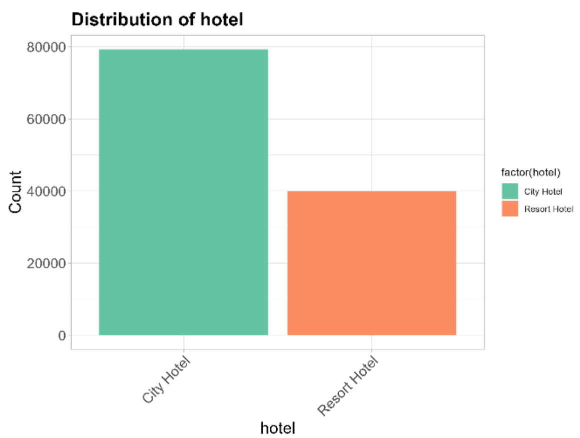
Hotel Reservation Data Report

Mario Getaw

Exploratory Data Analysis

Categorical Variables:

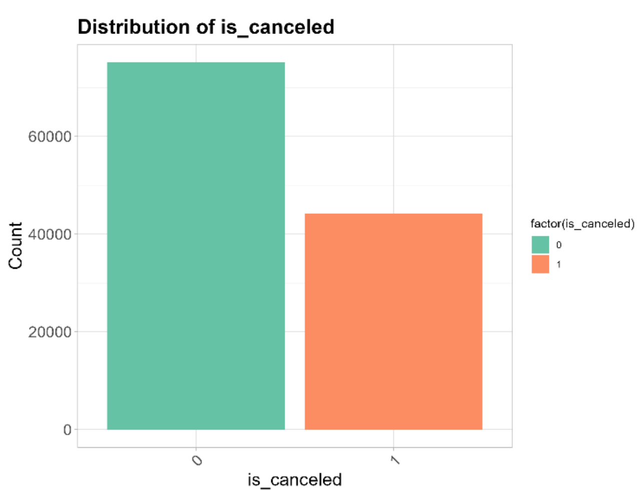
1. Hotel – There are two hotel's booking information in this dataset – a city hotel, and a resort hotel



	Variable	Level	Freq
1	hotel	City Hotel	79330
2	hotel	Resort Hotel	40060

These visuals show the summary of the hotel variable, and the count of each option in the data frame. There are 79,330 accounts of City Hotel, and 40,060 accounts of Resort Hotel.

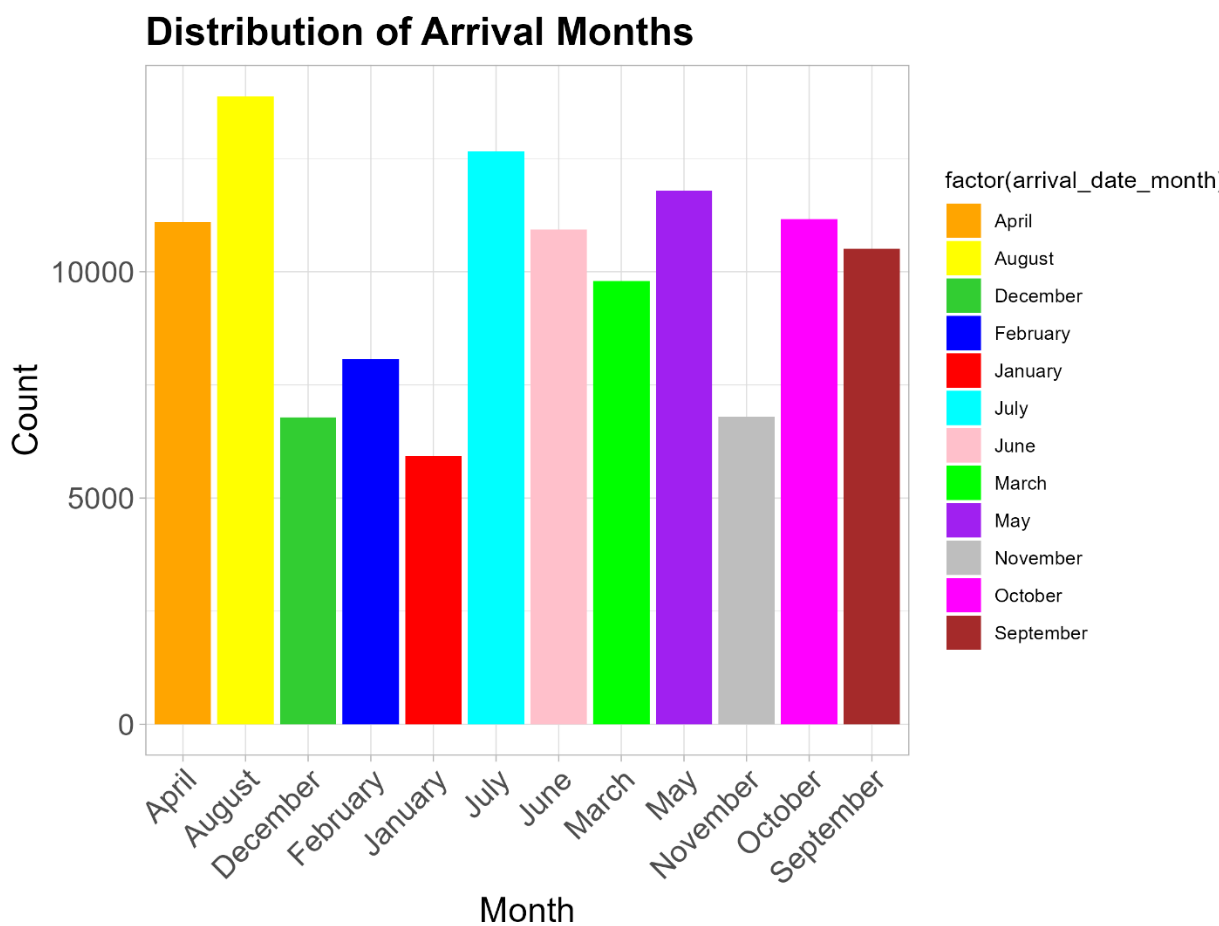
2. is_canceled – Value indicating if the booking was canceled (1) or not (0).



	Status	Count
1	0	75166
2	1	44224

Roughly 37% of the rooms booked in these hotels were cancelled. (1 means they were cancelled)

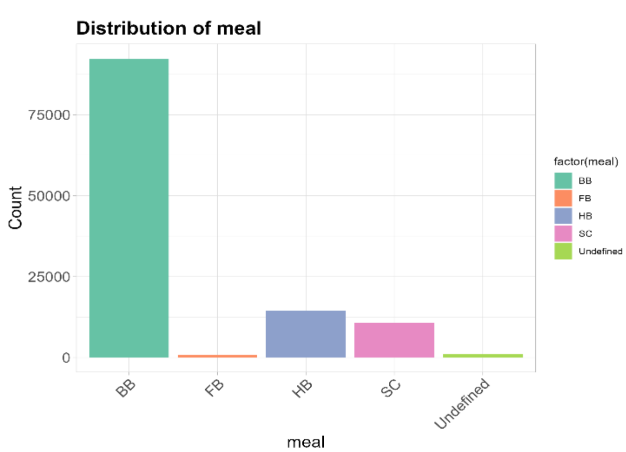
3. arrival_date_month - Month of arrival date with 12 categories: “January” to “December.”



	Month	Count
1	April	11089
2	August	13877
3	December	6780
4	February	8068
5	January	5929
6	July	12661
7	June	10939
8	March	9794
9	May	11791
10	November	6794
11	October	11160
12	September	10508

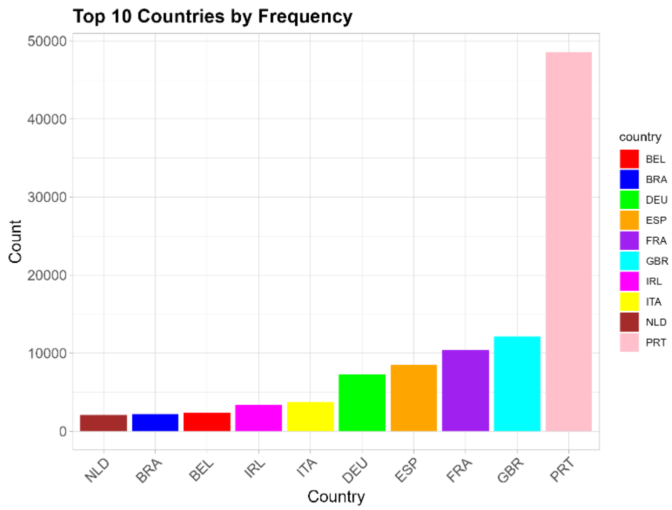
The most popular months to book for are in the summer and fall seasons.

4. Meal - BB – Bed & Breakfast, FB-full board, HB-half board



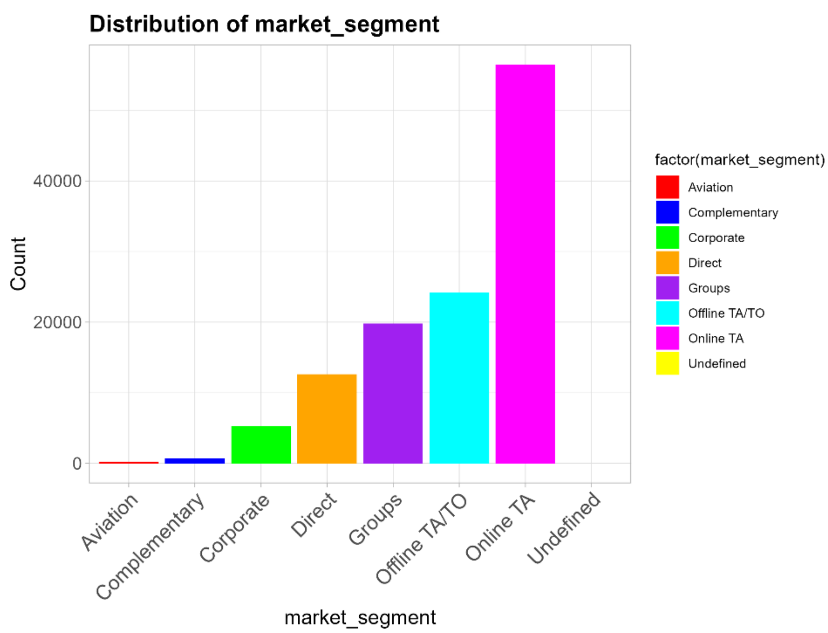
The most popular meal plan is Bed and Breakfast by a wide margin.

