

# TP-output

2025-05-14

## R Markdown

```
#####
# PARTE 01: CONFIGURACIÓN INICIAL Y CARGA DE DATOS
#####
## PROJECT WD
# Determinar el directorio del proyecto correctamente
if (exists(".rs.getProjectDirectory") &&
    is.function(get(".rs.getProjectDirectory"))) {
  # Si estamos en RStudio, usar la función de RStudio
  project_dir <- .rs.getProjectDirectory()
  print(paste("Directorio del proyecto (RStudio):", project_dir))
} else {
  # Si no estamos en RStudio o la función no existe
  current_dir <- getwd()

  # Verificar si estamos dentro de la carpeta 'code'
  if (basename(current_dir) == "code") {
    # Subir un nivel para llegar a la raíz del proyecto
    project_dir <- dirname(current_dir)
    print(paste("Detectada ejecución desde carpeta 'code', subiendo un nivel:", project_dir))
  } else {
    # Asumir que estamos en la raíz del proyecto
    project_dir <- current_dir
    print(paste("Directorio actual (asumido como raíz del proyecto):", project_dir))
  }
}

## [1] "Detectada ejecución desde carpeta 'code', subiendo un nivel: D:/@aleec02/CC216-TP-2025-1"

# Definir las rutas de los directorios (SIEMPRE relativas a la raíz del proyecto)
data_dir <- file.path(project_dir, "data")
code_dir <- file.path(project_dir, "code")

# Verificar que las rutas existen
if (!dir.exists(data_dir)) {
  stop(paste("El directorio de datos no existe:", data_dir))
}
if (!dir.exists(code_dir)) {
  stop(paste("El directorio de código no existe:", code_dir))
}

# Mostrar la información de directorios
cat("Directorio del proyecto:", project_dir, "\n")

## Directorio del proyecto: D:/@aleec02/CC216-TP-2025-1
```

```

cat("Directorio de datos:", data_dir, "\n")

## Directorio de datos: D:/@aleec02/CC216-TP-2025-1/data

cat("Directorio de código:", code_dir, "\n")

## Directorio de código: D:/@aleec02/CC216-TP-2025-1/code

# Ruta al archivo CSV (relativa a la raíz del proyecto)
CSV_original <- file.path(data_dir, "hotel_bookings.csv")
if (!file.exists(CSV_original)) {
  stop(paste("El archivo CSV no existe:", CSV_original, "\nVerificar ruta completa."))
}

## LIBRARIES
if (!require("tidyverse", quietly = TRUE)) install.packages("tidyverse")

## -- Attaching core tidyverse packages ----- tidyverse 2.0.0 --
## v dplyr      1.1.4      v readr      2.1.5
## v forcats    1.0.0      v stringr   1.5.1
## v ggplot2    3.5.2      v tibble    3.2.1
## v lubridate  1.9.4      v tidyr     1.3.1
## v purrr      1.0.4
## -- Conflicts ----- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()     masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

if (!require("naniar", quietly = TRUE)) install.packages("naniar")
if (!require("skimr", quietly = TRUE)) install.packages("skimr")

##
## Adjuntando el paquete: 'skimr'
##
## The following object is masked from 'package:naniar':
##
##   n_complete

if (!require("knitr", quietly = TRUE)) install.packages("knitr")
if (!require("crayon", quietly = TRUE)) install.packages("crayon")

##
## Adjuntando el paquete: 'crayon'
##
## The following object is masked from 'package:ggplot2':
##
##   %+%

if (!require("ggplot2", quietly = TRUE)) install.packages("ggplot2")
if (!require("gridExtra", quietly = TRUE)) install.packages("gridExtra")

##

```

```

## Adjuntando el paquete: 'gridExtra'
##
## The following object is masked from 'package:dplyr':
##
##     combine

## INITIAL SETUP
library(tidyverse)
library(naniar)    # Para análisis de datos faltantes
library(skimr)     # Para resúmenes estadísticos
library(knitr)     # Para tablas bonitas
library(crayon)    # Para colorear la salida en consola
library(ggplot2)   # Para visualizaciones
library(gridExtra) # Para organizar múltiples gráficos

# Definir las rutas de los directorios
data_dir <- file.path(project_dir, "data")
code_dir <- file.path(project_dir, "code")

# Verificar que las rutas existen
if (!dir.exists(data_dir)) {
  stop(red("El directorio de datos no existe:", data_dir))
}
if (!dir.exists(code_dir)) {
  stop(red("El directorio de código no existe:", code_dir))
}

# Mostrar la información de directorios
cat(green("Directorio del proyecto:"), project_dir, "\n")

## Directorio del proyecto: D:/@aleec02/CC216-TP-2025-1
cat(green("Directorio de datos:"), data_dir, "\n")

## Directorio de datos: D:/@aleec02/CC216-TP-2025-1/data
cat(green("Directorio de código:"), code_dir, "\n")

## Directorio de código: D:/@aleec02/CC216-TP-2025-1/code
# Ruta al archivo CSV
CSV_original <- file.path(data_dir, "hotel_bookings.csv")
if (!file.exists(CSV_original)) {
  stop(red("El archivo CSV no existe:", CSV_original))
}

# Cargar los datos
cat(green("Cargando datos desde:"), CSV_original, "\n")

## Cargando datos desde: D:/@aleec02/CC216-TP-2025-1/data/hotel_bookings.csv
hotel_data <- read.csv(CSV_original, header = TRUE, stringsAsFactors = FALSE)

# Inspección inicial
cat(yellow("\n--- DIMENSIONES DEL DATASET ---\n"))

##
## --- DIMENSIONES DEL DATASET ---

```

```

print(dim(hotel_data))

## [1] 119390      32
cat(yellow("\n--- PRIMERAS FILAS DEL DATASET ---\n"))

##
## --- PRIMERAS FILAS DEL DATASET ---
print(head(hotel_data, 5))

##      hotel is_canceled lead_time arrival_date_year arrival_date_month
## 1 Resort Hotel         0      342          2015          July
## 2 Resort Hotel         0      737          2015          July
## 3 Resort Hotel         0        7          2015          July
## 4 Resort Hotel         0       13          2015          July
## 5 Resort Hotel         0       14          2015          July
## arrival_date_week_number arrival_date_day_of_month stays_in_weekend_nights
## 1                27                1                0
## 2                27                1                0
## 3                27                1                0
## 4                27                1                0
## 5                27                1                0
## stays_in_week_nights adults children babies meal country market_segment
## 1                0      2        0      0  BB      PRT      Direct
## 2                0      2        0      0  BB      PRT      Direct
## 3                1      1        0      0  BB      GBR      Direct
## 4                1      1        0      0  BB      GBR      Corporate
## 5                2      2        0      0  BB      GBR      Online TA
## distribution_channel is_repeated_guest previous_cancellations
## 1                Direct                0                0
## 2                Direct                0                0
## 3                Direct                0                0
## 4                Corporate                0                0
## 5                TA/TO                0                0
## previous_bookings_not_canceled reserved_room_type assigned_room_type
## 1                0                C                C
## 2                0                C                C
## 3                0                A                C
## 4                0                A                A
## 5                0                A                A
## booking_changes deposit_type agent company days_in_waiting_list customer_type
## 1                3  No Deposit  NULL  NULL                0  Transient
## 2                4  No Deposit  NULL  NULL                0  Transient
## 3                0  No Deposit  NULL  NULL                0  Transient
## 4                0  No Deposit  304  NULL                0  Transient
## 5                0  No Deposit  240  NULL                0  Transient
## adr required_car_parking_spaces total_of_special_requests reservation_status
## 1      0                0                0      Check-Out
## 2      0                0                0      Check-Out
## 3     75                0                0      Check-Out
## 4     75                0                0      Check-Out
## 5     98                0                1      Check-Out
## reservation_status_date
## 1      2015-07-01

```

```
## 2          2015-07-01
## 3          2015-07-02
## 4          2015-07-02
## 5          2015-07-03
```

```
cat(yellow("\n--- ESTRUCTURA DEL DATASET ---\n"))
```

```
##
## --- ESTRUCTURA DEL DATASET ---
```

```
str(hotel_data)
```

```
## 'data.frame':    119390 obs. of  32 variables:
## $ hotel          : chr  "Resort Hotel" "Resort Hotel" "Resort Hotel" "Resort Hotel"
## $ is_canceled    : int   0 0 0 0 0 0 0 0 1 1 ...
## $ lead_time      : int   342 737 7 13 14 14 0 9 85 75 ...
## $ arrival_date_year : int   2015 2015 2015 2015 2015 2015 2015 2015 2015 2015 ...
## $ arrival_date_month : chr  "July" "July" "July" "July" ...
## $ arrival_date_week_number : int   27 27 27 27 27 27 27 27 27 27 ...
## $ arrival_date_day_of_month : int   1 1 1 1 1 1 1 1 1 1 ...
## $ stays_in_weekend_nights : int   0 0 0 0 0 0 0 0 0 0 ...
## $ stays_in_week_nights : int   0 0 1 1 2 2 2 2 3 3 ...
## $ adults          : int   2 2 1 1 2 2 2 2 2 2 ...
## $ children        : int   0 0 0 0 0 0 0 0 0 0 ...
## $ babies          : int   0 0 0 0 0 0 0 0 0 0 ...
## $ meal            : chr  "BB" "BB" "BB" "BB" ...
## $ country         : chr  "PRT" "PRT" "GBR" "GBR" ...
## $ market_segment : chr  "Direct" "Direct" "Direct" "Corporate" ...
## $ distribution_channel : chr  "Direct" "Direct" "Direct" "Corporate" ...
## $ is_repeated_guest : int   0 0 0 0 0 0 0 0 0 0 ...
## $ previous_cancellations : int   0 0 0 0 0 0 0 0 0 0 ...
## $ previous_bookings_not_canceled : int   0 0 0 0 0 0 0 0 0 0 ...
## $ reserved_room_type : chr  "C" "C" "A" "A" ...
## $ assigned_room_type : chr  "C" "C" "C" "A" ...
## $ booking_changes   : int   3 4 0 0 0 0 0 0 0 0 ...
## $ deposit_type      : chr  "No Deposit" "No Deposit" "No Deposit" "No Deposit" ...
## $ agent             : chr  "NULL" "NULL" "NULL" "304" ...
## $ company           : chr  "NULL" "NULL" "NULL" "NULL" ...
## $ days_in_waiting_list : int   0 0 0 0 0 0 0 0 0 0 ...
## $ customer_type     : chr  "Transient" "Transient" "Transient" "Transient" ...
## $ adr               : num   0 0 75 75 98 ...
## $ required_car_parking_spaces : int   0 0 0 0 0 0 0 0 0 0 ...
## $ total_of_special_requests : int   0 0 0 0 1 1 0 1 1 0 ...
## $ reservation_status : chr  "Check-Out" "Check-Out" "Check-Out" "Check-Out" ...
## $ reservation_status_date : chr  "2015-07-01" "2015-07-01" "2015-07-02" "2015-07-02" ...
```

```
cat(yellow("\n--- RESUMEN ESTADÍSTICO BÁSICO (TODAS LAS COLUMNAS) ---\n"))
```

```
##
## --- RESUMEN ESTADÍSTICO BÁSICO (TODAS LAS COLUMNAS) ---
```

```
print(summary(hotel_data))
```

```
##      hotel          is_canceled      lead_time  arrival_date_year
## Length:119390      Min.   :0.0000      Min.    : 0      Min.   :2015
## Class :character    1st Qu.:0.0000      1st Qu.: 18      1st Qu.:2016
## Mode  :character    Median :0.0000      Median : 69      Median :2016
```

```

##          Mean    :0.3704    Mean    :104    Mean    :2016
##          3rd Qu.:1.0000    3rd Qu.:160    3rd Qu.:2017
##          Max.    :1.0000    Max.    :737    Max.    :2017
##
## arrival_date_month arrival_date_week_number arrival_date_day_of_month
## Length:119390      Min.      : 1.00      Min.      : 1.0
## Class :character   1st Qu.:16.00      1st Qu.: 8.0
## Mode  :character   Median :28.00      Median :16.0
##                   Mean    :27.17      Mean    :15.8
##                   3rd Qu.:38.00      3rd Qu.:23.0
##                   Max.    :53.00      Max.    :31.0
##
## stays_in_weekend_nights stays_in_week_nights adults
## Min.      : 0.0000      Min.      : 0.0      Min.      : 0.000
## 1st Qu.: 0.0000      1st Qu.: 1.0      1st Qu.: 2.000
## Median : 1.0000      Median : 2.0      Median : 2.000
## Mean    : 0.9276      Mean    : 2.5      Mean    : 1.856
## 3rd Qu.: 2.0000      3rd Qu.: 3.0      3rd Qu.: 2.000
## Max.    :19.0000      Max.    :50.0      Max.    :55.000
##
## children babies meal country
## Min.      : 0.0000      Min.      : 0.000000      Length:119390      Length:119390
## 1st Qu.: 0.0000      1st Qu.: 0.000000      Class :character   Class :character
## Median : 0.0000      Median : 0.000000      Mode  :character   Mode  :character
## Mean    : 0.1039      Mean    : 0.007949
## 3rd Qu.: 0.0000      3rd Qu.: 0.000000
## Max.    :10.0000      Max.    :10.000000
## NA's      :4
## market_segment distribution_channel is_repeated_guest
## Length:119390      Length:119390      Min.      :0.00000
## Class :character   Class :character   1st Qu.:0.00000
## Mode  :character   Mode  :character   Median :0.00000
##                   Mean    :0.03191
##                   3rd Qu.:0.00000
##                   Max.    :1.00000
##
## previous_cancellations previous_bookings_not_canceled reserved_room_type
## Min.      : 0.00000      Min.      : 0.0000      Length:119390
## 1st Qu.: 0.00000      1st Qu.: 0.0000      Class :character
## Median : 0.00000      Median : 0.0000      Mode  :character
## Mean    : 0.08712      Mean    : 0.1371
## 3rd Qu.: 0.00000      3rd Qu.: 0.0000
## Max.    :26.00000      Max.    :72.0000
##
## assigned_room_type booking_changes deposit_type agent
## Length:119390      Min.      : 0.0000      Length:119390      Length:119390
## Class :character   1st Qu.: 0.0000      Class :character   Class :character
## Mode  :character   Median : 0.0000      Mode  :character   Mode  :character
##                   Mean    : 0.2211
##                   3rd Qu.: 0.0000
##                   Max.    :21.0000
##
## company days_in_waiting_list customer_type adr
## Length:119390      Min.      : 0.000      Length:119390      Min.      : -6.38

```

```
## Class :character 1st Qu.: 0.000 Class :character 1st Qu.: 69.29
## Mode :character Median : 0.000 Mode :character Median : 94.58
## Mean : 2.321 Mean : 101.83
## 3rd Qu.: 0.000 3rd Qu.: 126.00
## Max. :391.000 Max. :5400.00
##
## required_car_parking_spaces total_of_special_requests reservation_status
## Min. :0.00000 Min. :0.0000 Length:119390
## 1st Qu.:0.00000 1st Qu.:0.0000 Class :character
## Median :0.00000 Median :0.0000 Mode :character
## Mean :0.06252 Mean :0.5714
## 3rd Qu.:0.00000 3rd Qu.:1.0000
## Max. :8.00000 Max. :5.0000
##
## reservation_status_date
## Length:119390
## Class :character
## Mode :character
##
##
##
```

```
# Análisis adicional con skimr para obtener más detalles
cat(yellow("\n--- ANÁLISIS DETALLADO CON SKIMR ---\n"))
```

```
##
## --- ANÁLISIS DETALLADO CON SKIMR ---
```

```
print(skim(hotel_data))
```

```
## -- Data Summary -----
##                               Values
## Name                        hotel_data
## Number of rows              119390
## Number of columns           32
## -----
## Column type frequency:
##   character                  14
##   numeric                    18
## -----
## Group variables            None
##
## -- Variable type: character -----
##   skim_variable  n_missing complete_rate min max empty n_unique
## 1 hotel          0             1 10 12 0 2
## 2 arrival_date_month 0             1 3 9 0 12
## 3 meal             0             1 2 9 0 5
## 4 country          0             1 2 4 0 178
## 5 market_segment   0             1 6 13 0 8
## 6 distribution_channel 0             1 3 9 0 5
## 7 reserved_room_type 0             1 1 1 0 10
## 8 assigned_room_type 0             1 1 1 0 12
## 9 deposit_type      0             1 10 10 0 3
## 10 agent            0             1 1 4 0 334
```

```

## 11 company          0          1  1  4  0      353
## 12 customer_type    0          1  5 15  0        4
## 13 reservation_status 0          1  7  9  0        3
## 14 reservation_status_date 0        1 10 10  0      926
##      whitespace
## 1          0
## 2          0
## 3          0
## 4          0
## 5          0
## 6          0
## 7          0
## 8          0
## 9          0
## 10         0
## 11         0
## 12         0
## 13         0
## 14         0
##
## -- Variable type: numeric -----
##      skim_variable      n_missing complete_rate      mean      sd
## 1 is_canceled          0          1      0.370    0.483
## 2 lead_time            0          1     104.    107.
## 3 arrival_date_year      0          1    2016.    0.707
## 4 arrival_date_week_number 0          1     27.2    13.6
## 5 arrival_date_day_of_month 0          1     15.8     8.78
## 6 stays_in_weekend_nights 0          1     0.928    0.999
## 7 stays_in_week_nights    0          1     2.50     1.91
## 8 adults                0          1     1.86     0.579
## 9 children              4          1.00    0.104    0.399
## 10 babies                0          1     0.00795  0.0974
## 11 is_repeated_guest      0          1     0.0319   0.176
## 12 previous_cancellations 0          1     0.0871   0.844
## 13 previous_bookings_not_canceled 0          1     0.137    1.50
## 14 booking_changes        0          1     0.221    0.652
## 15 days_in_waiting_list   0          1     2.32    17.6
## 16 adr                    0          1    102.    50.5
## 17 required_car_parking_spaces 0          1     0.0625   0.245
## 18 total_of_special_requests 0          1     0.571    0.793
##      p0      p25      p50      p75 p100 hist
## 1  0        0        0        1    1
## 2  0        18       69      160  737
## 3 2015      2016      2016      2017 2017
## 4  1        16       28       38   53
## 5  1         8       16       23   31
## 6  0         0        1        2   19
## 7  0         1        2        3   50
## 8  0         2        2        2   55
## 9  0         0        0        0   10
## 10 0         0        0        0   10
## 11 0         0        0        0    1
## 12 0         0        0        0   26
## 13 0         0        0        0   72

```



```
## 14      0      0      0      0    21
## 15      0      0      0      0   391
## 16  -6.38   69.3   94.6  126 5400
## 17      0      0      0      0     8
## 18      0      0      0      1     5
```

```
# Rutas para guardar los datasets procesados
CSV_limpio <- file.path(data_dir, "hotel_bookings_limpio.csv")
CSV_final <- file.path(data_dir, "hotel_bookings_final.csv")

cat(green("\nRutas para guardar datasets procesados:"), "\n")
```

```
##
## Rutas para guardar datasets procesados:
```

```
cat("Dataset limpio:", CSV_limpio, "\n")
```

```
## Dataset limpio: D:/@aleec02/CC216-TP-2025-1/data/hotel_bookings_limpio.csv
```

```
cat("Dataset final (si es necesario):", CSV_final, "\n")
```

```
## Dataset final (si es necesario): D:/@aleec02/CC216-TP-2025-1/data/hotel_bookings_final.csv
```

```
#####
# PARTE 02: ANÁLISIS DE DATOS FALTANTES Y ATÍPICOS
#####
```

```
cat(green("\n=== RONDA 2: ANÁLISIS DE DATOS FALTANTES Y ATÍPICOS ===\n"))
```

```
##
## === RONDA 2: ANÁLISIS DE DATOS FALTANTES Y ATÍPICOS ===
```

```
#-----
# 2.1. ANÁLISIS DE DATOS FALTANTES (NA)
#-----
cat(yellow("\n--- ANÁLISIS DE DATOS FALTANTES ---\n"))
```

```
##
## --- ANÁLISIS DE DATOS FALTANTES ---
```

```
# Conteo de NA por columna
na_count <- colSums(is.na(hotel_data))
na_percentage <- round(na_count / nrow(hotel_data) * 100, 2)
```

```
na_summary <- data.frame(
  Variable = names(na_count),
  NA_Count = na_count,
  NA_Percentage = na_percentage
)
```

```
# Ordenar por cantidad de NA (descendente)
na_summary <- na_summary[order(-na_summary$NA_Count), ]
```

```
# Mostrar resumen de valores NA
print(na_summary)
```

```
##
## children      Variable NA_Count
## hotel         children      4
##                hotel       0
```

