```
import datetime
def find_optimal_booking_time():
    current_time = datetime.datetime.now()
    optimal_booking_time = current_time + datetime.timedelta(days=14)  # Booking 2 weeks in
    return optimal_booking_time


def find_ideal_length_of_stay():
    # Simulated rates for different lengths of stay
    rates = {
        1: 100,
        2: 90,
        3: 85,
        4: 80,
        5: 75,
        6: 70,
        7: 65
    }

    min_rate = min(rates.values())
    ideal_length_of_stay = [length for length, rate in rates.items() if rate == min_rate]
    return ideal_length_of_stay


if __name__ == "__main__":
    optimal_booking_time = find_optimal_booking_time()
    print("Optimal booking time:", optimal_booking_time.strftime("%Y-%m-%d"))
    ideal_length_of_stay = find_ideal_length_of_stay()
    print("Ideal length of stay for the best rates:", ideal_length_of_stay)
```

```
Optimal booking time: 2024-05-20
Ideal length of stay for the best rates: [7]
```

```
import random


def generate_special_requests(date):
    # Simulate factors that can lead to a surge in special requests
    holidays = ["New Year's Day", "Christmas", "Thanksgiving", "Easter", "Independence Day
    events = ["Music festival", "Business conference", "Sporting event", "Cultural festiva
    special_occasions = ["Anniversary", "Birthday", "Honeymoon", "Graduation"]
```

```
def generate_special_requests(date):
    # Simulate factors that can lead to a surge in special requests
    holidays = ["New Year's Day", "Christmas", "Thanksgiving", "Easter", "Independence Day
    events = ["Music festival", "Business conference", "Sporting event", "Cultural festiva
    special_occasions = ["Anniversary", "Birthday", "Honeymoon", "Graduation"]

    surge_probability = 0.3
    if date.month == 12 and date.day >= 20 and date.day <= 31:
        # Christmas and New Year holiday season
        return ["Decorations for Christmas", "Christmas dinner arrangement", "New Year's E
    elif date.month == 2 and date.day >= 12 and date.day <= 14:
        # Valentine's Day
        return ["Romantic dinner setup", "Flower bouquet in room", "Champagne and chocolat
    elif date.month == 7 and date.day >= 1 and date.day <= 4:
        # Independence Day (USA)
        return ["Fireworks view arrangement", "BBQ setup"]
    elif random.random() < surge_probability:
        # Other random events or special occasions
        special_requests = []
        if random.random() < 0.5:
            special_requests.append(random.choice(holidays))
        if random.random() < 0.5:
            special_requests.append(random.choice(events))
        if random.random() < 0.5:
            special_requests.append(random.choice(special_occasions))
        return special_requests
    else:
        return []


if __name__ == "__main__":
    # Simulate special requests for a specific date
    date = datetime.datetime(2024, 12, 24)  # Change the date as needed
    special_requests = generate_special_requests(date)

    if special_requests:
        print("Special requests for", date.strftime("%Y-%m-%d"), ":")
        for request in special_requests:
            print("-", request)
    else:
        print("No surge in special requests for", date.strftime("%Y-%m-%d"))
```

```
 Special requests for 2024-12-24 :
 - Decorations for Christmas
 - Christmas dinner arrangement
 - New Year's Eve party arrangement
```

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


data = pd.read_csv('/content/Hotel Bookings.csv')


print(data.head())
```

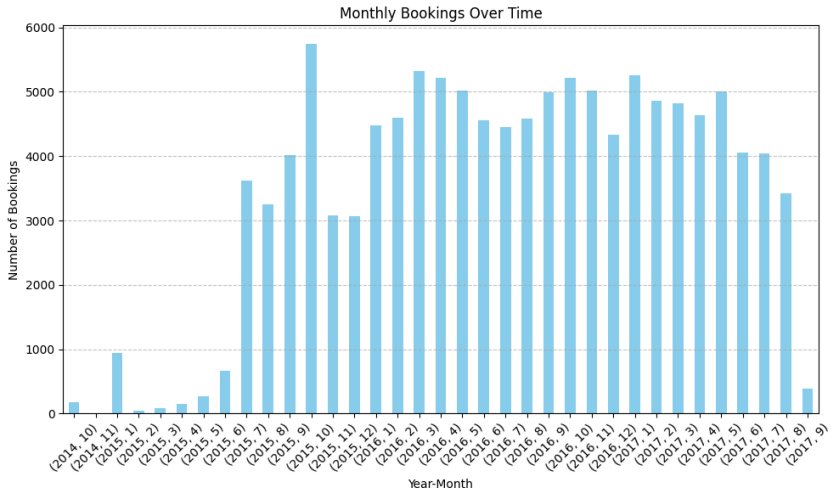
```
          hotel  is_canceled  lead_time  arrival_date_year arrival_date_month  \
0  Resort Hotel            0        342               2015               July
1  Resort Hotel            0        737               2015               July
2  Resort Hotel            0          7               2015 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