5. Country - Country of origin.



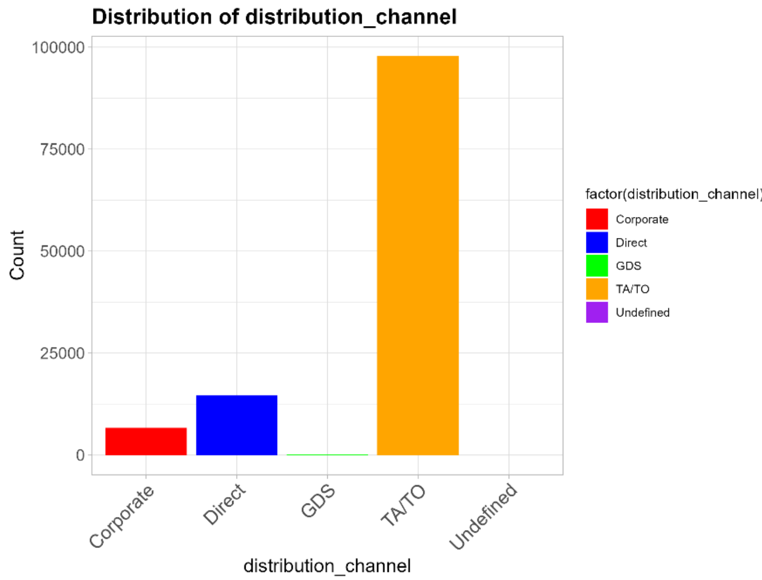
Here are the most frequent countries in the data frame. Portugal has the most bookers by well over 30,000 customers.

6. market_segment - Market segment designation. In categories, the term “TA” means “Travel Agents,” and “TO” means “Tour Operators.”



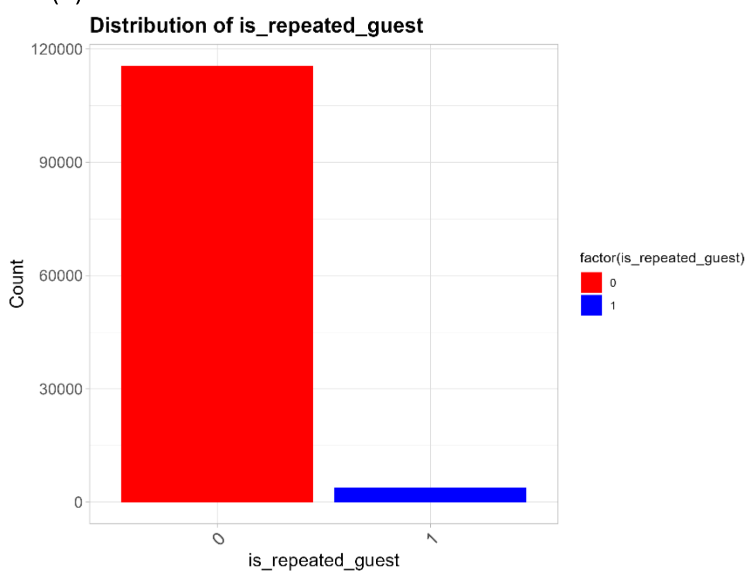
Of the different market segments, we can see that the vast majority of marketing segments is through Online travel agents.

7. distribution_channel - Booking distribution channel. The term “TA” means “Travel Agents,” and “TO” means “Tour Operators.”



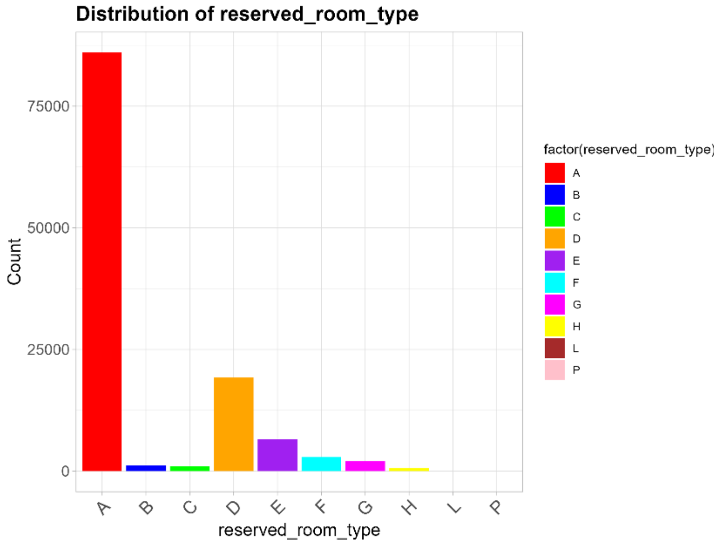
Travel agents and tour operators account for almost all of the booking distribution.

8. is_repeated_guest - Value indicating if the booking name was from a repeated guest (1) or not (0)



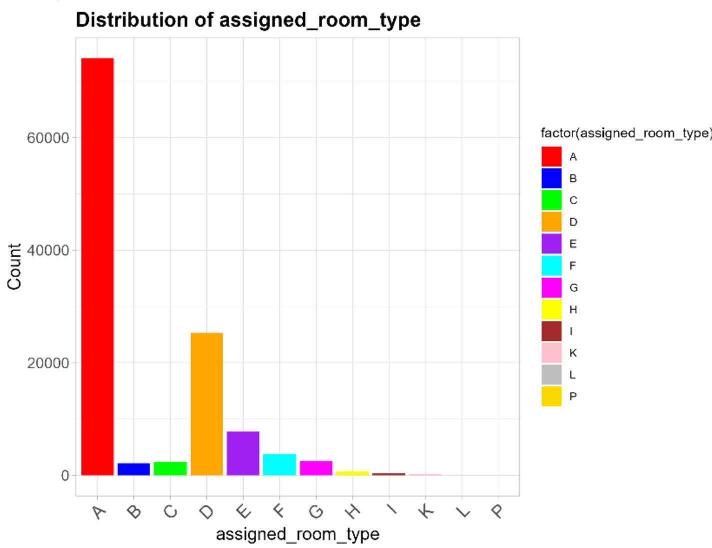
Almost all of the customers from this set of data are not repeated guests. These customers are first timers, for the most part

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



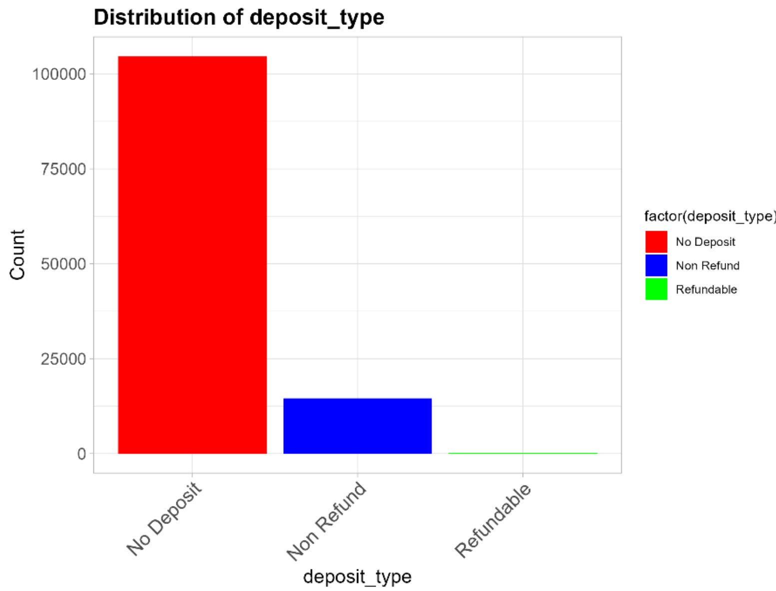
Because of the anonymity in the room types, the only things we can conclude is that A and D are the most popular choices for room type.

10. assigned_room_type - Code for the type of room assigned to the booking. Sometimes the assigned room type differs from the reserved room type.



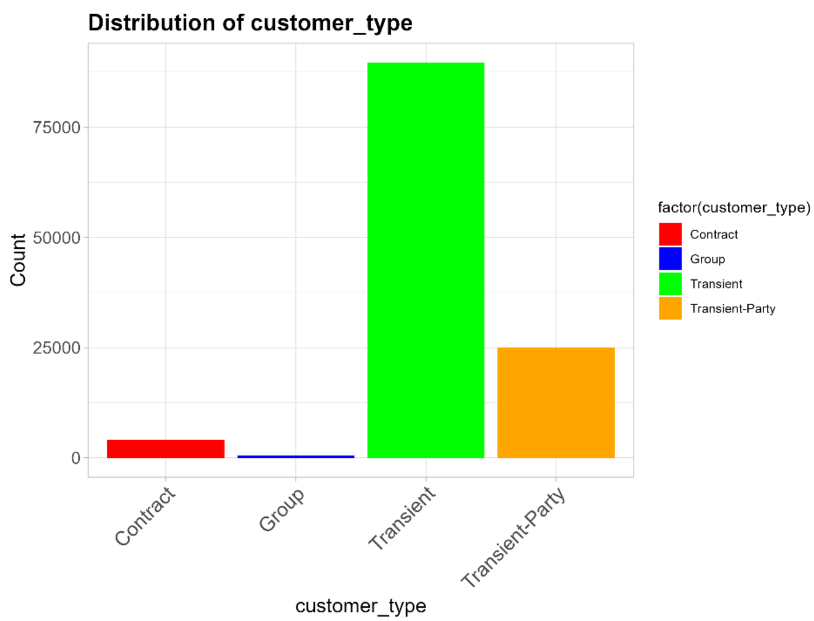
Similar to the reserved room type, types A and D are the most frequently assigned rooms. This means that these are the rooms that they have available the most

11. deposit_type - No Deposit – no deposit was made; Non-Refund – a deposit was made in the value of the total stay cost; Refundable



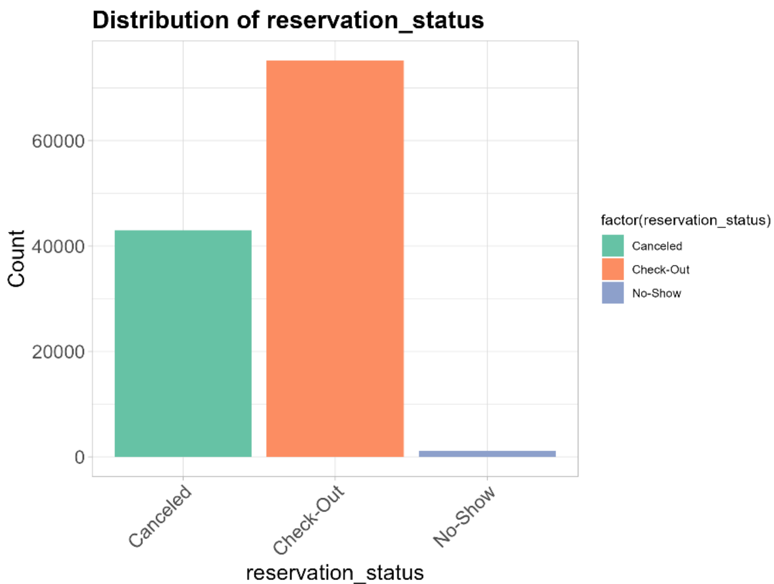
The most common type of deposit type is actually no deposit. This pool of customers typically don't pay ahead of time for their rooms.

12. customer_type - Group – when the booking is associated with a group



Most of the customers are transient customers, meaning that they are booking independently.

