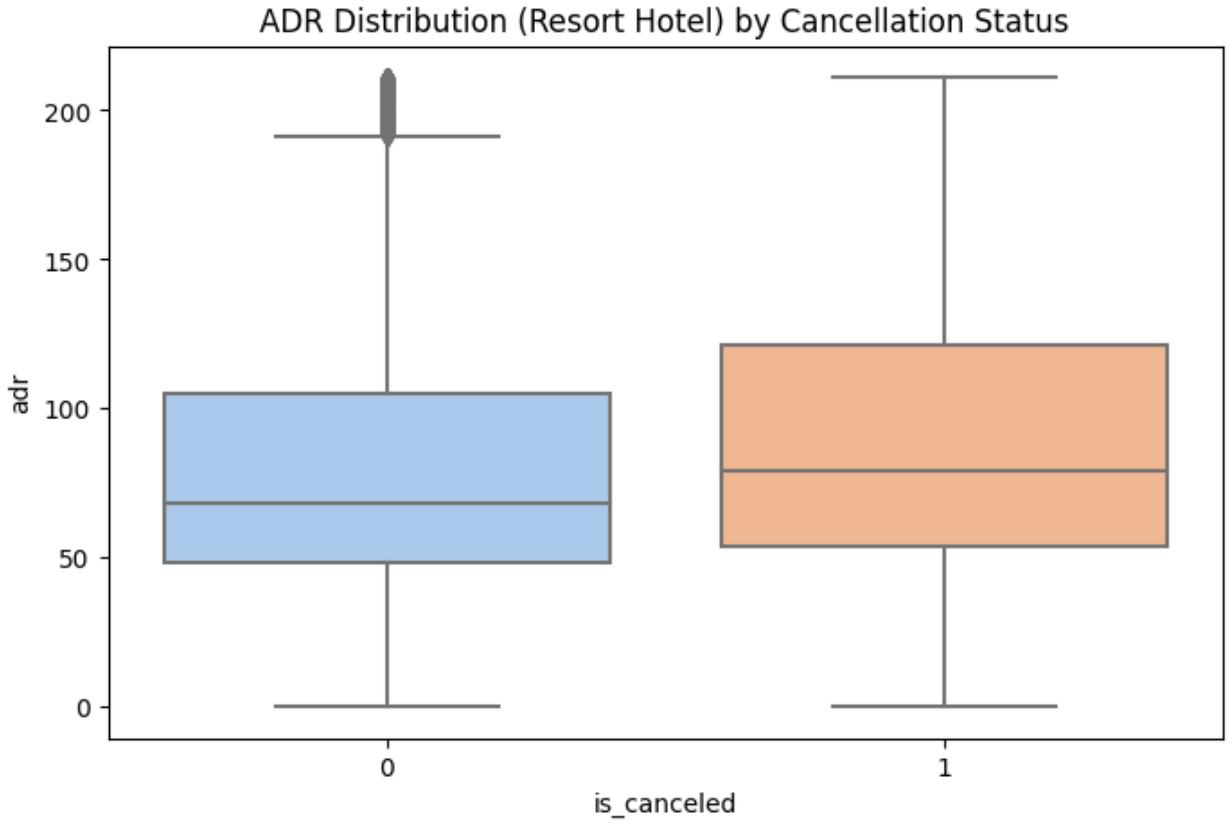
```
for var in categorical_vars:
    plt.figure(figsize=(8, 5))
    sns.countplot(x=var, hue='is_canceled', data=resort_hotel)
    plt.title(f'Cancellation Rate by {var} (Resort Hotel)')
    plt.xlabel(var)
    plt.ylabel('Count')
    plt.show()
```

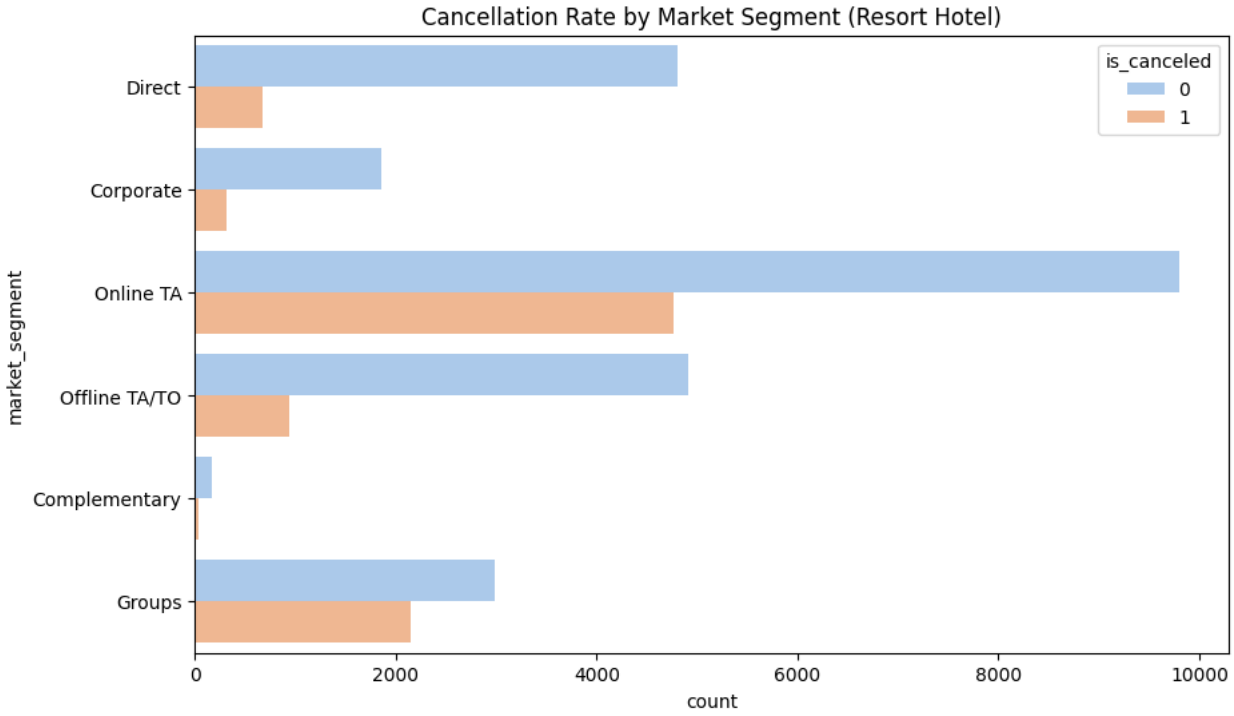
```
Resort Hotel Cancellation Rate:
is_canceled
0    73.458084
1    26.541916
Name: proportion, dtype: float64

Resort Hotel Average Lead Time (by cancellation status):
is_canceled
0     69.050459
1    120.381162
Name: lead_time, dtype: float64
```

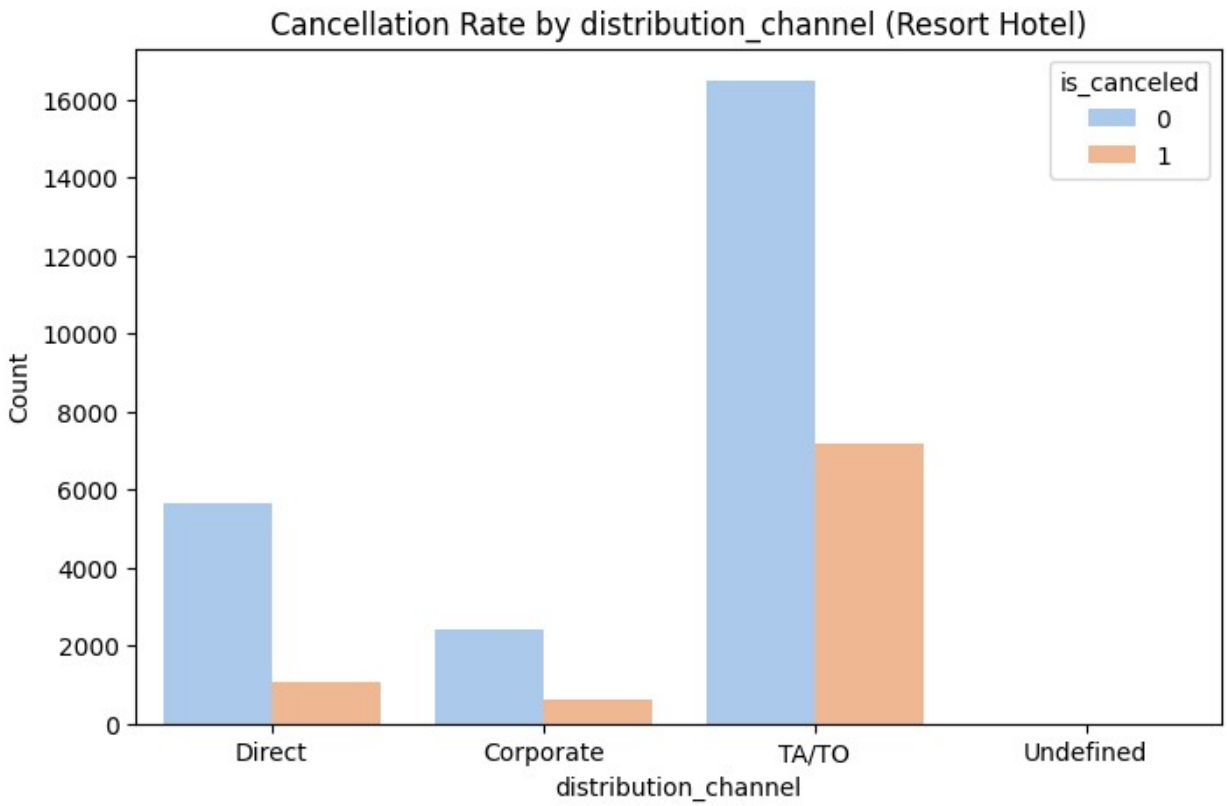


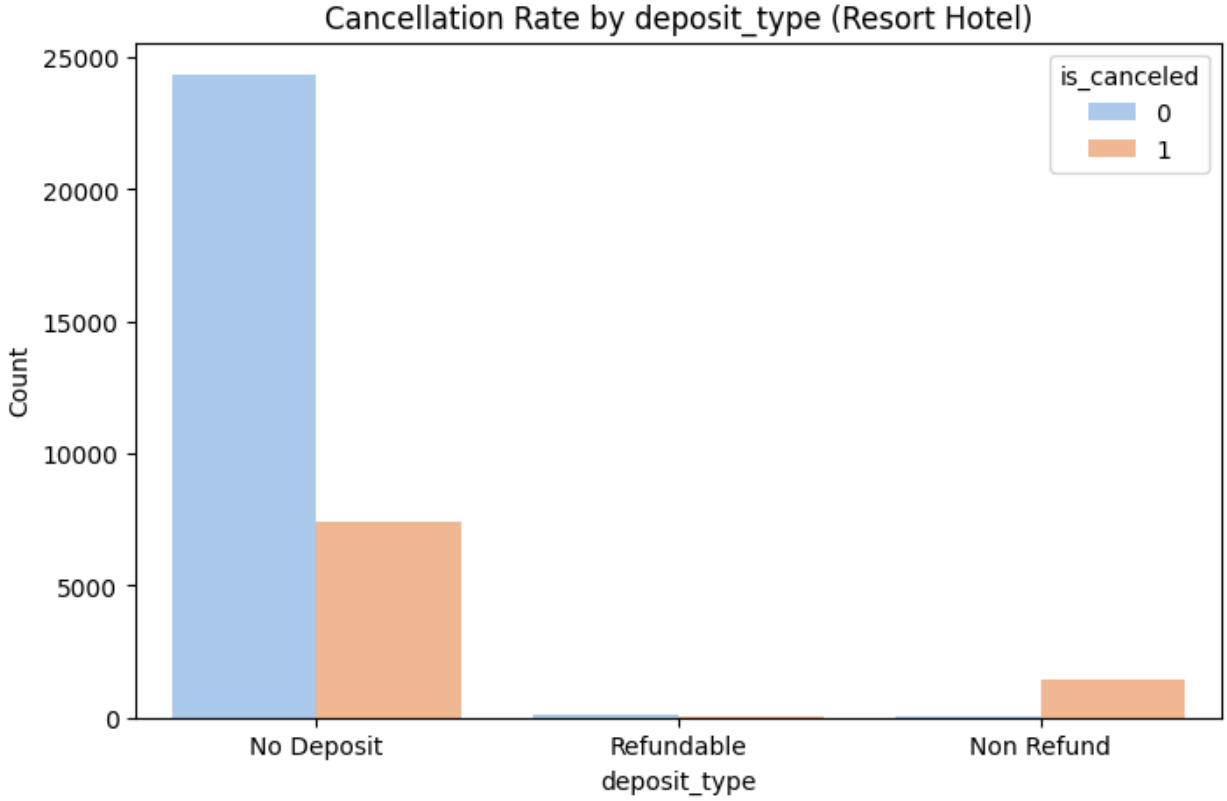

```
Resort Hotel Average ADR (by cancellation status):
is_canceled
0    81.038483
```

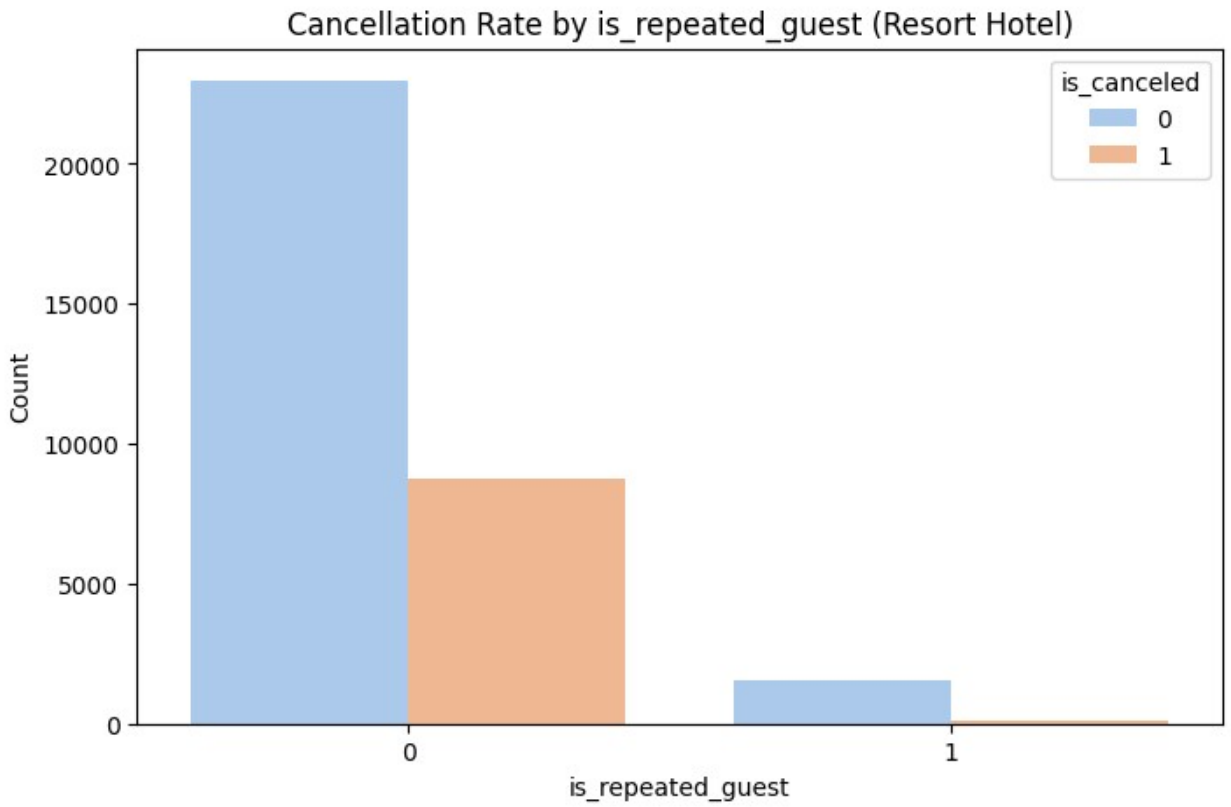
```
1    91.262109
Name: adr, dtype: float64
```