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

   stays_in_weekend_nights  stays_in_week_nights  adults  ...  deposit_type  \
0                        0                     0       2  ...    No Deposit
1                        0                     0       2  ...    No Deposit
2                        0                     1       1  ...    No Deposit
3                        0                     1       1  ...    No Deposit
4                        0                     2       2  ...    No Deposit

   agent company days_in_waiting_list customer_type   adr  \
0    NaN     NaN                    0     Transient   0.0
1    NaN     NaN                    0     Transient   0.0
2    NaN     NaN                    0     Transient  75.0
3  304.0     NaN                    0     Transient  75.0
4  240.0     NaN                    0     Transient  98.0

   required_car_parking_spaces  total_of_special_requests  reservation_status  \
0                            0                          0           Check-Out
1                            0                          0           Check-Out
2                            0                          0           Check-Out
3                            0                          0           Check-Out
4                            0                          1           Check-Out

  reservation_status_date
0              01-07-2015
1              01-07-2015
2              02-07-2015
3              02-07-2015
4              03-07-2015

[5 rows x 32 columns]
```

```
data['reservation_status_date'] = pd.to_datetime(data['reservation_status_date'], format="%
```

```
data['reservation_month'] = data['reservation_status_date'].dt.month
data['reservation_year'] = data['reservation_status_date'].dt.year
```

```
monthly_bookings = data.groupby(['reservation_year', 'reservation_month']).size()
```

```
plt.figure(figsize=(10, 6))
monthly_bookings.plot(kind='bar', color='skyblue')
plt.title('Monthly Bookings Over Time')
plt.xlabel('Year-Month')
plt.ylabel('Number of Bookings')
plt.xticks(rotation=45)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```

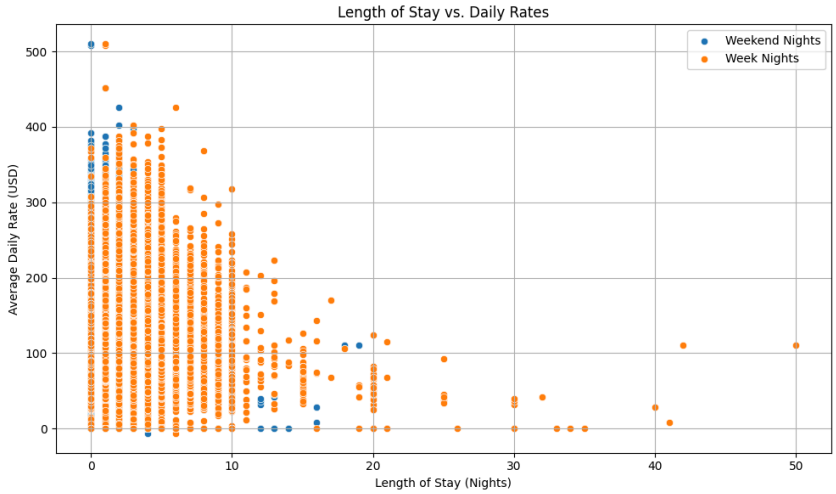



```
data = data[data['is_canceled'] == 0]

correlation = data[['stays_in_weekend_nights', 'stays_in_week_nights', 'adr']].corr()

print("Correlation Matrix:")
print(correlation)
```