13. reservation_status - Check-Out – customer has checked in but already departed; No-Show– the customer did not check in and did inform



Most of the reservations are either checked in and departed (check-out) or cancelled.

Quantitative variables:

1. arrival_date_year - Year of arrival date

```
arrival_date_year
Min.   :2015
1st Qu.:2016
Median :2016
Mean   :2016
3rd Qu.:2017
Max.   :2017
```

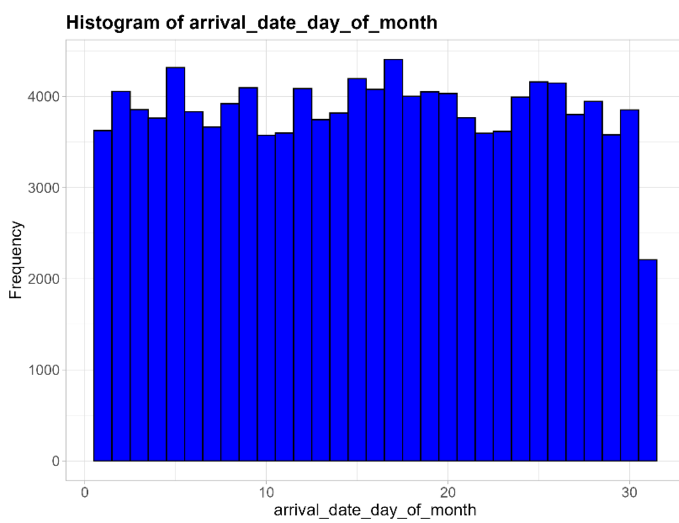
We can see that this data frame contains data from the years 2015-2017, and the average year is 2016.

2. arrival_date_week_number - The week number of the arrival date

```
arrival_date_week_number
Min.   : 1.00
1st Qu.:16.00
Median :28.00
Mean   :27.17
3rd Qu.:38.00
Max.   :53.00
```

The week number varies from 1-53, and the average week is 27.

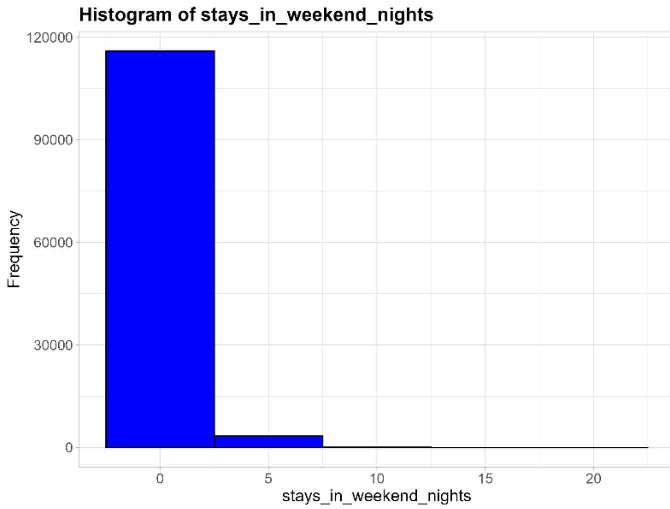
3. arrival_date_day_of_month - Month of arrival date with 12 categories: “January” to “December.”




```
arrival_date_day_of_month
Min.   : 1.0
1st Qu.: 8.0
Median :16.0
Mean   :15.8
3rd Qu.:23.0
Max.   :31.0
```

The days booked are on any day of the month, 1-31. Its easier to see with the histogram that there is no clear outlier in either direction, excluding day 31. That is only because not every month has 31 days in it.

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



	Weekend Nights	Count
1	0	51998
2	1	30626
3	2	33308
4	3	1259
5	4	1855
6	5	79
7	6	153
8	7	19
9	8	60
10	9	11
11	10	7
12	12	5
13	13	3
14	14	2
15	16	3
16	18	1
17	19	1

The histogram and the table both show that the largest group of vacationers does not stay on weekends. 51,998 is the number of vacationers in that group. Most other vacationers stay for one two weekend days. A combined 63,934 one and two weekend night travelers. In relation to the amount of customers in the data frame, there is a small amount that stay for more than 2 weekend days.

5. stays_in_week_nights - Number of weeknights (Monday to Friday) the guest stayed or booked to stay at the hotel BO and BL/Calculated by counting

	Week Nights	Count
1	0	7645
2	1	30310
3	2	33684
4	3	22258
5	4	9563
6	5	11077
7	6	1499
8	7	1029
9	8	656
10	9	231
11	10	1036
12	11	56
13	12	42
14	13	27
15	14	35
16	15	85
17	16	16
18	17	4
19	18	6
20	19	44
21	20	41
22	21	15
23	22	7
24	24	3
25	25	6
26	26	1
27	30	5
28	32	1

```
stays_in_week_nights
Min.   : 0.0
1st Qu.: 1.0
Median : 2.0
Mean   : 2.5
3rd Qu.: 3.0
Max.   :50.0
```

the values from 29-35 were excluded from this table because there were less than 3 instances of each amount in the entire data set. Most customers stay for 10 weeknights or less. The average is 2.5 days.

6. adults - Number of adults

Adults	Count
0	403
1	23027
2	89680
3	6202
4	62
5	2
6	1
10	1
20	2
26	5
27	2
40	1
50	1
55	1

```
     adults
 Min.   : 0.000
 1st Qu.: 2.000
 Median : 2.000
 Mean   : 1.856
 3rd Qu.: 2.000
 Max.   :55.000
```

The average number of adults staying in rooms is just under two – 1.8. the majority of visitors have one to three adults. There are a small number of larger parties though with higher adult counts.

7. children - Number of children

```
    children
 Min.   : 0.0000
 1st Qu.: 0.0000
 Median : 0.0000
 Mean   : 0.1039
 3rd Qu.: 0.0000
 Max.   :10.0000
```

	Number of Children	Count
1	0	110796
2	1	4861
3	2	3652
4	3	76
5	10	1

The average number of children in this data set is just barely over 0. Most of these travelers are not traveling with children.

8. babies - Number of babies

```
    babies
Min.   : 0.000000
1st Qu.: 0.000000
Median : 0.000000
Mean   : 0.007949
3rd Qu.: 0.000000
Max.   :10.000000
```

	Number of Babies	Count
1	0	118473
2	1	900
3	2	15
4	9	1
5	10	1

Nearly 100% of these travelers are traveling without a baby. This why the average is well under 1.

9. previous_cancellations - Number of previous bookings that the customer canceled prior to the current booking

Cancellations	Count
0	112906
1	6051
2	116
3	65
4	31
5	19
6	22
11	35
13	12
14	14
19	19
21	1
24	48
25	25
26	26

```
previous_cancellations
Min.   : 0.00000
1st Qu.: 0.00000
Median : 0.00000
Mean   : 0.08712
3rd Qu.: 0.00000
Max.   :26.00000
```

The large majority of customers are not cancelling their bookings. That is why the average is less than 1.

10. previous_bookings_not_canceled - Number of previous bookings not canceled by the customer prior to the current booking

11. booking_changes - Number of changes/amendments made to the booking from the moment the booking was entered on the PMS

```
   Booking Changes  Count
1                0 101314
2                1  12701
3                2   3805
4                3    927
5                4    376
6                5    118
7                6     63
8                7     31
9                8     17
10               9      8
11              10      6
12              11      2
13              12      2
14              13      5
15              14      5
16              15      3
17              16      2
18              17      2
19              18      1
20              20      1
21              21      1
```

```
booking_changes
Min.   : 0.0000
1st Qu.: 0.0000
Median : 0.0000
Mean   : 0.2211
3rd Qu.: 0.0000
Max.   :21.0000
```

Most customers are not changing their bookings, and if they do its typically under 5 changes. The average is less than 1, about .22 changes.

12. days_in_waiting_list - Number of days the booking was on the waiting list before it was confirmed to the customer

```
days_in_waiting_list
Min.   :  0.000
1st Qu.:  0.000
Median :  0.000
Mean   :  2.321
3rd Qu.:  0.000
Max.   :391.000
```

This variable has a very wide range of occurrences. These customers have spent Anywhere from 0-391 days on the waiting list; however, most customers are not ever on the waiting list, and the average reflects that – being only 2.3 days.

13. adr (average daily rate) - Calculated by dividing the sum of all lodging transactions by the total number of staying nights

```
      adr
Min.   :  -6.38
1st Qu.:  69.29
Median :  94.58
Mean   : 101.83
3rd Qu.: 126.00
Max.   :5400.00
```

The ADR also sees a wide range of values. A minimum value of $-6.38 indicates that a customer was refunded to some extent, and the maximum ADR a client paid was $5400. The average is $101.83 a day.

14. required_car_parking_spaces - Number of car parking spaces required by the customer

```
required_car_parking_spaces
Min.   :0.00000
1st Qu.:0.00000
Median :0.00000
Mean   :0.06252
3rd Qu.:0.00000
Max.   :8.00000
```

	Required Parking Spaces	Count
1	0	111974
2	1	7383
3	2	28
4	3	3
5	8	2

Most customers don't require any parking spaces. If they need any, it's typically only one spot. The average is less than .1 per customer.

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

```
total_of_special_requests
Min.   :0.0000
1st Qu.:0.0000
Median :0.0000
Mean   :0.5714
3rd Qu.:1.0000
Max.   :5.0000
```

	Number of Special Requests	Count
1	0	70318
2	1	33226
3	2	12969
4	3	2497
5	4	340
6	5	40

Most customers are not making special requests, but there is still a large group of customers that is making 1-3 requests a trip. Very few are making more than that. In this data frame, the range is 1-5 requests. The average is .5714 per customer.