Cancellation Rate by distribution_channel (Resort Hotel)
is_canceled
0
1
Count
16000
14000
12000
10000
8000
6000
4000
2000
0
Direct
Corporate
TA/TO
Undefined
distribution_channel


Cancellation Rate by deposit_type (Resort Hotel)
is_canceled
0
1
Count
25000
20000
15000
10000
5000
0
No Deposit
Refundable
Non Refund
deposit_type

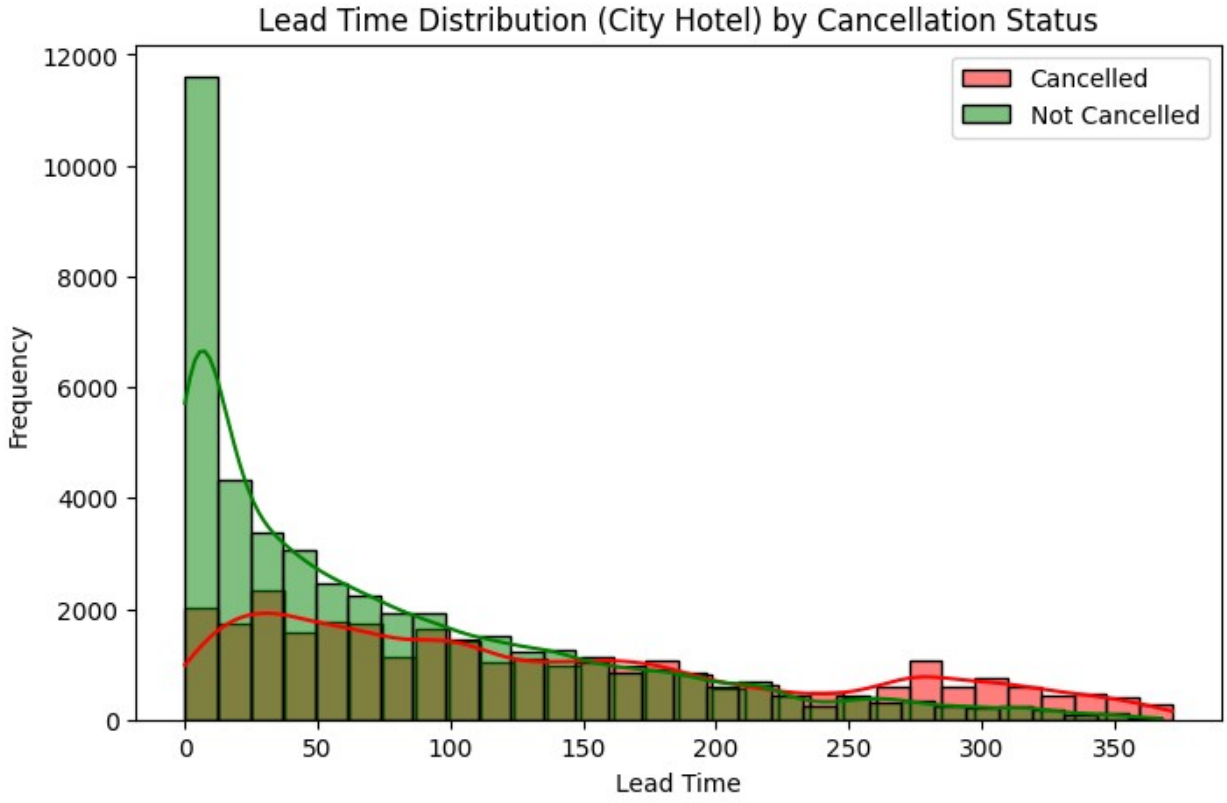



```
import matplotlib.pyplot as plt
import seaborn as sns

# City Hotel Subset
city_hotel = df[df['hotel'] == 'City Hotel']

# 1. Cancellation rate for City Hotel
city_cancellation_rate =
city_hotel['is_canceled'].value_counts(normalize=True) * 100
print("City Hotel Cancellation Rate:")
print(city_cancellation_rate)

# 2. Average Lead Time and its distribution (canceled vs not canceled)
city_avg_lead_time = city_hotel.groupby('is_canceled')
['lead_time'].mean()
print("\nCity Hotel Average Lead Time (by cancellation status):")
print(city_avg_lead_time)

# Plotting distribution of lead_time for City Hotel
plt.figure(figsize=(8, 5))
sns.histplot(city_hotel[city_hotel['is_canceled'] == 1]['lead_time'],
kde=True, color='red', label='Cancelled', bins=30)
sns.histplot(city_hotel[city_hotel['is_canceled'] == 0]['lead_time'],
kde=True, color='green', label='Not Cancelled', bins=30)
plt.title('Lead Time Distribution (City Hotel) by Cancellation
```

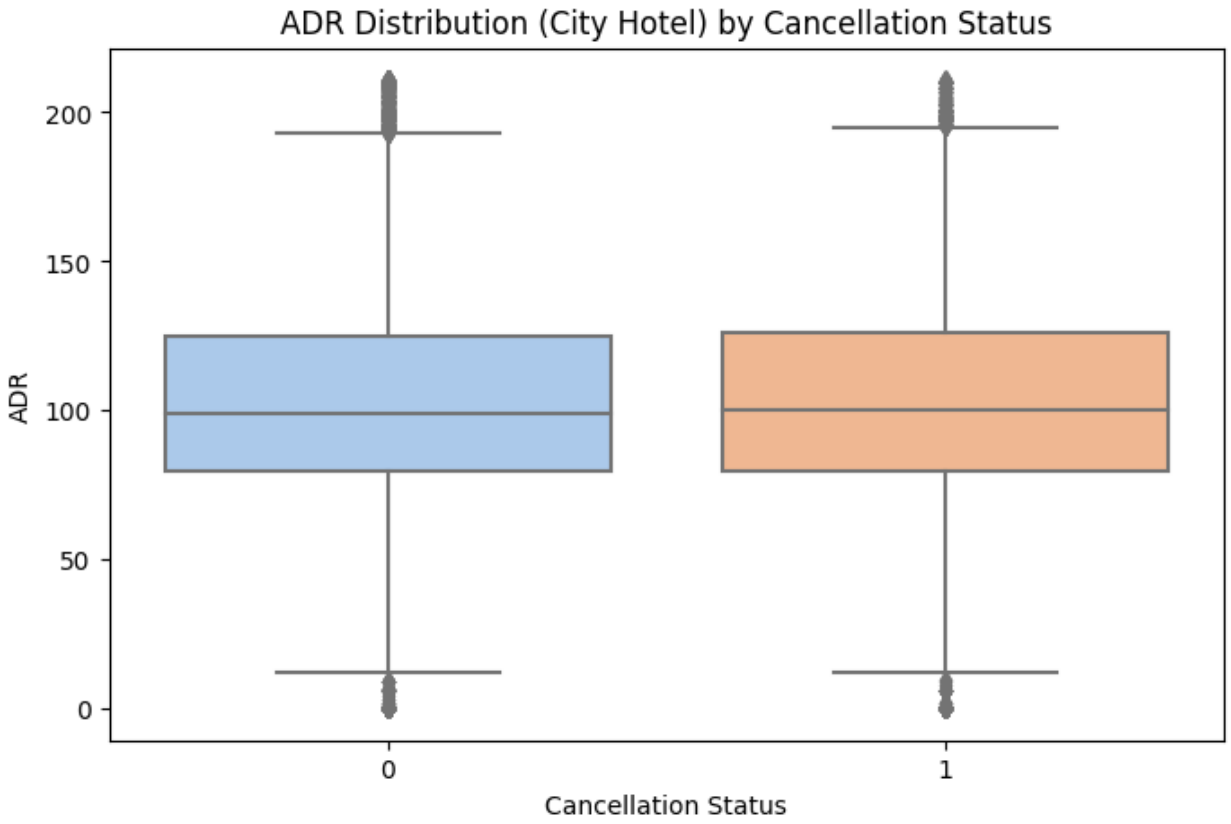
```
Status')
plt.xlabel('Lead Time')
plt.ylabel('Frequency')
plt.legend()
plt.show()

# 3. Average ADR (canceled vs not canceled)
city_avg_adr = city_hotel.groupby('is_canceled')['adr'].mean()
print("\nCity Hotel Average ADR (by cancellation status):")
print(city_avg_adr)

# Plotting ADR for City Hotel
plt.figure(figsize=(8, 5))
sns.boxplot(x='is_canceled', y='adr', data=city_hotel)
plt.title('ADR Distribution (City Hotel) by Cancellation Status')
plt.xlabel('Cancellation Status')
plt.ylabel('ADR')
plt.show()

# 4. Cancellation Rate by Key Categorical Variables for City Hotel

# --- SPECIAL CASE: Market Segment (Horizontal / Swapped Axes) ---
plt.figure(figsize=(10, 6))
sns.countplot(y='market_segment', hue='is_canceled', data=city_hotel)
plt.title('Cancellation Rate by Market Segment (City Hotel)')
plt.show()

# --- OTHER VARIABLES (Vertical / Standard) ---
# We define the list here to ensure it excludes market_segment
categorical_vars = ['distribution_channel', 'deposit_type',
'is_repeated_guest']

for var in categorical_vars:
    plt.figure(figsize=(8, 5))
    sns.countplot(x=var, hue='is_canceled', data=city_hotel)
    plt.title(f'Cancellation Rate by {var} (City Hotel)')
    plt.xlabel(var)
    plt.ylabel('Count')
    plt.show()
```

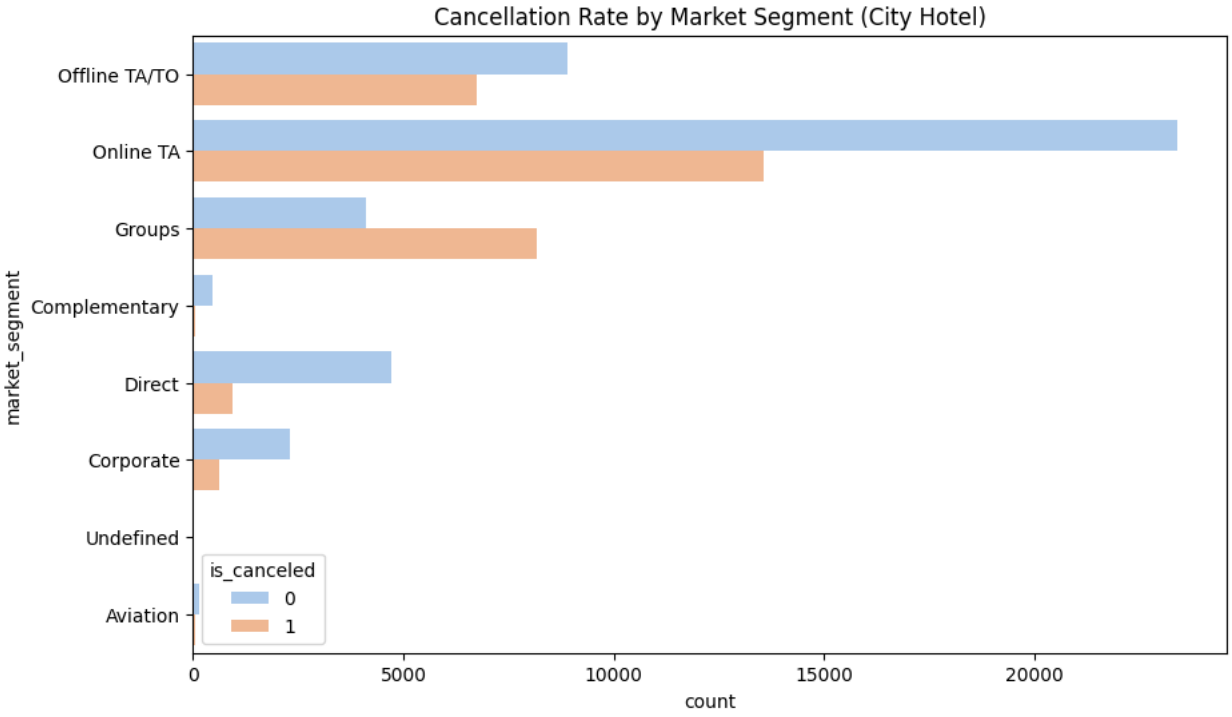
```
City Hotel Cancellation Rate:
is_canceled
0    59.407795
1    40.592205
Name: proportion, dtype: float64

City Hotel Average Lead Time (by cancellation status):
is_canceled
0     75.009229
```

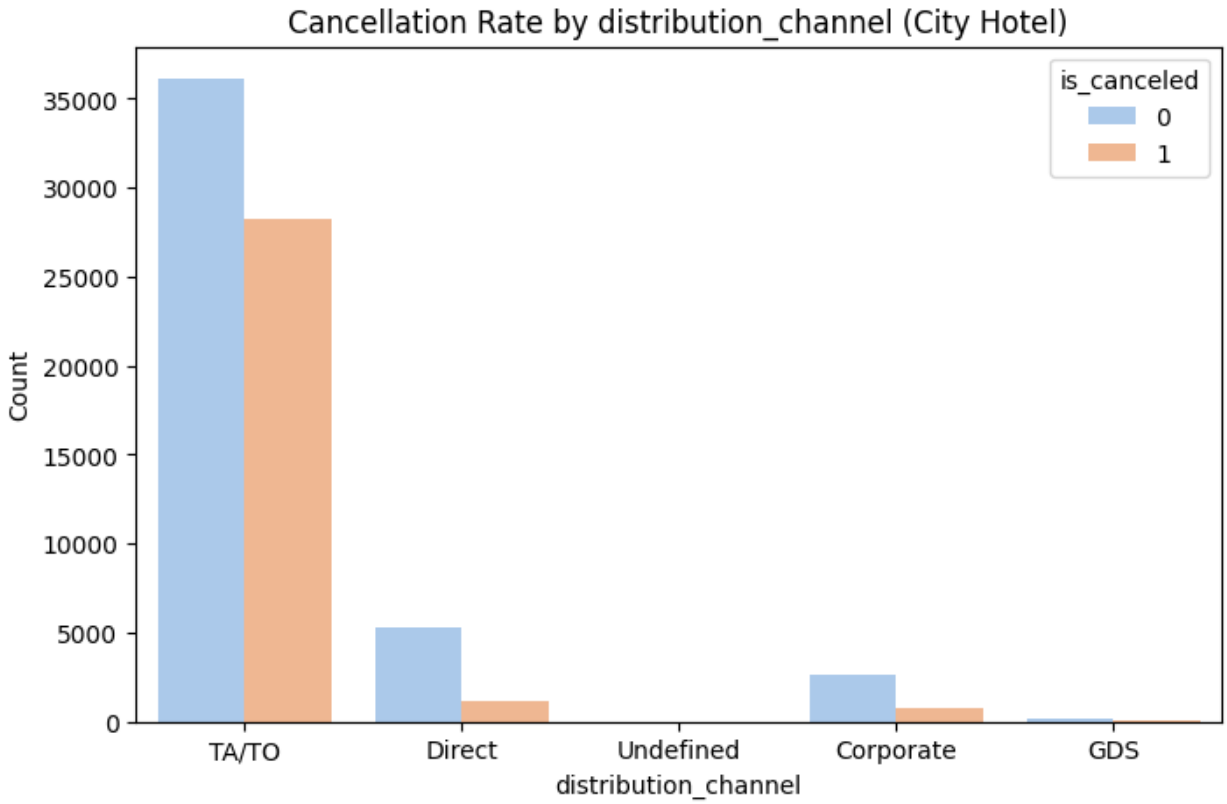
```
1    133.991836
Name: lead_time, dtype: float64
```