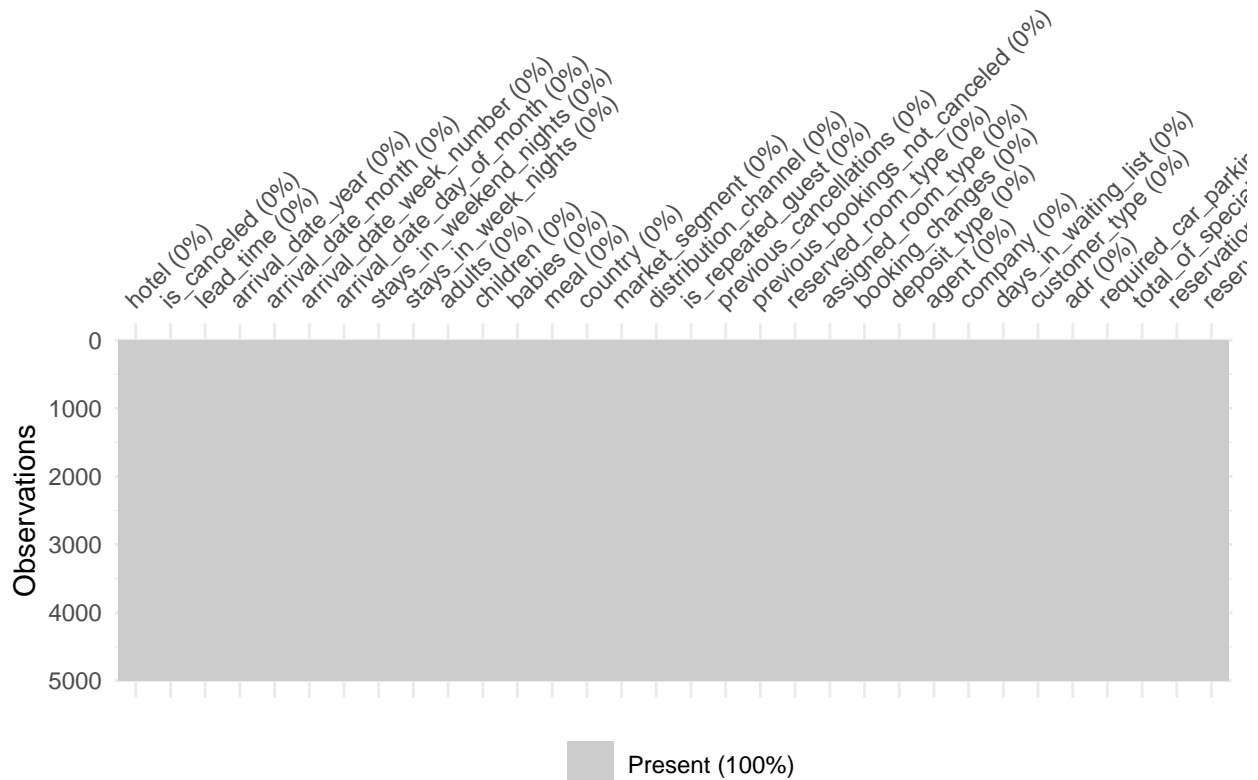
## is_canceled	is_canceled	0
## lead_time	lead_time	0
## arrival_date_year	arrival_date_year	0
## arrival_date_month	arrival_date_month	0
## arrival_date_week_number	arrival_date_week_number	0
## arrival_date_day_of_month	arrival_date_day_of_month	0
## stays_in_weekend_nights	stays_in_weekend_nights	0
## stays_in_week_nights	stays_in_week_nights	0
## adults	adults	0
## babies	babies	0
## meal	meal	0
## country	country	0
## market_segment	market_segment	0
## distribution_channel	distribution_channel	0
## is_repeated_guest	is_repeated_guest	0
## previous_cancellations	previous_cancellations	0
## previous_bookings_not_canceled	previous_bookings_not_canceled	0
## reserved_room_type	reserved_room_type	0
## assigned_room_type	assigned_room_type	0
## booking_changes	booking_changes	0
## deposit_type	deposit_type	0
## agent	agent	0
## company	company	0
## days_in_waiting_list	days_in_waiting_list	0
## customer_type	customer_type	0
## adr	adr	0
## required_car_parking_spaces	required_car_parking_spaces	0
## total_of_special_requests	total_of_special_requests	0
## reservation_status	reservation_status	0
## reservation_status_date	reservation_status_date	0
##	NA_Percentage	
## children	0	
## 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	
## babies	0	
## meal	0	
## country	0	
## market_segment	0	
## distribution_channel	0	
## is_repeated_guest	0	
## previous_cancellations	0	
## previous_bookings_not_canceled	0	
## reserved_room_type	0	
## assigned_room_type	0	
## booking_changes	0	
## deposit_type	0	

```
## agent 0
## company 0
## days_in_waiting_list 0
## customer_type 0
## adr 0
## required_car_parking_spaces 0
## total_of_special_requests 0
## reservation_status 0
## reservation_status_date 0

# Visualización de NA - usando una muestra representativa
set.seed(123) # Para reproducibilidad
muestra_datos <- hotel_data %>%
  slice_sample(n = 5000) # Tomar una muestra de 5000 registros

cat(yellow("\nVisualización de datos faltantes (muestra de 5000 registros):"))

##
## Visualización de datos faltantes (muestra de 5000 registros):
print(vis_miss(muestra_datos))
```



```
# Análisis específico para variable 'children' (la única con NA)
cat(yellow("\nAnálisis específico para la variable 'children':"))
```

```
##
## Análisis específico para la variable 'children':
```

```

cat("\nDistribución de valores no-NA en 'children':\n")

##
## Distribución de valores no-NA en 'children':
print(table(hotel_data$children, useNA = "ifany"))

##
##      0      1      2      3     10  <NA>
## 110796  4861  3652   76     1     4

#-----
# 2.2. ANÁLISIS DE VALORES ATÍPICOS (OUTLIERS)
#-----
cat(yellow("\n--- ANÁLISIS DE VALORES ATÍPICOS ---\n"))

##
## --- ANÁLISIS DE VALORES ATÍPICOS ---
# Análisis estadístico de outliers usando el método IQR para variables clave
cat(yellow("\nAnálisis estadístico de outliers en variables clave:\n"))

##
## Análisis estadístico de outliers en variables clave:
outlier_stats <- function(data, var_name) {
  var <- data[[var_name]]
  var <- var[!is.na(var)]

  Q1 <- quantile(var, 0.25)
  Q3 <- quantile(var, 0.75)
  IQR <- Q3 - Q1

  lower_bound <- Q1 - 1.5 * IQR
  upper_bound <- Q3 + 1.5 * IQR

  outliers <- sum(var < lower_bound | var > upper_bound)
  outlier_percent <- round(outliers / length(var) * 100, 2)

  return(data.frame(
    Variable = var_name,
    Q1 = Q1,
    Q3 = Q3,
    IQR = IQR,
    Lower_Bound = lower_bound,
    Upper_Bound = upper_bound,
    Outlier_Count = outliers,
    Outlier_Percentage = outlier_percent
  ))
}

# Variables numéricas que pueden tener outliers
numeric_vars <- c(
  "lead_time", "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"
)

# Aplicar la función a variables numéricas clave
outlier_summary <- do.call(rbind, lapply(numeric_vars, function(var) {
  outlier_stats(hotel_data, var)
})))

print(outlier_summary)

##           Variable      Q1  Q3   IQR Lower_Bound Upper_Bound
## 25%           lead_time 18.00 160 142.00   -195.000    373.000
## 25%1      stays_in_weekend_nights 0.00  2  2.00    -3.000     5.000
## 25%2      stays_in_week_nights  1.00  3  2.00    -2.000     6.000
## 25%3           adults  2.00  2  0.00     2.000     2.000
## 25%4          children  0.00  0  0.00     0.000     0.000
## 25%5           babies  0.00  0  0.00     0.000     0.000
## 25%6    previous_cancellations  0.00  0  0.00     0.000     0.000
## 25%7 previous_bookings_not_canceled  0.00  0  0.00     0.000     0.000
## 25%8           booking_changes  0.00  0  0.00     0.000     0.000
## 25%9      days_in_waiting_list  0.00  0  0.00     0.000     0.000
## 25%10              adr 69.29 126  56.71   -15.775    211.065
## 25%11 required_car_parking_spaces  0.00  0  0.00     0.000     0.000
## 25%12    total_of_special_requests  0.00  1  1.00    -1.500     2.500
##      Outlier_Count Outlier_Percentage
## 25%              3005              2.52
## 25%1              265              0.22
## 25%2              3354              2.81
## 25%3             29710             24.88
## 25%4              8590              7.20
## 25%5              917              0.77
## 25%6             6484              5.43
## 25%7             3620              3.03
## 25%8            18076             15.14
## 25%9             3698              3.10
## 25%10            3793              3.18
## 25%11            7416              6.21
## 25%12            2877              2.41

# Valores extremos específicos para variables de interés
cat(yellow("\nValores extremos en variables clave:\n"))

##
## Valores extremos en variables clave:

mostrar_extremos <- function(data, var_name, n = 5) {
  cat("\nVariable:", var_name, "\n")
  sorted_values <- sort(data[[var_name]], decreasing = TRUE)
  cat("Top", n, "valores más altos:", head(sorted_values, n), "\n")

  if (min(data[[var_name]], na.rm = TRUE) < 0) {
    cat("Valores negativos:", sort(data[[var_name]][data[[var_name]] < 0]), "\n")
  }
}

```

```

# Variables de particular interés
variables_interes <- c("lead_time", "adults", "adr", "stays_in_week_nights", "stays_in_weekend_nights")

for (var in variables_interes) {
  mostrar_extremos(hotel_data, var)
}

##
## Variable: lead_time
## Top 5 valores más altos: 737 709 629 629 629
##
## Variable: adults
## Top 5 valores más altos: 55 50 40 27 27
##
## Variable: adr
## Top 5 valores más altos: 5400 510 508 451.5 450
## Valores negativos: -6.38
##
## Variable: stays_in_week_nights
## Top 5 valores más altos: 50 42 41 40 40
##
## Variable: stays_in_weekend_nights
## Top 5 valores más altos: 19 18 16 16 16

# Crear histogramas individuales para variables clave
cat(yellow("\nHistogramas para variables clave:"))

##
## Histogramas para variables clave:

crear_histogramas_mejorados <- function(data, variables) {
  for (var in variables) {
    # Para variables con valores extremos, usar zoom
    if(var == "lead_time") {
      p <- ggplot(data, aes(x = .data[[var]])) +
        geom_histogram(bins = 30, fill = "steelblue", color = "black") +
        labs(title = paste("Histograma de", var), x = var, y = "Frecuencia") +
        theme_minimal() +
        xlim(0, 400) # Zoom para ver mejor la distribución principal
    } else if(var == "adults") {
      p <- ggplot(data, aes(x = .data[[var]])) +
        geom_histogram(bins = 30, fill = "steelblue", color = "black") +
        labs(title = paste("Histograma de", var), x = var, y = "Frecuencia") +
        theme_minimal() +
        xlim(0, 5) # Enfocarse en valores razonables
    } else if(var == "adr") {
      p <- ggplot(data, aes(x = .data[[var]])) +
        geom_histogram(bins = 30, fill = "steelblue", color = "black") +
        labs(title = paste("Histograma de", var), x = var, y = "Frecuencia") +
        theme_minimal() +
        xlim(0, 500) # Enfocarse en el rango principal
    } else {
      p <- ggplot(data, aes(x = .data[[var]])) +
        geom_histogram(bins = 30, fill = "steelblue", color = "black") +
        labs(title = paste("Histograma de", var), x = var, y = "Frecuencia") +

```

```

    theme_minimal()
  }
  print(p)
}
}

cat("\nCreando histogramas mejorados para variables clave...\n")

```

```
##
```

```
## Creando histogramas mejorados para variables clave...
```

```
crear_histogramas_mejorados(hotel_data, variables_interes)
```

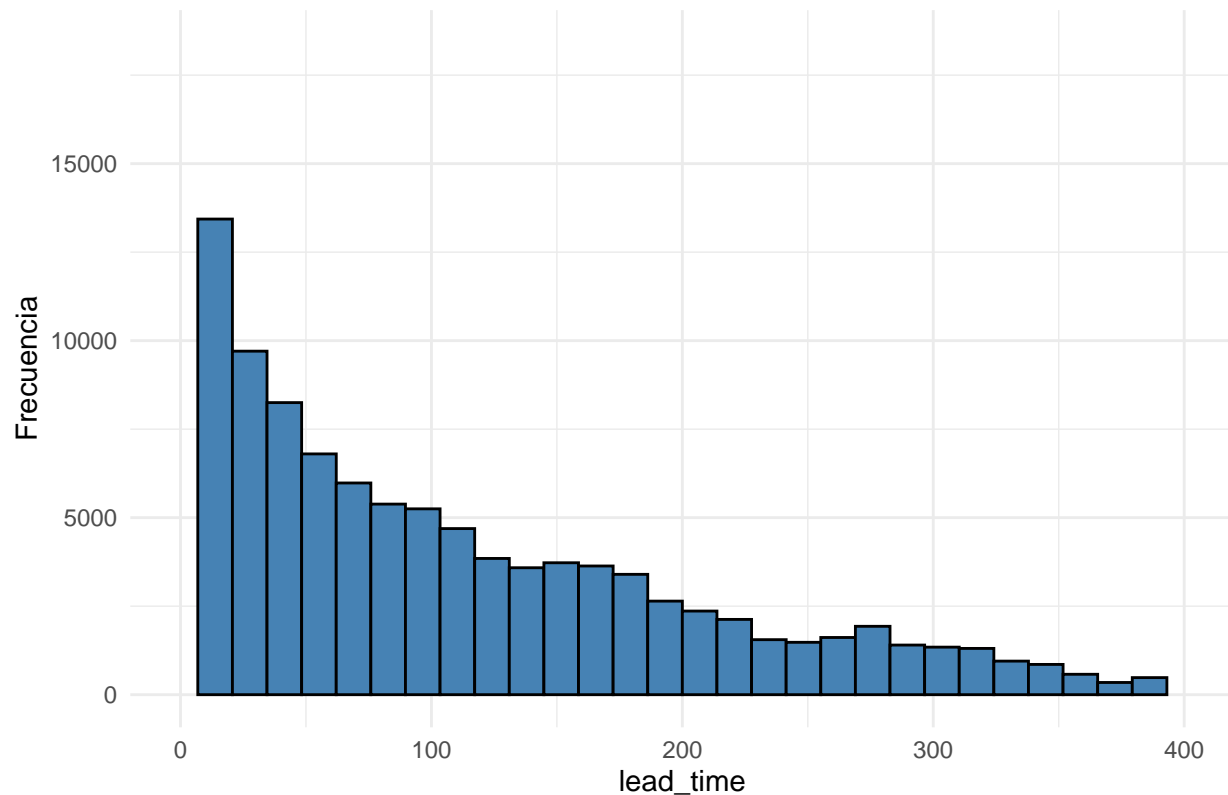
```
## Warning: Removed 2114 rows containing non-finite outside the scale range
```

```
## (`stat_bin()`).
```

```
## Warning: Removed 2 rows containing missing values or values outside the scale range
```

```
## (`geom_bar()`).
```

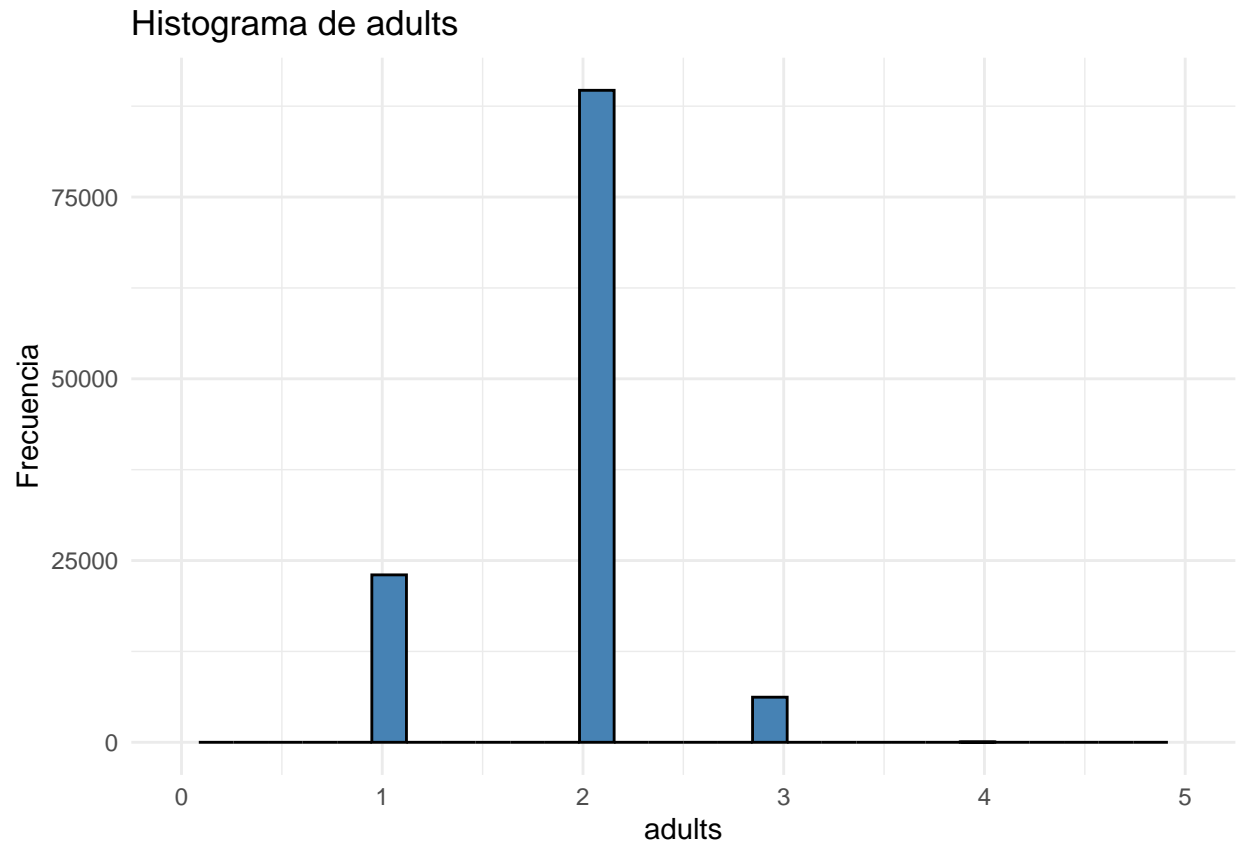
Histograma de lead\_time



```
## Warning: Removed 14 rows containing non-finite outside the scale range (`stat_bin()`).
```

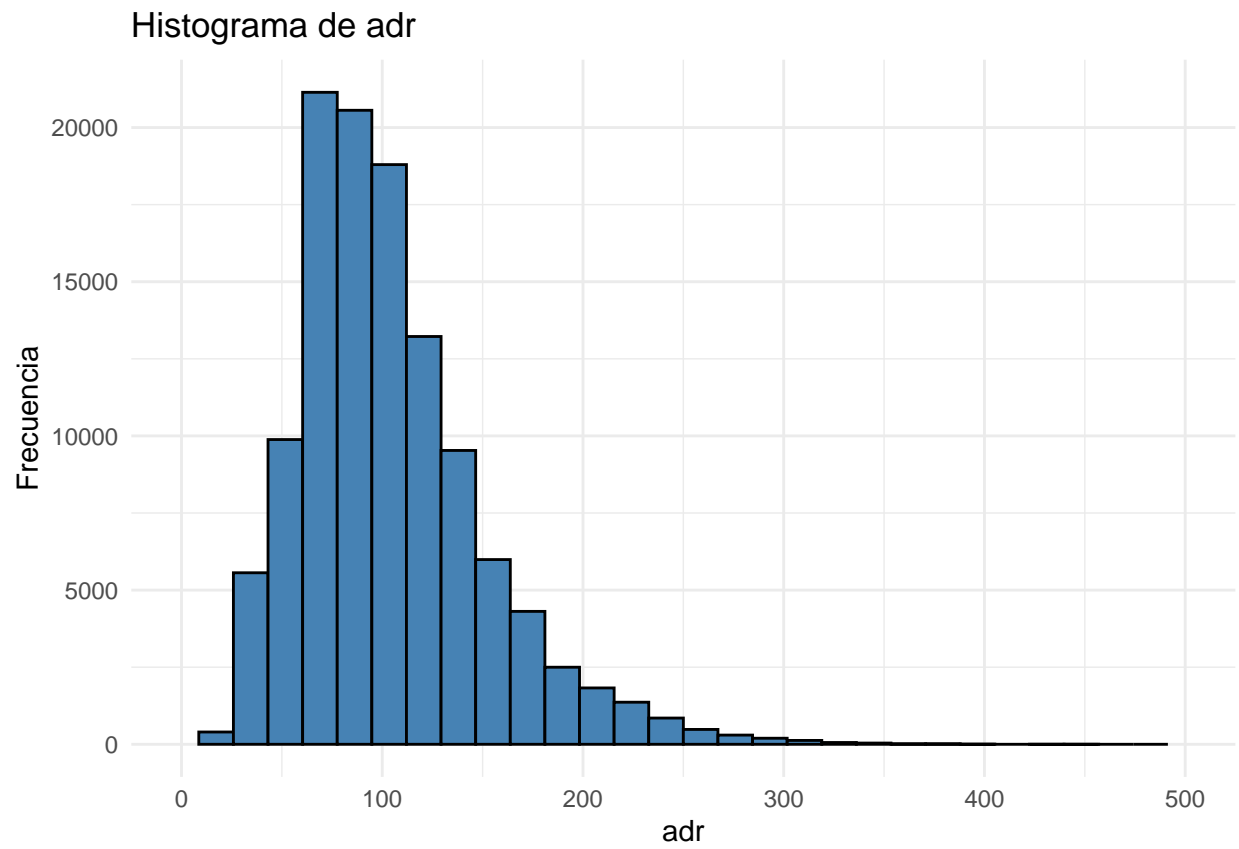
```
## Removed 2 rows containing missing values or values outside the scale range
```

```
## (`geom_bar()`).
```

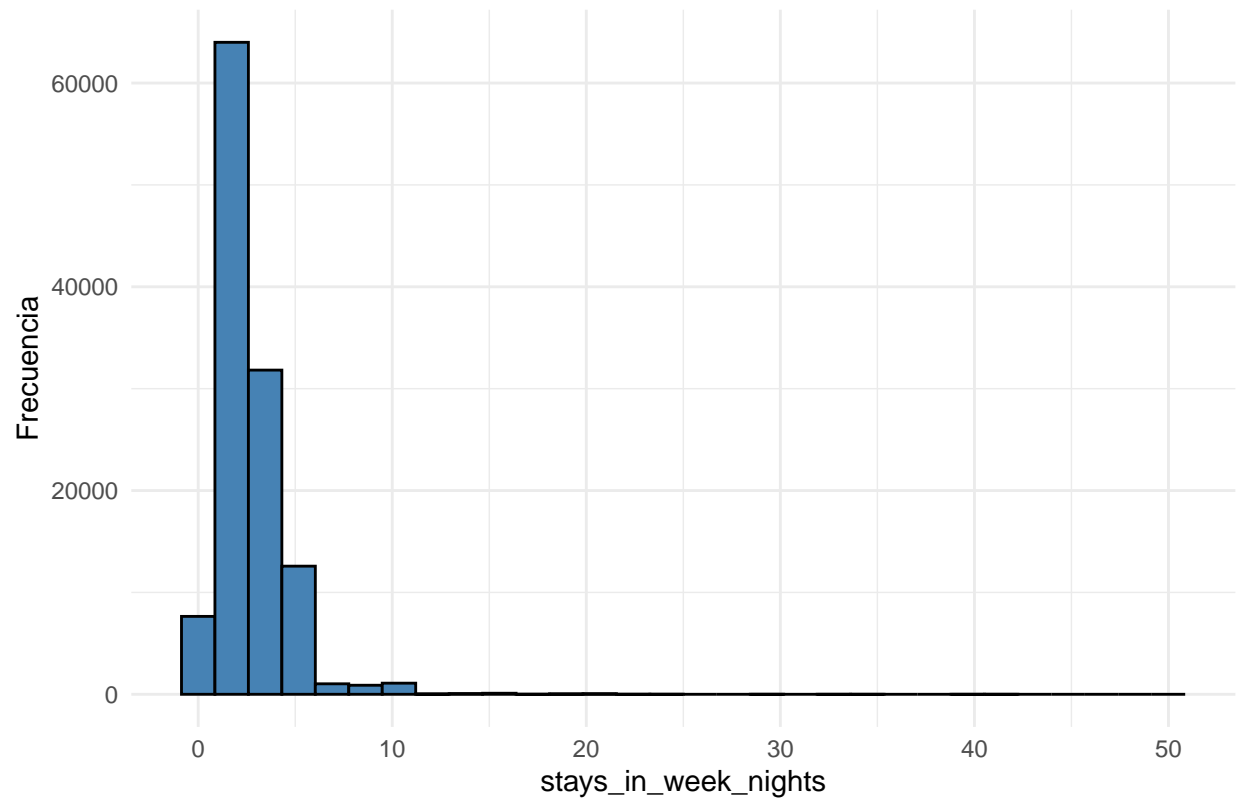


```
## Warning: Removed 4 rows containing non-finite outside the scale range (`stat_bin()`).  
## Removed 2 rows containing missing values or values outside the scale range  
## (`geom_bar()`).
```