Data Analysis

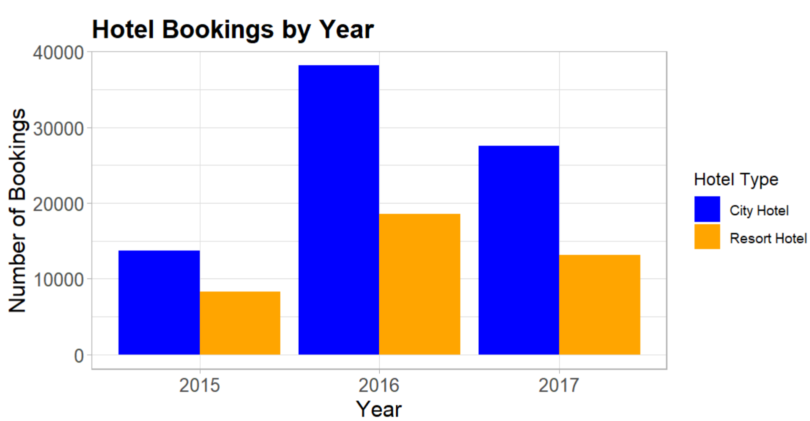
(visualize if possible and explain/discuss each of the following results)

Variable “hotel” (that means city hotel and resort hotel)

1. Hotel – done in EDA
2. Previous_cancellations and previous_bookings_not_cancelled with the variable hotel

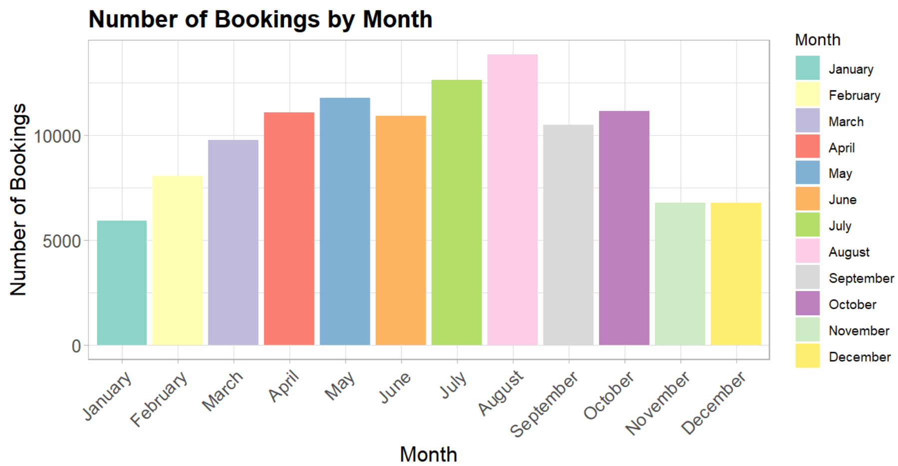
City hotel has almost double the total bookings than Resort hotel, but in the case of both hotels, customers seem to not cancel bookings more often than they do cancel bookings, prior to finally coming in to stay. In relative terms to the number of bookings each hotel has, a higher percentage of Resort hotel customers have previous cancellations than they do at City Hotel.

3. Hotel Bookings by Year



Both hotels had their most bookings in 2016, and their least bookings in 2015.

4. Check the number of bookings by month



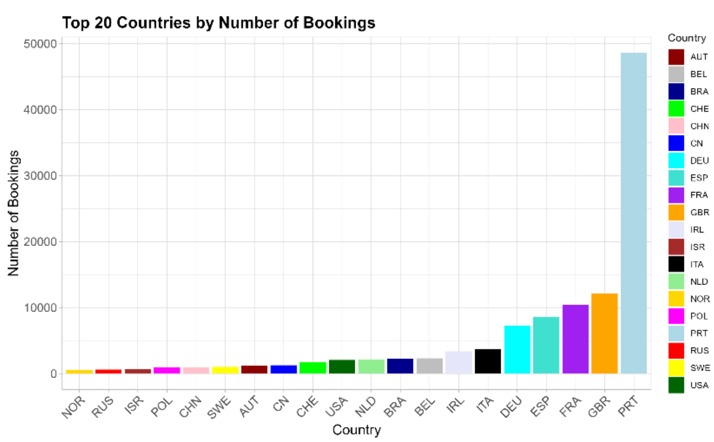
The most popular months to book are in late spring and late summer. There are not many booking in the winter months of November-February.

5. Find out the unique countries

```
Number of unique countries: 178
"PRT" "GBR" "USA" "ESP" "IRL" "FRA" NA    "ROU" "NOR" "OMN" "ARG" "POL" "DEU" "BEL" "CHE" "CN"  "GRC" "ITA"
"NLD" "DNK" "RUS" "SWE" "AUS" "EST" "CZE" "BRA" "FIN" "MOZ" "BWA" "LUX" "SVN" "ALB" "IND" "CHN" "MEX" "MAR"
"UKR" "SMR" "LVA" "PRI" "SRB" "CHL" "AUT" "BLR" "LTU" "TUR" "ZAF" "AGO" "ISR" "CYM" "ZMB" "CPV" "ZWE" "DZA"
"KOR" "CRI" "HUN" "ARE" "TUN" "JAM" "HRV" "HKG" "IRN" "GEO" "AND" "GIB" "URY" "JEY" "CAF" "CYP" "COL" "GGY"
"KWT" "NGA" "MDV" "VEN" "SVK" "FJI" "KAZ" "PAK" "IDN" "LBN" "PHL" "SEN" "SYC" "AZE" "BHR" "NZL" "THA" "DOM"
"MKD" "MYS" "ARM" "JPN" "LKA" "CUB" "CMR" "BIH" "MUS" "COM" "SUR" "UGA" "BGR" "CIV" "JOR" "SYR" "SGP" "BDI"
"SAU" "VNM" "PLW" "QAT" "EGY" "PER" "MLT" "MWI" "ECU" "MDG" "ISL" "UZB" "NPL" "BHS" "MAC" "TGO" "TWN" "DJI"
"STP" "KNA" "ETH" "IRQ" "HND" "RWA" "KHM" "MCO" "BGD" "IMN" "TJK" "NIC" "BEN" "VGB" "TZA" "GAB" "GHA" "TMP"
"GLP" "KEN" "LIE" "GNB" "MNE" "UMI" "MYT" "FRO" "MMR" "PAN" "BFA" "LBY" "MLI" "NAM" "BOL" "PRY" "BRB" "ABW"
"AIA" "SLV" "DMA" "PYF" "GUY" "LCA" "ATA" "GTM" "ASM" "MRT" "NCL" "KIR" "SDN" "ATF" "SLE" "LAO"
```

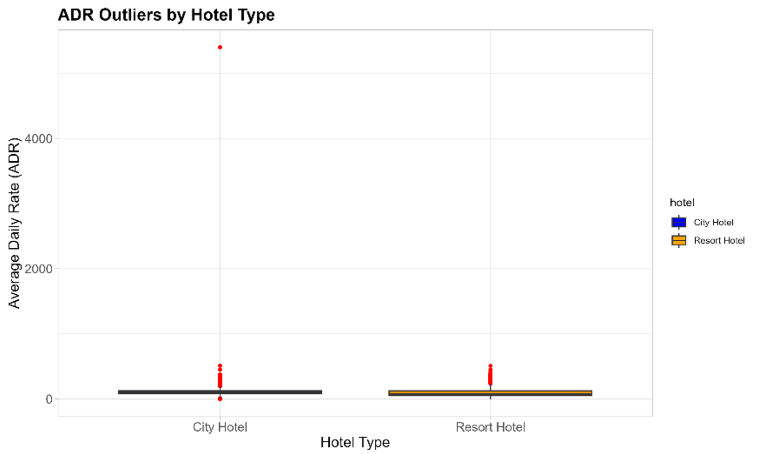
Listed in the screenshot are all 178 unique countries that appear in the dataset.

6. Number of bookings based on countries



I shortened the list to the top 20 countries, because 178 countries would be too many to look at the same time. We can conclude that These hotels are most popular to the people of Portugal.

7. Check outliers for average daily rate(adr) based on hotel types



	hotel	Total_Outliers	Min_ADR	Max_ADR	Avg_ADR
1	City Hotel	3387	0.0	5400	140.1254
2	Resort Hotel	1344	237.6	508	272.2067

The box plot and the table both represent only the outliers of the ADR variable of data frame. City hotel has over double the outliers than resort does, but City just having more overall data does play a role in that as well. City Hotel also has the biggest outlier in ADR, with an average cost of $5400 in one customer's case.

8. Check the average daily rate (adr) vs hotel.

	hotel	Mean_ADR
1	City Hotel	105.30447
2	Resort Hotel	94.95293

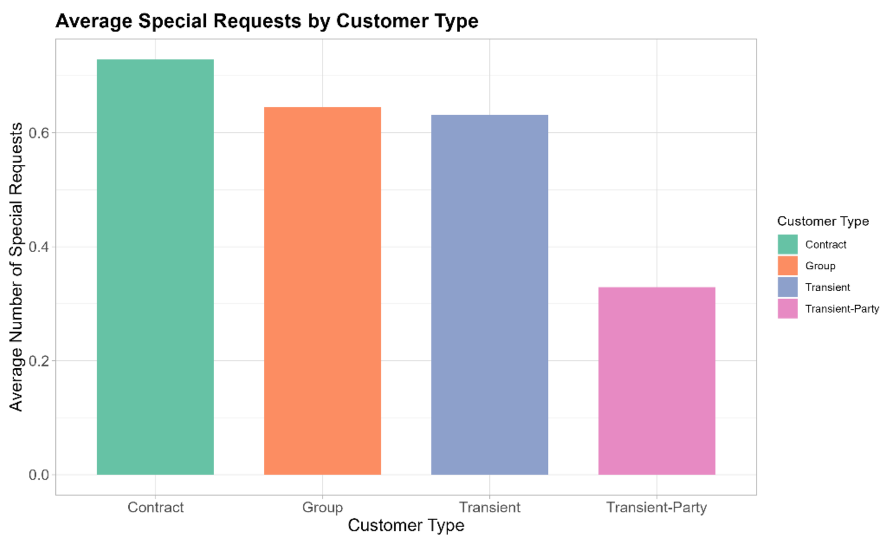
The average ADR for each hotel is very similar between both hotels, only a difference in about 10$ a day.

9. Customer type vs hotel type

	hotel	customer_type	Count
1	City Hotel	Contract	2300
2	City Hotel	Group	293
3	City Hotel	Transient	59404
4	City Hotel	Transient-Party	17333
5	Resort Hotel	Contract	1776
6	Resort Hotel	Group	284
7	Resort Hotel	Transient	30209
8	Resort Hotel	Transient-Party	7791

Based on the bar chart visual and the table, we can see that transient customer types are the most popular for both City and Resort Hotel. Transient-party is the second most prevalent customer type for both hotels. Contracts and groups aren't nearly as common as the transient customers.

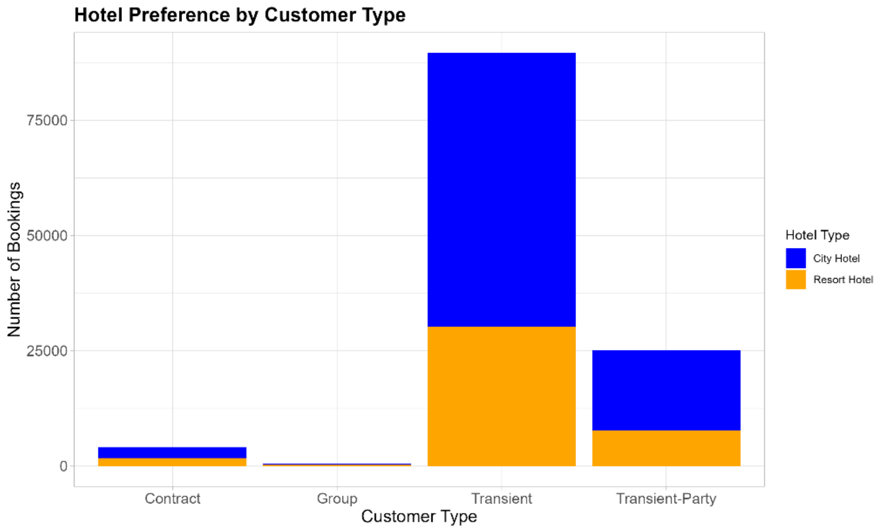
10. Customer type vs special request



	customer_type	Mean_Requests	Median_Requests	Max_Requests	Total_Bookings
1	Contract	0.7289009	0	5	4076
2	Group	0.6447140	0	5	577
3	Transient	0.6315825	0	5	89613
4	Transient-Party	0.3293265	0	5	25124

Based on the table and the bar chart, the highest average of special requests is coming from the contract customer type, followed closely by groups and transient customers. Contract and groups being the highest makes sense because contracted customers may not want to come unless they have their special requests granted. Group travelers also have higher chances of making special requests because they are traveling in larger amounts – giving more opportunities for someone to have a special request.