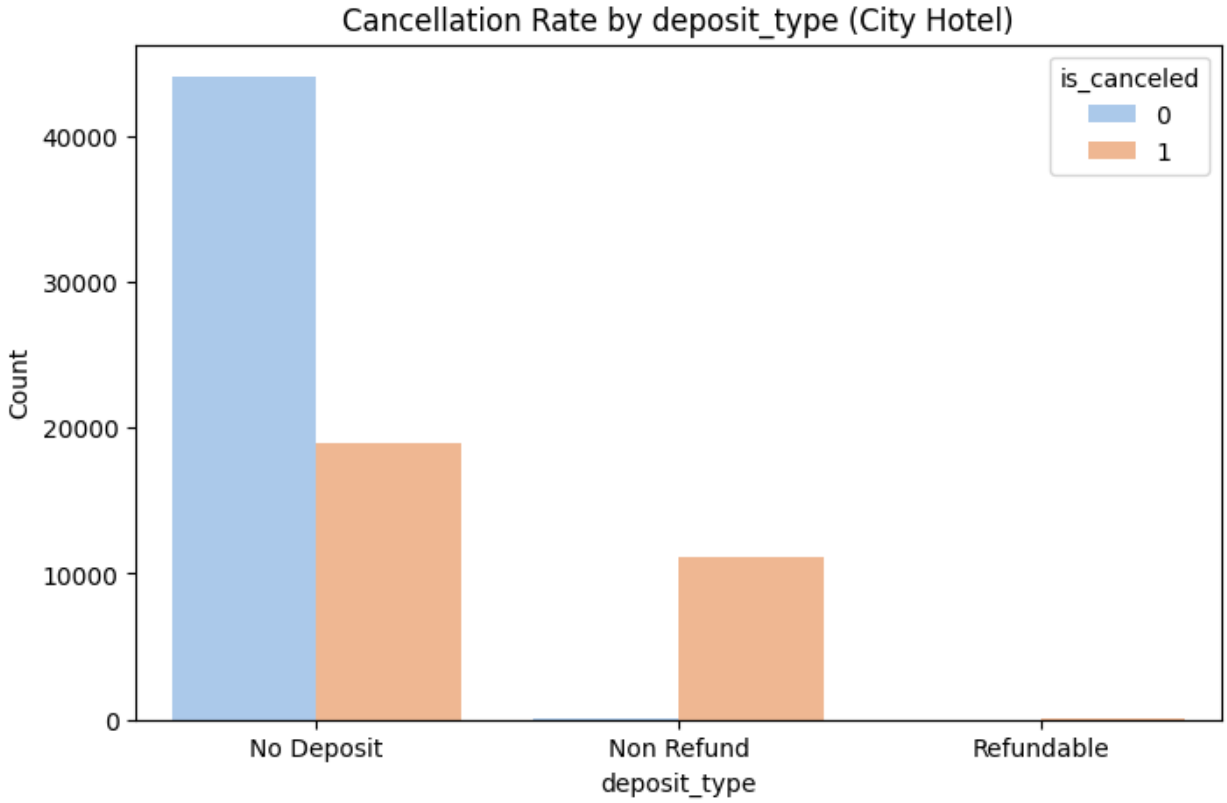


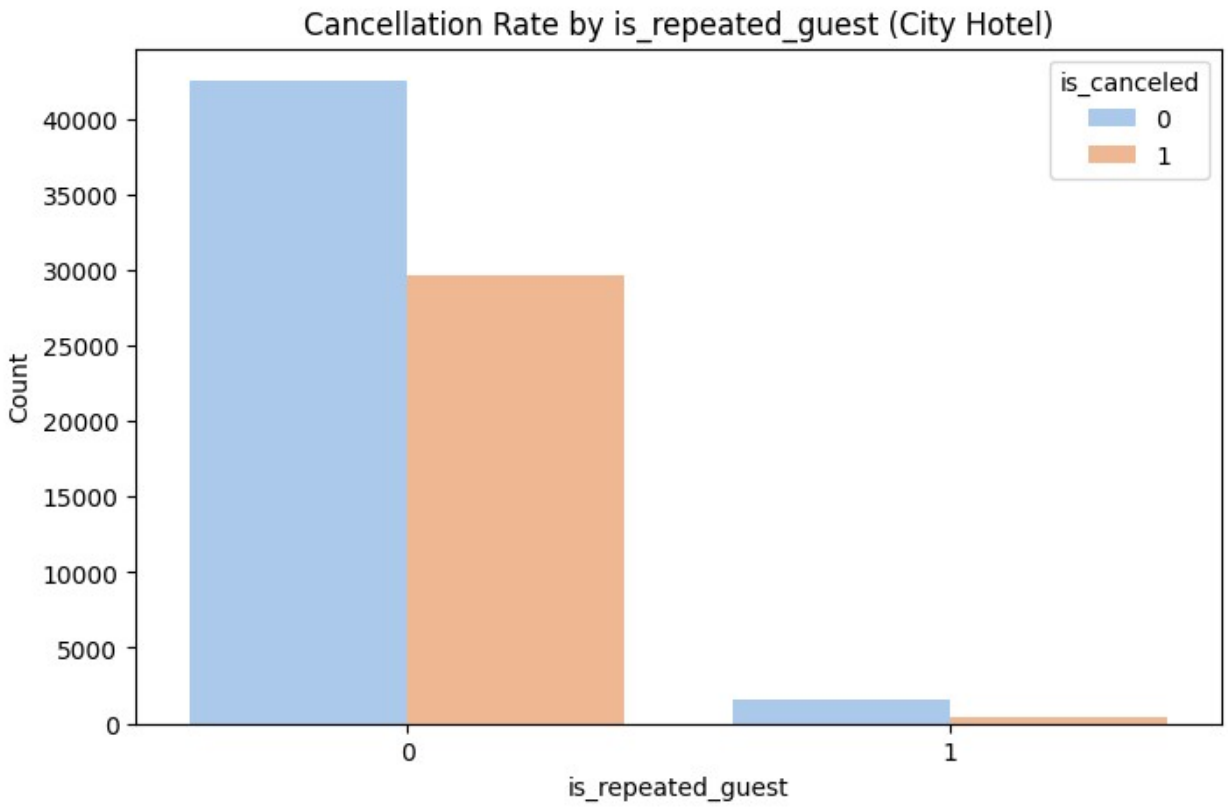

```
City Hotel Average ADR (by cancellation status):
is_canceled
0    103.292299
1    104.342688
Name: adr, dtype: float64
```


ADR Distribution (City Hotel) by Cancellation Status
ADR
200
150
100
50
0
0
1
Cancellation Status

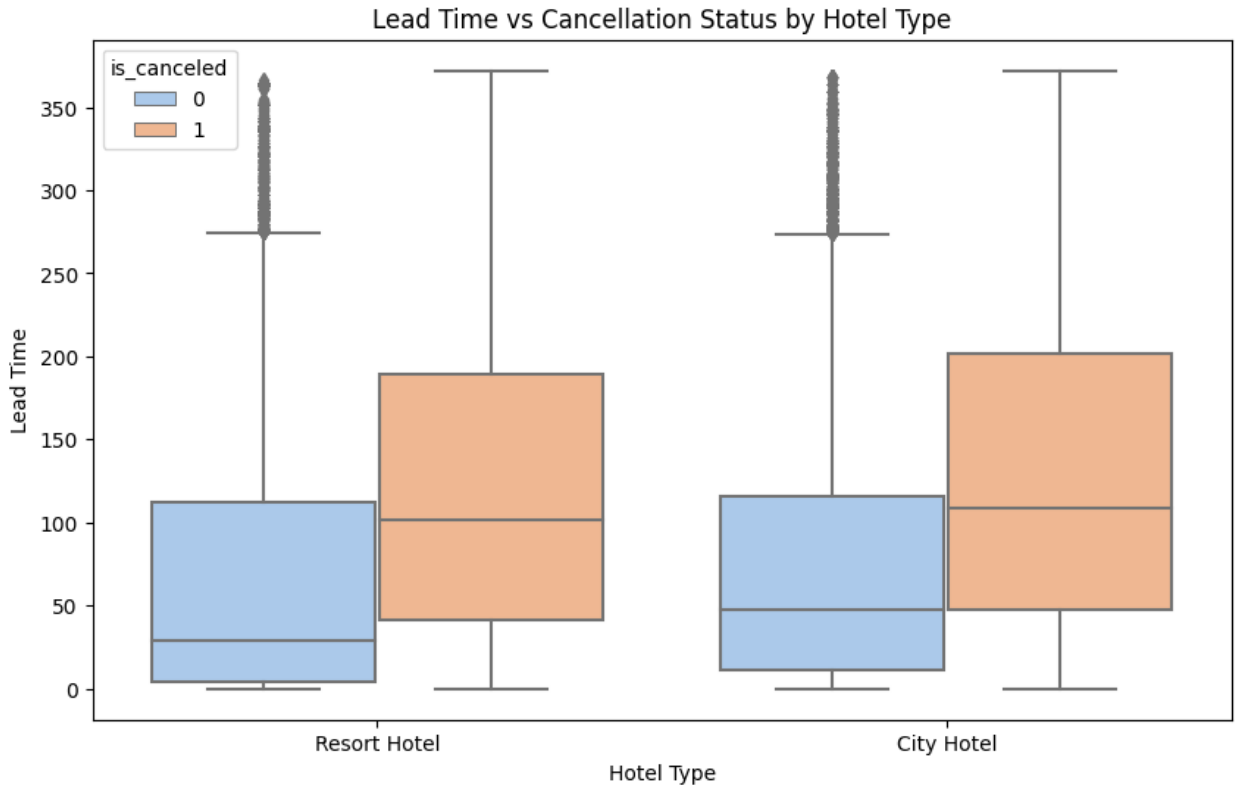

Cancellation Rate by Market Segment (City Hotel)
market_segment
Offline TA/TO
Online TA
Groups
Complementary
Direct
Corporate
Undefined
Aviation
is_canceled
0
1
0
5000
10000
15000
20000
count

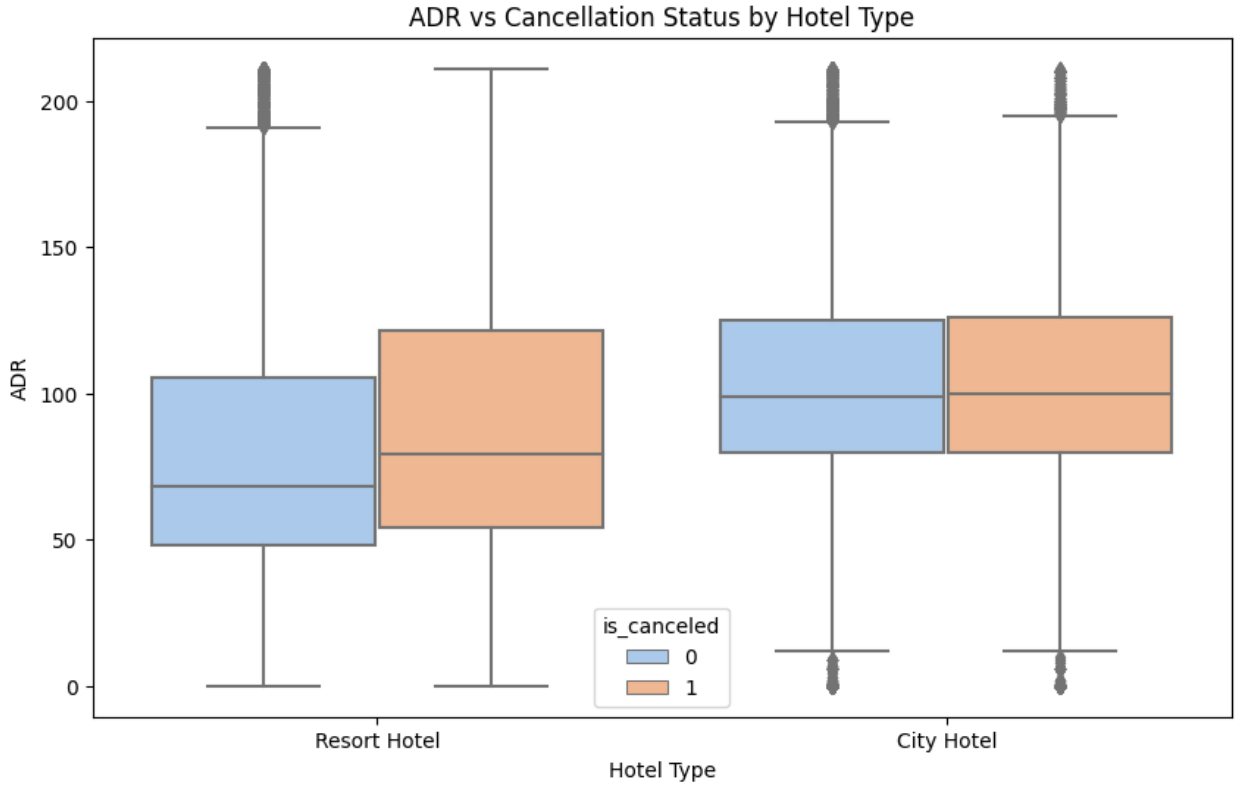

Cancellation Rate by distribution_channel (City Hotel)
is_canceled
0
1
Count
35000
30000
25000
20000
15000
10000
5000
0
TA/TO
Direct
Undefined
Corporate
GDS
distribution_channel