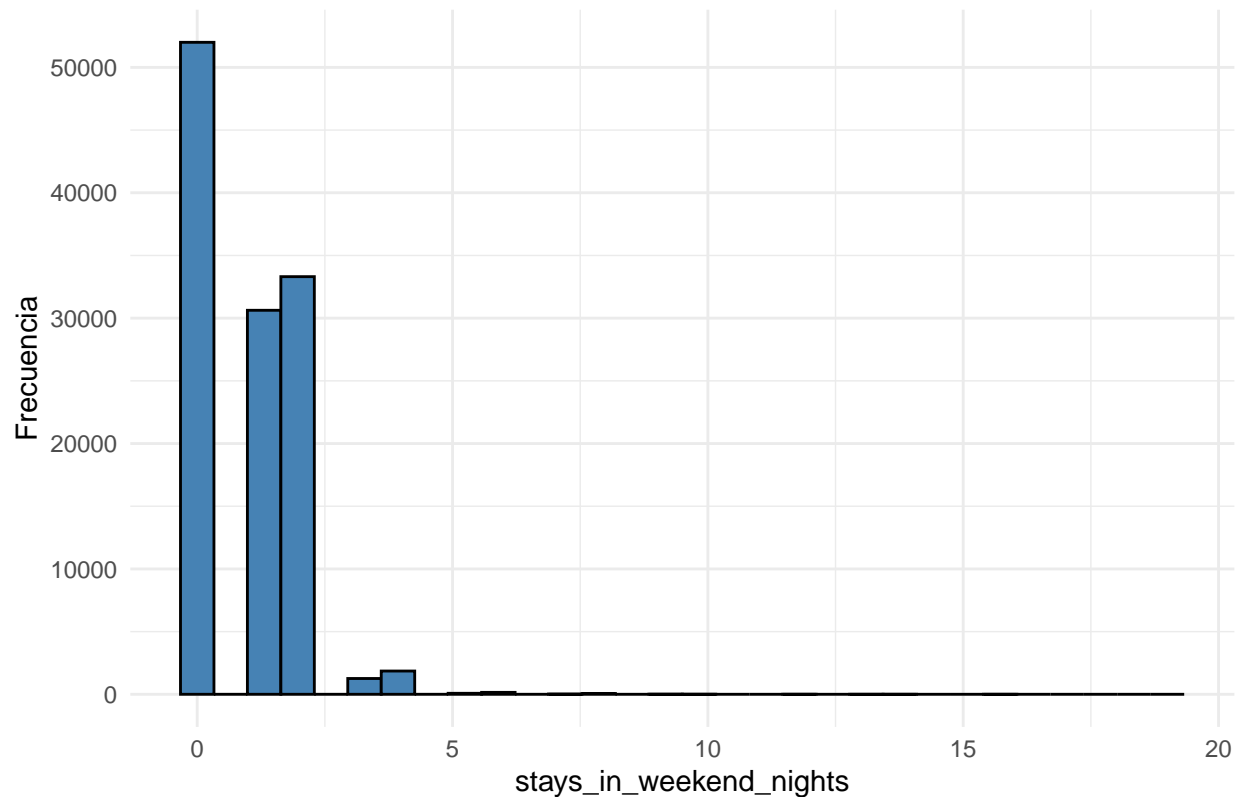




Histograma de stays\_in\_week\_nights



Histograma de stays\_in\_weekend\_nights



```
#-----
# 2.3. VERIFICACIÓN DE CONSISTENCIA LÓGICA
#-----
cat(yellow("\n--- VERIFICACIÓN DE CONSISTENCIA LÓGICA ---\n"))
```

```
##
## --- VERIFICACIÓN DE CONSISTENCIA LÓGICA ---
```

```
# Verificar reservas sin adultos
reservas_sin_adultos <- sum(hotel_data$adults == 0)
cat("Reservas sin adultos:", reservas_sin_adultos, "\n")
```

```
## Reservas sin adultos: 403
```

```
# Verificar total de noches = 0
reservas_sin_noches <- sum(hotel_data$stays_in_weekend_nights == 0 & hotel_data$stays_in_week_nights == 0)
cat("Reservas sin noches (0 días de estadía):", reservas_sin_noches, "\n")
```

```
## Reservas sin noches (0 días de estadía): 715
```

```
# Verificar consistencia entre estado de reserva y cancelación
inconsistencia_cancelacion <- sum(hotel_data$is_canceled == 1 & hotel_data$reservation_status == "Check-out")
cat("Inconsistencias entre cancelación y estado (canceladas pero con check-out):", inconsistencia_cancelacion, "\n")
```

```
## Inconsistencias entre cancelación y estado (canceladas pero con check-out): 0
```

```
# Verificar valores atípicos en número de adultos
cat("\nPosibles errores en 'adults':")
```

```
##
```

```
## Posibles errores en 'adults':
```

```
adultos_tabla <- table(hotel_data$adults)
print(adultos_tabla)
```

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

```
#-----
# 2.4. DEFINIR ESTRATEGIAS PARA DATOS FALTANTES Y ATÍPICOS
#-----
cat(yellow("\n--- ESTRATEGIAS PROPUESTAS ---\n"))
```

```
##
## --- ESTRATEGIAS PROPUESTAS ---
cat("1. Estrategia para datos faltantes:\n")
```

```
## 1. Estrategia para datos faltantes:
```

```
cat("    - Para 'children': Imputar con la moda (0) ya que la mayoría de reservas no tienen niños\n")
```

```
##    - Para 'children': Imputar con la moda (0) ya que la mayoría de reservas no tienen niños
```

```
cat("\n2. Estrategia para outliers:\n")
```

```
##
## 2. Estrategia para outliers:
```

```
cat("    - lead_time: Mantener valores hasta 365 días (1 año), recortar valores superiores\n")
```

```
##    - lead_time: Mantener valores hasta 365 días (1 año), recortar valores superiores
```

```
cat("    - adults: Valores superiores a 4 parecen errores, considerar recortar a un máximo razonable\n")
```

```
##    - adults: Valores superiores a 4 parecen errores, considerar recortar a un máximo razonable
```

```
cat("    - stays_in_weekend_nights y stays_in_week_nights: Establecer límites razonables (ej. máximo 14 días)\n")
```

```
##    - stays_in_weekend_nights y stays_in_week_nights: Establecer límites razonables (ej. máximo 14 días)
```

```
cat("    - adr: Eliminar valores negativos y recortar valores extremadamente altos (ej. > 1000)\n")
```

```
##    - adr: Eliminar valores negativos y recortar valores extremadamente altos (ej. > 1000)
```

```
#####
# PARTE 03: PREPROCESAMIENTO DE DATOS
#####
```

```
cat(green("\n=== RONDA 3: PREPROCESAMIENTO DE DATOS ===\n"))
```

```
##
## === RONDA 3: PREPROCESAMIENTO DE DATOS ===
```

```
# Crear una copia para no modificar los datos originales
hotel_data_limpio <- hotel_data
```

```
#-----
# 3.1. TRATAMIENTO DE DATOS FALTANTES
```

```

#-----
cat(yellow("\n--- TRATAMIENTO DE DATOS FALTANTES ---\n"))

##
## --- TRATAMIENTO DE DATOS FALTANTES ---
# Imputar los NA en 'children' con la moda (0)
hotel_data_limpio$children[is.na(hotel_data_limpio$children)] <- 0
cat("Valores NA en 'children' después de imputación:", sum(is.na(hotel_data_limpio$children)), "\n")

## Valores NA en 'children' después de imputación: 0
#-----
# 3.2. TRATAMIENTO DE VALORES ATÍPICOS (OUTLIERS)
#-----
cat(yellow("\n--- TRATAMIENTO DE VALORES ATÍPICOS ---\n"))

##
## --- TRATAMIENTO DE VALORES ATÍPICOS ---
# Función para aplicar winsorización en una variable
winsorizar <- function(x, lower_limit, upper_limit) {
  x[x < lower_limit] <- lower_limit
  x[x > upper_limit] <- upper_limit
  return(x)
}

# 3.2.1. Winsorizar lead_time (tiempo de anticipación)
cat("\nTratamiento de 'lead_time':")

##
## Tratamiento de 'lead_time':
cat("\n Antes - Max:", max(hotel_data_limpio$lead_time), "Min:", min(hotel_data_limpio$lead_time))

##
## Antes - Max: 737 Min: 0
hotel_data_limpio$lead_time <- winsorizar(hotel_data_limpio$lead_time, 0, 365)
cat("\n Después - Max:", max(hotel_data_limpio$lead_time), "Min:", min(hotel_data_limpio$lead_time), "\n")

##
## Después - Max: 365 Min: 0
# 3.2.2. Corregir adults (adultos)
cat("\nTratamiento de 'adults':")

##
## Tratamiento de 'adults':
cat("\n Antes - Max:", max(hotel_data_limpio$adults), "Min:", min(hotel_data_limpio$adults))

##
## Antes - Max: 55 Min: 0
# Reemplazar valores 0 con 1 (no tiene sentido una reserva sin adultos)
hotel_data_limpio$adults[hotel_data_limpio$adults == 0] <- 1
# Winsorizar a un máximo de 4 adultos por habitación
hotel_data_limpio$adults <- winsorizar(hotel_data_limpio$adults, 1, 4)
cat("\n Después - Max:", max(hotel_data_limpio$adults), "Min:", min(hotel_data_limpio$adults), "\n")

```

```

##
## Después - Max: 4 Min: 1
# 3.2.3. Winsorizar stays_in_weekend_nights y stays_in_week_nights
cat("\nTratamiento de 'stays_in_weekend_nights':")

##
## Tratamiento de 'stays_in_weekend_nights':
cat("\n Antes - Max:", max(hotel_data_limpio$stays_in_weekend_nights))

##
## Antes - Max: 19
hotel_data_limpio$stays_in_weekend_nights <- winsorizar(hotel_data_limpio$stays_in_weekend_nights, 0, 19)
cat("\n Después - Max:", max(hotel_data_limpio$stays_in_weekend_nights), "\n")

##
## Después - Max: 14
cat("\nTratamiento de 'stays_in_week_nights':")

##
## Tratamiento de 'stays_in_week_nights':
cat("\n Antes - Max:", max(hotel_data_limpio$stays_in_week_nights))

##
## Antes - Max: 50
hotel_data_limpio$stays_in_week_nights <- winsorizar(hotel_data_limpio$stays_in_week_nights, 0, 50)
cat("\n Después - Max:", max(hotel_data_limpio$stays_in_week_nights), "\n")

##
## Después - Max: 14
# 3.2.4. Tratar adr (tarifa diaria promedio)
cat("\nTratamiento de 'adr':")

##
## Tratamiento de 'adr':
cat("\n Antes - Max:", max(hotel_data_limpio$adr), "Min:", min(hotel_data_limpio$adr))

##
## Antes - Max: 5400 Min: -6.38
# Reemplazar valores negativos con 0
hotel_data_limpio$adr[hotel_data_limpio$adr < 0] <- 0
# Winsorizar valores extremadamente altos
hotel_data_limpio$adr <- winsorizar(hotel_data_limpio$adr, 0, 1000)
cat("\n Después - Max:", max(hotel_data_limpio$adr), "Min:", min(hotel_data_limpio$adr), "\n")

##
## Después - Max: 1000 Min: 0
# 3.2.5. Winsorizar otros valores numéricos
cat("\nTratamiento de 'children':")

##

```

```

## Tratamiento de 'children':
hotel_data_limpio$children <- winsorizar(hotel_data_limpio$children, 0, 3)
cat("\n Después - Max:", max(hotel_data_limpio$children), "\n")

##
## Después - Max: 3
cat("\nTratamiento de 'babies':")

##
## Tratamiento de 'babies':
hotel_data_limpio$babies <- winsorizar(hotel_data_limpio$babies, 0, 2)
cat("\n Después - Max:", max(hotel_data_limpio$babies), "\n")

##
## Después - Max: 2
#-----
# 3.3. TRATAMIENTO DE INCONSISTENCIAS LÓGICAS
#-----
cat(yellow("\n--- TRATAMIENTO DE INCONSISTENCIAS LÓGICAS ---\n"))

##
## --- TRATAMIENTO DE INCONSISTENCIAS LÓGICAS ---
# Identificar reservas sin noches (estancia de 0 días)
reservas_sin_noches <- hotel_data_limpio$stays_in_weekend_nights == 0 &
  hotel_data_limpio$stays_in_week_nights == 0
cat("Reservas con estancia de 0 días:", sum(reservas_sin_noches), "\n")

## Reservas con estancia de 0 días: 715
# Como no tiene sentido una reserva sin estadía, establecemos al menos 1 noche
hotel_data_limpio$stays_in_week_nights[reservas_sin_noches] <- 1
cat("Reservas con estancia de 0 días después de corrección:",
    sum(hotel_data_limpio$stays_in_weekend_nights == 0 &
        hotel_data_limpio$stays_in_week_nights == 0), "\n")

## Reservas con estancia de 0 días después de corrección: 0
#-----
# 4. VERIFICACIÓN DE LIMPIEZA
#-----
cat(yellow("\n--- VERIFICACIÓN DE LIMPIEZA ---\n"))

##
## --- VERIFICACIÓN DE LIMPIEZA ---
# Verificar NA después de limpieza
na_count_limpio <- colSums(is.na(hotel_data_limpio))
cat("\nCantidad de NA después de limpieza:", sum(na_count_limpio), "\n")

##
## Cantidad de NA después de limpieza: 0
# Verificar outliers después de limpieza
outlier_summary_limpio <- do.call(rbind, lapply(numeric_vars, function(var) {
  outlier_stats(hotel_data_limpio, var)

```

```

}))

cat("\nResumen de outliers después de limpieza:\n")

##
## Resumen de outliers después de limpieza:
print(outlier_summary_limpio)

##
##          Variable    Q1  Q3    IQR Lower_Bound Upper_Bound
## 25%          lead_time 18.00 160 142.00   -195.000    373.000
## 25%1      stays_in_weekend_nights 0.00  2   2.00    -3.000     5.000
## 25%2          stays_in_week_nights 1.00  3   2.00    -2.000     6.000
## 25%3              adults 2.00  2   0.00     2.000     2.000
## 25%4              children 0.00  0   0.00     0.000     0.000
## 25%5              babies 0.00  0   0.00     0.000     0.000
## 25%6      previous_cancellations 0.00  0   0.00     0.000     0.000
## 25%7 previous_bookings_not_canceled 0.00  0   0.00     0.000     0.000
## 25%8          booking_changes 0.00  0   0.00     0.000     0.000
## 25%9          days_in_waiting_list 0.00  0   0.00     0.000     0.000
## 25%10              adr 69.29 126  56.71   -15.775    211.065
## 25%11      required_car_parking_spaces 0.00  0   0.00     0.000     0.000
## 25%12      total_of_special_requests 0.00  1   1.00    -1.500     2.500
##
##      Outlier_Count Outlier_Percentage
## 25%              0              0.00
## 25%1             265              0.22
## 25%2            3354              2.81
## 25%3           29710             24.88
## 25%4            8590              7.19
## 25%5             917              0.77
## 25%6            6484              5.43
## 25%7            3620              3.03
## 25%8           18076             15.14
## 25%9            3698              3.10
## 25%10           3793              3.18
## 25%11           7416              6.21
## 25%12          2877              2.41

#-----
# 4.5 GUARDAR GRÁFICAS COMO JPG EN DATA FOLDER
#-----
cat(yellow("\n--- GUARDANDO GRÁFICAS EN FORMATO JPG ---\n"))

##
## --- GUARDANDO GRÁFICAS EN FORMATO JPG ---
# Crear subfolder para las gráficas si no existe
graphics_dir <- file.path(data_dir, "graficas")
if (!dir.exists(graphics_dir)) {
  dir.create(graphics_dir)
  cat("Creado directorio para gráficas:", graphics_dir, "\n")
} else {
  cat("Usando directorio existente para gráficas:", graphics_dir, "\n")
}

## Usando directorio existente para gráficas: D:/@aleec02/CC216-TP-2025-1/data/graficas

```



```

# Crear subfolder para gráficas de datos limpios
graphics_clean_dir <- file.path(graphics_dir, "limpios")
if (!dir.exists(graphics_clean_dir)) {
  dir.create(graphics_clean_dir)
  cat("Creado directorio para gráficas de datos limpios:", graphics_clean_dir, "\n")
} else {
  cat("Usando directorio existente para gráficas limpias:", graphics_clean_dir, "\n")
}

```

## Usando directorio existente para gráficas limpias: D:/@aleec02/CC216-TP-2025-1/data/graficas/limpios

```

# Función para crear y guardar histogramas
crear_y_guardar_histogramas <- function(data, variables, directorio) {
  graficas_guardadas <- c()

  for (var in variables) {
    # Nombre del archivo para guardar
    filename <- file.path(directorio, paste0("histograma_", var, ".jpg"))

    # Crear histograma según la variable
    if(var == "lead_time") {
      p <- ggplot(data, aes(x = .data[[var]])) +
        geom_histogram(bins = 30, fill = "steelblue", color = "black") +
        labs(title = paste("Histograma de", var), x = var, y = "Frecuencia") +
        theme_minimal() +
        xlim(0, 400) # Zoom para ver mejor la distribución principal
    } else if(var == "adults") {
      p <- ggplot(data, aes(x = .data[[var]])) +
        geom_histogram(bins = 30, fill = "steelblue", color = "black") +
        labs(title = paste("Histograma de", var), x = var, y = "Frecuencia") +
        theme_minimal() +
        xlim(0, 5) # Enfocarse en valores razonables
    } else if(var == "adr") {
      p <- ggplot(data, aes(x = .data[[var]])) +
        geom_histogram(bins = 30, fill = "steelblue", color = "black") +
        labs(title = paste("Histograma de", var), x = var, y = "Frecuencia") +
        theme_minimal() +
        xlim(0, 500) # Enfocarse en el rango principal
    } else {
      p <- ggplot(data, aes(x = .data[[var]])) +
        geom_histogram(bins = 30, fill = "steelblue", color = "black") +
        labs(title = paste("Histograma de", var), x = var, y = "Frecuencia") +
        theme_minimal()
    }

    # Guardar el gráfico como JPG
    ggsave(filename = filename, plot = p, width = 8, height = 6, dpi = 300)
    graficas_guardadas <- c(graficas_guardadas, filename)
    cat(" - Guardado:", filename, "\n")
  }

  return(graficas_guardadas)
}

```

```

# Guardar histogramas para variables de interés
cat("\nGuardando histogramas para datos originales...\n")

##
## Guardando histogramas para datos originales...
histogramas_originales <- crear_y_guardar_histogramas(
  hotel_data,
  variables_interes,
  graphics_dir
)

## Warning: Removed 2114 rows containing non-finite outside the scale range (`stat_bin()`).
## Removed 2 rows containing missing values or values outside the scale range
## (`geom_bar()`).
## - Guardado: D:/@aleec02/CC216-TP-2025-1/data/graficas/histograma_lead_time.jpg
## Warning: Removed 14 rows containing non-finite outside the scale range (`stat_bin()`).
## Removed 2 rows containing missing values or values outside the scale range
## (`geom_bar()`).
## - Guardado: D:/@aleec02/CC216-TP-2025-1/data/graficas/histograma_adults.jpg
## Warning: Removed 4 rows containing non-finite outside the scale range (`stat_bin()`).
## Removed 2 rows containing missing values or values outside the scale range
## (`geom_bar()`).
## - Guardado: D:/@aleec02/CC216-TP-2025-1/data/graficas/histograma_adr.jpg
## - Guardado: D:/@aleec02/CC216-TP-2025-1/data/graficas/histograma_stays_in_week_nights.jpg
## - Guardado: D:/@aleec02/CC216-TP-2025-1/data/graficas/histograma_stays_in_weekend_nights.jpg
# Guardar histogramas para datos limpios
cat("\nGuardando histogramas para datos limpios...\n")

##
## Guardando histogramas para datos limpios...
histogramas_limpios <- crear_y_guardar_histogramas(
  hotel_data_limpio,
  variables_interes,
  graphics_clean_dir
)

## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_bar()`).
## - Guardado: D:/@aleec02/CC216-TP-2025-1/data/graficas/limpios/histograma_lead_time.jpg
## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_bar()`).
## - Guardado: D:/@aleec02/CC216-TP-2025-1/data/graficas/limpios/histograma_adults.jpg
## Warning: Removed 3 rows containing non-finite outside the scale range (`stat_bin()`).
## Removed 2 rows containing missing values or values outside the scale range
## (`geom_bar()`).
## - Guardado: D:/@aleec02/CC216-TP-2025-1/data/graficas/limpios/histograma_adr.jpg
## - Guardado: D:/@aleec02/CC216-TP-2025-1/data/graficas/limpios/histograma_stays_in_week_nights.jpg

```

```

## - Guardado: D:/@aleec02/CC216-TP-2025-1/data/graficas/limpios/histograma_stays_in_weekend_nights.jp
# Resumir gráficas guardadas
cat("\nTotal de archivos guardados:",
    length(c(histogramas_originales, histogramas_limpios)), "\n")

##
## Total de archivos guardados: 10

#-----
# PARTE 05. GUARDAR DATASET LIMPIO
#-----
cat(yellow("\n--- GUARDANDO DATASET LIMPIO ---\n"))

##
## --- GUARDANDO DATASET LIMPIO ---
# Guardar dataset limpio
write.csv(hotel_data_limpio, CSV_limpio, row.names = FALSE)
cat("Dataset limpio guardado en:", CSV_limpio, "\n")

## Dataset limpio guardado en: D:/@aleec02/CC216-TP-2025-1/data/hotel_bookings_limpio.csv
# Mostrar comparación de dimensiones antes y después de la limpieza
cat(yellow("\n--- COMPARACIÓN DE DIMENSIONES ---\n"))

##
## --- COMPARACIÓN DE DIMENSIONES ---
cat("Dataset original:", dim(hotel_data)[1], "filas x", dim(hotel_data)[2], "columnas\n")

## Dataset original: 119390 filas x 32 columnas
cat("Dataset limpio:", dim(hotel_data_limpio)[1], "filas x", dim(hotel_data_limpio)[2], "columnas\n")

## Dataset limpio: 119390 filas x 32 columnas
# Verificar si hubo cambios
filas_diff <- dim(hotel_data_limpio)[1] - dim(hotel_data)[1]
cols_diff <- dim(hotel_data_limpio)[2] - dim(hotel_data)[2]

if (filas_diff != 0 || cols_diff != 0) {
  cat(red(";Alerta! Las dimensiones cambiaron durante la limpieza:\n"))
  if (filas_diff != 0) cat(" - Diferencia en filas:", filas_diff, "\n")
  if (cols_diff != 0) cat(" - Diferencia en columnas:", cols_diff, "\n")
} else {
  cat(green(" Las dimensiones se mantuvieron intactas durante la limpieza.\n"))
  cat(" No se eliminaron filas ni se agregaron/eliminaron columnas.\n")
  cat(" Solo se modificaron los valores para corregir outliers y datos faltantes.\n")
}

## Las dimensiones se mantuvieron intactas durante la limpieza.
## No se eliminaron filas ni se agregaron/eliminaron columnas.
## Solo se modificaron los valores para corregir outliers y datos faltantes.

#####
# PARTE 06: ANÁLISIS EDA
#####

cat(green("\n=== RONDA 4: ANÁLISIS EXPLORATORIO - PREGUNTAS CLAVE (PARTE 1) ===\n"))