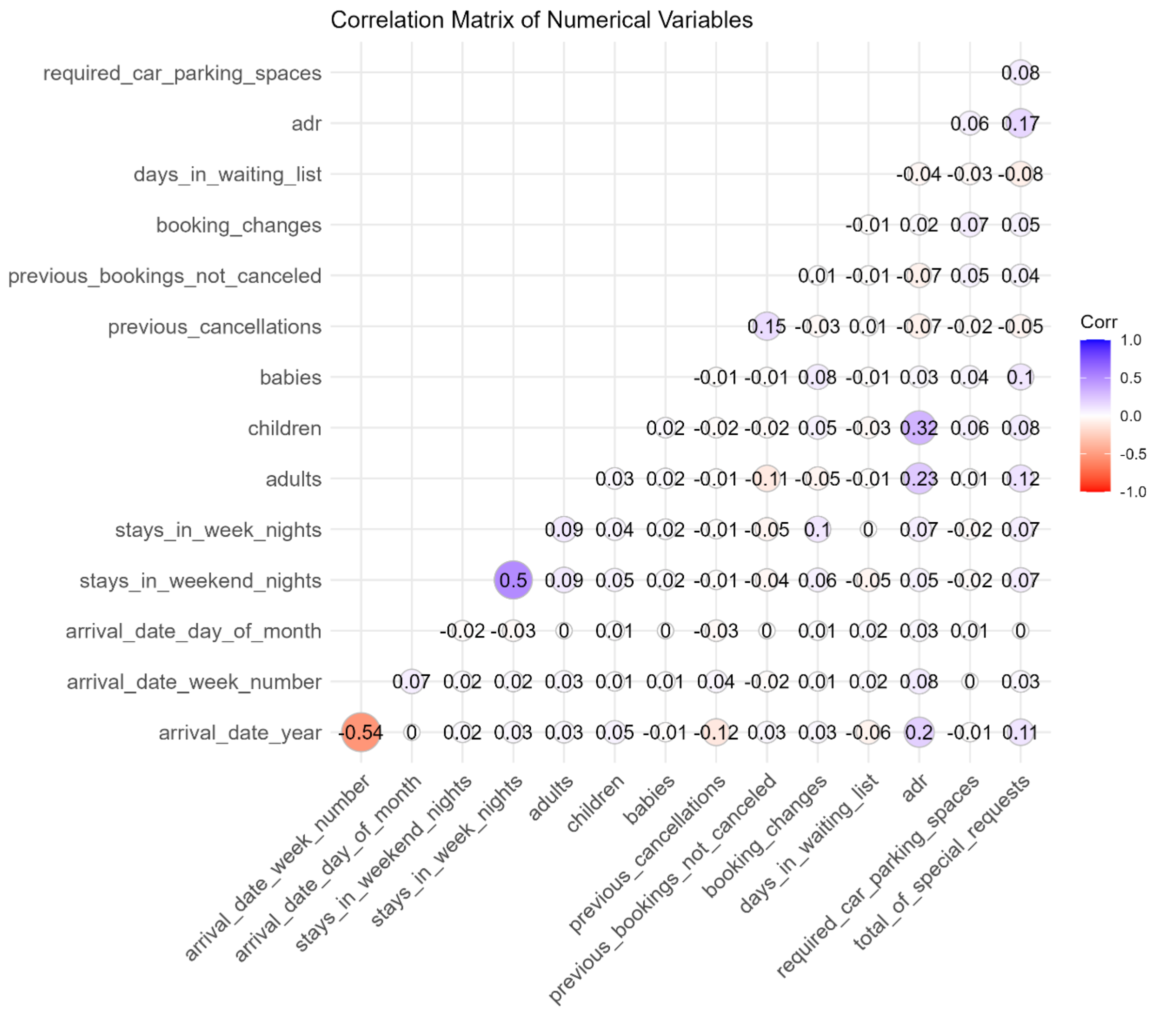
11. Hotel Preference by customer type



	customer_type	hotel	Count
1	Contract	City Hotel	2300
2	Contract	Resort Hotel	1776
3	Group	City Hotel	293
4	Group	Resort Hotel	284
5	Transient	City Hotel	59404
6	Transient	Resort Hotel	30209
7	Transient-Party	City Hotel	17333
8	Transient-Party	Resort Hotel	7791

As established earlier, most customers in this database are transient customers. City Hotel has more transient customers than Resort Hotel. They both have more than any other customer type.

12. Discuss the correlation of the dataset
(If your Microsoft word/pdf reader/browser is in dark mode, the correlation matrix graph is very hard to read).



In general, this graph can be hard to read, but here are the main takeaways:

Strong Correlations

- **stays_in_weekend_nights - stays_in_week_nights**: 0.498
    - There is a moderate positive correlation between the number of weekend and weekday nights, which makes sense because longer stays will most often include both weekdays and weekends.
- **adr ↔ adults**: 0.231
    - There is a weak positive correlation, suggesting that ADR increases slightly with the number of adults, likely due to higher rates for larger bookings as well.

- **adr - children**: 0.325
    - A stronger correlation compared to adults, possibly reflecting additional charges for children in bookings.
- **total_of_special_requests - adr**: 0.172
    - A weak positive correlation, indicating that higher ADR bookings may involve more special requests, potentially reflecting premium services.

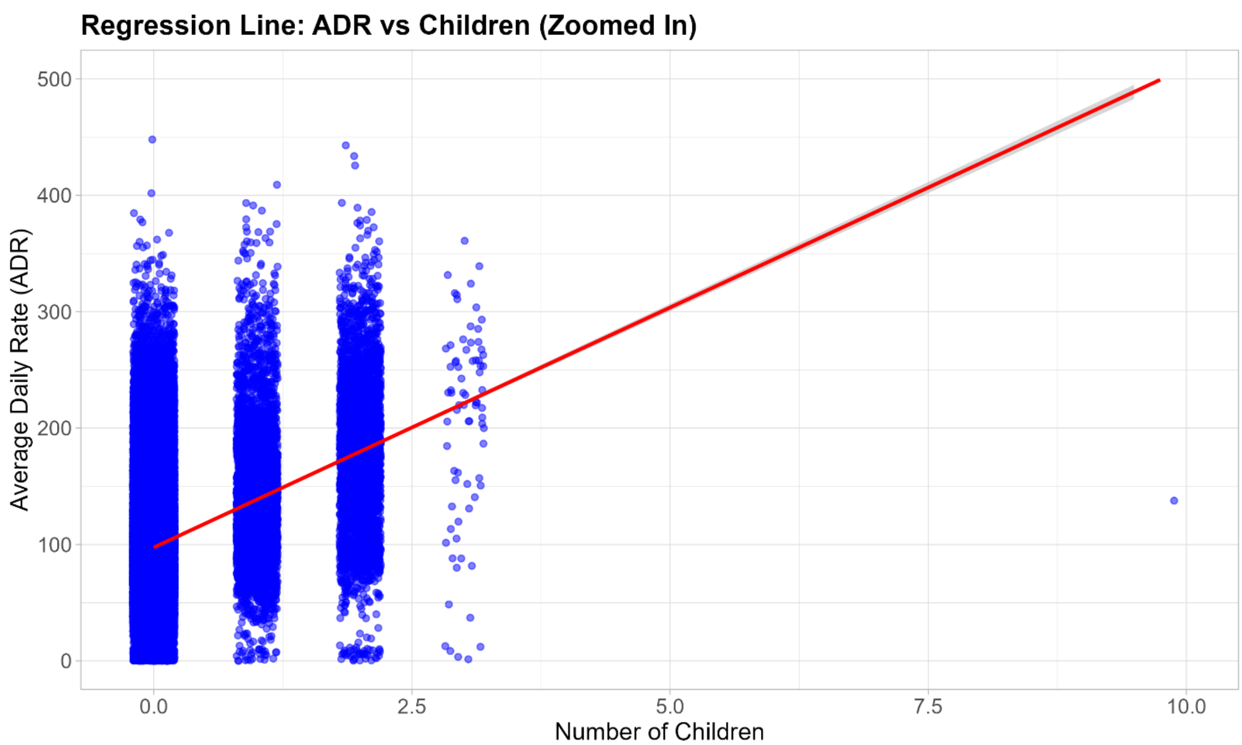
2. Weak or No Correlations

- **arrival_date_day_of_month – All Other Variables**:
    - Correlations are very close to zero, indicating that the specific day of the month has no meaningful relationship with other variables.
- **previous_cancellations - previous_bookings_not_canceled**: 0.153
    - A weak positive correlation, suggesting that customers with more cancellations may also have more non-canceled bookings in the past.

3. Negative Correlations

- **arrival_date_year - arrival_date_week_number**: −0.541
    - A moderate negative correlation, possibly reflecting that the dataset includes bookings distributed unevenly across years and weeks.
- **previous_cancellations - adr**: −0.066
    - A weak negative correlation, indicating that bookings with previous cancellations are slightly associated with lower ADRs
- **days_in_waiting_list - total_of_special_requests**: −0.083
    - There is a weak negative correlation, suggesting that longer waiting times may be associated with fewer special requests.

13. Pick any two variables and fit a regression line> for this situation, you may use the above results in order to select which two variables show the “best” relationship



Note: for visualization purposes, this graph does not include one customer's ADR which was $5400.

We can conclude that there is not a big relationship between these two variables. There ais not a direct correlation between number of children, and ADR.

14. Average daily rate trend over three years

	arrival_date_year	Average_ADR
1	2015	87.18
2	2016	98.33
3	2017	114.64

It is safe to conclude that the average ADR increases each year. 2015-2016 it increases by about 12%, and then from 2016-2017 it increases by about 16%.

Conclusion/Reflection

The EDA results provided valuable insights into the dataset. It was interesting to see the differences in number of bookings between City and Resort Hotel, such as the higher number of bookings for City Hotels and the most frequent bookings occurring during late spring and summer. The correlation analysis highlighted some predictable relationships, like the positive correlation between ADR and the number of children or adults in a booking. On the contrary, it was surprising to find weak or no correlation between certain variables, such as arrival_date_day_of_month and other booking factors. This lack of correlation might be because the day of the month has little impact on the type of customer or their booking habits. Similarly, the weak correlation between previous_cancellations and adr was unexpected, as one might assume that customers with a history of cancellations could be associated with lower ADRs due to last-minute or less reliable bookings. These insights suggest that not all variables have meaningful relationships, possibly due to the dataset's context or how these variables interact in real life.