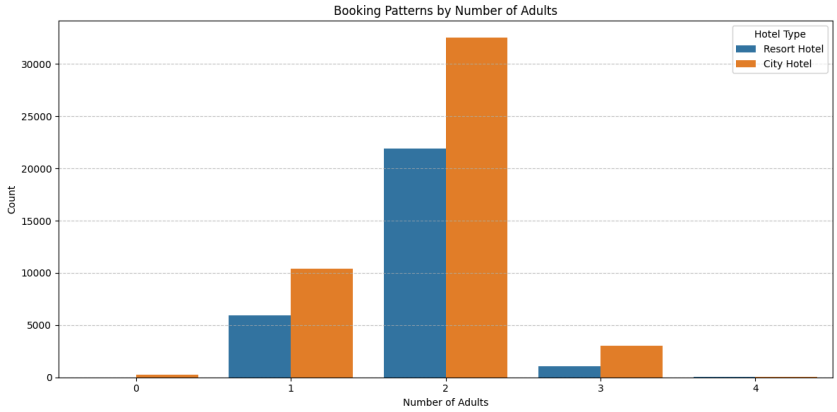
```
Correlation Matrix:
                        stays_in_weekend_nights  stays_in_week_nights  \
stays_in_weekend_nights                 1.000000              0.510640
stays_in_week_nights                    0.510640              1.000000
adr                                     0.037395              0.050289
```

```
                                adr
stays_in_weekend_nights  0.037395
stays_in_week_nights     0.050289
adr                      1.000000
```

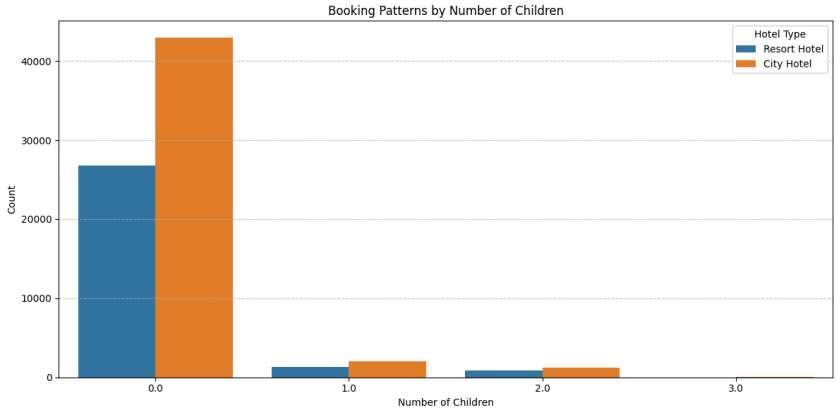
```
import seaborn as sns
plt.figure(figsize=(10, 6))
sns.scatterplot(data=data, x='stays_in_weekend_nights', y='adr', label='Weekend Nights')
sns.scatterplot(data=data, x='stays_in_week_nights', y='adr', label='Week Nights')
plt.title('Length of Stay vs. Daily Rates')
plt.xlabel('Length of Stay (Nights)')
plt.ylabel('Average Daily Rate (USD)')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```



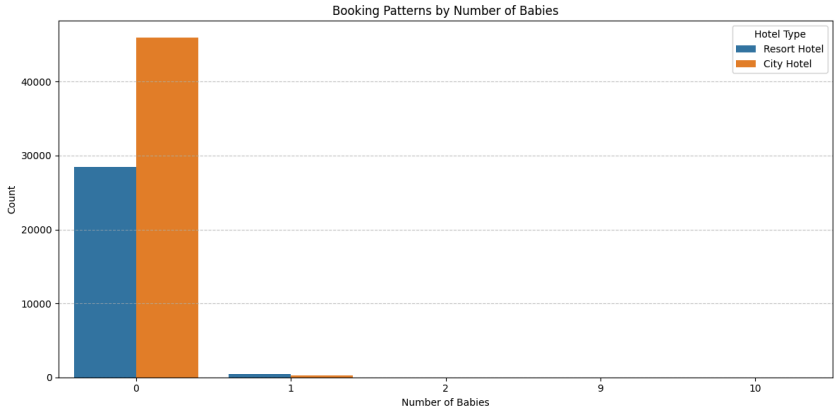
```
plt.figure(figsize=(12, 6))
sns.countplot(data=data, x='adults', hue='hotel')
plt.title('Booking Patterns by Number of Adults')
plt.xlabel('Number of Adults')
plt.ylabel('Count')
plt.legend(title='Hotel Type')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```




```
plt.figure(figsize=(12, 6))
sns.countplot(data=data, x='children', hue='hotel')
plt.title('Booking Patterns by Number of Children')
plt.xlabel('Number of Children')
plt.ylabel('Count')
plt.legend(title='Hotel Type')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```