```

```
##
## === RONDA 4: ANÁLISIS EXPLORATORIO - PREGUNTAS CLAVE (PARTE 1) ===
# Cargar el dataset limpio
hotel_data_limpio <- read.csv(CSV_limpio, header = TRUE, stringsAsFactors = FALSE)

# Crear directorio para gráficas de análisis si no existe
graphics_analysis_dir <- file.path(graphics_dir, "analisis")
if (!dir.exists(graphics_analysis_dir)) {
  dir.create(graphics_analysis_dir)
  cat("Creado directorio para gráficas de análisis:", graphics_analysis_dir, "\n")
}

#-----
# EDA 01: ¿CUÁNTAS RESERVAS SE REALIZAN POR TIPO DE HOTEL? ¿QUÉ TIPO DE HOTEL PREFIERE LA GENTE?
#-----
cat(yellow("\n--- ANÁLISIS POR TIPO DE HOTEL ---\n"))

##
## --- ANÁLISIS POR TIPO DE HOTEL ---
# Contar reservas por tipo de hotel
reservas_por_hotel <- table(hotel_data_limpio$hotel)
reservas_por_hotel_df <- as.data.frame(reservas_por_hotel)
names(reservas_por_hotel_df) <- c("Tipo_Hotel", "Cantidad")

# Calcular porcentajes
reservas_por_hotel_df$Porcentaje <- round(
  reservas_por_hotel_df$Cantidad / sum(reservas_por_hotel_df$Cantidad) * 100, 2
)

# Mostrar resultados
cat("\nDistribución de reservas por tipo de hotel:\n")

##
## Distribución de reservas por tipo de hotel:
print(reservas_por_hotel_df)

##      Tipo_Hotel Cantidad Porcentaje
## 1   City Hotel    79330      66.45
## 2 Resort Hotel   40060      33.55

# Visualizar distribución
plot_hoteles <- ggplot(reservas_por_hotel_df, aes(x = Tipo_Hotel, y = Cantidad, fill = Tipo_Hotel)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = paste0(Cantidad, "\n(", Porcentaje, "%)")),
    position = position_stack(vjust = 0.5), color = "white", size = 4) +
  labs(title = "Cantidad de reservas por tipo de hotel",
    x = "Tipo de hotel",
    y = "Cantidad de reservas") +
  theme_minimal() +
  theme(legend.position = "none")

print(plot_hoteles)
```



```
# Guardar gráfica
ggsave(file.path(graphics_analysis_dir, "reservas_por_hotel.jpg"),
        plot_hoteles, width = 8, height = 6, dpi = 300)

# Análisis por estado de cancelación para ver preferencia real
reservas_completadas <- hotel_data_limpio[hotel_data_limpio$is_canceled == 0, ]
reservas_completadas_por_hotel <- table(reservas_completadas$hotel)
reservas_completadas_df <- as.data.frame(reservas_completadas_por_hotel)
names(reservas_completadas_df) <- c("Tipo_Hotel", "Reservas_Completadas")

# Calcular porcentajes de reservas completadas
reservas_completadas_df$Porcentaje <- round(
  reservas_completadas_df$Reservas_Completadas / sum(reservas_completadas_df$Reservas_Completadas) * 100
)

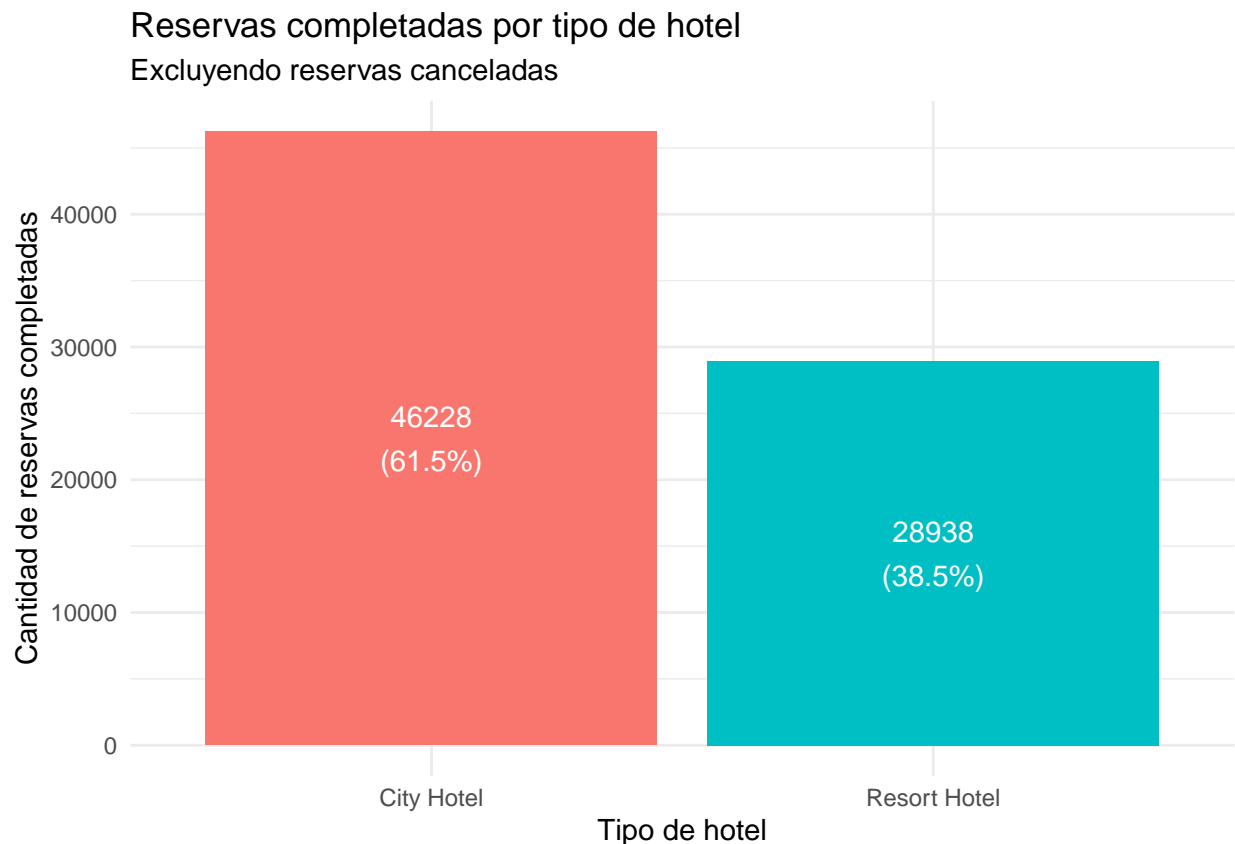
cat("\nDistribución de reservas completadas (no canceladas) por tipo de hotel:\n")

##
## Distribución de reservas completadas (no canceladas) por tipo de hotel:
print(reservas_completadas_df)

##      Tipo_Hotel Reservas_Completadas Porcentaje
## 1   City Hotel           46228           61.5
## 2 Resort Hotel           28938           38.5
```

```
# Visualizar reservas completadas
plot_completadas <- ggplot(reservas_completadas_df, aes(x = Tipo_Hotel, y = Reservas_Completadas, fill = 
  geom_bar(stat = "identity") +
  geom_text(aes(label = paste0(Reservas_Completadas, "\n(", Porcentaje, "%)")),
    position = position_stack(vjust = 0.5), color = "white", size = 4) +
  labs(title = "Reservas completadas por tipo de hotel",
    subtitle = "Excluyendo reservas canceladas",
    x = "Tipo de hotel",
    y = "Cantidad de reservas completadas") +
  theme_minimal() +
  theme(legend.position = "none")

print(plot_completadas)
```



```
# Guardar gráfica
ggsave(file.path(graphics_analysis_dir, "reservas_completadas_por_hotel.jpg"),
  plot_completadas, width = 8, height = 6, dpi = 300)

#-----
# EDA 02: ¿ESTÁ AUMENTANDO LA DEMANDA CON EL TIEMPO?
#-----
cat(yellow("\n--- ANÁLISIS DE TENDENCIA DE DEMANDA ---\n"))

##
## --- ANÁLISIS DE TENDENCIA DE DEMANDA ---
```

```

# Análisis por año y mes
# Convertir mes a factor ordenado
hotel_data_limpio$arrival_date_month <- factor(
  hotel_data_limpio$arrival_date_month,
  levels = c("January", "February", "March", "April", "May", "June",
             "July", "August", "September", "October", "November", "December")
)

# Crear columna de fecha combinada (año-mes)
hotel_data_limpio$arrival_yearmonth <- paste(
  hotel_data_limpio$arrival_date_year,
  sprintf("%02d", as.numeric(factor(hotel_data_limpio$arrival_date_month,
                                   levels = c("January", "February", "March", "April", "May", "June",
                                               "July", "August", "September", "October", "November", "December"))
  sep = "-"
)

# Agregación por año-mes
reservas_por_tiempo <- hotel_data_limpio %>%
  group_by(arrival_yearmonth) %>%
  summarise(
    Total_Reservas = n(),
    Reservas_Completadas = sum(is_canceled == 0)
  ) %>%
  arrange(arrival_yearmonth)

# Mostrar tendencia de reservas de forma más clara
cat(yellow("\n--- TENDENCIA DE RESERVAS A LO LARGO DEL TIEMPO ---\n"))

##
## --- TENDENCIA DE RESERVAS A LO LARGO DEL TIEMPO ---

# Calcular tasa de cancelación para cada período
reservas_por_tiempo <- reservas_por_tiempo %>%
  mutate(
    Tasa_Cancelacion = round((Total_Reservas - Reservas_Completadas) / Total_Reservas * 100, 1),
    Periodo = paste(substr(arrival_yearmonth, 1, 4), substr(arrival_yearmonth, 6, 7), sep = "-")
  ) %>%
  select(Periodo, Total_Reservas, Reservas_Completadas, Tasa_Cancelacion)

# Mostrar tabla bonita
cat("\nDatos de reservas por período (primeros 10 meses):\n\n")

##
## Datos de reservas por período (primeros 10 meses):
print(knitr::kable(head(reservas_por_tiempo, 10),
  col.names = c("Período", "Total Reservas", "Reservas Completadas", "Tasa Cancelación"),
  align = c("l", "r", "r", "r"),
  format = "simple"))

##
##

```

## Período	Total Reservas	Reservas Completadas	Tasa Cancelación (%)
## -----	-----	-----	-----
## 2015-07	2776	1517	45.4
## 2015-08	3889	2291	41.1
## 2015-09	5114	3020	40.9
## 2015-10	4957	3225	34.9
## 2015-11	2340	1854	20.8
## 2015-12	2920	1947	33.3
## 2016-01	2248	1691	24.8
## 2016-02	3891	2554	34.4
## 2016-03	4824	3347	30.6
## 2016-04	5428	3367	38.0

```
# Mostrar estadísticas de resumen
```

```
cat("\nEstadísticas de resumen:\n")
```

```
##
```

```
## Estadísticas de resumen:
```

```
cat("- Período con mayor cantidad de reservas:",
    reservas_por_tiempo$Periodo[which.max(reservas_por_tiempo$Total_Reservas)],
    "con", max(reservas_por_tiempo$Total_Reservas), "reservas\n")
```

```
## - Período con mayor cantidad de reservas: 2017-05 con 6313 reservas
```

```
cat("- Período con menor cantidad de reservas:",
    reservas_por_tiempo$Periodo[which.min(reservas_por_tiempo$Total_Reservas)],
    "con", min(reservas_por_tiempo$Total_Reservas), "reservas\n")
```

```
## - Período con menor cantidad de reservas: 2016-01 con 2248 reservas
```

```
cat("- Tasa de cancelación promedio:",
    round(mean(reservas_por_tiempo$Tasa_Cancelacion), 1), "%\n")
```

```
## - Tasa de cancelación promedio: 36.3 %
```

```
# Mostrar tendencia principal
```

```
primero <- head(reservas_por_tiempo, 1)
```

```
ultimo <- tail(reservas_por_tiempo, 1)
```

```
cambio_porcent <- round((ultimo$Total_Reservas - primero$Total_Reservas) / primero$Total_Reservas * 100, 1)
```

```
cat("\nTendencia general:", ifelse(cambio_porcent > 0, "AUMENTO", "DISMINUCIÓN"),
    "del", abs(cambio_porcent), "% en reservas totales",
    "desde", primero$Periodo, "hasta", ultimo$Periodo, "\n")
```

```
##
```

```
## Tendencia general: AUMENTO del 77.4 % en reservas totales desde 2015-07 hasta 2017-08
```

```
# Primero, preparo los datos calculando porcentajes
```

```
reservas_por_tiempo <- hotel_data_limpio %>%
```

```
  group_by(arrival_yearmonth) %>%
```

```
  summarise(
```

```
    Total_Reservas = n(),
```

```
    Reservas_Completadas = sum(is_canceled == 0),
```

```
    Reservas_Canceladas = sum(is_canceled == 1)
```

```
  ) %>%
```

```
  mutate(
```

```
    Porcentaje_Completadas = round(Reservas_Completadas / Total_Reservas * 100, 1),
```



```

    Porcentaje_Canceladas = round(Reservas_Canceladas / Total_Reservas * 100, 1)
  ) %>%
  arrange(arrival_yearmonth)

# Reorganizo los datos para facilitar la visualización apilada
datos_para_grafico <- reservas_por_tiempo %>%
  pivot_longer(
    cols = c(Reservas_Completadas, Reservas_Canceladas),
    names_to = "Estado",
    values_to = "Cantidad"
  ) %>%
  mutate(
    Porcentaje = ifelse(
      Estado == "Reservas_Completadas",
      Porcentaje_Completadas,
      Porcentaje_Canceladas
    )
  )

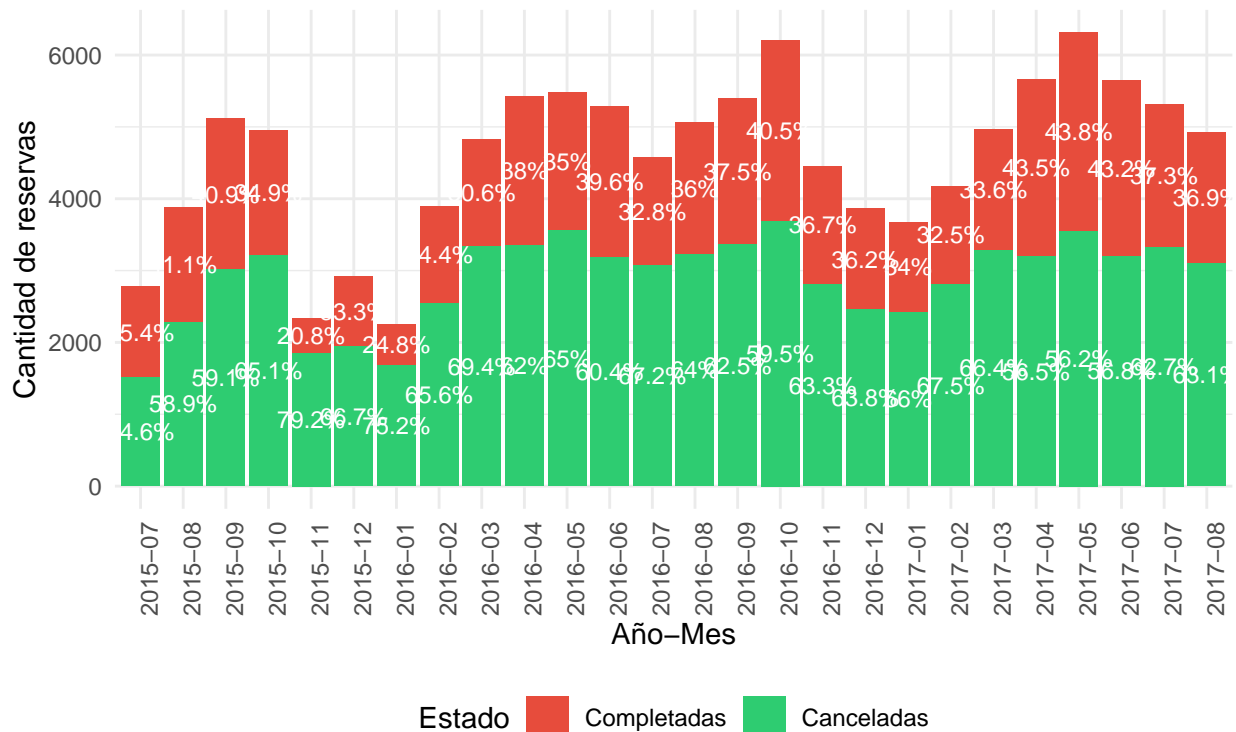
# Creo un gráfico que muestra valores absolutos y porcentajes
plot_tendencia_porcentajes <- ggplot(datos_para_grafico,
                                     aes(x = arrival_yearmonth, y = Cantidad,
                                         fill = Estado, group = Estado)) +
  geom_bar(stat = "identity", position = "stack") +
  geom_text(aes(label = paste0(Porcentaje, "%")),
            position = position_stack(vjust = 0.5),
            color = "white", size = 3) +
  labs(title = "Tendencia de reservas a lo largo del tiempo",
        subtitle = "Mostrando porcentajes de reservas completadas y canceladas",
        x = "Año-Mes",
        y = "Cantidad de reservas",
        fill = "Estado") +
  scale_fill_manual(values = c("Reservas_Completadas" = "#2ecc71",
                               "Reservas_Canceladas" = "#e74c3c"),
                    labels = c("Completadas", "Canceladas")) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1),
        legend.position = "bottom")

print(plot_tendencia_porcentajes)

```

## Tendencia de reservas a lo largo del tiempo

Mostrando porcentajes de reservas completadas y canceladas



```
# Guardar gráfica
```

```
ggsave(file.path(graphics_analysis_dir, "tendencia_reservas_porcentajes.jpg"),
        plot_tendencia_porcentajes, width = 12, height = 8, dpi = 300)
```

```
# También podemos crear un gráfico de líneas que muestre el porcentaje de cancelación a lo largo del tiempo
```

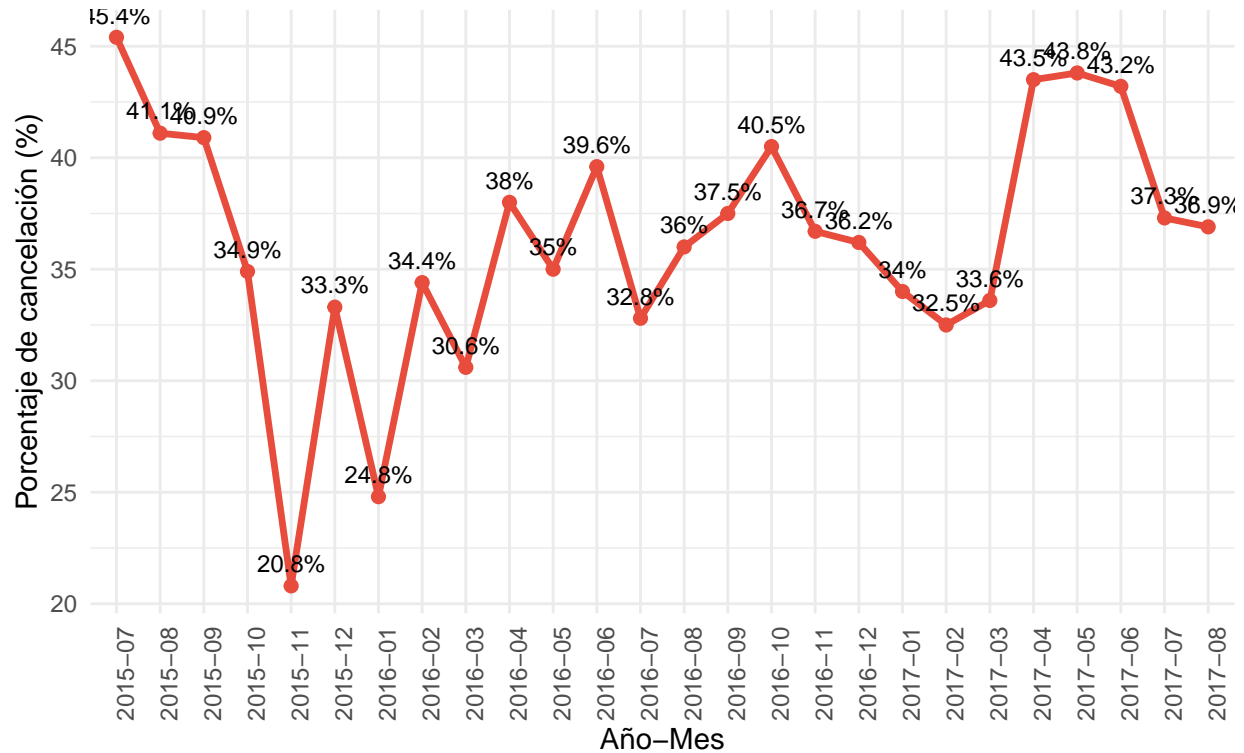
```
plot_porcentaje_cancelacion <- ggplot(reservas_por_tiempo,
                                     aes(x = arrival_yearmonth, y = Porcentaje_Canceladas, group = 1))
  geom_line(color = "#e74c3c", size = 1.2) +
  geom_point(color = "#e74c3c", size = 2) +
  geom_text(aes(label = paste0(Porcentaje_Canceladas, "%")),
            vjust = -0.8, size = 3) +
  labs(title = "Tendencia del porcentaje de cancelaciones",
        subtitle = "Porcentaje de reservas canceladas por mes",
        x = "Año-Mes",
        y = "Porcentaje de cancelación (%)") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
print(plot_porcentaje_cancelacion)
```