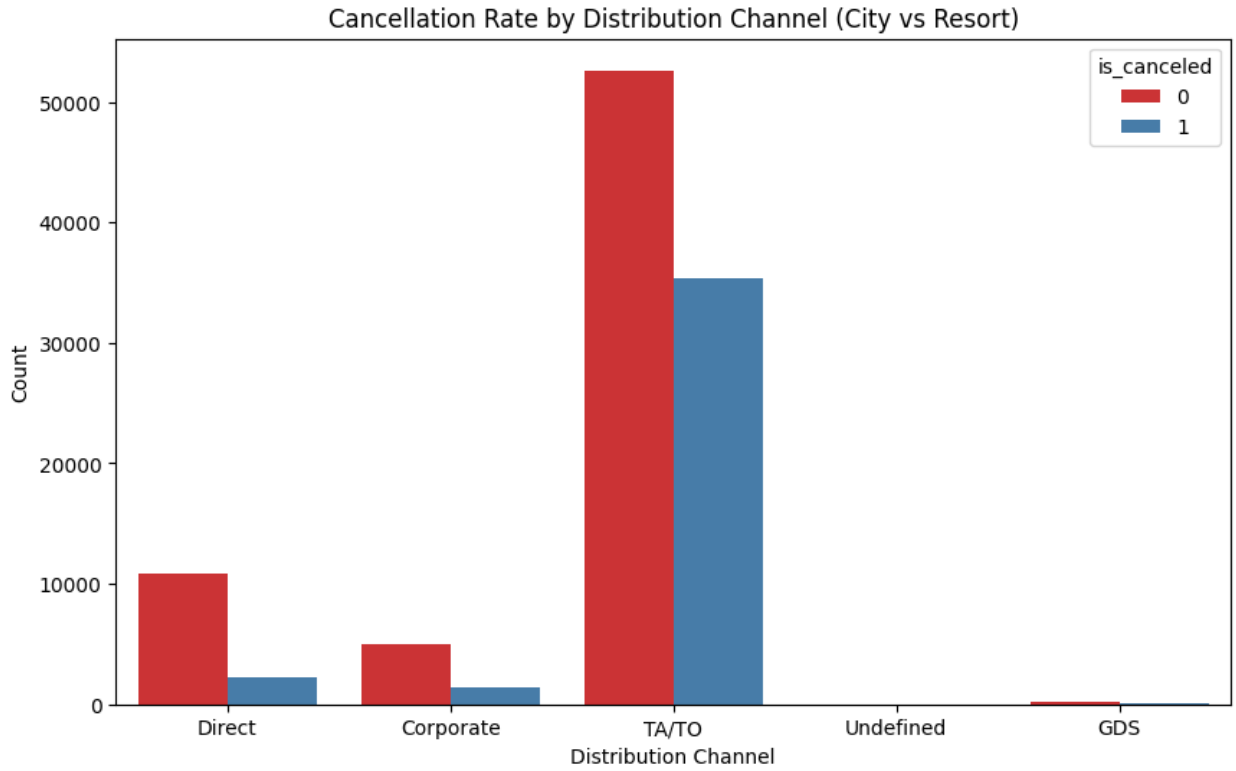

Cancellation Rate by deposit_type (City Hotel)
is_canceled
0
1
Count
40000
30000
20000
10000
0
No Deposit
Non Refund
Refundable
deposit_type




```
# Comparing Long Lead Times and Cancellations by Hotel Type
plt.figure(figsize=(10, 6))
sns.boxplot(x='hotel', y='lead_time', hue='is_canceled', data=df)
plt.title('Lead Time vs Cancellation Status by Hotel Type')
plt.xlabel('Hotel Type')
plt.ylabel('Lead Time')
plt.show()

# Comparing ADR by Hotel Type and Cancellation Status
plt.figure(figsize=(10, 6))
sns.boxplot(x='hotel', y='adr', hue='is_canceled', data=df)
plt.title('ADR vs Cancellation Status by Hotel Type')
plt.xlabel('Hotel Type')
plt.ylabel('ADR')
plt.show()

# Distribution Channel Behavior by Hotel Type
plt.figure(figsize=(10, 6))
sns.countplot(x='distribution_channel', hue='is_canceled', data=df,
palette='Set1')
plt.title('Cancellation Rate by Distribution Channel (City vs
Resort)')
plt.xlabel('Distribution Channel')
plt.ylabel('Count')
plt.show()
```

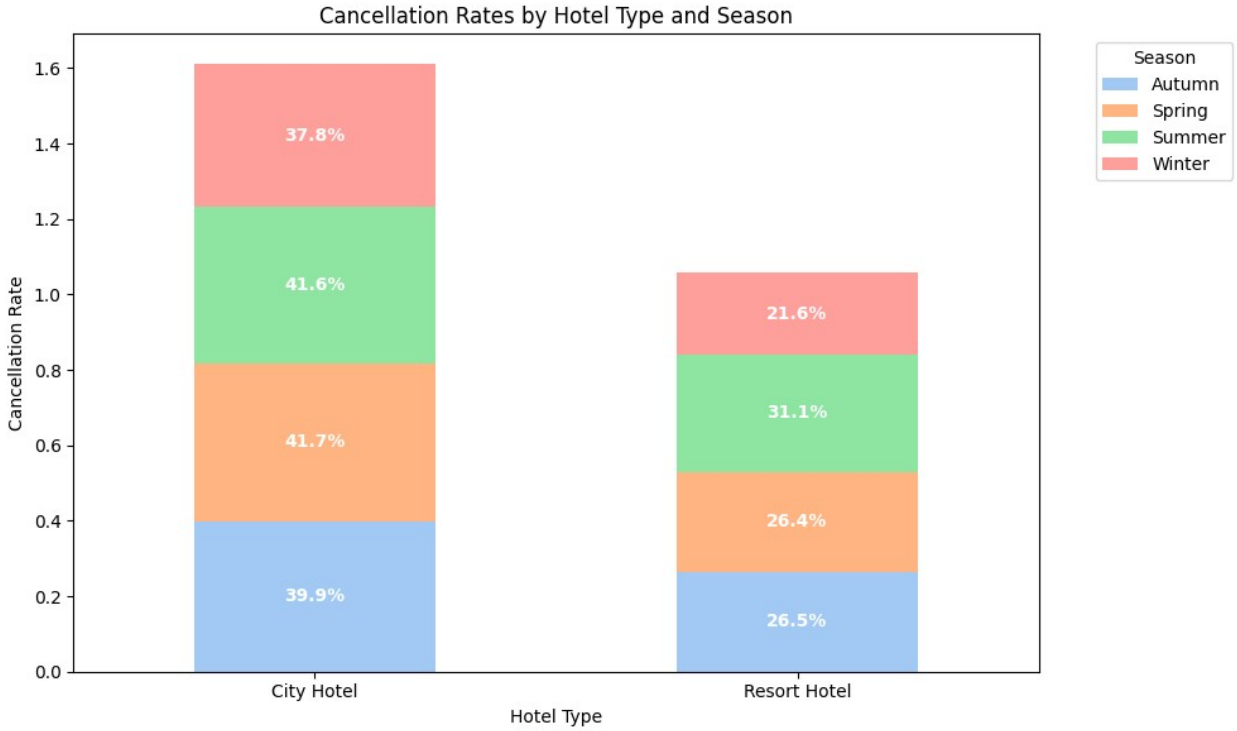

Lead Time vs Cancellation Status by Hotel Type
is_canceled
0
1
Lead Time
0
50
100
150
200
250
300
350
Resort Hotel
City Hotel
Hotel Type


ADR vs Cancellation Status by Hotel Type
ADR
0
50
100
150
200
is_canceled
0
1
Resort Hotel
City Hotel
Hotel Type

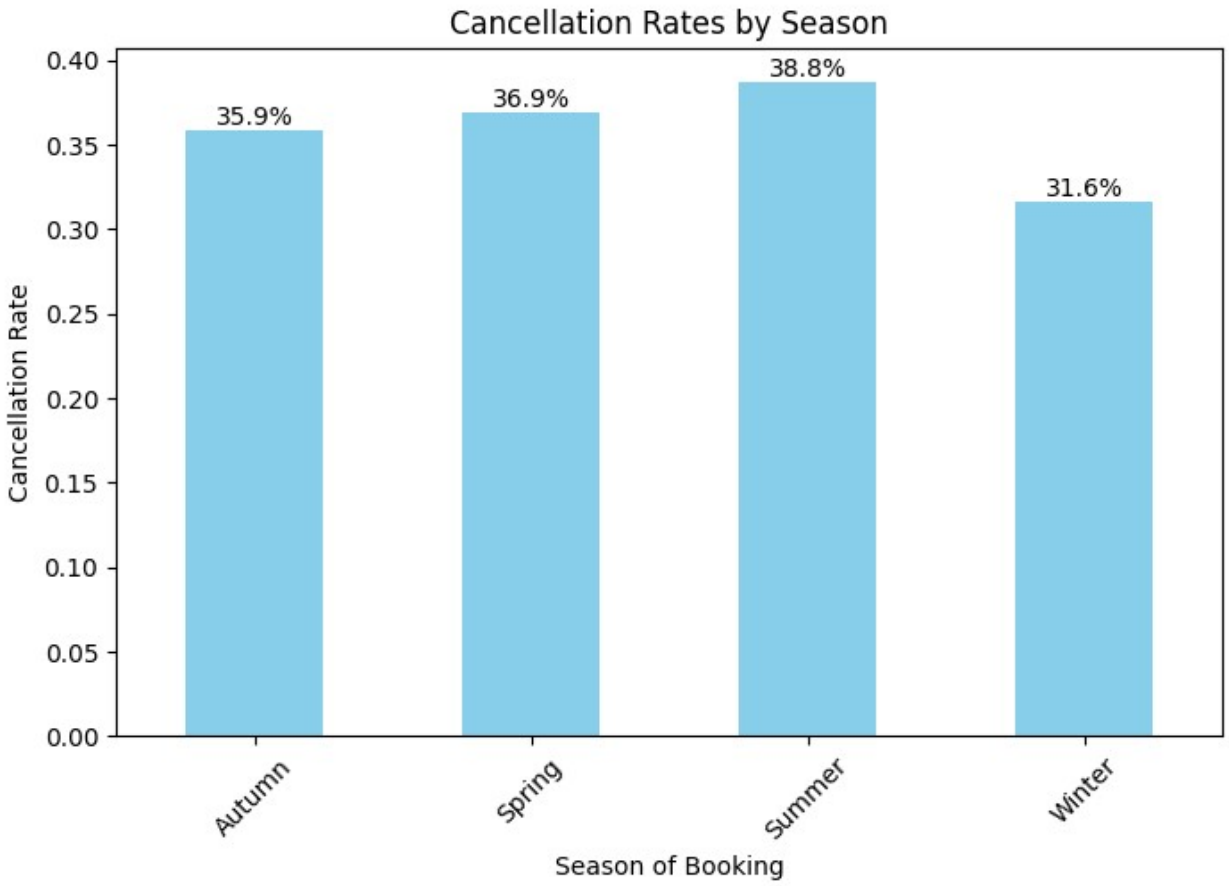



```
# 1. Define the missing variable first
hotel_season_cancellation = df.groupby(['hotel', 'season_of_booking'])
['is_canceled'].mean().unstack()

# 2. Create the plot
ax = hotel_season_cancellation.plot(kind='bar', stacked=True,
title='Cancellation Rates by Hotel Type and Season', figsize=(10, 6))

plt.ylabel('Cancellation Rate')
plt.xlabel('Hotel Type')
plt.xticks(rotation=0)

# 3. Add percentage labels to the stacked bars
for container in ax.containers:
    # Create labels: format as percentage, hide if 0 (to avoid
clutter)
    labels = [f'{val*100:.1f}%' if val > 0 else '' for val in
container.datavalues]

    # Place labels in the center of each bar segment
    ax.bar_label(container, labels=labels, label_type='center',
color='white', fontsize=10, weight='bold')

# Move legend outside to keep the chart clean
plt.legend(title='Season', bbox_to_anchor=(1.05, 1), loc='upper left')
```