This study had both strong and weak points. One of the good things was how the analysis covered a wide range of variables, allowing for a deeper understanding of customer behavior and trends like ADR increasing over the years. The use of visualizations made it easier to interpret complex relationships, such as the differences in customer types between the two hotels. However, there were areas for improvement. The dataset had some limitations, like anonymized room types and countries, which restricted the ability to draw more detailed conclusions. If I could redo the study, I would focus on refining the dataset by removing unnecessary variables, gathering more detailed data, and exploring additional factors like customer satisfaction or external influences (e.g., economic trends). From a backend perspective, I would change the way I wrote the code and use more functions. I used them some, but I could have saved myself hundreds of lines of code if I used them in every situation possible. These improvements could provide even more useful insights and a clearer picture of the booking patterns.

Appendix

```
library(tidyverse)

library(ggplot2)

library(dplyr)

#install.packages("readr")

library(readr)

library(tidyr)

#install.packages("ggcorrplot")

library(ggcorrplot)



# on my school computer

hotel_data <- read_csv("//bengal2/dfs/home/getawm/Downloads/hotel_booking.csv")


# on my laptop

hotel_data <- read_csv("C://Users/dante/Documents/Data229_Local_Work/hotel_booking.csv")



# Exploratory Data Analysis

str(hotel_data) # Show each column's data type


# Categorical Variable EDA


# Correct code to summarize each categorical variable

hotel_data$is_canceled <- as.factor(hotel_data$is_canceled) #is_cancelled was not being
accounted for
```

```
#as a categorical. This line makes sure it is
categorical_vars <- hotel_data[, sapply(hotel_data, function(x) is.factor(x) || is.character(x))]

# Apply summary to each column individually
cat_var_summaries <- lapply(categorical_vars, function(var) summary(as.factor(var)))
cat_var_summaries

#Put the categorical Variable data in a table
cat_var_summaries_table <- lapply(categorical_vars, table)
cat_var_summaries_table

# Put the categorical variable summary table into a more readable data
# frame to read for certain variable summaries
cat_var_summary_df <- do.call(
 rbind,
 lapply(names(cat_var_summaries), function(var) {
  # Convert to a data frame
  data.frame(
   Variable = var,
   Level = names(cat_var_summaries[[var]]),
   Freq = as.numeric(cat_var_summaries[[var]]),
   stringsAsFactors = FALSE
  )
 })
)

# Put variables results in data frames to easily see
# their values. This will be used alongside the plots in
# the report. I didn't do hotel because it was the first one
```

```
# in the whole categorical variable frame and i just screen snipped

# Organized tables for necessary variables
# is_cancelled
is_canceled_table <- as.data.frame(table(hotel_data$is_canceled))
colnames(is_canceled_table) <- c("Status", "Count")

# arrival_date_month
arrival_date_month_table <- as.data.frame(table(hotel_data$arrival_date_month))
colnames(arrival_date_month_table) <- c("Month", "Count")

# Here below I Group the categorical variables to lighten the load on R
# (My application crashed everytime I tried to loop through
# all the categoricals at once)

# Each loop saves each plot to the current working directory with ggsave()

# Group 1: Hotel and Booking Details
group_1 <- c("hotel", "is_canceled", "reservation_status", "meal", "reservation_status")
# I'm considering the variable "arrival_date_month" as a group 1 variable, but
# for plotting purposes it will be separate from the group 1
# for loop to avoid issues with the colors

for (var in group_1) {
  p <- ggplot(hotel_data, aes_string(x = paste0("factor(", var, ")"), fill = paste0("factor(", var, ")"))) +
    geom_bar() +
    labs(
      title = paste("Distribution of", var),
      x = var,
```

```
    y = "Count"
   ) +
   theme_light() +  # Use a light theme for better visibility
   theme(
    axis.text.x = element_text(size = 14, angle = 45, hjust = 1),  # Larger, rotated x-axis labels
    axis.text.y = element_text(size = 14),  # Larger y-axis labels
    axis.title = element_text(size = 16),  # Larger axis titles
    plot.title = element_text(size = 18, face = "bold")  # Larger, bold plot title
   ) +
   scale_fill_brewer(palette = "Set2")  # Use a color palette for the bars

  # Save the plot to the current working directory
  ggsave(filename = paste0("plot_", var, ".png"), plot = p, width = 8, height = 6)
}

# Define a custom palette with named colors for the months
month_colors <- c(
  "January" = "red",
  "February" = "blue",
  "March" = "green",
  "April" = "orange",
  "May" = "purple",
  "June" = "pink",
  "July" = "cyan",
  "August" = "yellow",
  "September" = "brown",
  "October" = "magenta",
  "November" = "gray",
  "December" = "limegreen"
```

```
)

# Plot `arrival_date_month`
p <- ggplot(hotel_data, aes(x = factor(arrival_date_month), fill = factor(arrival_date_month))) +
  geom_bar() +
  labs(
    title = "Distribution of Arrival Months",
    x = "Month",
    y = "Count"
  ) +
  theme_light() +  # Use a light theme for better visibility
  theme(
    axis.text.x = element_text(size = 14, angle = 45, hjust = 1),  # Larger, rotated x-axis labels
    axis.text.y = element_text(size = 14),  # Larger y-axis labels
    axis.title = element_text(size = 16),  # Larger axis titles
    plot.title = element_text(size = 18, face = "bold")  # Larger, bold plot title
  ) +
  scale_fill_manual(values = month_colors)  # Use custom colors

# Save
ggsave(filename = "plot_arrival_date_month.png", plot = p, width = 8, height = 6)

# The country variable has too many options to graph or put in
# a table, so I will grab the most frequent countries and plot

# Count and sort countries by frequency
top_countries <- hotel_data %>%
  count(country, sort = TRUE) %>%
  top_n(10, n)  # Select top 10 countries
```

```r
# Define a custom color palette with 10 distinct colors
custom_colors <- c(
  "red", "blue", "green", "orange", "purple",
  "cyan", "magenta", "yellow", "brown", "pink"
)

# Create the plot for the top countries
p <- ggplot(top_countries, aes(x = reorder(country, n), y = n, fill = country)) +
  geom_bar(stat = "identity") +
  labs(
    title = "Top 10 Countries by Frequency",
    x = "Country",
    y = "Count"
  ) +
  theme_light() +
  theme(
    axis.text.x = element_text(size = 12, angle = 45, hjust = 1),  # Rotate x-axis labels
    axis.text.y = element_text(size = 12),
    axis.title = element_text(size = 14),
    plot.title = element_text(size = 16, face = "bold")
  ) +
  scale_fill_manual(values = custom_colors)  # Use custom colors

# Save the plot to a file
ggsave(filename = "top_countries_plot.png", plot = p, width = 8, height = 6)
```

```r
# Group 2: Customer Demographics
group_2 <- c("market_segment", "distribution_channel", "is_repeated_guest")

# Define custom colors (can be reused across all variables)
custom_colors <- c(
  "red", "blue", "green", "orange", "purple",
  "cyan", "magenta", "yellow", "brown", "pink"
)

for (var in group_2) {
  p <- ggplot(hotel_data, aes_string(x = paste0("factor(", var, ")"), fill = paste0("factor(", var, ")"))) +
    geom_bar() +
    labs(
      title = paste("Distribution of", var),
      x = var,
      y = "Count"
    ) +
    theme_light() +
    theme(
      axis.text.x = element_text(size = 14, angle = 45, hjust = 1),  # Rotate x-axis labels
      axis.text.y = element_text(size = 12),  # Larger y-axis labels
      axis.title = element_text(size = 14),  # Larger axis titles
      plot.title = element_text(size = 16, face = "bold")  # Larger, bold plot title
    ) +
    scale_fill_manual(values = custom_colors)  # Custom colors

  # Save the plot to the current working directory
  ggsave(filename = paste0("plot_", var, ".png"), plot = p, width = 8, height = 6)
}
```

```
# Group 3: Booking Specifics
group_3 <- c("reserved_room_type", "assigned_room_type", "deposit_type", "customer_type")

# Custom Colors
custom_colors <- c(
  "red", "blue", "green", "orange", "purple",
  "cyan", "magenta", "yellow", "brown", "pink", "gray",
  "gold"
)

for (var in group_3) {
  p <- ggplot(hotel_data, aes_string(x = paste0("factor(", var, ")"), fill = paste0("factor(", var, ")"))) +
    geom_bar() +
    labs(
      title = paste("Distribution of", var),
      x = var,
      y = "Count"
    ) +
    theme_light() +
    theme(
      axis.text.x = element_text(size = 14, angle = 45, hjust = 1),  # Rotate x-axis labels
      axis.text.y = element_text(size = 12),
      axis.title = element_text(size = 14),
      plot.title = element_text(size = 16, face = "bold")
    ) +
    scale_fill_manual(values = custom_colors)
```

```
  # Save the plot to the current working directory
  ggsave(filename = paste0("plot_", var, ".png"), plot = p, width = 8, height = 6)
}
```

```
# Quantitative Variables
quantitative_vars <- hotel_data[, sapply(hotel_data, is.numeric)]
summary(quantitative_vars)

# Count frequencies of specific values for a quantitative variable
table(hotel_data$arrival_date_year)

# Group quantitative variables to mitigate crashes in R
group_1 <- c("arrival_date_year", "arrival_date_week_number", "arrival_date_day_of_month")
group_2 <- c("stays_in_weekend_nights", "stays_in_week_nights", "adults", "children")
group_3 <- c("babies", "previous_cancellations", "previous_bookings_not_canceled",
"booking_changes")
group_4 <- c("days_in_waiting_list", "adr", "required_car_parking_spaces",
"total_of_special_requests")

# Use a function this time instead of 3 separate for loops for less code
create_boxplots <- function(var_group, data) {
  for (var in var_group) {
    p <- ggplot(data, aes_string(y = var)) +
      geom_boxplot(fill = "blue", color = "black") +
      labs(
```

```
    title = paste("Boxplot of", var),
    y = var
   ) +
   theme_light() +
   theme(
    axis.text.y = element_text(size = 12),
    axis.title = element_text(size = 14),
    plot.title = element_text(size = 16, face = "bold")
   )

  # Save each plot to a file
  ggsave(filename = paste0("boxplot_", var, ".png"), plot = p, width = 8, height = 6)
 }
}

# Apply the function to each group of quantitative variables
create_boxplots(group_1, hotel_data)