## Tendencia del porcentaje de cancelaciones

Porcentaje de reservas canceladas por mes



```
# Guardar segunda gráfica
```

```
ggsave(file.path(graphics_analysis_dir, "tendencia_porcentaje_cancelacion.jpg"),
        plot_porcentaje_cancelacion, width = 12, height = 8, dpi = 300)
```

```
# Análisis por año con porcentajes
```

```
reservas_por_anio <- hotel_data_limpio %>%
  group_by(arrival_date_year) %>%
  summarise(
    Total_Reservas = n(),
    Reservas_Completadas = sum(is_canceled == 0),
    Reservas_Canceladas = sum(is_canceled == 1)
  ) %>%
  mutate(
    Porcentaje_Completadas = round(Reservas_Completadas / Total_Reservas * 100, 1),
    Porcentaje_Canceladas = round(Reservas_Canceladas / Total_Reservas * 100, 1)
  )
```

```
cat("\nReservas por año con porcentajes:\n")
```

```
##
```

```
## Reservas por año con porcentajes:
```

```

print(reservas_por_anio)

## # A tibble: 3 x 6
##   arrival_date_year Total_Reservas Reservas_Completadas Reservas_Canceladas
##           <int>         <int>             <int>             <int>
## 1           2015         21996             13854             8142
## 2           2016         56707             36370            20337
## 3           2017         40687             24942            15745
## # i 2 more variables: Porcentaje_Completadas <dbl>, Porcentaje_Canceladas <dbl>

# Preparar datos para el gráfico horizontal
datos_anio_apilado <- reservas_por_anio %>%
  pivot_longer(
    cols = c(Reservas_Completadas, Reservas_Canceladas),
    names_to = "Estado",
    values_to = "Cantidad"
  ) %>%
  mutate(
    Porcentaje = ifelse(
      Estado == "Reservas_Completadas",
      Porcentaje_Completadas,
      Porcentaje_Canceladas
    )
  )

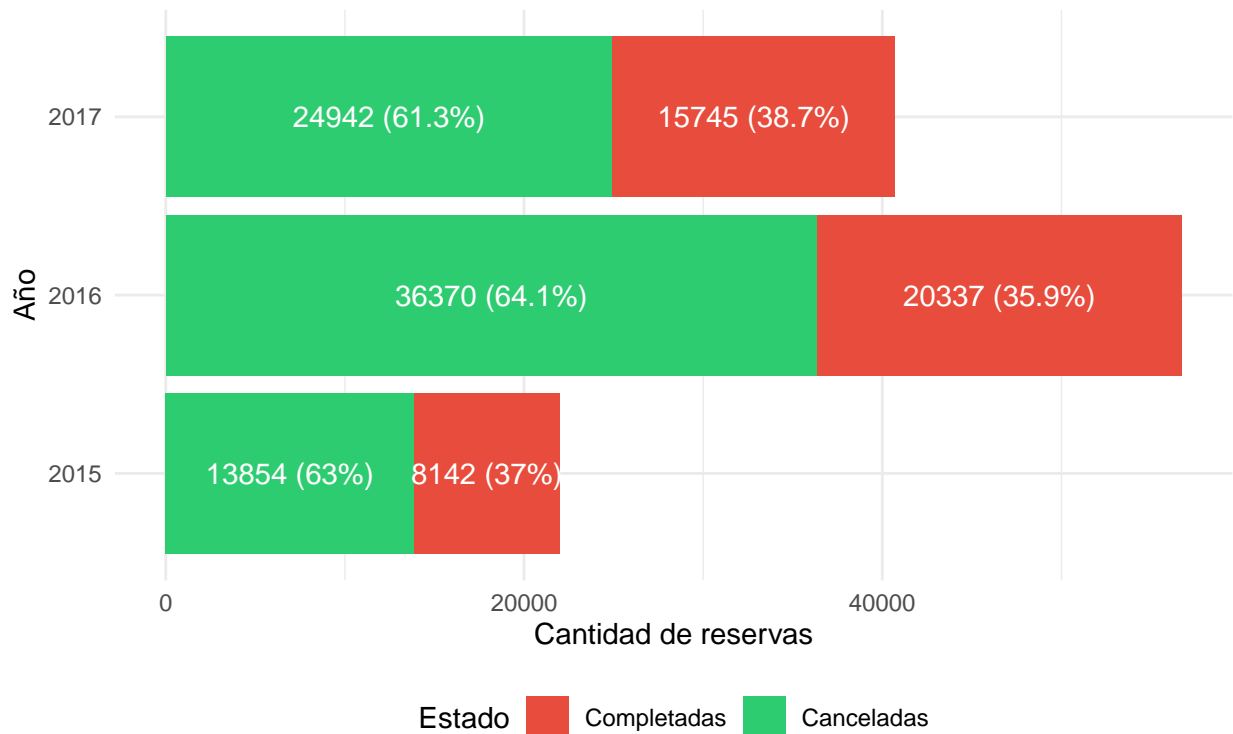
# Crear el gráfico horizontal de barras apiladas
plot_anios_horizontal <- ggplot(datos_anio_apilado,
                                aes(x = as.factor(arrival_date_year),
                                    y = Cantidad,
                                    fill = Estado,
                                    group = Estado)) +
  geom_bar(stat = "identity", position = "stack") +
  geom_text(aes(label = paste0(Cantidad, " (", Porcentaje, "%)")),
            position = position_stack(vjust = 0.5),
            color = "white", size = 4) +
  coord_flip() + # Hace el gráfico horizontal
  labs(title = "Reservas por año",
        subtitle = "Mostrando cantidades y porcentajes de reservas",
        y = "Cantidad de reservas",
        x = "Año",
        fill = "Estado") +
  scale_fill_manual(values = c("Reservas_Completadas" = "#2ecc71",
                               "Reservas_Canceladas" = "#e74c3c"),
                    labels = c("Completadas", "Canceladas")) +
  theme_minimal() +
  theme(legend.position = "bottom")

print(plot_anios_horizontal)

```

## Reservas por año

Mostrando cantidades y porcentajes de reservas



```
# Guardar gráfica horizontal
ggsave(file.path(graphics_analysis_dir, "reservas_por_anio_horizontal.jpg"),
        plot_anios_horizontal, width = 8, height = 6, dpi = 300)
```

```
#-----
# EDA 03: ¿CUÁLES SON LAS TEMPORADAS DE RESERVAS (ALTA, MEDIA, BAJA)?
#-----
cat(yellow("\n--- ANÁLISIS DE TEMPORADAS ---\n"))
```

```
##
```

```
## --- ANÁLISIS DE TEMPORADAS ---
```

```
# Agregación por mes
```

```
reservas_por_mes <- hotel_data_limpio %>%
```

```
  group_by(arrival_date_month) %>%
```

```
  summarise(
```

```
    Total_Reservas = n(),
```

```
    Reservas_Completadas = sum(is_canceled == 0),
```

```
    Tasa_Cancelacion = round(sum(is_canceled) / n() * 100, 2)
```

```
  ) %>%
```

```
  arrange(match(arrival_date_month, c("January", "February", "March", "April", "May", "June",
                                     "July", "August", "September", "October", "November", "December")))
```

```

cat("\nReservas por mes:\n")

##
## Reservas por mes:
print(reservas_por_mes)

## # A tibble: 12 x 4
##   arrival_date_month Total_Reservas Reservas_Completadas Tasa_Cancelacion
##   <fct>                <int>          <int>          <dbl>
## 1 January              5929            4122            30.5
## 2 February             8068            5372            33.4
## 3 March                9794            6645            32.2
## 4 April               11089            6565            40.8
## 5 May                 11791            7114            39.7
## 6 June                10939            6404            41.5
## 7 July                12661            7919            37.4
## 8 August              13877            8638            37.8
## 9 September           10508            6392            39.2
## 10 October            11160            6914            38.0
## 11 November           6794            4672            31.2
## 12 December           6780            4409            35.0

# Determinar temporadas basadas en cantidad de reservas
media_reservas <- mean(reservas_por_mes$Total_Reservas)
sd_reservas <- sd(reservas_por_mes$Total_Reservas)

reservas_por_mes$Temporada <- case_when(
  reservas_por_mes$Total_Reservas >= (media_reservas + 0.5 * sd_reservas) ~ "Alta",
  reservas_por_mes$Total_Reservas <= (media_reservas - 0.5 * sd_reservas) ~ "Baja",
  TRUE ~ "Media"
)

cat("\nClasificación de temporadas por mes:\n")

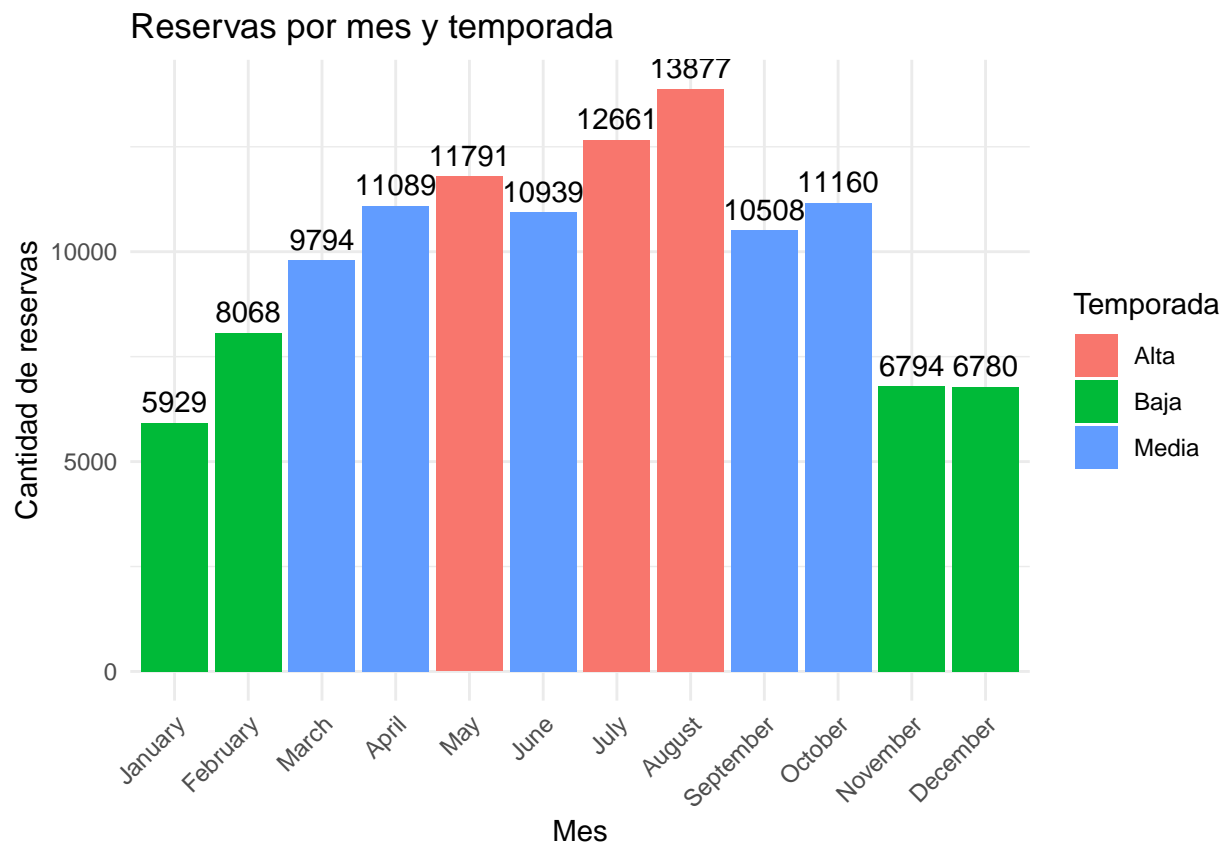
##
## Clasificación de temporadas por mes:
print(reservas_por_mes[, c("arrival_date_month", "Total_Reservas", "Temporada")])

## # A tibble: 12 x 3
##   arrival_date_month Total_Reservas Temporada
##   <fct>                <int> <chr>
## 1 January              5929 Baja
## 2 February             8068 Baja
## 3 March                9794 Media
## 4 April               11089 Media
## 5 May                 11791 Alta
## 6 June                10939 Media
## 7 July                12661 Alta
## 8 August              13877 Alta
## 9 September           10508 Media
## 10 October            11160 Media
## 11 November           6794 Baja
## 12 December           6780 Baja

```

```
# Visualizar reservas por mes
plot_meses <- ggplot(reservas_por_mes, aes(x = arrival_date_month, y = Total_Reservas, fill = Temporada)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = Total_Reservas), vjust = -0.5, color = "black") +
  labs(title = "Reservas por mes y temporada",
       x = "Mes",
       y = "Cantidad de reservas") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

print(plot_meses)
```



```
# Guardar gráfica
ggsave(file.path(graphics_analysis_dir, "reservas_por_mes.jpg"),
        plot_meses, width = 10, height = 6, dpi = 300)
```

```
#-----
# EDA 04: ¿CUÁNDO ES MENOR LA DEMANDA DE RESERVAS?
#-----
cat(yellow("\n--- ANÁLISIS DE PERÍODOS DE BAJA DEMANDA ---\n"))
```

```
##
```

```
## --- ANÁLISIS DE PERÍODOS DE BAJA DEMANDA ---
# Ya identificamos los meses de baja demanda en el análisis anterior
meses_baja_demanda <- reservas_por_mes %>%
  filter(Temporada == "Baja") %>%
  arrange(Total_Reservas)

cat("\nMeses con menor demanda de reservas:\n")

##
## Meses con menor demanda de reservas:

print(meses_baja_demanda)

## # A tibble: 4 x 5
##   arrival_date_month Total_Reservas Reservas_Completadas Tasa_Cancelacion
##   <fct>                <int>                <int>                <dbl>
## 1 January              5929                  4122                 30.5
## 2 December             6780                  4409                 35.0
## 3 November             6794                  4672                 31.2
## 4 February            8068                  5372                 33.4
## # i 1 more variable: Temporada <chr>

# Análisis por combinación de mes y día del mes
reservas_por_dia_mes <- hotel_data_limpio %>%
  group_by(arrival_date_month, arrival_date_day_of_month) %>%
  summarise(Total_Reservas = n()) %>%
  arrange(Total_Reservas)

## `summarise()` has grouped output by 'arrival_date_month'. You can override
## using the `.groups` argument.

cat("\nCombinaciones de mes y día con menor demanda (10 primeros):\n")

##
## Combinaciones de mes y día con menor demanda (10 primeros):

print(head(reservas_por_dia_mes, 10))

## # A tibble: 10 x 3
## # Groups:   arrival_date_month [4]
##   arrival_date_month arrival_date_day_of_month Total_Reservas
##   <fct>                <int>                <int>
## 1 December              11                 89
## 2 February             29                 96
## 3 December             13                 96
## 4 January              11                101
## 5 January              17                106
## 6 January              31                107
## 7 December             18                110
## 8 November             22                115
## 9 December             14                116
## 10 December            20                116

# Visualizar los 10 días con menor demanda
dias_menor_demanda <- reservas_por_dia_mes %>%
  arrange(Total_Reservas) %>%
  head(10)
```



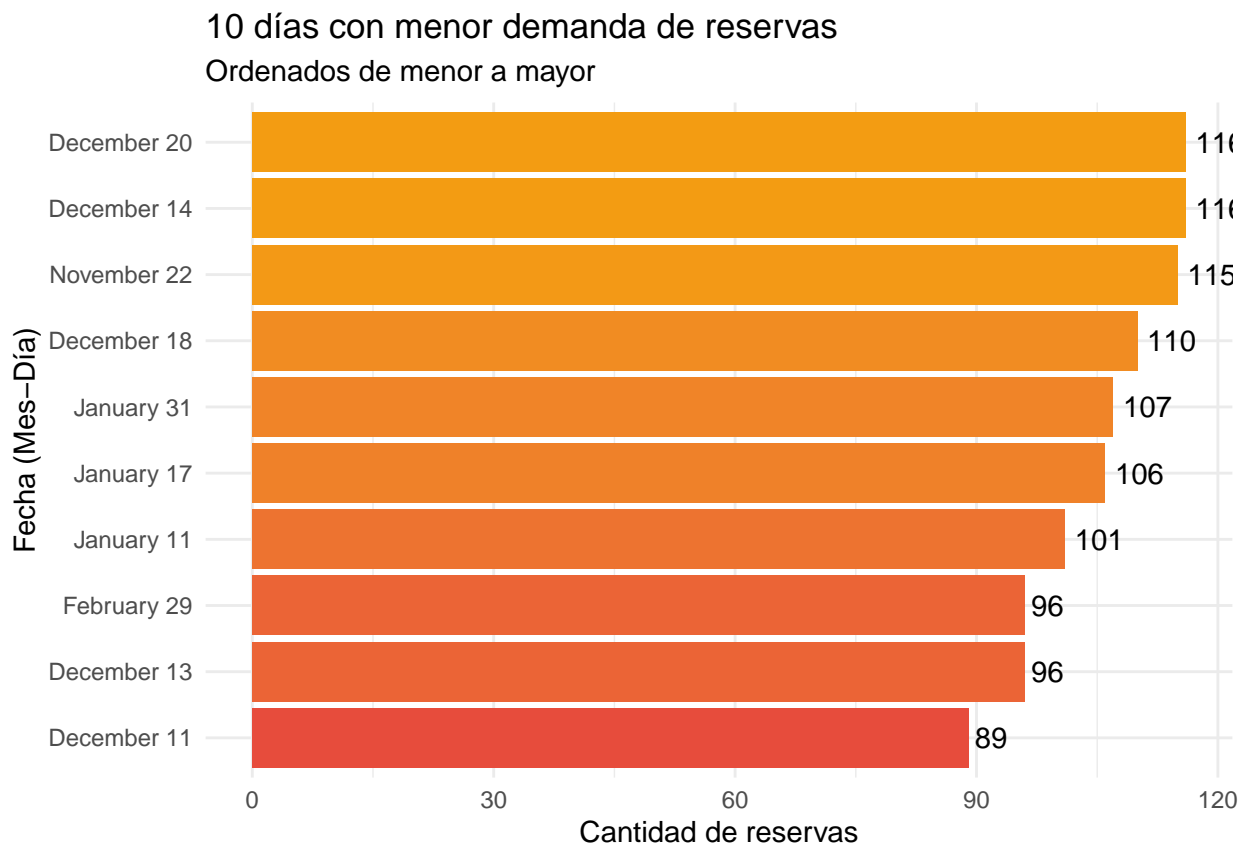
```

dias_menor_demanda$Fecha <- paste(dias_menor_demanda$arrival_date_month,
                                   dias_menor_demanda$arrival_date_day_of_month)

# Gráfico con el menor valor arriba
plot_menor_demanda <- ggplot(dias_menor_demanda,
                             aes(x = reorder(Fecha, Total_Reservas),
                                 y = Total_Reservas,
                                 fill = Total_Reservas)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = Total_Reservas), hjust = -0.2) +
  labs(title = "10 días con menor demanda de reservas",
       subtitle = "Ordenados de menor a mayor",
       x = "Fecha (Mes-Día)",
       y = "Cantidad de reservas") +
  coord_flip() +
  scale_fill_gradient(low = "#e74c3c", high = "#f39c12") +
  theme_minimal() +
  theme(legend.position = "none")

print(plot_menor_demanda)

```



```

# Guardar gráfica
ggsave(file.path(graphics_analysis_dir, "dias_menor_demanda.jpg"),
        plot_menor_demanda, width = 8, height = 6, dpi = 300)

```

```

#-----
# EDA 05: ¿CUÁNDO ES MAYOR LA DEMANDA DE RESERVAS?
#-----
cat(yellow("\n--- ANÁLISIS DE PERÍODOS DE ALTA DEMANDA ---\n"))

##
## --- ANÁLISIS DE PERÍODOS DE ALTA DEMANDA ---
# Identificar meses de alta demanda
meses_alta_demanda <- reservas_por_mes %>%
  filter(Temporada == "Alta") %>%
  arrange(desc(Total_Reservas))

cat("\nMeses con mayor demanda de reservas:\n")

##
## Meses con mayor demanda de reservas:
print(meses_alta_demanda)

## # A tibble: 3 x 5
##   arrival_date_month Total_Reservas Reservas_Completadas Tasa_Cancelacion
##   <fct>                <int>                <int>                <dbl>
## 1 August              13877                8638                37.8
## 2 July                12661                7919                37.4
## 3 May                 11791                7114                39.7
## # i 1 more variable: Temporada <chr>
# Análisis por combinación de mes y día del mes para alta demanda
cat("\nCombinaciones de mes y día con mayor demanda (10 primeros):\n")

##
## Combinaciones de mes y día con mayor demanda (10 primeros):
dias_mayor_demanda <- reservas_por_dia_mes %>%
  arrange(desc(Total_Reservas)) %>%
  head(10)

print(dias_mayor_demanda)

## # A tibble: 10 x 3
## # Groups:   arrival_date_month [6]
##   arrival_date_month arrival_date_day_of_month Total_Reservas
##   <fct>                <int>                <int>
## 1 October              16                637
## 2 August               14                632
## 3 April                29                596
## 4 December             5                590
## 5 May                  19                580
## 6 June                 17                560
## 7 August               8                555
## 8 April                28                553
## 9 June                 8                537