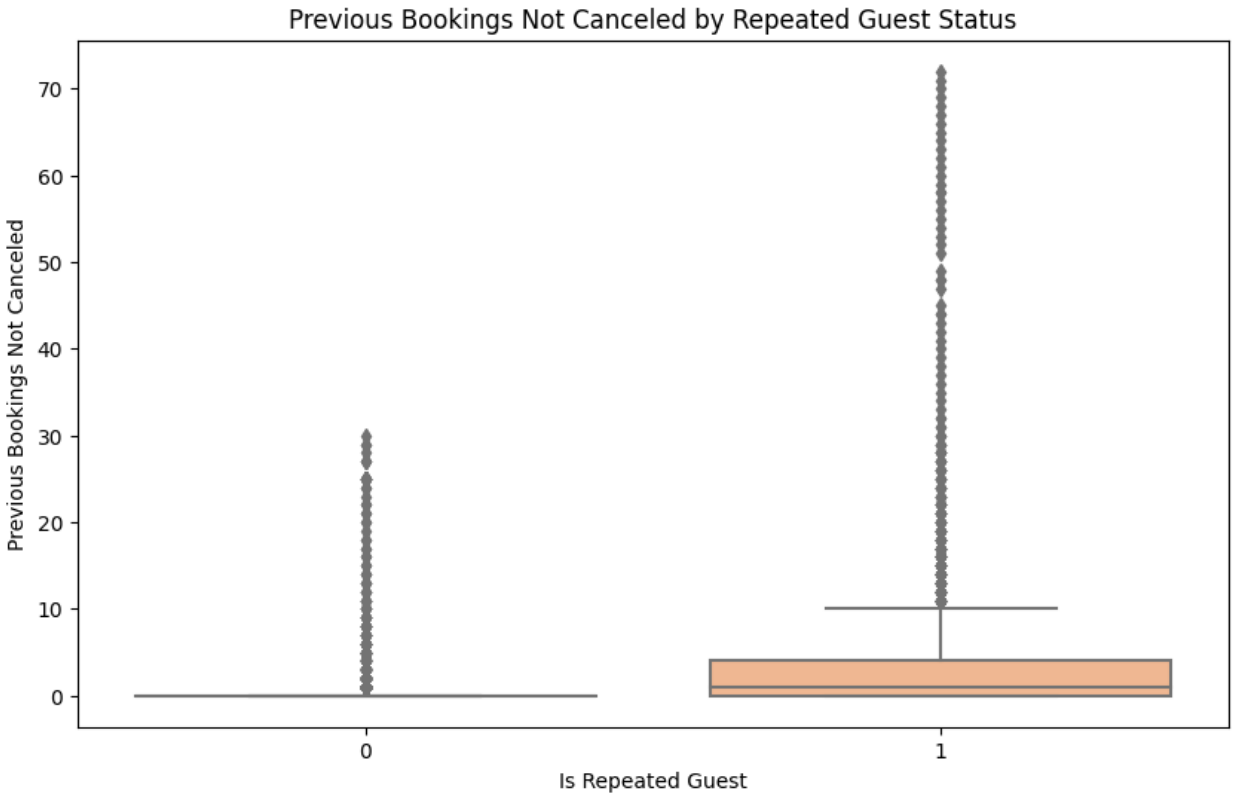
```
plt.tight_layout()
plt.show()
```

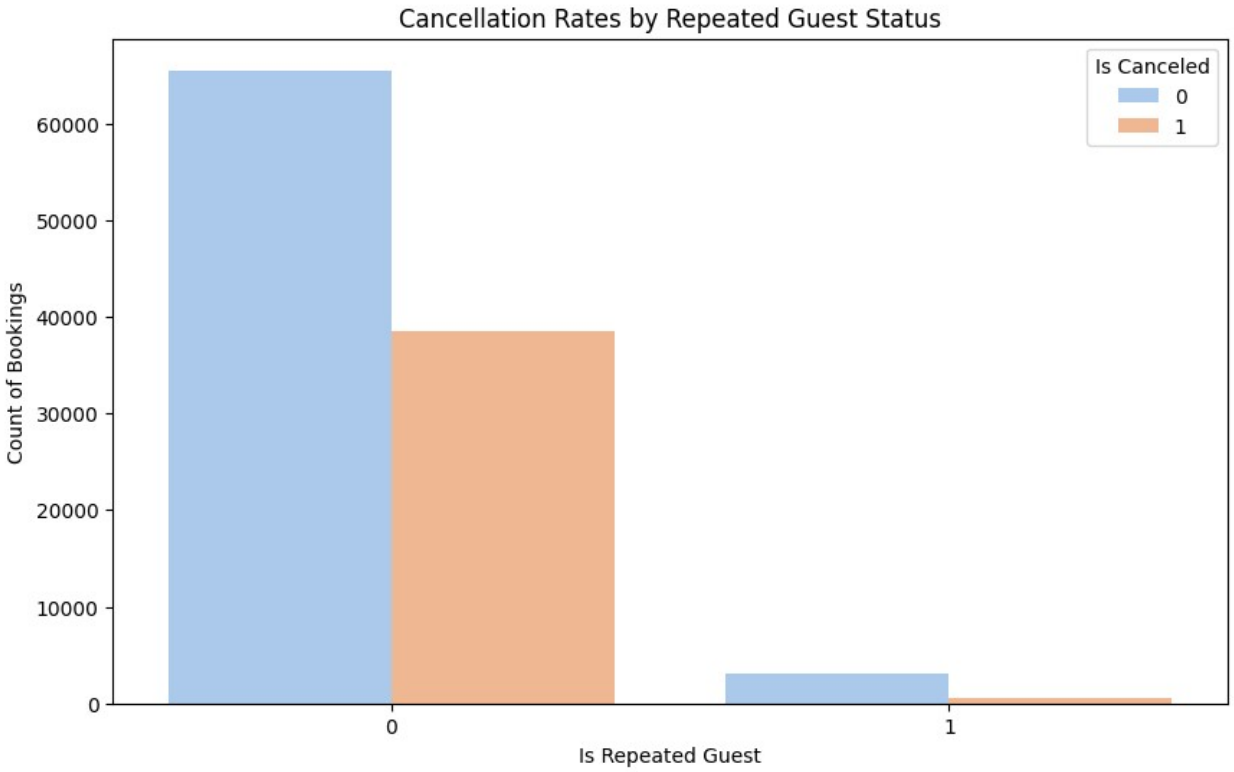



```
season_cancellation_rate = df.groupby('season_of_booking')
['is_canceled'].mean()
# Create the plot
ax = season_cancellation_rate.plot(kind='bar', title='Cancellation
Rates by Season', color='skyblue', figsize=(8, 5))

plt.ylabel('Cancellation Rate')
plt.xlabel('Season of Booking')
plt.xticks(rotation=45)

# Loop through patches to add labels
for p in ax.patches:
    value = p.get_height()
    percentage = f'{value*100:.1f}%'

    x = p.get_x() + p.get_width() / 2
    y = p.get_height()

    ax.annotate(percentage, (x, y), ha='center', va='bottom')

plt.show()
```

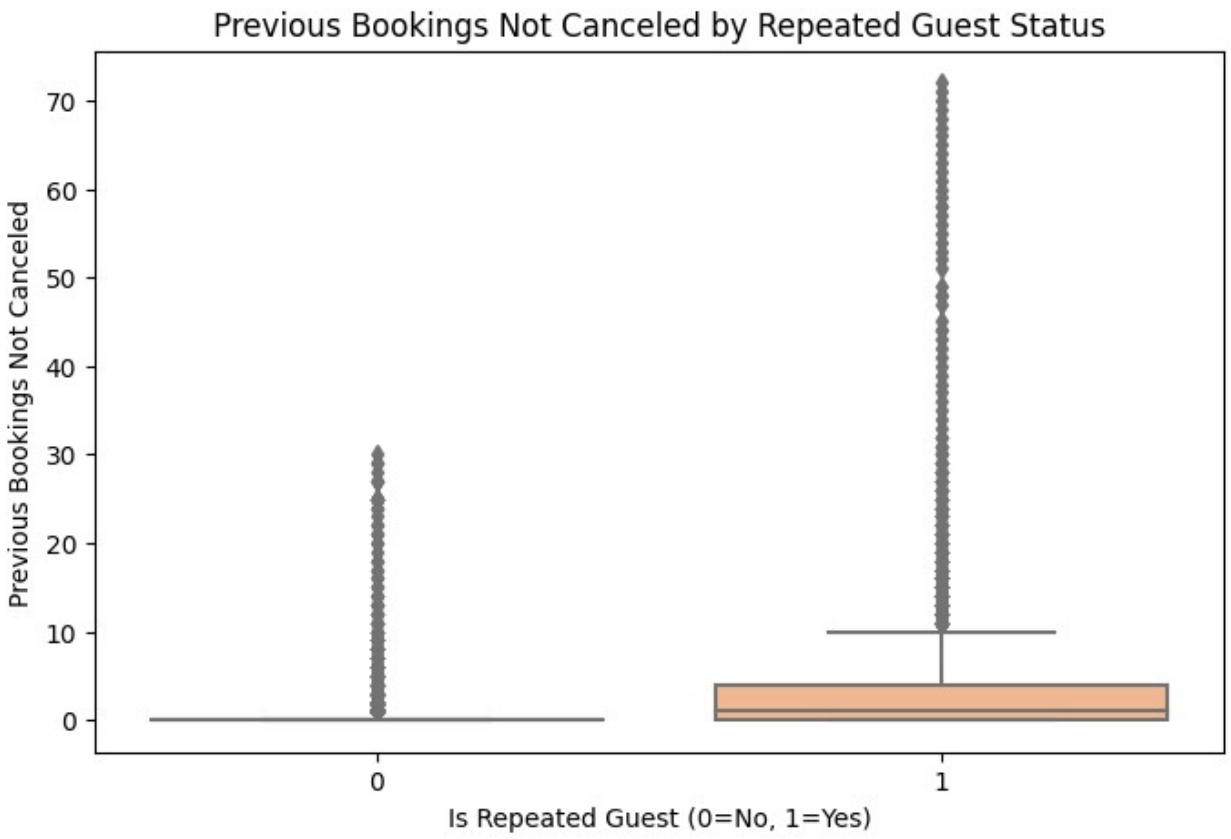


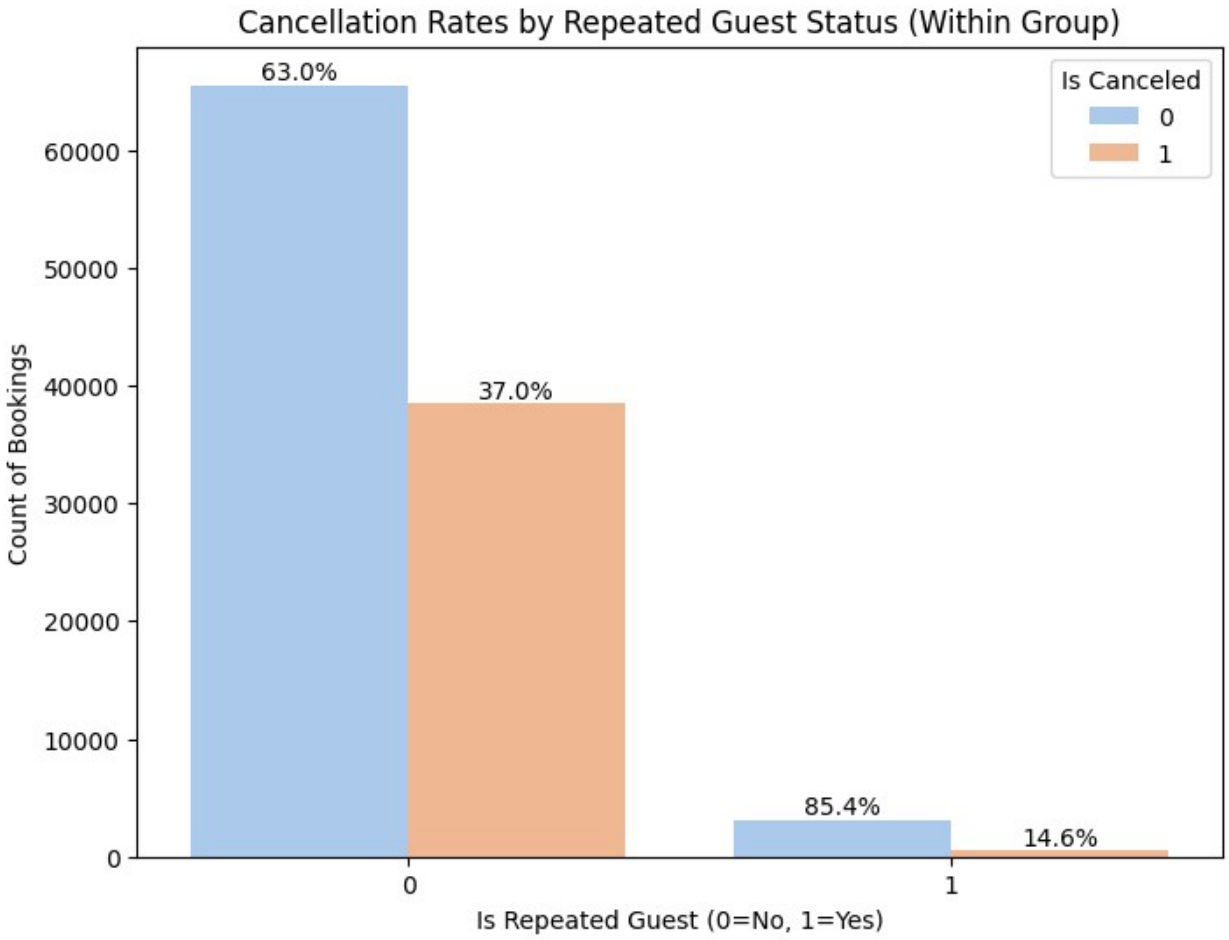

```
# Boxplot of previous bookings not canceled by repeated guest status
sns.boxplot(x='is_repeated_guest', y='previous_bookings_not_canceled',
data=df)
plt.title('Previous Bookings Not Canceled by Repeated Guest Status')
plt.xlabel('Is Repeated Guest')
plt.ylabel('Previous Bookings Not Canceled')
plt.show()

# Countplot for is_repeated_guest vs is_canceled
sns.countplot(x='is_repeated_guest', hue='is_canceled', data=df)
plt.title('Cancellation Rates by Repeated Guest Status')
plt.xlabel('Is Repeated Guest')
plt.ylabel('Count of Bookings')
plt.legend(title='Is Canceled', loc='upper right')
plt.show()
```


Previous Bookings Not Canceled by Repeated Guest Status
Previous Bookings Not Canceled
70
60
50
40
30
20
10
0
0
1
Is Repeated Guest


Cancellation Rates by Repeated Guest Status
Is Canceled
0
1
Count of Bookings
60000
50000
40000
30000
20000
10000
0
0
1
Is Repeated Guest

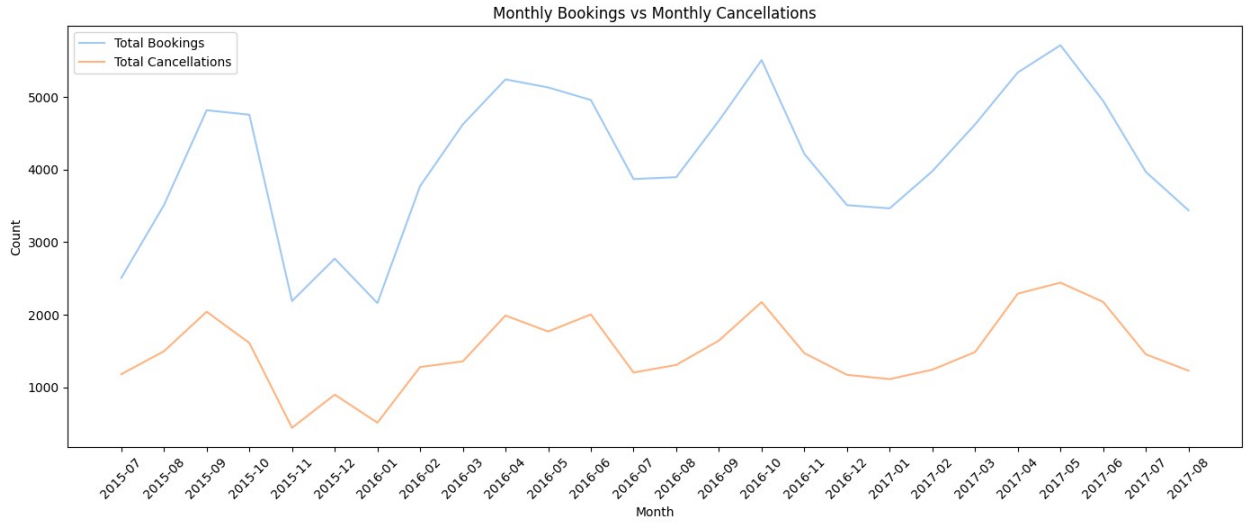
```
# 1. Boxplot (Unchanged - Boxplots don't support percentage labels)
plt.figure(figsize=(8, 5))
sns.boxplot(x='is_repeated_guest', y='previous_bookings_not_canceled',
data=df)
plt.title('Previous Bookings Not Canceled by Repeated Guest Status')
plt.xlabel('Is Repeated Guest (0=No, 1=Yes)')
plt.ylabel('Previous Bookings Not Canceled')
plt.show()


plt.figure(figsize=(8, 6))

# 1. Create the plot
ax = sns.countplot(x='is_repeated_guest', hue='is_canceled', data=df)
plt.title('Cancellation Rates by Repeated Guest Status (Within
Group)')
plt.xlabel('Is Repeated Guest (0=No, 1=Yes)')
plt.ylabel('Count of Bookings')
plt.legend(title='Is Canceled')

# 2. Calculate the total count for each x-axis category (0 and 1)
# This gives us: {0: Total_New_Guests, 1: Total_Repeated_Guests}
group_totals = df['is_repeated_guest'].value_counts().sort_index()

# 3. Add percentage labels relative to each group's total
for container in ax.containers:
    # Iterate through each bar in the container
    for i, bar in enumerate(container):
        # Get the height (count) of the bar
        height = bar.get_height()

        # Get the total count for this specific x-axis group (0 or 1)
        # We use 'i' because the bars are ordered by x-axis index
        total = group_totals.iloc[i]

        # Calculate percentage
        percentage = f'{height / total * 100:.1f}%'

        # Annotate
        ax.annotate(percentage,
                    (bar.get_x() + bar.get_width() / 2, height),
                    ha='center', va='bottom')

plt.show()
```