```
plt.figure(figsize=(12, 6))
sns.countplot(data=data, x='babies', hue='hotel')
plt.title('Booking Patterns by Number of Babies')
plt.xlabel('Number of Babies')
plt.ylabel('Count')
plt.legend(title='Hotel Type')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```




```
print("Data Exploration:")
print("Shape of the dataset:", data.shape)
print("Columns in the dataset:", data.columns)
```

```
Data Exploration:
Shape of the dataset: (75166, 34)
Columns in the dataset: Index(['hotel', 'is_canceled', 'lead_time', 'arrival_date_year
       'arrival_date_month', 'arrival_date_week_number',
       'arrival_date_day_of_month', 'stays_in_weekend_nights',
       'stays_in_week_nights', 'adults', 'children', 'babies', 'meal',
       'country', 'market_segment', 'distribution_channel',
       'is_repeated_guest', 'previous_cancellations',
       'previous_bookings_not_canceled', 'reserved_room_type',
       'assigned_room_type', 'booking_changes', 'deposit_type', 'agent',
       'company', 'days_in_waiting_list', 'customer_type', 'adr',
       'required_car_parking_spaces', 'total_of_special_requests',
       'reservation_status', 'reservation_status_date', 'reservation_month',
       'reservation_year'],
      dtype='object')
```

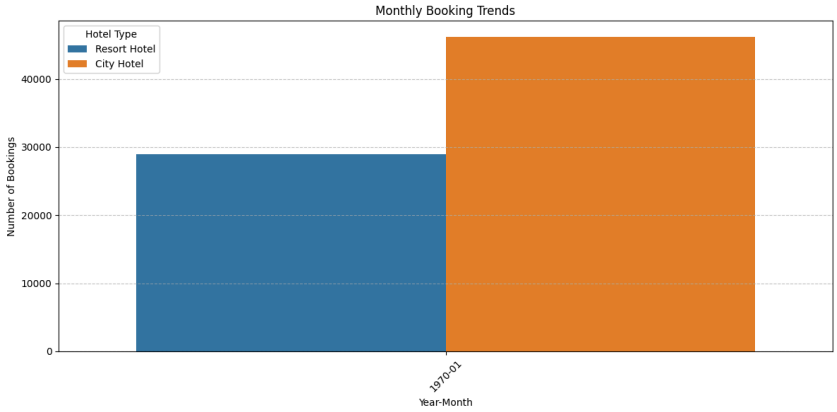
```
print("\nMissing Values:")
print(data.isnull().sum())
```

```
Missing Values:
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                            421
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                            12310
company                          69560
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
reservation_month                    0
reservation_year                     0
dtype: int64
```

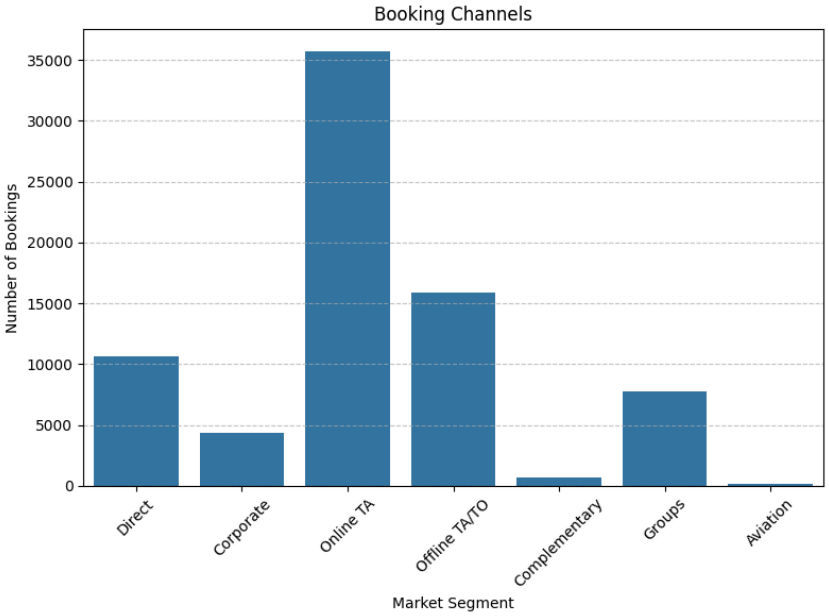
```
try:
    data['arrival_date'] = pd.to_datetime(data['arrival_date_day_of_month'])
    data['arrival_year_month'] = data['arrival_date'].dt.to_period('M')
except KeyError:
    print("Column 'arrival_date' not found in the DataFrame.")
```

```
data['arrival_date'] = pd.to_datetime(data['arrival_date'])
data['arrival_year_month'] = data['arrival_date'].dt.to_period('M')
```