```

```
## 10 May
```

```
5
```

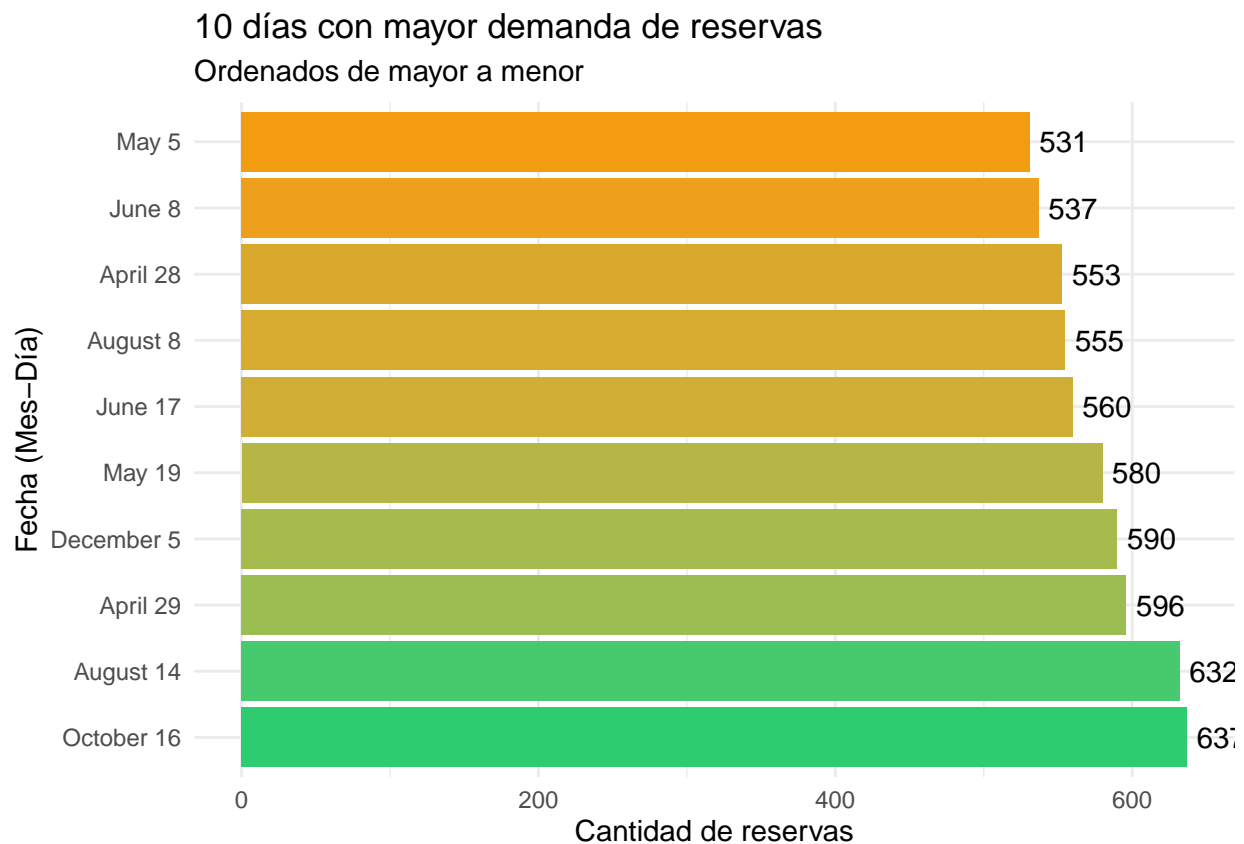
```
531
```

```
# Crear etiqueta de fecha
```

```
dias_mayor_demanda$Fecha <- paste(dias_mayor_demanda$arrival_date_month,  
                                   dias_mayor_demanda$arrival_date_day_of_month)
```

```
# Visualizar los 10 días con mayor demanda
```

```
plot_mayor_demanda <- ggplot(dias_mayor_demanda,  
                             aes(x = reorder(Fecha, -Total_Reservas),  
                                 y = Total_Reservas,  
                                 fill = Total_Reservas)) +  
  geom_bar(stat = "identity") +  
  geom_text(aes(label = Total_Reservas), hjust = -0.2) +  
  labs(title = "10 días con mayor demanda de reservas",  
       subtitle = "Ordenados de mayor a menor",  
       x = "Fecha (Mes-Día)",  
       y = "Cantidad de reservas") +  
  coord_flip() +  
  scale_fill_gradient(low = "#f39c12", high = "#2ecc71") + # Invertimos colores: amarillo a verde  
  theme_minimal() +  
  theme(legend.position = "none")  
  
print(plot_mayor_demanda)
```



```
# Guardar gráfica
```

```
ggsave(file.path(graphics_analysis_dir, "dias_mayor_demanda.jpg"),  
        plot_mayor_demanda, width = 8, height = 6, dpi = 300)
```

```

# Comparativa de meses extremos (mayor y menor demanda)
cat(yellow("\n--- COMPARATIVA DE MESES DE MAYOR Y MENOR DEMANDA ---\n"))

##
## --- COMPARATIVA DE MESES DE MAYOR Y MENOR DEMANDA ---
# Obtener el mes de mayor y menor demanda
mes_max_demanda <- reservas_por_mes %>%
  arrange(desc(Total_Reservas)) %>%
  slice(1)

mes_min_demanda <- reservas_por_mes %>%
  arrange(Total_Reservas) %>%
  slice(1)

cat("\nMes con MAYOR demanda:", mes_max_demanda$arrival_date_month,
    "con", mes_max_demanda$Total_Reservas, "reservas\n")

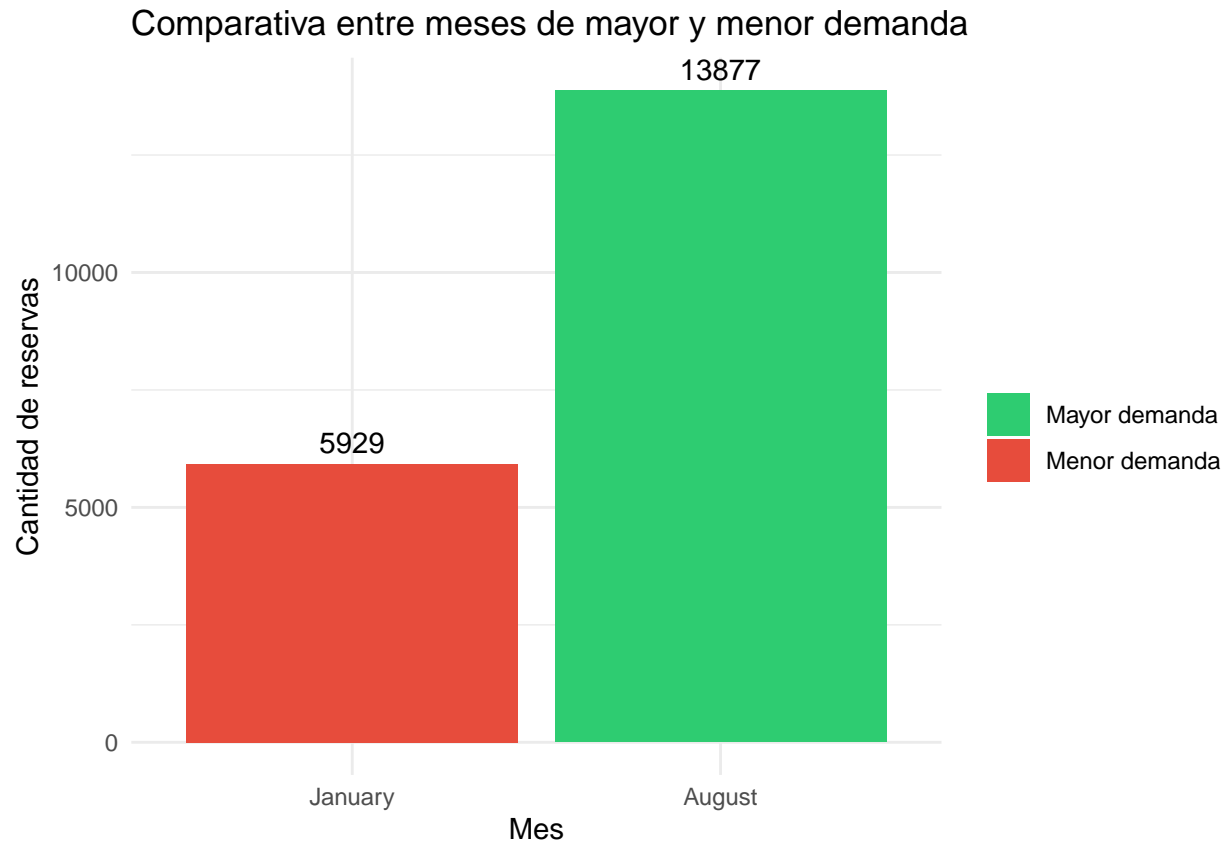
##
## Mes con MAYOR demanda: 8 con 13877 reservas
cat("Mes con MENOR demanda:", mes_min_demanda$arrival_date_month,
    "con", mes_min_demanda$Total_Reservas, "reservas\n")

## Mes con MENOR demanda: 1 con 5929 reservas
# Crear dataframe para comparativa
meses_extremos <- rbind(
  mes_max_demanda %>% mutate(Tipo = "Mayor demanda"),
  mes_min_demanda %>% mutate(Tipo = "Menor demanda")
)

# Visualizar comparativa
plot_meses_extremos <- ggplot(meses_extremos,
  aes(x = arrival_date_month, y = Total_Reservas, fill = Tipo)) +
  geom_bar(stat = "identity", position = "dodge") +
  geom_text(aes(label = Total_Reservas,
    position = position_dodge(width = 0.9), vjust = -0.5) +
  labs(title = "Comparativa entre meses de mayor y menor demanda",
    x = "Mes",
    y = "Cantidad de reservas",
    fill = "") +
  scale_fill_manual(values = c("Mayor demanda" = "#2ecc71", "Menor demanda" = "#e74c3c")) +
  theme_minimal()

print(plot_meses_extremos)

```



```
# Guardar gráfica
ggsave(file.path(graphics_analysis_dir, "comparativa_meses_extremos.jpg"),
        plot_meses_extremos, width = 8, height = 6, dpi = 300)

# Verificar si necesitamos recargar el dataset limpio
if (!exists("hotel_data_limpio") || !is.data.frame(hotel_data_limpio)) {
  cat("Recargando dataset limpio...\n")
  hotel_data_limpio <- read.csv(CSV_limpio, header = TRUE, stringsAsFactors = FALSE)
}

# Asegurar que el directorio para gráficas de análisis existe
graphics_analysis_dir <- file.path(graphics_dir, "analisis")
if (!dir.exists(graphics_analysis_dir)) {
  dir.create(graphics_analysis_dir)
  cat("Creado directorio para gráficas de análisis:", graphics_analysis_dir, "\n")
}

#-----
# EDA 06: ¿CUÁNTAS RESERVAS INCLUYEN NIÑOS Y/O BEBÉS?
```

```

#-----
cat(yellow("\n--- ANÁLISIS DE RESERVAS CON NIÑOS Y/O BEBÉS ---\n"))

##
## --- ANÁLISIS DE RESERVAS CON NIÑOS Y/O BEBÉS ---
# Crear variables categóricas para facilitar el análisis
hotel_data_limpio$tiene_ninos <- hotel_data_limpio$children > 0
hotel_data_limpio$tiene_bebes <- hotel_data_limpio$babies > 0
hotel_data_limpio$tiene_menores <- hotel_data_limpio$tiene_ninos | hotel_data_limpio$tiene_bebes

# Contar reservas con niños y/o bebés
reservas_con_ninos <- sum(hotel_data_limpio$tiene_ninos)
reservas_con_bebes <- sum(hotel_data_limpio$tiene_bebes)
reservas_con_menores <- sum(hotel_data_limpio$tiene_menores)
total_reservas <- nrow(hotel_data_limpio)

# Calcular porcentajes
porcentaje_ninos <- round(reservas_con_ninos / total_reservas * 100, 2)
porcentaje_bebes <- round(reservas_con_bebes / total_reservas * 100, 2)
porcentaje_menores <- round(reservas_con_menores / total_reservas * 100, 2)

# Mostrar resultados
cat("\nAnálisis de reservas con menores:\n")

##
## Análisis de reservas con menores:
cat("- Reservas con niños:", reservas_con_ninos, "(", porcentaje_ninos, "%)\n")

## - Reservas con niños: 8590 ( 7.19 %)
cat("- Reservas con bebés:", reservas_con_bebes, "(", porcentaje_bebes, "%)\n")

## - Reservas con bebés: 917 ( 0.77 %)
cat("- Reservas con niños y/o bebés:", reservas_con_menores, "(", porcentaje_menores, "%)\n")

## - Reservas con niños y/o bebés: 9332 ( 7.82 %)
cat("- Total de reservas:", total_reservas, "\n")

## - Total de reservas: 119390
# Crear dataframe para visualización
datos_menores <- data.frame(
  Categoria = c("Con niños", "Con bebés", "Con niños y/o bebés", "Sin menores"),
  Cantidad = c(reservas_con_ninos, reservas_con_bebes, reservas_con_menores,
    total_reservas - reservas_con_menores)
)

datos_menores$Porcentaje <- round(datos_menores$Cantidad / total_reservas * 100, 2)
datos_menores$Etiqueta <- paste0(datos_menores$Cantidad, "\n(", datos_menores$Porcentaje, "%)")

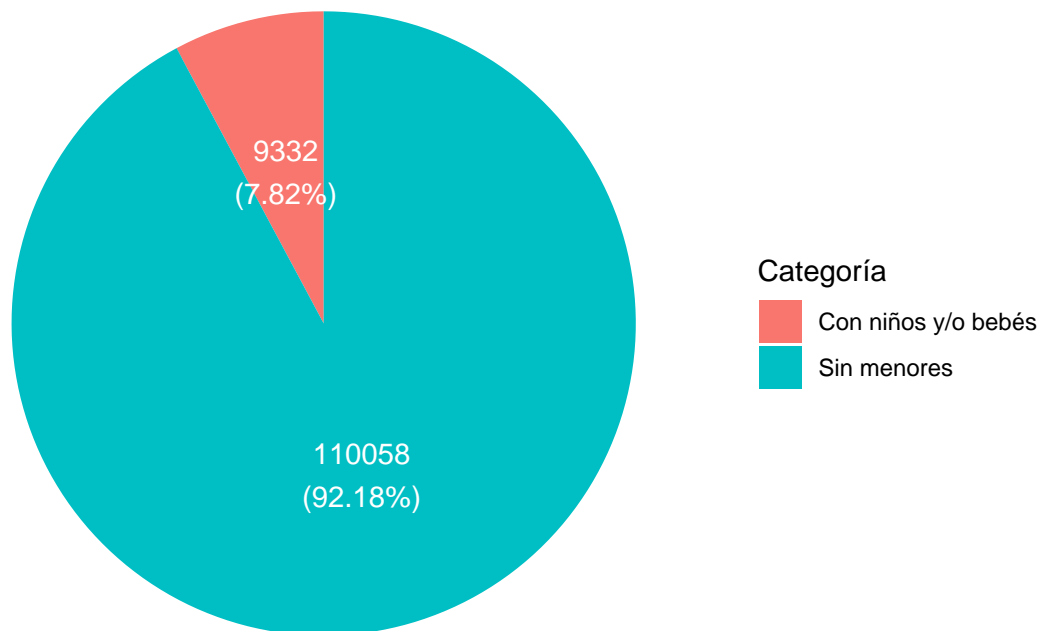
# Visualizar distribución
plot_menores <- ggplot(datos_menores[c(3,4),], aes(x = "", y = Cantidad, fill = Categoria)) +
  geom_bar(stat = "identity", width = 1) +
  geom_text(aes(label = Etiqueta), position = position_stack(vjust = 0.5), color = "white") +

```

```
coord_polar("y", start = 0) +
labs(title = "Proporción de reservas con y sin menores",
     fill = "Categoría") +
theme_minimal() +
theme(axis.title = element_blank(),
      axis.text = element_blank(),
      panel.grid = element_blank())

print(plot_menores)
```

Proporción de reservas con y sin menores



```
# Guardar gráfica
ggsave(file.path(graphics_analysis_dir, "reservas_con_menores.jpg"),
       plot_menores, width = 8, height = 6, dpi = 300)

# Analizar por tipo de hotel
reservas_menores_hotel <- hotel_data_limpio %>%
  group_by(hotel) %>%
  summarise(
    Total_Reservas = n(),
    Con_Ninos = sum(tiene_ninos),
    Con_Bebes = sum(tiene_bebes),
    Con_Menores = sum(tiene_menores),
    Porcentaje_Menores = round(sum(tiene_menores) / n() * 100, 2)
  )

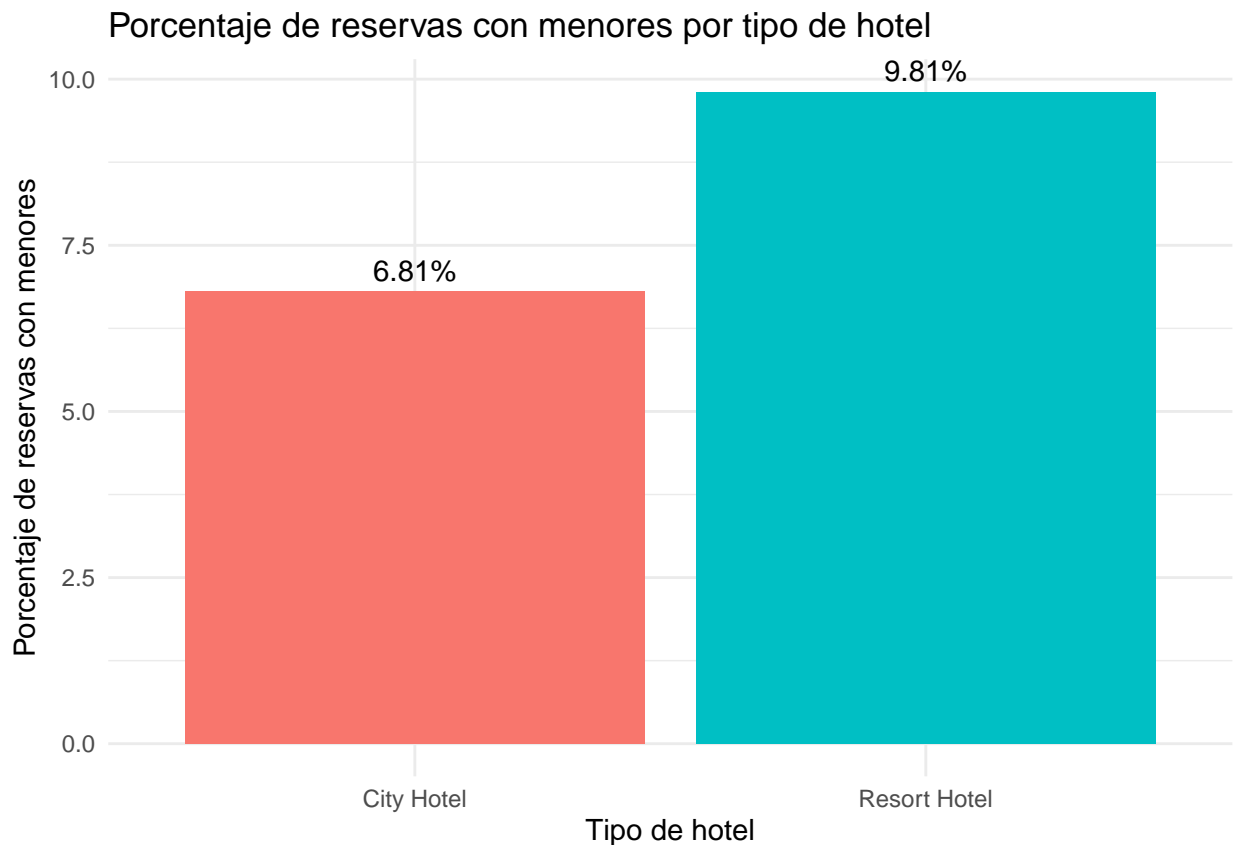
cat("\nReservas con menores por tipo de hotel:\n")
```

```
##
## Reservas con menores por tipo de hotel:
print(reservas_menores_hotel)

## # A tibble: 2 x 6
##   hotel      Total_Reservas Con_Ninos Con_Bebes Con_Menores Porcentaje_Menores
##   <chr>          <int>    <int>    <int>    <int>          <dbl>
## 1 City Hotel      79330     5106     369     5403           6.81
## 2 Resort Hotel   40060     3484     548     3929           9.81

# Visualizar proporción por hotel
plot_menores_hotel <- ggplot(reservas_menores_hotel,
                             aes(x = hotel, y = Porcentaje_Menores, fill = hotel)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = paste0(Porcentaje_Menores, "%")), vjust = -0.5) +
  labs(title = "Porcentaje de reservas con menores por tipo de hotel",
       x = "Tipo de hotel",
       y = "Porcentaje de reservas con menores") +
  theme_minimal() +
  theme(legend.position = "none")

print(plot_menores_hotel)
```



```
# Guardar gráfica
ggsave(file.path(graphics_analysis_dir, "menores_por_hotel.jpg"),
       plot_menores_hotel, width = 8, height = 6, dpi = 300)
```



```

#-----
# EDA 07. ¿ES IMPORTANTE CONTAR CON ESPACIOS DE ESTACIONAMIENTO?
#-----
cat(yellow("\n--- ANÁLISIS DE ESPACIOS DE ESTACIONAMIENTO ---\n"))

##
## --- ANÁLISIS DE ESPACIOS DE ESTACIONAMIENTO ---
# Crear variable categórica
hotel_data_limpio$requiere_estacionamiento <- hotel_data_limpio$required_car_parking_spaces > 0

# Contar reservas que requieren estacionamiento
reservas_con_estacionamiento <- sum(hotel_data_limpio$requiere_estacionamiento)
porcentaje_estacionamiento <- round(reservas_con_estacionamiento / total_reservas * 100, 2)

cat("\nAnálisis de necesidad de estacionamiento:\n")

##
## Análisis de necesidad de estacionamiento:
cat("- Reservas que requieren estacionamiento:", reservas_con_estacionamiento,
    "(", porcentaje_estacionamiento, "%)\n")

## - Reservas que requieren estacionamiento: 7416 ( 6.21 %)
cat("- Reservas sin requerimiento de estacionamiento:", total_reservas - reservas_con_estacionamiento,
    "(", 100 - porcentaje_estacionamiento, "%)\n")

## - Reservas sin requerimiento de estacionamiento: 111974 ( 93.79 %)
# Crear dataframe para visualización
datos_estacionamiento <- data.frame(
  Categoria = c("Requiere estacionamiento", "No requiere estacionamiento"),
  Cantidad = c(reservas_con_estacionamiento, total_reservas - reservas_con_estacionamiento)
)

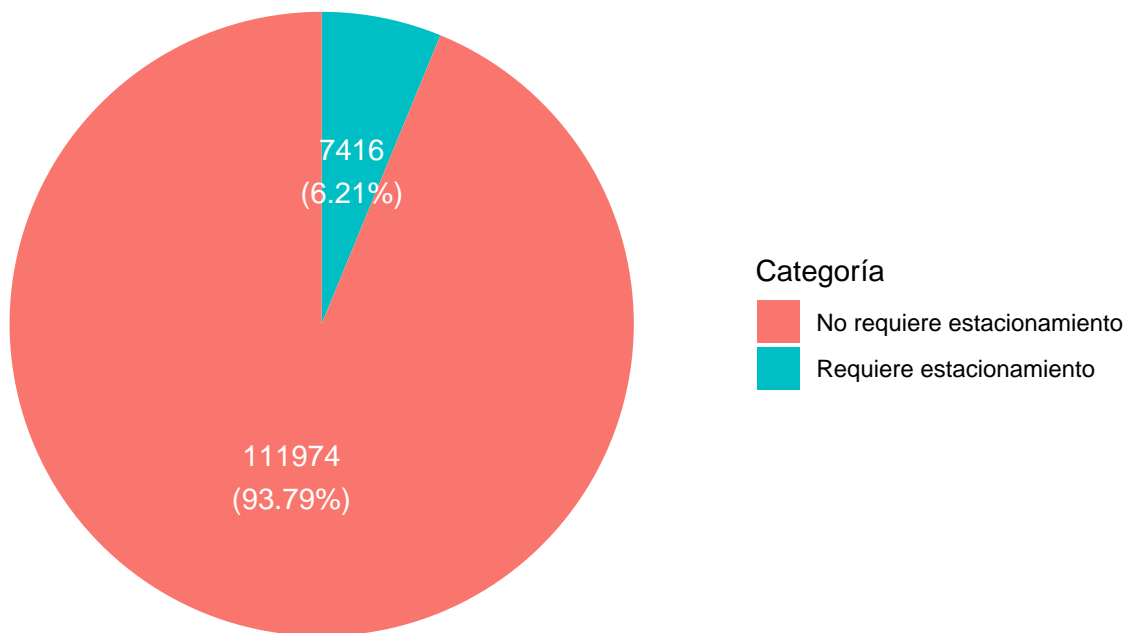
datos_estacionamiento$Porcentaje <- round(datos_estacionamiento$Cantidad / total_reservas * 100, 2)
datos_estacionamiento$Etiqueta <- paste0(datos_estacionamiento$Cantidad,
    "\n(", datos_estacionamiento$Porcentaje, "%)")

# Visualizar distribución
plot_estacionamiento <- ggplot(datos_estacionamiento, aes(x = "", y = Cantidad, fill = Categoria)) +
  geom_bar(stat = "identity", width = 1) +
  geom_text(aes(label = Etiqueta), position = position_stack(vjust = 0.5), color = "white") +
  coord_polar("y", start = 0) +
  labs(title = "Proporción de reservas que requieren estacionamiento",
    fill = "Categoría") +
  theme_minimal() +
  theme(axis.title = element_blank(),
    axis.text = element_blank(),
    panel.grid = element_blank())

print(plot_estacionamiento)