create_boxplots(group_2, hotel_data)
create_boxplots(group_3, hotel_data)
create_boxplots(group_4, hotel_data)

# Trying histograms
# Define a function to create histograms for a group of variables
create_histograms <- function(var_group, data) {
 for (var in var_group) {
  # Use a sensible default binwidth; adjust as needed
  bin_width <- ifelse(var %in% c("arrival_date_week_number", "arrival_date_day_of_month"), 1, 5)

  p <- ggplot(data, aes_string(x = var)) +
```

```r
    geom_histogram(binwidth = bin_width, fill = "blue", color = "black") +
    labs(
      title = paste("Histogram of", var),
      x = var,
      y = "Frequency"
    ) +
    theme_light() +
    theme(
      axis.text.x = element_text(size = 12),
      axis.text.y = element_text(size = 12),
      axis.title = element_text(size = 14),
      plot.title = element_text(size = 16, face = "bold")
    )

   # Save each plot to a file
   ggsave(filename = paste0("histogram_", var, ".png"), plot = p, width = 8, height = 6)
  }
}

# Apply the function to each group of quantitative variables
create_histograms(group_1, hotel_data)
create_histograms(group_2, hotel_data)
create_histograms(group_3, hotel_data)
create_histograms(group_4, hotel_data)

# Create a frequency table for stays_in_weekend_nights
freq_table_weekendnights <- as.data.frame(table(hotel_data$stays_in_weekend_nights))
# Rename the columns for clarity
colnames(freq_table_weekendnights) <- c("Weekend Nights", "Count")
```

```
print(freq_table_weekendnights)

# Frequency table for stays_in_week_nights
freq_table_weeknights <- as.data.frame(table(hotel_data$stays_in_week_nights))
# Rename the columns for clarity
colnames(freq_table_weeknights) <- c("Week Nights", "Count")
print(freq_table_weeknights)

# Frequency table for number of adults
freq_table_adults <- as.data.frame(table(hotel_data$adults))
colnames(freq_table_adults) <- c("Number of Adults", "Count")
print(freq_table_adults)

# Frequency table for children
freq_table_children <- as.data.frame(table(hotel_data$children))
colnames(freq_table_children) <- c("Number of Children", "Count")
print(freq_table_children)

# Frequency table for babies
freq_table_babies <- as.data.frame(table(hotel_data$babies))
colnames(freq_table_babies) <- c("Number of Babies", "Count")
print(freq_table_babies)

# Frequency table for previous_cancellations
freq_table_previous_cancellations  <- as.data.frame(table(hotel_data$previous_cancellations ))
colnames(freq_table_previous_cancellations ) <- c("Cancellations", "Count")
print(freq_table_previous_cancellations)

# Frequency table for previous_bookings_not_canceled
```

```
freq_table_previous_bookings_not_canceled  <-
as.data.frame(table(hotel_data$previous_bookings_not_canceled))

colnames(freq_table_previous_bookings_not_canceled) <- c("Previous not canceled", "Count")

print(freq_table_previous_bookings_not_canceled)


# Frequency Table for booking changes

freq_table_booking_changes  <- as.data.frame(table(hotel_data$booking_changes))

colnames(freq_table_booking_changes) <- c("Booking Changes", "Count")

print(freq_table_booking_changes)


# Frequency Table for days in waiting list

freq_table_days_in_waiting_list <- as.data.frame(table(hotel_data$days_in_waiting_list))

colnames(freq_table_days_in_waiting_list) <- c("Days in Waiting List", "Count")

print(freq_table_days_in_waiting_list)


# Frequency table for ADR

freq_table_adr <- as.data.frame(table(hotel_data$adr))

colnames(freq_table_adr) <- c("Average Rate", "Count")

print(freq_table_adr)


# Frequency table for required_car_parking_spaces

freq_table_required_car_parking_spaces <-
as.data.frame(table(hotel_data$required_car_parking_spaces))

colnames(freq_table_required_car_parking_spaces) <- c("Required Parking Spaces", "Count")

print(freq_table_required_car_parking_spaces)


# frequency table for special requests

freq_table_total_of_special_requests <-
as.data.frame(table(hotel_data$total_of_special_requests))

colnames(freq_table_total_of_special_requests) <- c("Number of Special Requests", "Count")
```

```
print(freq_table_total_of_special_requests)


# 1/2 -    Previous_cancellations and previous_bookings_not_cancelled with the variable hotel

# Summarize total counts by hotel
total_counts <- hotel_data %>%
  group_by(hotel) %>%
  summarise(
    Total_Previous_Cancellations = sum(previous_cancellations, na.rm = TRUE),
    Total_Previous_Bookings_Not_Canceled = sum(previous_bookings_not_canceled, na.rm = TRUE)
  )

# Convert to long format for visualization
long_counts <- total_counts %>%
  pivot_longer(cols = c("Total_Previous_Cancellations", "Total_Previous_Bookings_Not_Canceled"),
               names_to = "Metric", values_to = "Count")

# Bar plot
ggplot(long_counts, aes(x = hotel, y = Count, fill = Metric)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(
    title = "Total Previous Cancellations and Bookings Not Canceled by Hotel Type",
    x = "Hotel Type",
    y = "Total Count",
    fill = "Metric"
  ) +
  theme_light() +
```

```r
  scale_fill_manual(values = c("Total_Previous_Cancellations" = "red",
"Total_Previous_Bookings_Not_Canceled" = "green"))


# 3 summarize the hotel variable based on year
hotel_by_year <- hotel_data %>%
  group_by(arrival_date_year, hotel) %>%
  summarise(Count = n(), .groups = "drop") %>%
  mutate(Proportion = Count / sum(Count))


ggplot(hotel_by_year, aes(x = factor(arrival_date_year), y = Count, fill = hotel)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(
    title = "Hotel Bookings by Year",
    x = "Year",
    y = "Number of Bookings",
    fill = "Hotel Type"
  ) +
  theme_light() +
  theme(
    axis.text.x = element_text(size = 12),
    axis.text.y = element_text(size = 12),
    axis.title = element_text(size = 14),
    plot.title = element_text(size = 16, face = "bold")
  ) +
  scale_fill_manual(values = c("City Hotel" = "blue", "Resort Hotel" = "orange"))

# 4 Bookings by month
bookings_by_month <- hotel_data %>%
  group_by(arrival_date_month) %>%
```

```
  summarise(Count = n(), .groups = "drop")

print(bookings_by_month)

# Reorder months in the correct order
bookings_by_month <- bookings_by_month %>%
  mutate(arrival_date_month = factor(
    arrival_date_month,
    levels = c("January", "February", "March", "April", "May", "June",
               "July", "August", "September", "October", "November", "December")
  ))


ggplot(bookings_by_month, aes(x = arrival_date_month, y = Count, fill = arrival_date_month)) +
  geom_bar(stat = "identity") +
  labs(
    title = "Number of Bookings by Month",
    x = "Month",
    y = "Number of Bookings",
    fill = "Month"
  ) +
  theme_light() +
  theme(
    axis.text.x = element_text(size = 12, angle = 45, hjust = 1),
    axis.text.y = element_text(size = 12),
    axis.title = element_text(size = 14),
    plot.title = element_text(size = 16, face = "bold")
  ) +
  scale_fill_brewer(palette = "Set3")  # Use a colorful palette
```

```
# 5 find the unique countries
# Find unique countries
unique_countries <- unique(hotel_data$country)

# Print the unique countries
print(unique_countries)
# Count the total number of unique countries
num_unique_countries <- length(unique_countries)

# Print the total count
cat("Number of unique countries:", num_unique_countries, "\n")

# 6 Number of bookings based on countries

# Count bookings by country
bookings_by_country <- hotel_data %>%
  group_by(country) %>%
  summarise(Count = n(), .groups = "drop") %>%
  arrange(desc(Count))

# View the top 10 countries
head(bookings_by_country, 10)

# Count bookings by country and select the top 20
top_20_countries <- hotel_data %>%
  group_by(country) %>%
  summarise(Count = n(), .groups = "drop") %>%
  arrange(desc(Count)) %>%
```

```
  top_n(20, Count)

# Define a custom color palette with 20 distinct colors and randomize the order
set.seed(123)  # Set seed for reproducibility
custom_colors <- sample(c(
  "red", "blue", "green", "orange", "purple",
  "cyan", "magenta", "yellow", "brown", "pink",
  "turquoise", "gold", "darkgreen", "darkblue", "darkred",
  "lightblue", "lightgreen", "lavender", "gray", "black"
))

# Bar plot for top 20 countries
p <- ggplot(top_20_countries, aes(x = reorder(country, Count), y = Count, fill = country)) +
  geom_bar(stat = "identity") +
  labs(
    title = "Top 20 Countries by Number of Bookings",
    x = "Country",
    y = "Number of Bookings",
    fill = "Country"
  ) +
  theme_light() +
  theme(
    axis.text.x = element_text(size = 12, angle = 45, hjust = 1),
    axis.text.y = element_text(size = 12),
    axis.title = element_text(size = 14),
    plot.title = element_text(size = 16, face = "bold")
  ) +
  scale_fill_manual(values = custom_colors)
```

```
# Save the plot to a file
ggsave("top_20_countries_bookings.png", plot = p, width = 10, height = 6)

# 7 check outliers for average daily rate (adr) based on hotel types
# Boxplot of ADR by hotel type
p <- ggplot(hotel_data, aes(x = hotel, y = adr, fill = hotel)) +
  geom_boxplot(outlier.color = "red", outlier.shape = 16) +  # Highlight outliers in red
  labs(
    title = "ADR Outliers by Hotel Type",
    x = "Hotel Type",
    y = "Average Daily Rate (ADR)"
  ) +
  theme_light() +
  theme(
    axis.text.x = element_text(size = 12),
    axis.text.y = element_text(size = 12),
    axis.title = element_text(size = 14),
    plot.title = element_text(size = 16, face = "bold")
  ) +
  scale_fill_manual(values = c("City Hotel" = "blue", "Resort Hotel" = "orange"))

# Save the plot
ggsave("adr_outliers_by_hotel.png", plot = p, width = 10, height = 6)

library(dplyr)

# Summarize ADR statistics by hotel type
adr_summary <- hotel_data %>%
  group_by(hotel) %>%
```

```
  summarise(
    Min = min(adr, na.rm = TRUE),
    Q1 = quantile(adr, 0.25, na.rm = TRUE),
    Median = median(adr, na.rm = TRUE),
    Q3 = quantile(adr, 0.75, na.rm = TRUE),
    Max = max(adr, na.rm = TRUE),
    IQR = IQR(adr, na.rm = TRUE)
  )

# View summary statistics
print(adr_summary)

# Calculate IQR and boundaries for outliers by hotel type
outlier_data <- hotel_data %>%
  group_by(hotel) %>%
  summarise(
    Q1 = quantile(adr, 0.25, na.rm = TRUE),
    Q3 = quantile(adr, 0.75, na.rm = TRUE),
    IQR = Q3 - Q1,
    Lower_Bound = Q1 - 1.5 * IQR,
    Upper_Bound = Q3 + 1.5 * IQR
  ) %>%
  left_join(hotel_data, by = "hotel") %>%