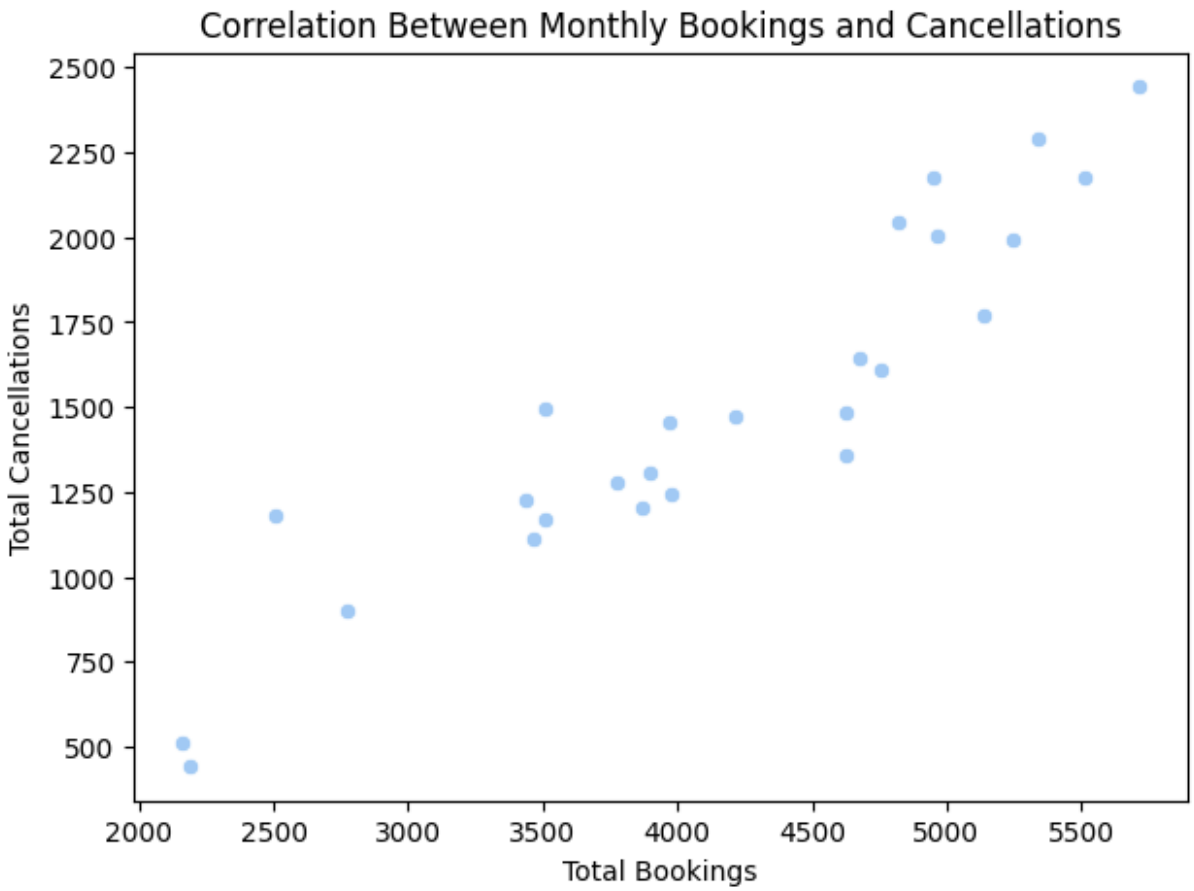

Previous Bookings Not Canceled by Repeated Guest Status
Previous Bookings Not Canceled
70
60
50
40
30
20
10
0
0
1
Is Repeated Guest (0=No, 1=Yes)




```
# Create a year-month column for grouping
df['year_month'] = df['date_of_arrival'].dt.to_period('M').astype(str)

# Group by month
monthly_data = df.groupby('year_month').agg(
    total_bookings = ('is_canceled', 'count'),
    total_cancellations = ('is_canceled', 'sum')
).reset_index()

monthly_data.head()

plt.figure(figsize=(14,6))

sns.lineplot(x='year_month', y='total_bookings', data=monthly_data,
label='Total Bookings')
sns.lineplot(x='year_month', y='total_cancellations',
data=monthly_data, label='Total Cancellations')

plt.xticks(rotation=45)
plt.title("Monthly Bookings vs Monthly Cancellations")
plt.xlabel("Month")
```

```
plt.ylabel("Count")
plt.legend()
plt.tight_layout()
plt.show()

plt.figure(figsize=(7,5))

sns.scatterplot(
    x='total_bookings',
    y='total_cancellations',
    data=monthly_data
)

plt.title("Correlation Between Monthly Bookings and Cancellations")
plt.xlabel("Total Bookings")
plt.ylabel("Total Cancellations")
plt.show()

corr =
monthly_data['total_bookings'].corr(monthly_data['total_cancellations'
])
print("Correlation between monthly bookings and cancellations:", corr)
```

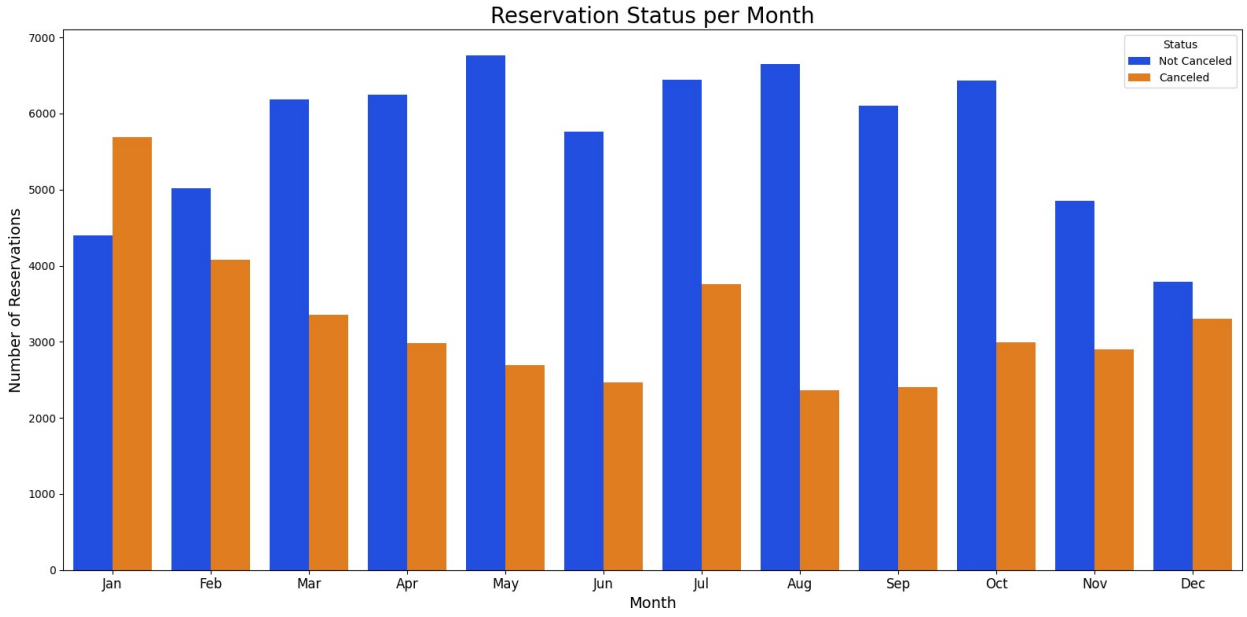





```
Correlation between monthly bookings and cancellations:
0.9261697170193572
```

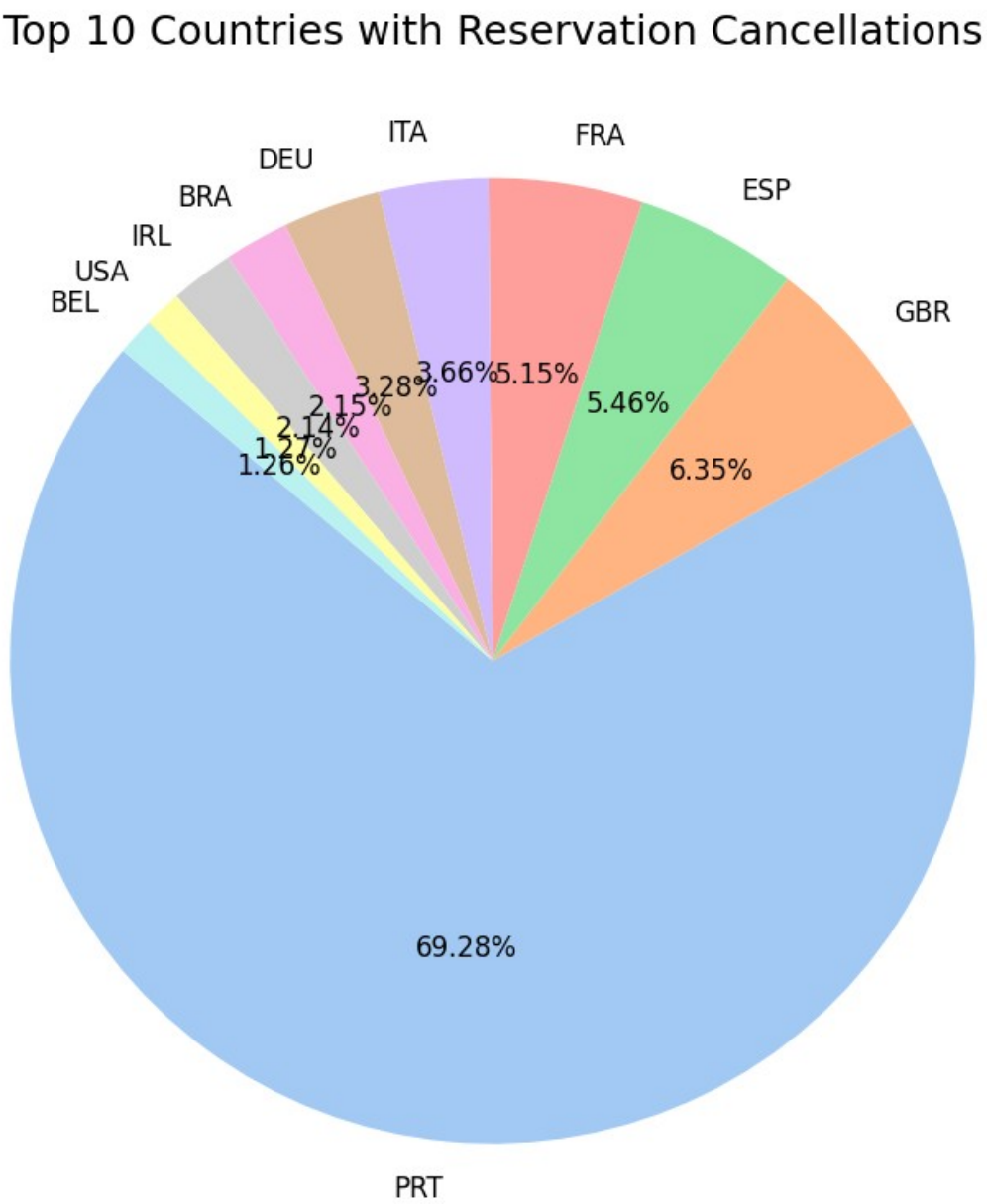
```
import calendar
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# 1. FIX: Convert the column to datetime first
df['reservation_status_date'] =
pd.to_datetime(df['reservation_status_date'])

# 2. Now you can safely extract the month
df['month'] = df['reservation_status_date'].dt.month

# 3. Plotting
plt.figure(figsize=(16, 8))
ax1 = sns.countplot(
    x='month',
    hue='is_canceled',
    data=df,
    palette='bright'
```

```
)

# Set legend and labels
legend_labels, _ = ax1.get_legend_handles_labels()
ax1.legend(legend_labels, ['Not Canceled', 'Canceled'],
title='Status')
ax1.set_title('Reservation Status per Month', fontsize=20)
ax1.set_xlabel('Month', fontsize=14)
ax1.set_ylabel('Number of Reservations', fontsize=14)

# Map numeric months (1, 2) to names (Jan, Feb) for the X-axis labels
# We use a list comprehension to ensure labels match the data present
in the column
ax1.set_xticklabels([calendar.month_abbr[int(m)] for m in
sorted(df['month'].unique())], fontsize=12)

plt.tight_layout()
plt.show()
```

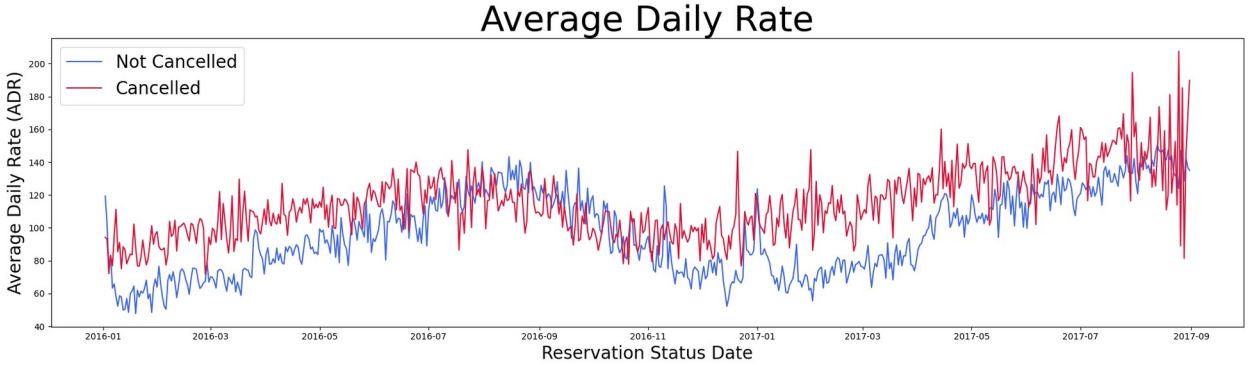



```
# Filter for canceled reservations
cancelled_data = df[df['is_canceled'] == 1]

# Get top 10 countries by number of cancellations
top_10_country = cancelled_data['country'].value_counts().head(10)

plt.figure(figsize=(8, 8))
plt.title('Top 10 Countries with Reservation Cancellations',
fontsize=18)

# Pie chart
```

```
plt.pie(
    top_10_country,
    autopct='%.2f%%',
    labels=top_10_country.index,
    startangle=140,
    textprops={'fontsize': 12}
)

plt.tight_layout()
plt.show()
```



```
# Compute ADR time series for cancelled reservations
cancelled_data = df[df['is_canceled'] == 1]
cancelled_df_adr = cancelled_data.groupby('reservation_status_date')
[['adr']].mean()
cancelled_df_adr.reset_index(inplace=True)
cancelled_df_adr.sort_values('reservation_status_date', inplace=True)
# Compute ADR time series for not cancelled reservations
not_cancelled_data = df[df['is_canceled'] == 0]
not_cancelled_df_adr =
not_cancelled_data.groupby('reservation_status_date')[['adr']].mean()
not_cancelled_df_adr.reset_index(inplace=True)
not_cancelled_df_adr.sort_values('reservation_status_date',
inplace=True)
#  Average Daily Rate (Filtered Date Range): Cancelled vs Not
Cancelled
# Filter for reservation_status_date between '2016-01-01' and '2017-
09-01'
cancelled_df_adr = cancelled_df_adr[
    (cancelled_df_adr['reservation_status_date'] > '2016-01-01') &
    (cancelled_df_adr['reservation_status_date'] < '2017-09-01')
]

not_cancelled_df_adr = not_cancelled_df_adr[
    (not_cancelled_df_adr['reservation_status_date'] > '2016-01-01') &
    (not_cancelled_df_adr['reservation_status_date'] < '2017-09-01')
]

plt.figure(figsize=(20, 6))
plt.title('Average Daily Rate', fontsize=40)
plt.plot(not_cancelled_df_adr['reservation_status_date'],
not_cancelled_df_adr['adr'], label='Not Cancelled', color='royalblue')
plt.plot(cancelled_df_adr['reservation_status_date'],
cancelled_df_adr['adr'], label='Cancelled', color='crimson')
plt.xlabel('Reservation Status Date', fontsize=20)
plt.ylabel('Average Daily Rate (ADR)', fontsize=20)
plt.legend(fontsize=20)
plt.tight_layout()
plt.show()
```