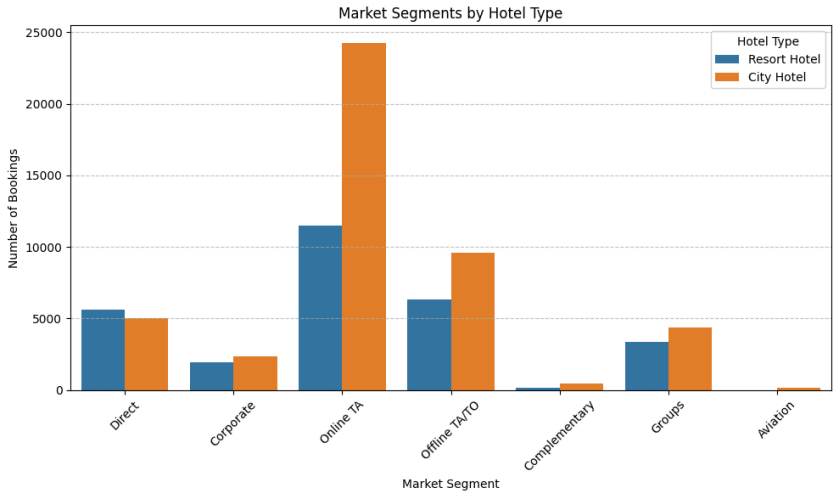
```
plt.figure(figsize=(12, 6))
sns.countplot(data=data, x='arrival_year_month', hue='hotel')
plt.title('Monthly Booking Trends')
plt.xlabel('Year-Month')
plt.ylabel('Number of Bookings')
plt.xticks(rotation=45)
plt.legend(title='Hotel Type')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```



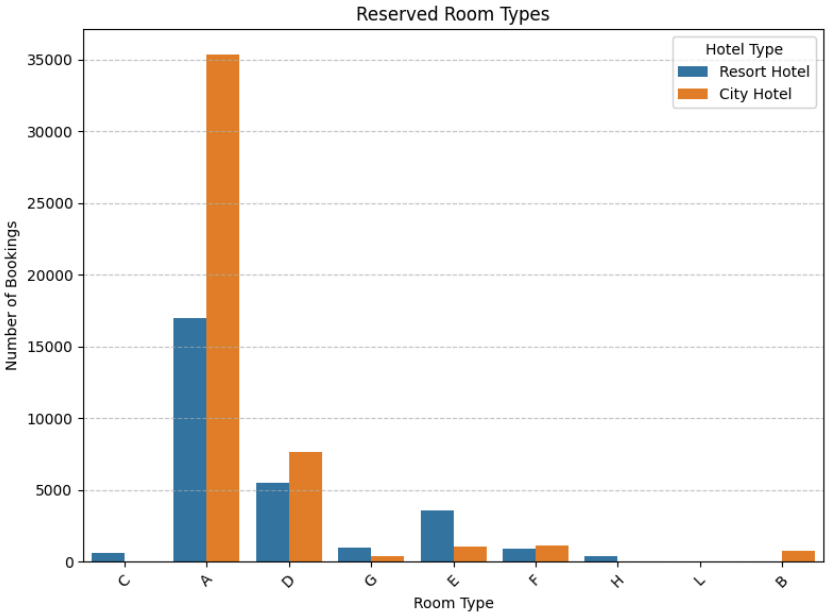

```
plt.figure(figsize=(8, 6))
sns.countplot(data=data, x='market_segment')
plt.title('Booking Channels')
plt.xlabel('Market Segment')
plt.ylabel('Number of Bookings')
plt.xticks(rotation=45)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```