```

## Proporción de reservas que requieren estacionamiento



```
# Guardar gráfica
ggsave(file.path(graphics_analysis_dir, "necesidad_estacionamiento.jpg"),
        plot_estacionamiento, width = 8, height = 6, dpi = 300)

# Análisis por tipo de hotel
estacionamiento_por_hotel <- hotel_data_limpio %>%
  group_by(hotel) %>%
  summarise(
    Total_Reservas = n(),
    Requiere_Estacionamiento = sum(requiere_estacionamiento),
    Porcentaje_Estacionamiento = round(sum(requiere_estacionamiento) / n() * 100, 2)
  )

cat("\nNecesidad de estacionamiento por tipo de hotel:\n")
```

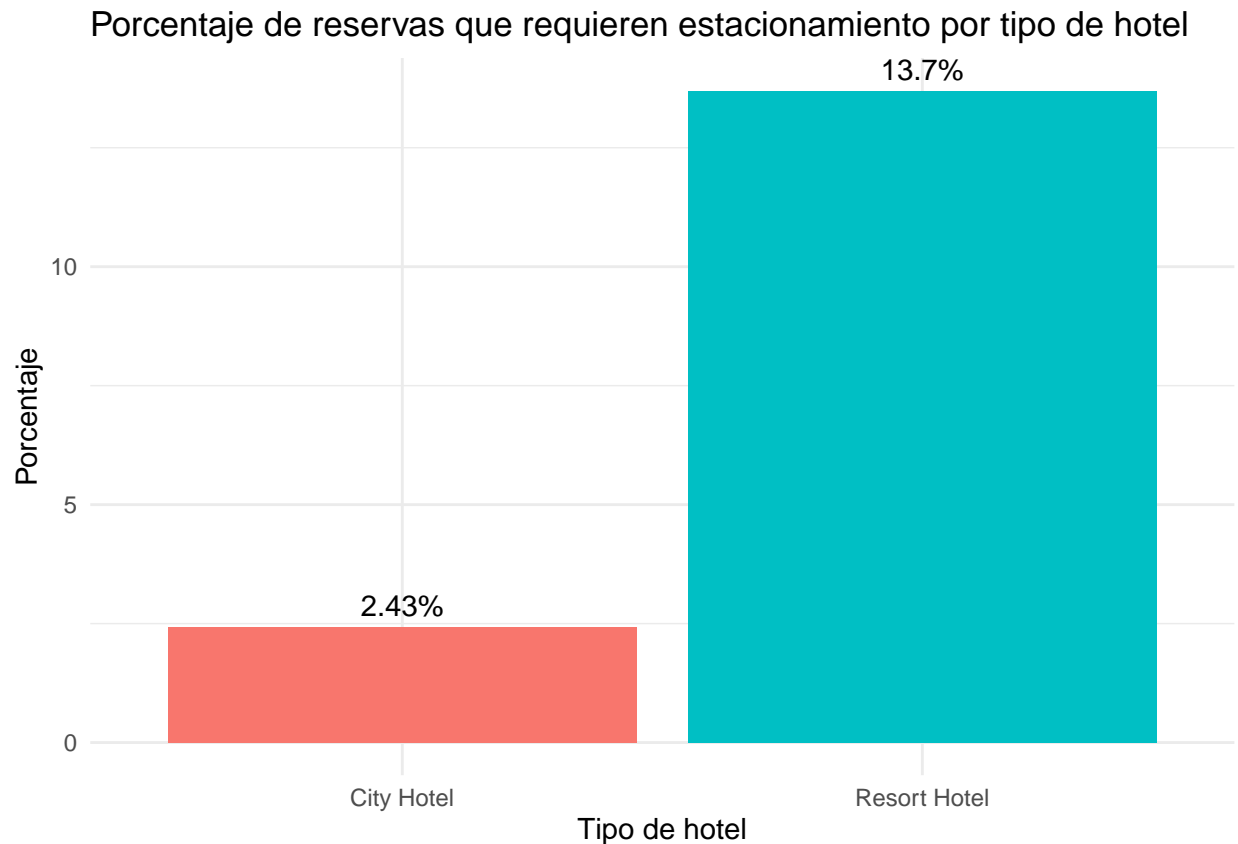
```
##
## Necesidad de estacionamiento por tipo de hotel:
```

```
print(estacionamiento_por_hotel)
```

```
## # A tibble: 2 x 4
##   hotel      Total_Reservas Requiere_Estacionamiento Porcentaje_Estacionamie-1
##   <chr>          <int>              <int>                <dbl>
## 1 City Hotel      79330                1926                2.43
## 2 Resort Hotel   40060                5490               13.7
## # i abbreviated name: 1: Porcentaje_Estacionamiento
```

```
# Visualizar proporción por hotel
plot_estacionamiento_hotel <- ggplot(estacionamiento_por_hotel,
                                     aes(x = hotel, y = Porcentaje_Estacionamiento, fill = hotel)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = paste0(Porcentaje_Estacionamiento, "%"), vjust = -0.5) +
  labs(title = "Porcentaje de reservas que requieren estacionamiento por tipo de hotel",
       x = "Tipo de hotel",
       y = "Porcentaje") +
  theme_minimal() +
  theme(legend.position = "none")

print(plot_estacionamiento_hotel)
```



```
# Guardar gráfica
ggsave(file.path(graphics_analysis_dir, "estacionamiento_por_hotel.jpg"),
        plot_estacionamiento_hotel, width = 8, height = 6, dpi = 300)

# Análisis adicional: relación entre estacionamiento y tipo de cliente
estacionamiento_por_cliente <- hotel_data_limpio %>%
  group_by(customer_type) %>%
  summarise(
    Total_Reservas = n(),
    Requiere_Estacionamiento = sum(requiere_estacionamiento),
    Porcentaje_Estacionamiento = round(sum(requiere_estacionamiento) / n() * 100, 2)
  ) %>%
  arrange(desc(Porcentaje_Estacionamiento))
```

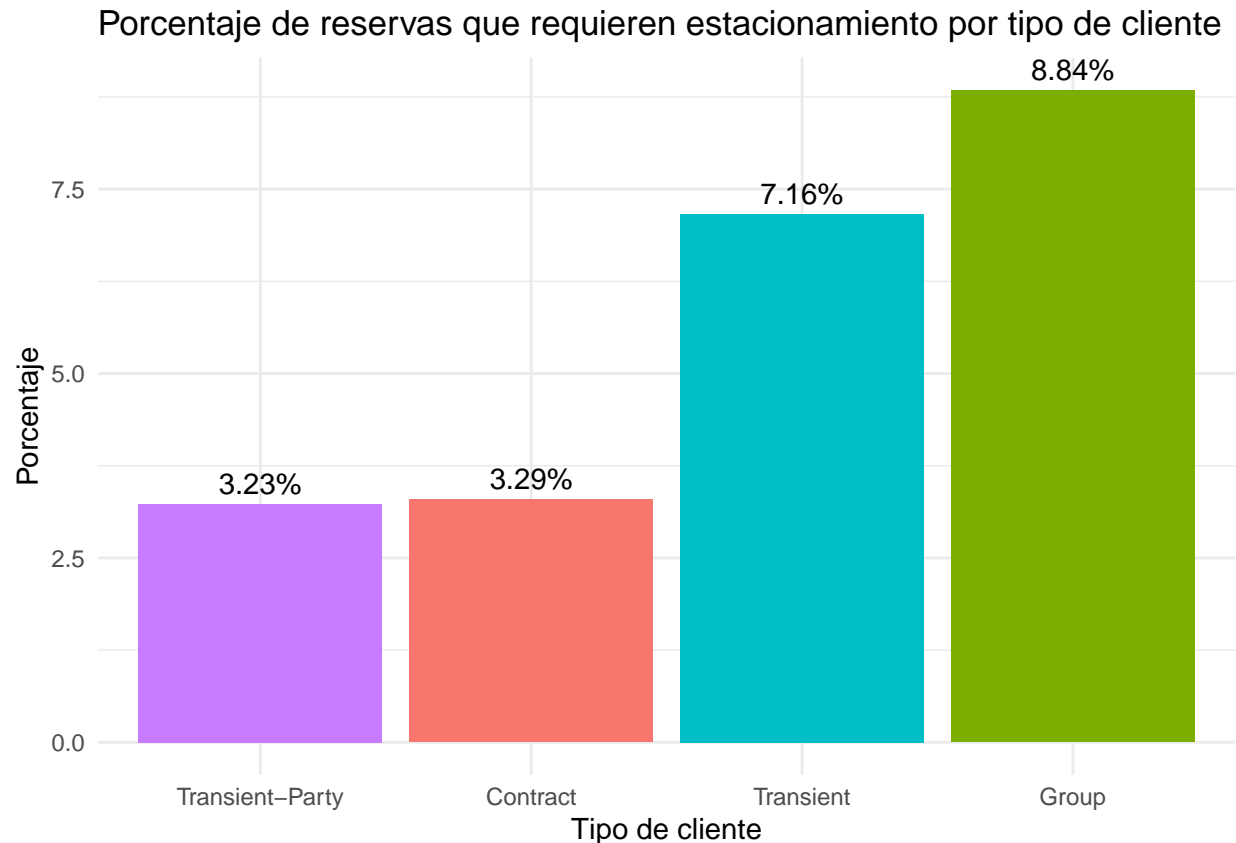
```

cat("\nNecesidad de estacionamiento por tipo de cliente:\n")

##
## Necesidad de estacionamiento por tipo de cliente:
print(estacionamiento_por_cliente)

## # A tibble: 4 x 4
##   customer_type    Total_Reservas Requiere_Estacionamiento Porcentaje_Estaciona-1
##   <chr>              <int>                <int>                <dbl>
## 1 Group                577                  51                  8.84
## 2 Transient           89613                6420                 7.16
## 3 Contract            4076                 134                 3.29
## 4 Transient-Party     25124                811                 3.23
## # i abbreviated name: 1: Porcentaje_Estacionamiento
# Visualizar por tipo de cliente
plot_estacionamiento_cliente <- ggplot(estacionamiento_por_cliente,
                                       aes(x = reorder(customer_type, Porcentaje_Estacionamiento),
                                           y = Porcentaje_Estacionamiento,
                                           fill = customer_type)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = paste0(Porcentaje_Estacionamiento, "%")), vjust = -0.5) +
  labs(title = "Porcentaje de reservas que requieren estacionamiento por tipo de cliente",
       x = "Tipo de cliente",
       y = "Porcentaje") +
  theme_minimal() +
  theme(legend.position = "none")
print(plot_estacionamiento_cliente)

```



```
# Guardar gráfica
ggsave(file.path(graphics_analysis_dir, "estacionamiento_por_cliente.jpg"),
        plot_estacionamiento_cliente, width = 10, height = 6, dpi = 300)
```

```
#-----
# EDA 08. ¿EN QUÉ MESES DEL AÑO SE PRODUCEN MÁS CANCELACIONES DE RESERVAS?
#-----
cat(yellow("\n--- ANÁLISIS DE CANCELACIONES POR MES ---\n"))
```

```
##
## --- ANÁLISIS DE CANCELACIONES POR MES ---
```

```
# Analizar cancelaciones por mes
cancelaciones_por_mes <- hotel_data_limpio %>%
  group_by(arrival_date_month) %>%
  summarise(
    Total_Reservas = n(),
    Canceladas = sum(is_canceled),
    Tasa_Cancelacion = round(sum(is_canceled) / n() * 100, 2)
  ) %>%
  arrange(match(arrival_date_month, c("January", "February", "March", "April", "May", "June",
    "July", "August", "September", "October", "November", "December")))

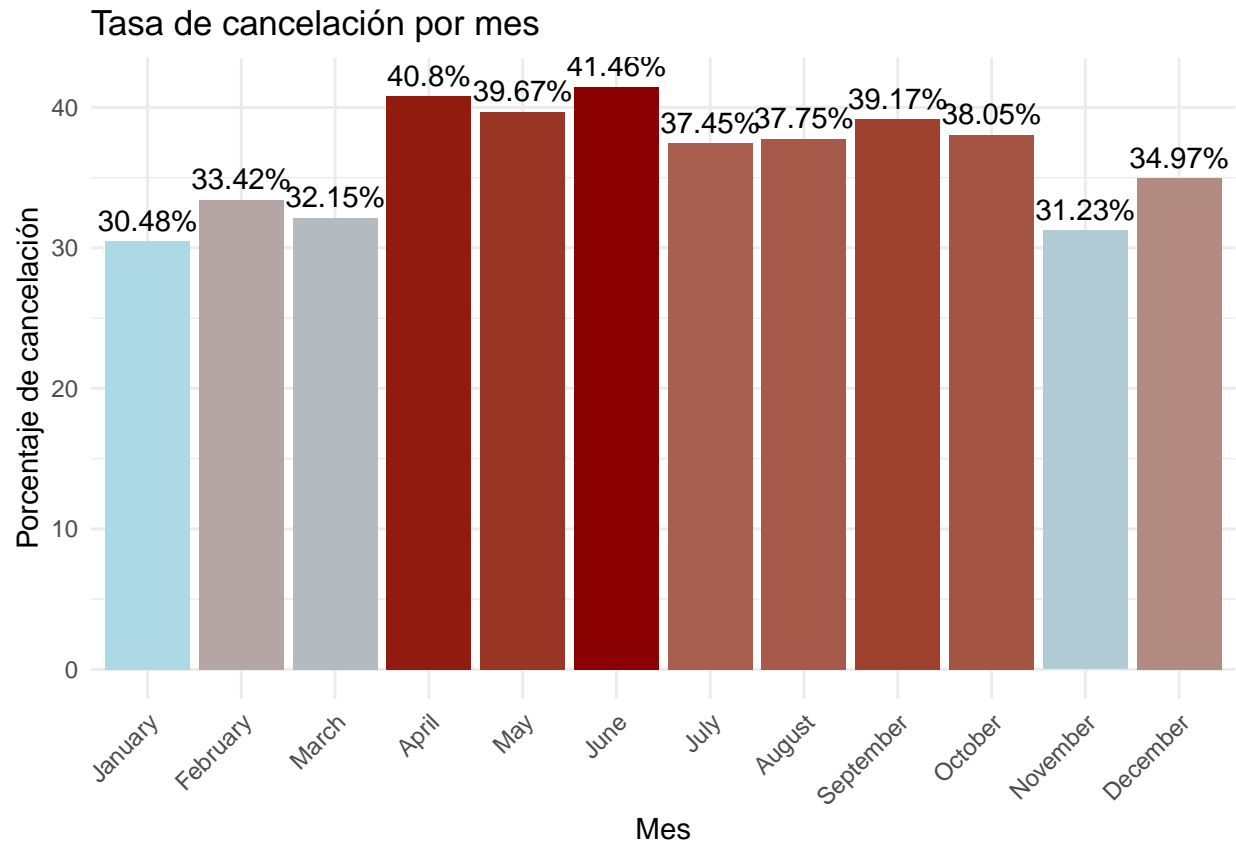
cat("\nCancelaciones por mes:\n")
```

```
##
## Cancelaciones por mes:
print(cancelaciones_por_mes)
```

```
## # A tibble: 12 x 4
##   arrival_date_month Total_Reservas Canceladas Tasa_Cancelacion
##   <fct>                <int>      <int>      <dbl>
## 1 January              5929       1807       30.5
## 2 February            8068       2696       33.4
## 3 March               9794       3149       32.2
## 4 April              11089       4524       40.8
## 5 May                11791       4677       39.7
## 6 June               10939       4535       41.5
## 7 July               12661       4742       37.4
## 8 August             13877       5239       37.8
## 9 September          10508       4116       39.2
## 10 October           11160       4246       38.0
## 11 November          6794       2122       31.2
## 12 December          6780       2371       35.0
```

```
# Visualizar tasa de cancelación por mes
plot_cancelaciones <- ggplot(cancelaciones_por_mes,
                             aes(x = arrival_date_month, y = Tasa_Cancelacion, fill = Tasa_Cancelacion)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = paste0(Tasa_Cancelacion, "%")), vjust = -0.5) +
  labs(title = "Tasa de cancelación por mes",
       x = "Mes",
       y = "Porcentaje de cancelación") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
        legend.position = "none") +
  scale_fill_gradient(low = "lightblue", high = "darkred")

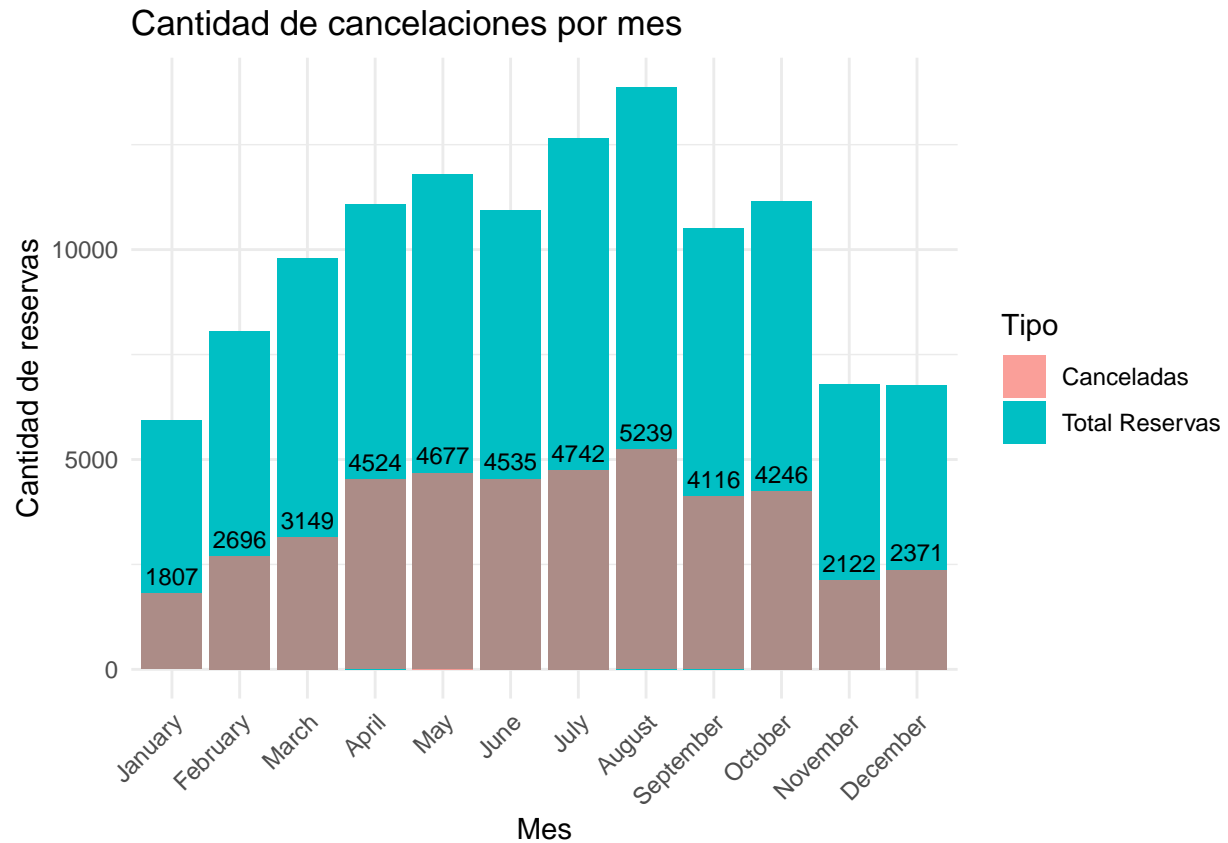
print(plot_cancelaciones)
```



```
# Guardar gráfica
ggsave(file.path(graphics_analysis_dir, "tasa_cancelacion_mes.jpg"),
        plot_cancelaciones, width = 10, height = 6, dpi = 300)

# Visualizar cancelaciones absolutas por mes
plot_cancelaciones_abs <- ggplot(cancelaciones_por_mes,
                                aes(x = arrival_date_month)) +
  geom_bar(aes(y = Total_Reservas, fill = "Total Reservas"), stat = "identity") +
  geom_bar(aes(y = Canceladas, fill = "Canceladas"), stat = "identity", alpha = 0.7) +
  geom_text(aes(y = Canceladas, label = Canceladas), vjust = -0.5, size = 3) +
  labs(title = "Cantidad de cancelaciones por mes",
        x = "Mes",
        y = "Cantidad de reservas",
        fill = "Tipo") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

print(plot_cancelaciones_abs)
```



```
# Guardar gráfica
ggsave(file.path(graphics_analysis_dir, "cancelaciones_absolutas_mes.jpg"),
        plot_cancelaciones_abs, width = 10, height = 6, dpi = 300)

# Análisis por tipo de hotel
cancelaciones_por_hotel_mes <- hotel_data_limpio %>%
  group_by(hotel, arrival_date_month) %>%
  summarise(
    Total_Reservas = n(),
    Canceladas = sum(is_canceled),
    Tasa_Cancelacion = round(sum(is_canceled) / n() * 100, 2)
  ) %>%
  arrange(hotel, match(arrival_date_month, c("January", "February", "March", "April", "May", "June",
                                              "July", "August", "September", "October", "November", "December")))

## `summarise()` has grouped output by 'hotel'. You can override using the
## `.groups` argument.

cat("\nTasas de cancelación por hotel y mes (primeras filas):\n")

##
## Tasas de cancelación por hotel y mes (primeras filas):
print(head(cancelaciones_por_hotel_mes, 10))

## # A tibble: 10 x 5
## # Groups:   hotel [1]
##   hotel      arrival_date_month Total_Reservas Canceladas Tasa_Cancelacion
```