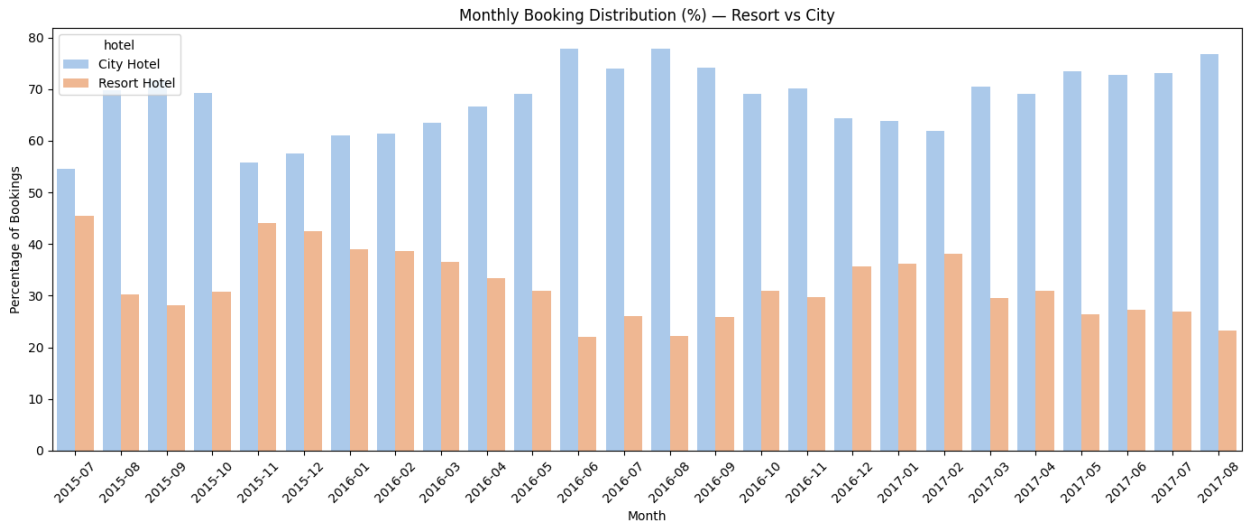
```python
# Absolute counts per market segment
market_segment_counts = df['market_segment'].value_counts()
print("Market Segment Counts:\n", market_segment_counts)
print('-' * 40)

# Proportion (percentage) per market segment
market_segment_perc =
df['market_segment'].value_counts(normalize=True) * 100
print("Market Segment Percentage:\n", market_segment_perc)
print('-' * 40)
```

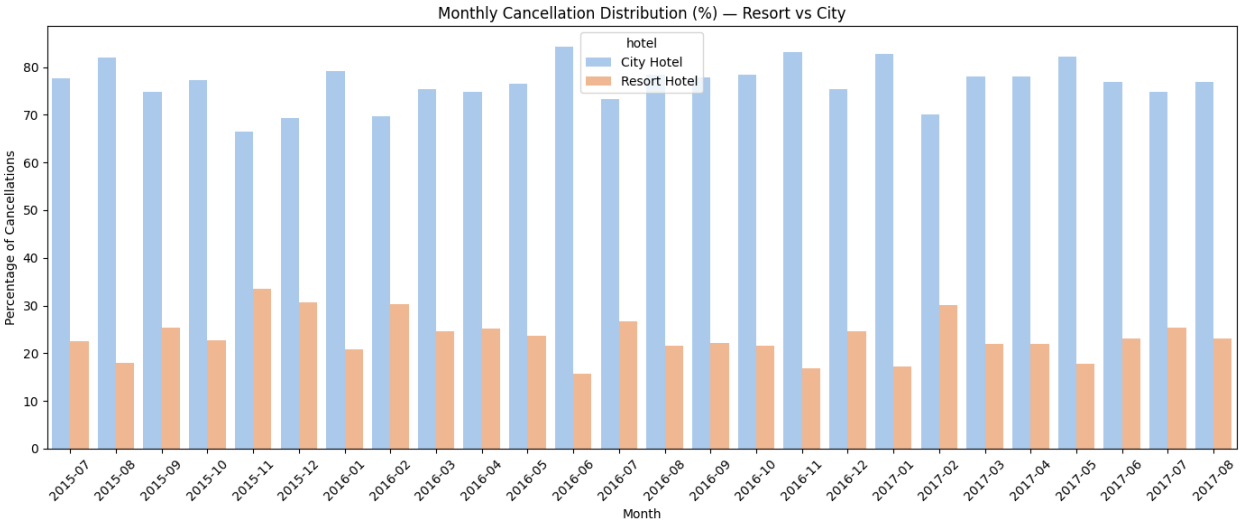
```
Market Segment Counts:
 market_segment
Online TA        51526
Offline TA/TO    21496
Groups           17397
Direct           11140
Corporate         5123
Complementary      735
Aviation           212
Undefined            2
Name: count, dtype: int64
----------------------------------------
Market Segment Percentage:
 market_segment
Online TA        47.872825
Offline TA/TO    19.971941
Groups           16.163559
Direct           10.350178
Corporate         4.759781
Complementary     0.682889
Aviation          0.196969
Undefined         0.001858
Name: proportion, dtype: float64
----------------------------------------
```

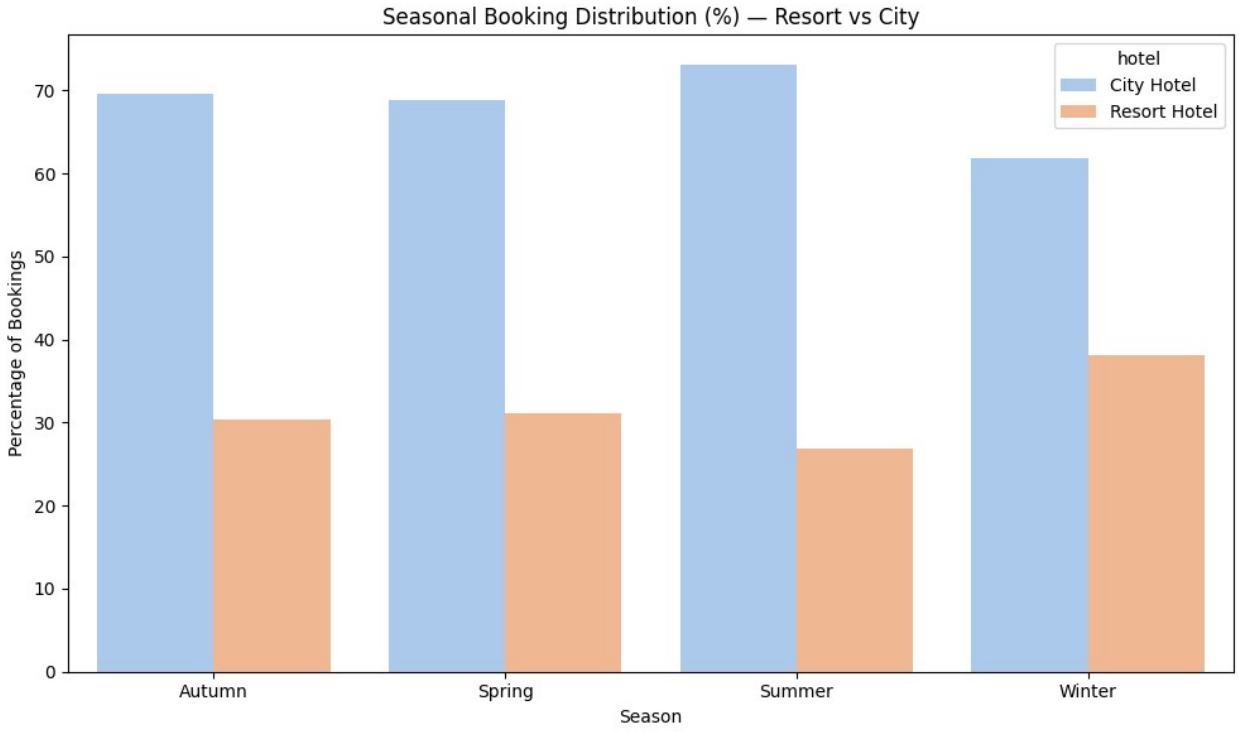
```python
# 1 PREP — CREATE YEAR-MONTH COLUMN (if not already created)
df['year_month'] = df['date_of_arrival'].dt.to_period('M').astype(str)

# Subset of only canceled bookings
cancel_df = df[df['is_canceled'] == 1]


# ========================================
# 2 MONTHLY BOOKING DISTRIBUTION (%)
# ========================================
monthly_bookings = df.groupby(['year_month',
'hotel']).size().reset_index(name='count')

# Use transform instead of apply to avoid index issues
```

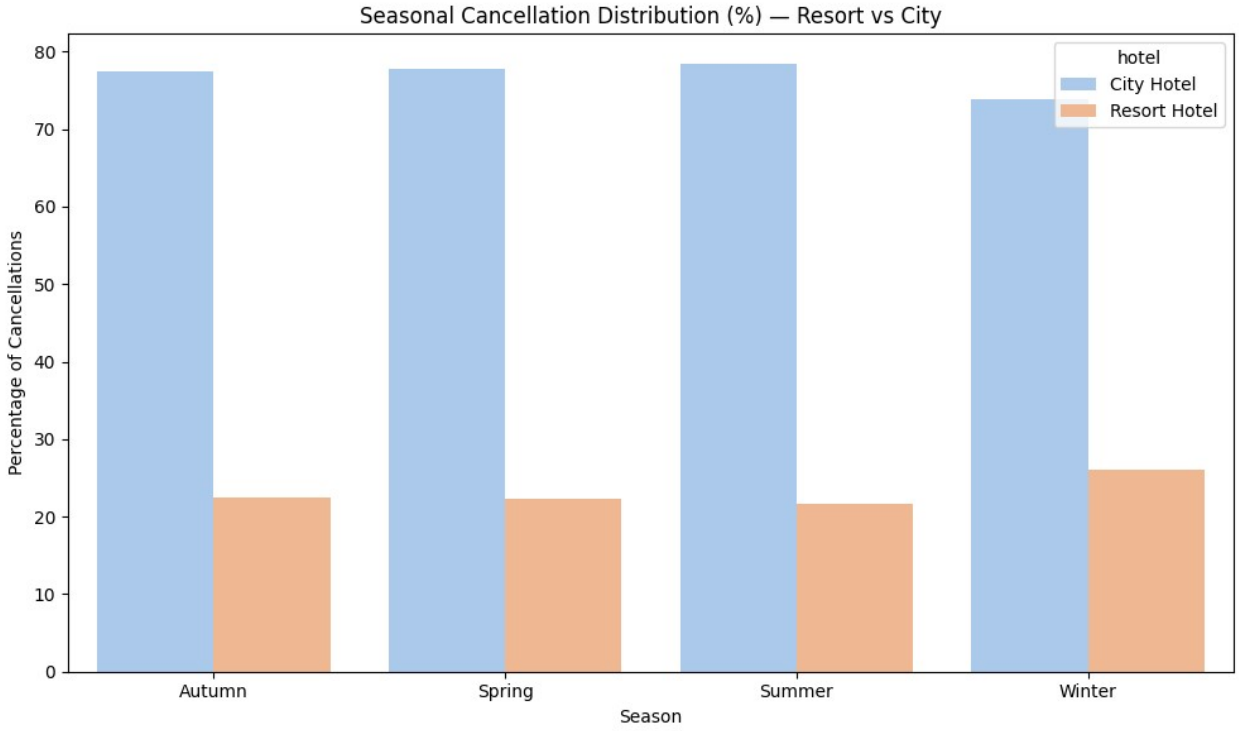
```
monthly_bookings['percent'] = (
    monthly_bookings['count'] /
    monthly_bookings.groupby('year_month')['count'].transform('sum') *
100
)

plt.figure(figsize=(14, 6))
sns.barplot(data=monthly_bookings, x='year_month', y='percent',
hue='hotel')
plt.title("Monthly Booking Distribution (%) — Resort vs City")
plt.xticks(rotation=45)
plt.ylabel("Percentage of Bookings")
plt.xlabel("Month")
plt.tight_layout()
plt.show()


# ========================================
# 3 MONTHLY CANCELLATION DISTRIBUTION (%)
# ========================================
monthly_cancels = cancel_df.groupby(['year_month',
'hotel']).size().reset_index(name='count')

monthly_cancels['percent'] = (
    monthly_cancels['count'] /
    monthly_cancels.groupby('year_month')['count'].transform('sum') *
100
)

plt.figure(figsize=(14, 6))
sns.barplot(data=monthly_cancels, x='year_month', y='percent',
hue='hotel')
plt.title("Monthly Cancellation Distribution (%) — Resort vs City")
plt.xticks(rotation=45)
plt.ylabel("Percentage of Cancellations")
plt.xlabel("Month")
plt.tight_layout()
plt.show()


# ========================================
# 4 SEASONAL BOOKING DISTRIBUTION (%)
# ========================================
season_bookings = df.groupby(['season_of_booking',
'hotel']).size().reset_index(name='count')

season_bookings['percent'] = (
    season_bookings['count'] /
    season_bookings.groupby('season_of_booking')
['count'].transform('sum') * 100
```

```
)

plt.figure(figsize=(10, 6))
sns.barplot(data=season_bookings, x='season_of_booking', y='percent',
hue='hotel')
plt.title("Seasonal Booking Distribution (%) — Resort vs City")
plt.ylabel("Percentage of Bookings")
plt.xlabel("Season")
plt.tight_layout()
plt.show()


# ========================================
# 5 SEASONAL CANCELLATION DISTRIBUTION (%)
# ========================================
season_cancels = cancel_df.groupby(['season_of_booking',
'hotel']).size().reset_index(name='count')

season_cancels['percent'] = (
    season_cancels['count'] /
    season_cancels.groupby('season_of_booking')
['count'].transform('sum') * 100
)

plt.figure(figsize=(10, 6))
sns.barplot(data=season_cancels, x='season_of_booking', y='percent',
hue='hotel')
plt.title("Seasonal Cancellation Distribution (%) — Resort vs City")
plt.ylabel("Percentage of Cancellations")
plt.xlabel("Season")
plt.tight_layout()
plt.show()
```