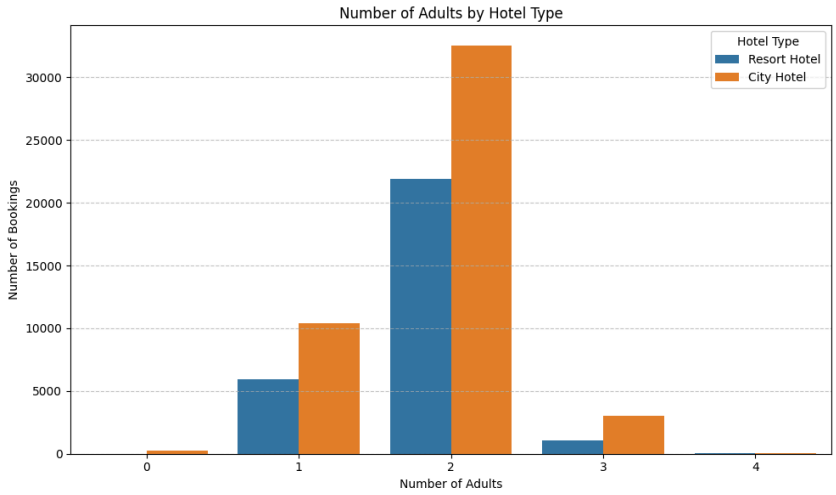

```
plt.figure(figsize=(10, 6))
sns.countplot(data=data, x='market_segment', hue='hotel')
plt.title('Market Segments by Hotel Type')
plt.xlabel('Market Segment')
plt.ylabel('Number of Bookings')
plt.xticks(rotation=45)
plt.legend(title='Hotel Type')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```



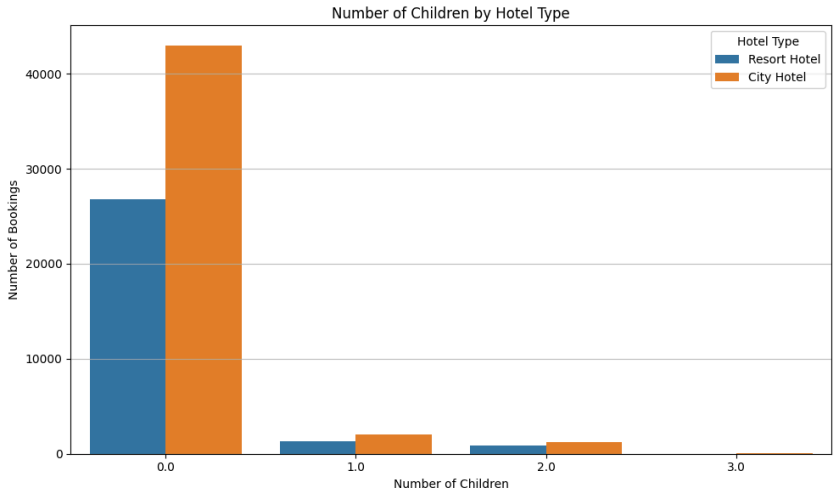

```
plt.figure(figsize=(8, 6))
sns.countplot(data=data, x='reserved_room_type', hue='hotel')
plt.title('Reserved Room Types')
plt.xlabel('Room Type')
plt.ylabel('Number of Bookings')
plt.xticks(rotation=45)
plt.legend(title='Hotel Type')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```




```
plt.figure(figsize=(10, 6))
sns.countplot(data=data, x='adults', hue='hotel')
plt.title('Number of Adults by Hotel Type')
plt.xlabel('Number of Adults')
plt.ylabel('Number of Bookings')
plt.legend(title='Hotel Type')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```




```
plt.figure(figsize=(10, 6))
sns.countplot(data=data, x='children', hue='hotel')
plt.title('Number of Children by Hotel Type')
plt.xlabel('Number of Children')
plt.ylabel('Number of Bookings')
plt.legend(title='Hotel Type')
plt.grid(axis='y', linestyle='-', alpha=0.77)
plt.tight_layout()
plt.show()
```




```
plt.figure(figsize=(10, 6))
sns.countplot(data=data, x='babies', hue='hotel')
plt.title('Number of Babies by Hotel Type')
plt.xlabel('Number of Babies')
import matplotlib.pyplot as plt
plt.ylabel('Number of Bookings')
plt.legend(title='Hotel Type')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```