  filter(adr < Lower_Bound | adr > Upper_Bound)

# View outliers
print(outlier_data)
# Summarize outliers by hotel type
outlier_summary <- outlier_data %>%
```

```
  group_by(hotel) %>%
  summarise(
    Total_Outliers = n(),
    Min_ADR = min(adr),
    Max_ADR = max(adr),
    Avg_ADR = mean(adr)
  )

# View summary
print(outlier_summary)
# Convert to a data frame
outlier_summary_df <- as.data.frame(outlier_summary)

# Print the data frame
print(outlier_summary_df)

# 8 Check the average daily rate (adr) vs hotel.
# Summarize ADR by hotel type
adr_vs_hotel_summary <- hotel_data %>%
  group_by(hotel) %>%
  summarise(
    Mean_ADR = mean(adr, na.rm = TRUE),
    Median_ADR = median(adr, na.rm = TRUE),
    Min_ADR = min(adr, na.rm = TRUE),
    Max_ADR = max(adr, na.rm = TRUE),
    Std_Dev_ADR = sd(adr, na.rm = TRUE)
  )

adr_vs_hotel_summary_df <- as.data.frame(adr_vs_hotel_summary)
```

```
# 9 Customer type vs Hotel Type
# Count customer types for each hotel
customer_vs_hotel_summary <- hotel_data %>%
  group_by(hotel, customer_type) %>%
  summarise(Count = n(), .groups = "drop")

# View the summary table
customer_vs_hotel_summary_df <- as.data.frame(customer_vs_hotel_summary)
# Bar chart for customer type vs hotel
ggplot(customer_vs_hotel_summary, aes(x = hotel, y = Count, fill = customer_type)) +
  geom_bar(stat = "identity", position = "stack") +
  labs(
    title = "Customer Type Distribution by Hotel Type",
    x = "Hotel Type",
    y = "Number of Bookings",
    fill = "Customer Type"
  ) +
  theme_light() +
  theme(
    axis.text.x = element_text(size = 12),
    axis.text.y = element_text(size = 12),
    axis.title = element_text(size = 14),
    plot.title = element_text(size = 16, face = "bold")
  ) +
  scale_fill_brewer(palette = "Set3")
ggsave("customer_type_vs_hotel_stacked.png", width = 10, height = 6)

# 10 customer type vs special request
```

```r
requests_summary <- hotel_data %>%
  group_by(customer_type) %>%
  summarise(
    Mean_Requests = mean(total_of_special_requests, na.rm = TRUE),
    Median_Requests = median(total_of_special_requests, na.rm = TRUE),
    Max_Requests = max(total_of_special_requests, na.rm = TRUE),
    Total_Bookings = n()
  )
requests_summary_df <- as.data.frame(requests_summary)


# Bar plot of mean special requests by customer type
ggplot(requests_summary, aes(x = customer_type, y = Mean_Requests, fill = customer_type)) +
  geom_bar(stat = "identity", width = 0.7) +
  labs(
    title = "Average Special Requests by Customer Type",
    x = "Customer Type",
    y = "Average Number of Special Requests",
    fill = "Customer Type"
  ) +
  theme_light() +
  theme(
    axis.text.x = element_text(size = 12),
    axis.text.y = element_text(size = 12),
    axis.title = element_text(size = 14),
    plot.title = element_text(size = 16, face = "bold")
  ) +
  scale_fill_brewer(palette = "Set2")
ggsave("average_special_requests_by_customer_type_barplot.png", width = 10, height = 6)
```

```
# 11. Hotel preference by customer type

hotel_preference_summary <- hotel_data %>%
  group_by(customer_type, hotel) %>%
  summarise(Count = n(), .groups = "drop")

# View the summary
hotel_preference_summary_df <- as.data.frame(hotel_preference_summary)

ggplot(hotel_preference_summary, aes(x = customer_type, y = Count, fill = hotel)) +
  geom_bar(stat = "identity", position = "stack") +
  labs(
    title = "Hotel Preference by Customer Type",
    x = "Customer Type",
    y = "Number of Bookings",
    fill = "Hotel Type"
  ) +
  theme_light() +
  theme(
    axis.text.x = element_text(size = 12),
    axis.text.y = element_text(size = 12),
    axis.title = element_text(size = 14),
    plot.title = element_text(size = 16, face = "bold")
  ) +
  scale_fill_manual(values = c("City Hotel" = "blue", "Resort Hotel" = "orange"))
ggsave("hotel_preference_by_customer_type_stacked.png", width = 10, height = 6)

# 12 discuss the correlation of dataset
# only numericals are chosen for this
```

```r
# Select only numerical variables
numerical_data <- hotel_data %>%
  select(arrival_date_year, arrival_date_week_number, arrival_date_day_of_month,
         stays_in_weekend_nights, stays_in_week_nights, adults, children, babies,
         previous_cancellations, previous_bookings_not_canceled, booking_changes,
         days_in_waiting_list, adr, required_car_parking_spaces, total_of_special_requests)

# Calculate the correlation matrix
correlation_matrix <- cor(numerical_data, use = "complete.obs")

# View the correlation matrix
print(correlation_matrix)



library(ggcorrplot)

# Plot the correlation matrix with enhanced readability
ggcorrplot(correlation_matrix,
           method = "circle",
           type = "lower",
           lab = TRUE,
           title = "Correlation Matrix of Numerical Variables",
           colors = c("red", "white", "blue"),
           lab_size = 4,          # Increase label size
           ggtheme = theme_minimal()) # Use a light, minimal theme



ggsave("correlation_matrix_heatmap_readable.png", width = 10, height = 8)
```

```r
# 13. pick any two variables and fit a regression line (ADR and children)
# Fit the linear regression model
adr_children_model <- lm(adr ~ children, data = hotel_data)

# Summarize the model
summary(adr_children_model)


ggplot(hotel_data, aes(x = children, y = adr)) +
 geom_jitter(alpha = 0.5, color = "blue", width = 0.2, height = 10) +  # Add jitter
 geom_smooth(method = "lm", color = "red", se = TRUE) +  # Regression line
 scale_y_continuous(limits = c(0, 500)) +  # Focus on ADR values between 0 and 500
 labs(
  title = "Regression Line: ADR vs Children (Zoomed In)",
  x = "Number of Children",
  y = "Average Daily Rate (ADR)"
 ) +
 theme_light() +
 theme(
  axis.text.x = element_text(size = 12),
  axis.text.y = element_text(size = 12),
  axis.title = element_text(size = 14),
  plot.title = element_text(size = 16, face = "bold")
 )
```

```
ggsave("adr_vs_children_zoomed.png", width = 10, height = 6)



# 14. Find the average daily rate trend over the three years


# Summarize average ADR by year
adr_trend_df <- hotel_data %>%
  group_by(arrival_date_year) %>%
  summarise(Average_ADR = mean(adr, na.rm = TRUE))
  as.data.frame()
adr_trend_df$Average_ADR <- round(adr_trend_df$Average_ADR, 2)


  # Code needed to complete the PowerPoint



# Compute the correlation between two variables
correlation_value <- cor(hotel_data$stays_in_weekend_nights, hotel_data$stays_in_week_nights,
use = "complete.obs")


# Print the correlation value
print(correlation_value)



scatter_plot <- ggplot(hotel_data, aes(x = stays_in_weekend_nights, y = stays_in_week_nights)) +
  geom_point(alpha = 0.5, color = "blue") +  # Scatter points
  geom_smooth(method = "lm", color = "red", se = TRUE) +
  labs(
    title = "Correlation: Weekend Nights vs Week Nights",
```

```
   x = "Stays in Weekend Nights",
   y = "Stays in Week Nights"
  ) +
  theme_minimal()
ggsave("weekend_vs_week_nights_correlation.png", plot = scatter_plot, width = 8, height = 6)



###

adr_children_plot <- ggplot(hotel_data, aes(x = children, y = adr)) +
  geom_point(alpha = 0.5, color = "blue") +  # Scatter points
  geom_smooth(method = "lm", color = "red", se = TRUE) +
  labs(
   title = "Correlation: ADR vs Children",
   x = "Number of Children",
   y = "Average Daily Rate (ADR)"
  ) +
  theme_minimal()

ggsave("adr_vs_children_correlation.png", plot = adr_children_plot, width = 8, height = 6)


# Filter out the outlier where ADR is $5400
filtered_data <- hotel_data %>% filter(adr < 5400)


adr_children_plot_filtered <- ggplot(filtered_data, aes(x = children, y = adr)) +
  geom_point(alpha = 0.5, color = "blue") +
```

```
 geom_smooth(method = "lm", color = "red", se = TRUE) +
 labs(
  title = "Correlation: ADR vs Children (Outlier Removed)",
  x = "Number of Children",
  y = "Average Daily Rate (ADR)"
 ) +
 theme_minimal()


ggsave("adr_vs_children_correlation_filtered.png", plot = adr_children_plot_filtered, width = 8,
height = 6)

# Find correlation value now without the outlier
correlation_value <- cor(filtered_data$adr, filtered_data$children, use = "complete.obs")
print(correlation_value)


###
adr_cancellations_plot <- ggplot(hotel_data, aes(x = previous_cancellations, y = adr)) +
 geom_point(alpha = 0.5, color = "blue") +
 geom_smooth(method = "lm", color = "red", se = TRUE) +
 labs(
  title = "Correlation: ADR vs Previous Cancellations",
  x = "Previous Cancellations",
  y = "Average Daily Rate (ADR)"
 ) +
 theme_minimal()
```

```
# Save the plot to a file
ggsave("adr_vs_previous_cancellations.png", plot = adr_cancellations_plot, width = 8, height = 6)


# Without the outlier
filtered_data <- hotel_data %>% filter(adr < 5400)


adr_cancellations_plot <- ggplot(filtered_data, aes(x = previous_cancellations, y = adr)) +
 geom_point(alpha = 0.5, color = "blue") +  # Scatter points
 geom_smooth(method = "lm", color = "red", se = TRUE) +  # Regression line
 labs(
  title = "Correlation: ADR vs Previous Cancellations",
  x = "Previous Cancellations",
  y = "Average Daily Rate (ADR)"
 ) +
 theme_minimal()

# Save the plot to a file
ggsave("adr_vs_previous_cancellations_filtered.png", plot = adr_cancellations_plot, width = 8,
height = 6)


###


year_week_correlation_plot <- ggplot(hotel_data, aes(x = arrival_date_year, y =
arrival_date_week_number)) +
 geom_point(alpha = 0.5, color = "blue") +  # Scatter points
 geom_smooth(method = "lm", color = "red", se = TRUE) +  # Regression line
```

```
 labs(
   title = "Correlation: Arrival Year vs Week Number (r = −0.541)",
   x = "Arrival Year",
   y = "Arrival Week Number"
 ) +
 theme_minimal()

# Save the plot to a file
ggsave("arrival_year_vs_week_number_correlation.png", plot = year_week_correlation_plot, width = 8, height = 6)

###

waiting_special_requests_plot <- ggplot(hotel_data, aes(x = days_in_waiting_list, y = total_of_special_requests)) +
 geom_point(alpha = 0.5, color = "blue") +  # Scatter points
 geom_smooth(method = "lm", color = "red", se = TRUE) +  # Regression line
 labs(
   title = "Correlation: Days in Waiting List vs Total of Special Requests (r = −0.083)",
   x = "Days in Waiting List",
   y = "Total of Special Requests"
 ) +
 theme_minimal()

# Save the plot to a file
ggsave("waiting_list_vs_special_requests_correlation.png", plot = waiting_special_requests_plot, width = 8, height = 6)
```