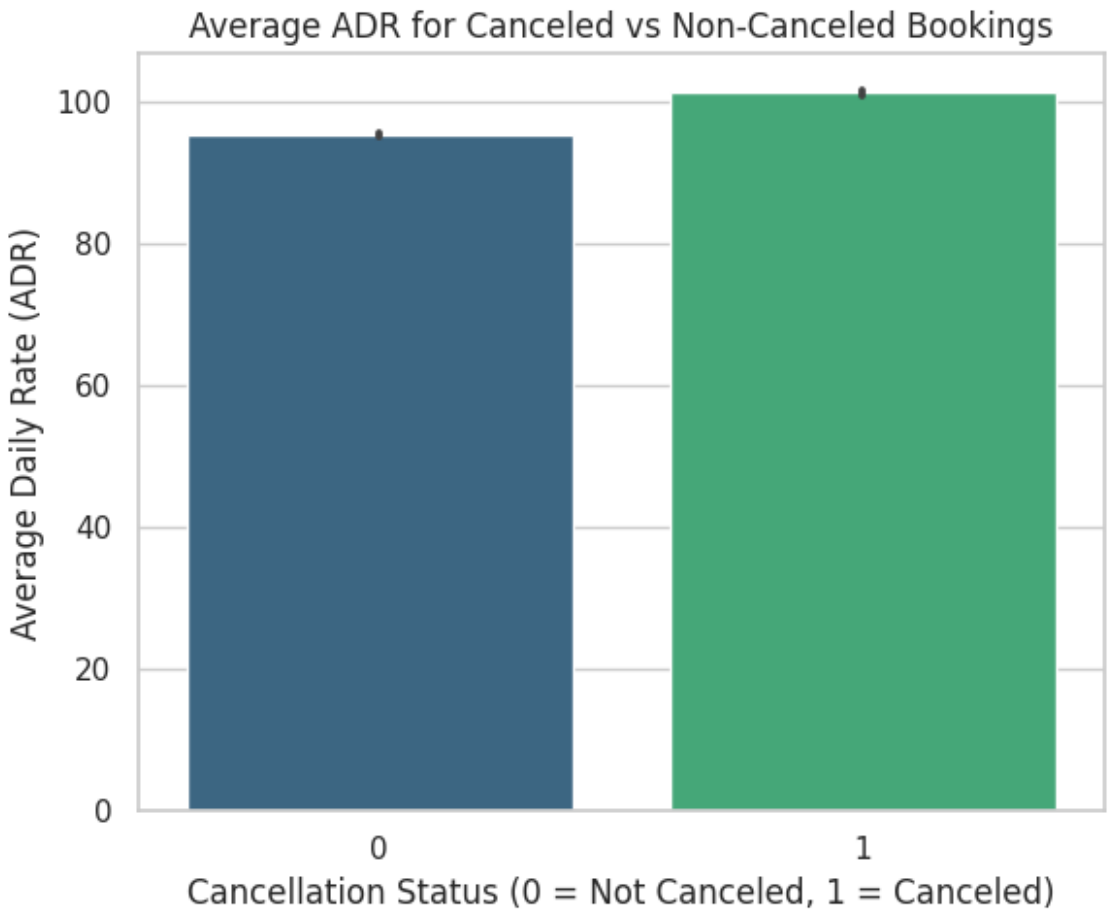



Monthly Cancellation Distribution (%) — Resort vs City
hotel
City Hotel
Resort Hotel
Percentage of Cancellations
0
10
20
30
40
50
60
70
80
2015-07
2015-08
2015-09
2015-10
2015-11
2015-12
2016-01
2016-02
2016-03
2016-04
2016-05
2016-06
2016-07
2016-08
2016-09
2016-10
2016-11
2016-12
2017-01
2017-02
2017-03
2017-04
2017-05
2017-06
2017-07
2017-08
Month

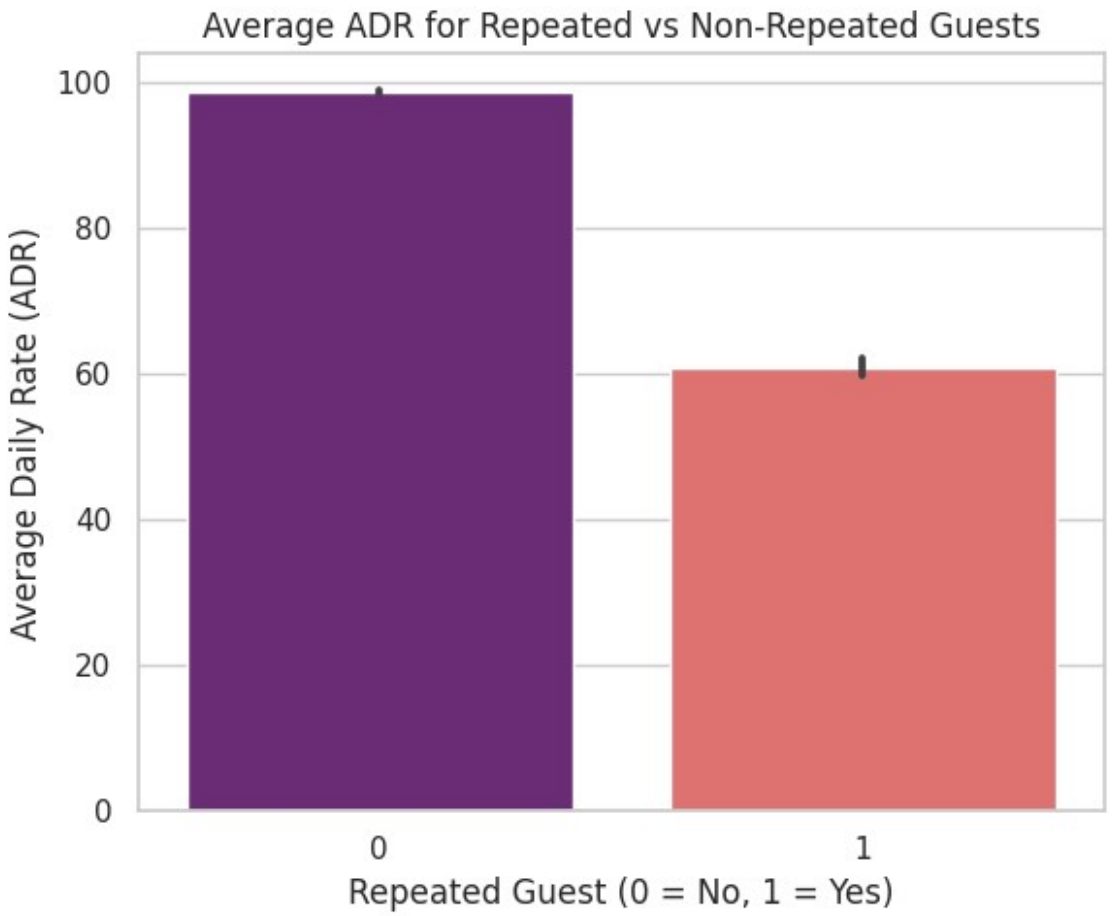

Seasonal Booking Distribution (%) — Resort vs City
hotel
City Hotel
Resort Hotel
Percentage of Bookings
0
10
20
30
40
50
60
70
Autumn
Spring
Summer
Winter
Season



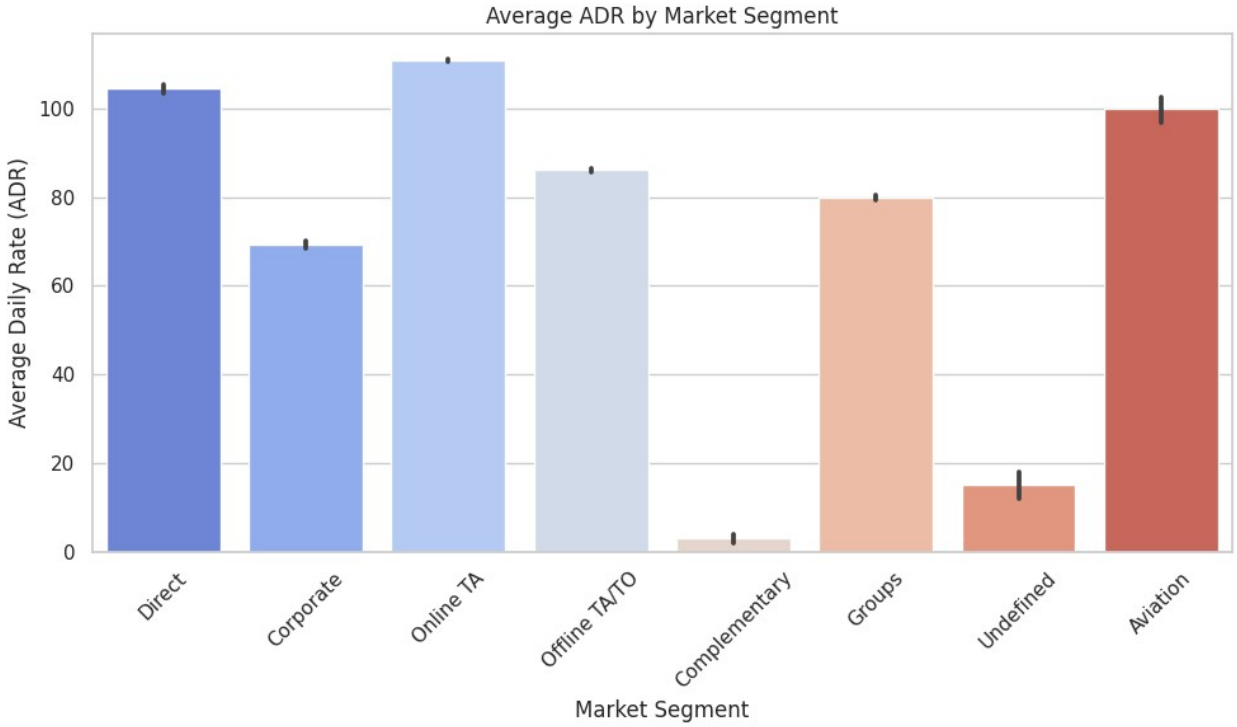

```
plt.figure(figsize=(6,5))
sns.barplot(data=df, x='is_canceled', y='adr', estimator='mean',
palette='viridis')

plt.title("Average ADR for Canceled vs Non-Canceled Bookings")
plt.xlabel("Cancellation Status (0 = Not Canceled, 1 = Canceled)")
plt.ylabel("Average Daily Rate (ADR)")
plt.tight_layout()
plt.show()
```



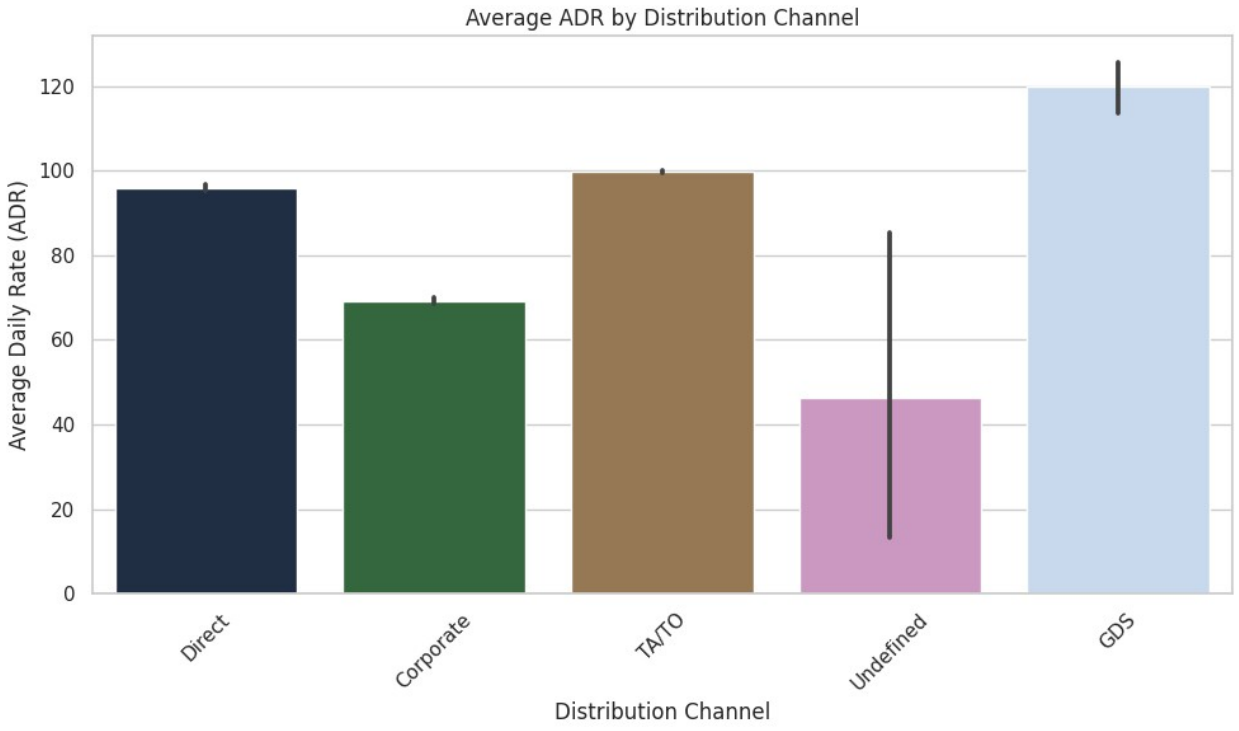

```
plt.figure(figsize=(6,5))
sns.barplot(data=df, x='is_repeated_guest', y='adr', estimator='mean',
palette='magma')

plt.title("Average ADR for Repeated vs Non-Repeated Guests")
plt.xlabel("Repeated Guest (0 = No, 1 = Yes)")
plt.ylabel("Average Daily Rate (ADR)")
plt.tight_layout()
plt.show()
```




```
plt.figure(figsize=(10,6))
sns.barplot(data=df, x='market_segment', y='adr', estimator='mean',
palette='coolwarm')

plt.title("Average ADR by Market Segment")
plt.xlabel("Market Segment")
plt.ylabel("Average Daily Rate (ADR)")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```




```
plt.figure(figsize=(10,6))
sns.barplot(data=df, x='distribution_channel', y='adr',
estimator='mean', palette='cubehelix')

plt.title("Average ADR by Distribution Channel")
plt.xlabel("Distribution Channel")
plt.ylabel("Average Daily Rate (ADR)")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```


Average ADR by Distribution Channel
Average Daily Rate (ADR)
120
100
80
60
40
20
0
Direct
Corporate
TA/TO
Undefined
GDS
Distribution Channel