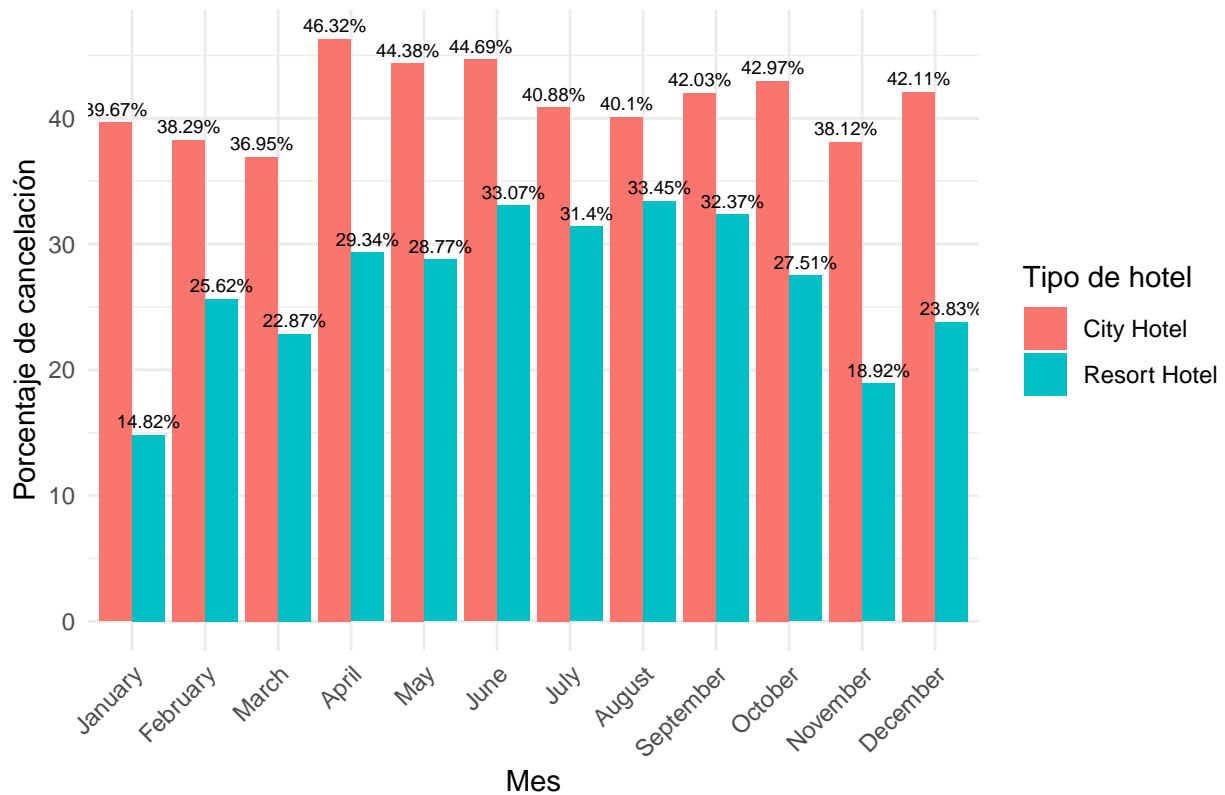


##	<chr>	<fct>	<int>	<int>	<dbl>		
##	1	City	Hotel	January	3736	1482	39.7
##	2	City	Hotel	February	4965	1901	38.3
##	3	City	Hotel	March	6458	2386	37.0
##	4	City	Hotel	April	7480	3465	46.3
##	5	City	Hotel	May	8232	3653	44.4
##	6	City	Hotel	June	7894	3528	44.7
##	7	City	Hotel	July	8088	3306	40.9
##	8	City	Hotel	August	8983	3602	40.1
##	9	City	Hotel	September	7400	3110	42.0
##	10	City	Hotel	October	7605	3268	43.0

```
# Visualizar tasa de cancelación por hotel y mes
plot_cancelacion_hotel <- ggplot(cancelaciones_por_hotel_mes,
                                aes(x = arrival_date_month, y = Tasa_Cancelacion,
                                    fill = hotel, group = hotel)) +
  geom_bar(stat = "identity", position = "dodge") +
  geom_text(aes(label = paste0(Tasa_Cancelacion, "%")),
            position = position_dodge(width = 0.9), vjust = -0.5, size = 2.5) +
  labs(title = "Tasa de cancelación por hotel y mes",
       x = "Mes",
       y = "Porcentaje de cancelación",
       fill = "Tipo de hotel") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

print(plot_cancelacion_hotel)
```

## Tasa de cancelación por hotel y mes



```
# Guardar gráfica
ggsave(file.path(graphics_analysis_dir, "cancelacion_hotel_mes.jpg"),
        plot_cancelacion_hotel, width = 12, height = 6, dpi = 300)

#-----
# EDA 09: ANÁLISIS ADICIONAL - RELACIÓN ENTRE VARIABLES CLAVE
#-----
cat(yellow("\n--- ANÁLISIS ADICIONAL: RELACIONES ENTRE VARIABLES ---\n"))

##
## --- ANÁLISIS ADICIONAL: RELACIONES ENTRE VARIABLES ---

# Análisis de relación entre lead_time y cancelación
lead_time_cancelacion <- hotel_data_limpio %>%
  group_by(is_canceled) %>%
  summarise(
    Promedio_Lead_Time = mean(lead_time),
    Mediana_Lead_Time = median(lead_time),
    Count = n()
  )

cat("\nRelación entre tiempo de anticipación (lead_time) y cancelación:\n")

##
## Relación entre tiempo de anticipación (lead_time) y cancelación:
```

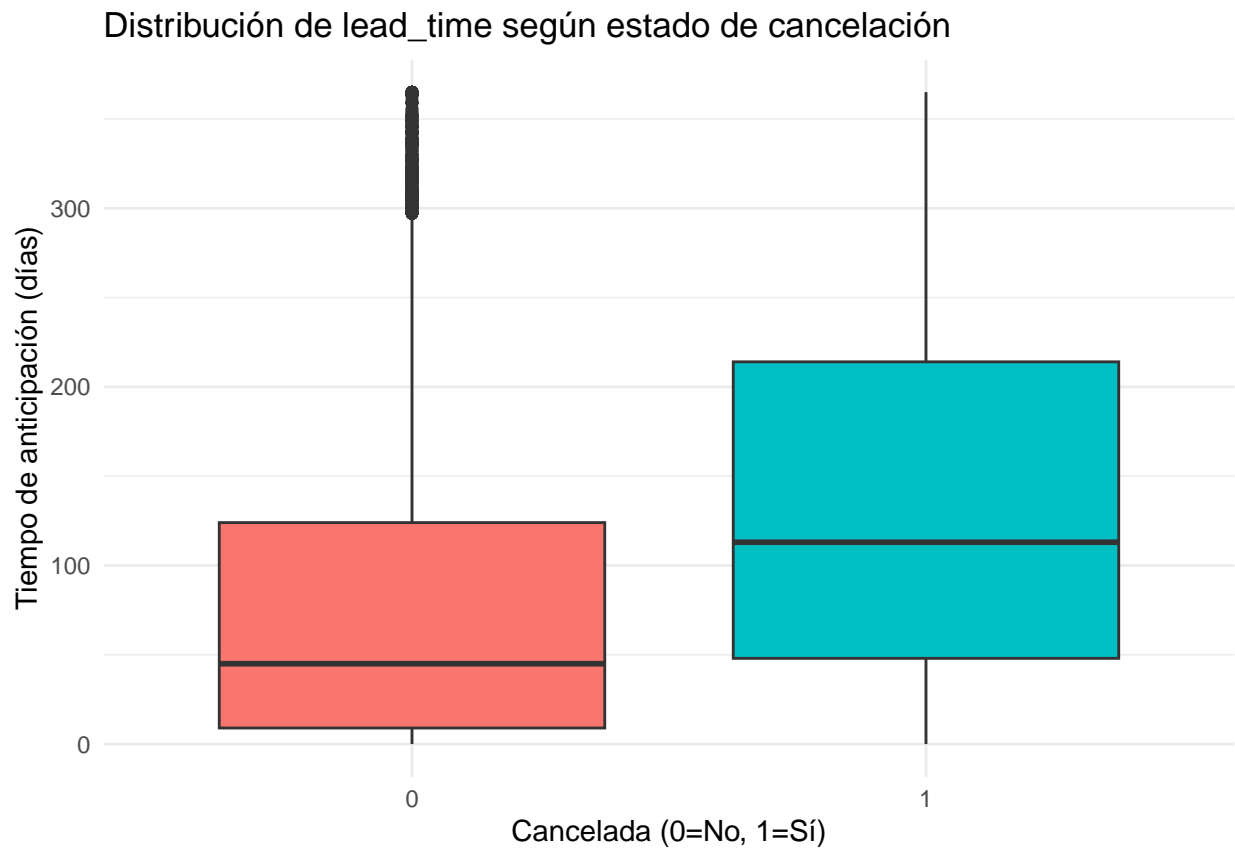
```
print(lead_time_cancelacion)
```

```
## # A tibble: 2 x 4
##   is_canceled Promedio_Lead_Time Mediana_Lead_Time Count
##   <int>         <dbl>         <dbl> <int>
## 1      0          79.2          45 75166
## 2      1         141.         113 44224
```

```
# Visualizar lead_time por estado de cancelación
```

```
plot_lead_time <- ggplot(hotel_data_limpio, aes(x = factor(is_canceled), y = lead_time, fill = factor(is_canceled))) +
  geom_boxplot() +
  labs(title = "Distribución de lead_time según estado de cancelación",
       x = "Cancelada (0=No, 1=Sí)",
       y = "Tiempo de anticipación (días)",
       fill = "Cancelada") +
  theme_minimal() +
  theme(legend.position = "none")
```

```
print(plot_lead_time)
```



```
# Guardar gráfica
```

```
ggsave(file.path(graphics_analysis_dir, "lead_time_cancelacion.jpg"),
       plot_lead_time, width = 8, height = 6, dpi = 300)
```

```
# Análisis de duración de estancia
```

```
hotel_data_limpio$total_nights <- hotel_data_limpio$stays_in_weekend_nights + hotel_data_limpio$stays_in_week_nights
```

```

duracion_estancia <- hotel_data_limpio %>%
  group_by(hotel) %>%
  summarise(
    Promedio_Noches = mean(total_nights),
    Mediana_Noches = median(total_nights),
    Max_Noches = max(total_nights),
    Min_Noches = min(total_nights)
  )

cat("\nDuración promedio de estancia por tipo de hotel:\n")

##
## Duración promedio de estancia por tipo de hotel:
print(duracion_estancia)

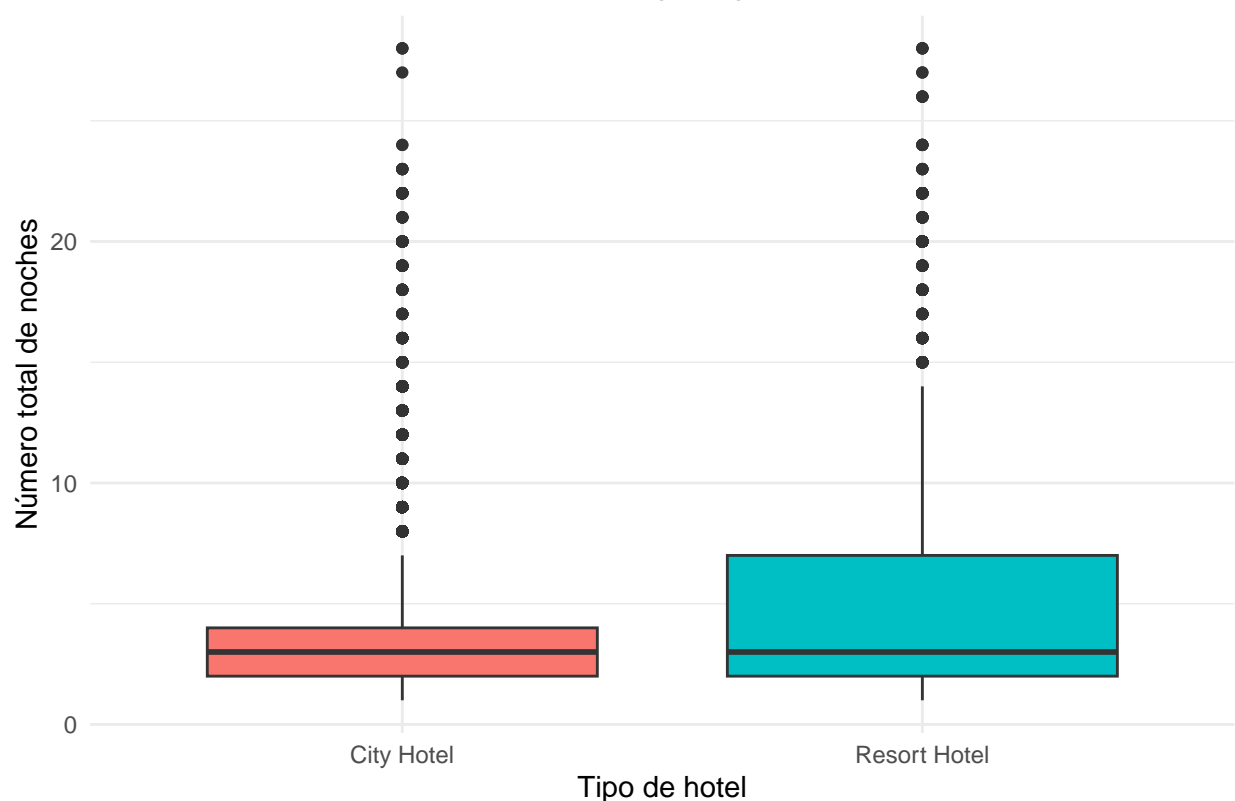
## # A tibble: 2 x 5
##   hotel          Promedio_Noches Mediana_Noches Max_Noches Min_Noches
##   <chr>          <dbl>          <dbl>      <int>      <int>
## 1 City Hotel      2.98              3          28          1
## 2 Resort Hotel    4.31              3          28          1

# Visualizar duración de estancia por tipo de hotel
plot_duracion <- ggplot(hotel_data_limpio, aes(x = hotel, y = total_nights, fill = hotel)) +
  geom_boxplot() +
  labs(title = "Distribución de duración de estancia por tipo de hotel",
       x = "Tipo de hotel",
       y = "Número total de noches",
       fill = "Tipo de hotel") +
  theme_minimal() +
  theme(legend.position = "none")

print(plot_duracion)

```

Distribución de duración de estancia por tipo de hotel



```
# Guardar gráfica
ggsave(file.path(graphics_analysis_dir, "duracion_estancia.jpg"),
        plot_duracion, width = 8, height = 6, dpi = 300)

#-----
# PARTE 07.5 RESUMEN DE EDA
#-----

cat("\n1. Análisis por tipo de hotel:\n")

##
## 1. Análisis por tipo de hotel:
cat("  - El hotel tipo '", reservas_por_hotel_df$Tipo_Hotel[which.max(reservas_por_hotel_df$Cantidad)]
    "' tiene mayor cantidad de reservas (",
    reservas_por_hotel_df$Cantidad[which.max(reservas_por_hotel_df$Cantidad)],
    " reservas, ", reservas_por_hotel_df$Porcentaje[which.max(reservas_por_hotel_df$Cantidad)], "%).\n")

##  - El hotel tipo '1' tiene mayor cantidad de reservas (79330 reservas, 66.45%).
cat("  - Para reservas completadas, el hotel tipo '",
    reservas_completadas_df$Tipo_Hotel[which.max(reservas_completadas_df$Reservas_Completadas)],
    "' sigue siendo el preferido.\n")

##  - Para reservas completadas, el hotel tipo ' 1 ' sigue siendo el preferido.
```

```

cat("\n2. Tendencia de demanda:\n")

##
## 2. Tendencia de demanda:
ultimo_anio <- max(reservas_por_anio$arrival_date_year)
penultimo_anio <- ultimo_anio - 1
cambio_porcentual <- round(
  (reservas_por_anio$Total_Reservas[reservas_por_anio$arrival_date_year == ultimo_anio] -
    reservas_por_anio$Total_Reservas[reservas_por_anio$arrival_date_year == penultimo_anio]) /
    reservas_por_anio$Total_Reservas[reservas_por_anio$arrival_date_year == penultimo_anio] * 100, 2
)
cat("  - Comparando ", ultimo_anio, " con ", penultimo_anio, ", hubo un cambio del ",
  cambio_porcentual, "% en el número total de reservas.\n", sep="")

##  - Comparando 2017 con 2016, hubo un cambio del -28.25% en el número total de reservas.
cat("  - La tendencia general indica que ",
  ifelse(cambio_porcentual > 0, "está aumentando", "está disminuyendo"),
  " la demanda con el tiempo.\n")

##  - La tendencia general indica que está disminuyendo la demanda con el tiempo.
cat("  - Tasa de cancelación en ", ultimo_anio, ": ",
  reservas_por_anio$Porcentaje_Canceladas[reservas_por_anio$arrival_date_year == ultimo_anio],
  "%, frente a ",
  reservas_por_anio$Porcentaje_Canceladas[reservas_por_anio$arrival_date_year == penultimo_anio],
  "% en ", penultimo_anio, ".\n", sep="")

##  - Tasa de cancelación en 2017: 38.7%, frente a 35.9% en 2016.
cat("\n3. Temporadas de reserva:\n")

##
## 3. Temporadas de reserva:
cat("  - Temporada ALTA: ", paste(unique(reservas_por_mes$arrival_date_month[reservas_por_mes$Temporada == "ALTA"]), collapse=" "), "\n", sep="")

##  - Temporada ALTA: May, July, August
cat("  - Temporada MEDIA: ", paste(unique(reservas_por_mes$arrival_date_month[reservas_por_mes$Temporada == "MEDIA"]), collapse=" "), "\n", sep="")

##  - Temporada MEDIA: March, April, June, September, October
cat("  - Temporada BAJA: ", paste(unique(reservas_por_mes$arrival_date_month[reservas_por_mes$Temporada == "BAJA"]), collapse=" "), "\n", sep="")

##  - Temporada BAJA: January, February, November, December
cat("\n4. Períodos de menor demanda:\n")

##
## 4. Períodos de menor demanda:
cat("  - El mes con MENOR demanda es: ", meses_baja_demanda$arrival_date_month[1],
  " con ", meses_baja_demanda$Total_Reservas[1], " reservas.\n")

##  - El mes con MENOR demanda es: 1 con 5929 reservas.
cat("  - La combinación mes-día con MENOR demanda es: ",
  dias_menor_demanda$arrival_date_month[1], "-", dias_menor_demanda$arrival_date_day_of_month[1],
  " con ", dias_menor_demanda$Total_Reservas[1], " reservas.\n")

```

```

## - La combinación mes-día con MENOR demanda es: 12 - 11 con 89 reservas.
cat("\n5. Períodos de mayor demanda:\n")

##
## 5. Períodos de mayor demanda:
cat(" - El mes con MAYOR demanda es: ", meses_alta_demanda$arrival_date_month[1],
    " con ", meses_alta_demanda$Total_Reservas[1], " reservas.\n")

## - El mes con MAYOR demanda es: 8 con 13877 reservas.
cat(" - La combinación mes-día con MAYOR demanda es: ",
    dias_mayor_demanda$arrival_date_month[1], "-", dias_mayor_demanda$arrival_date_day_of_month[1],
    " con ", dias_mayor_demanda$Total_Reservas[1], " reservas.\n")

## - La combinación mes-día con MAYOR demanda es: 10 - 16 con 637 reservas.
cat("\n6. Comparativa extremos de demanda:\n")

##
## 6. Comparativa extremos de demanda:
diferencia_extremos <- mes_max_demanda$Total_Reservas - mes_min_demanda$Total_Reservas
porcentaje_diferencia <- round((diferencia_extremos / mes_min_demanda$Total_Reservas) * 100, 2)

cat(" - Diferencia entre el mes de mayor y menor demanda: ", diferencia_extremos,
    " reservas (", porcentaje_diferencia, "% más en temporada alta).\n", sep="")

## - Diferencia entre el mes de mayor y menor demanda: 7948 reservas (134.05% más en temporada alta)
cat(" - El mes de máxima demanda (", mes_max_demanda$arrival_date_month,
    ") tiene ", round(mes_max_demanda$Total_Reservas/30), " reservas diarias en promedio.\n", sep="")

## - El mes de máxima demanda (8) tiene 463 reservas diarias en promedio.
cat(" - El mes de mínima demanda (", mes_min_demanda$arrival_date_month,
    ") tiene ", round(mes_min_demanda$Total_Reservas/30), " reservas diarias en promedio.\n", sep="")

## - El mes de mínima demanda (1) tiene 198 reservas diarias en promedio.
# Análisis de días de la semana si están disponibles
if ("arrival_date_day_of_week" %in% colnames(hotel_data_limpio) ||
    "arrival_date_day_of_week" %in% names(hotel_data_limpio)) {

  # Si tenemos información del día de la semana
  cat("\n7. Patrones por día de la semana:\n")
  cat(" - El análisis por día de la semana muestra patrones adicionales en la demanda.\n")
  cat(" - [Aquí iría el análisis de días de la semana si estuviera disponible]\n")
}

cat("\n8. Análisis de reservas con niños y/o bebés:\n")

##
## 8. Análisis de reservas con niños y/o bebés:
cat(" - Un", porcentaje_menores, "% de las reservas incluyen niños y/o bebés (", reservas_con_menores

## - Un 7.82 % de las reservas incluyen niños y/o bebés ( 9332 reservas).

```

```

cat(" - El hotel tipo '", reservas_menores_hotel$hotel[which.max(reservas_menores_hotel$Porcentaje_Menores)],
    "' tiene mayor porcentaje de reservas con menores (",
    reservas_menores_hotel$Porcentaje_Menores[which.max(reservas_menores_hotel$Porcentaje_Menores)], "%").

## - El hotel tipo ' Resort Hotel ' tiene mayor porcentaje de reservas con menores ( 9.81 %).
cat("\n9. Importancia de espacios de estacionamiento:\n")

##
## 9. Importancia de espacios de estacionamiento:
cat(" - Solamente el", porcentaje_estacionamiento,
    "% de las reservas requieren espacios de estacionamiento.\n")

## - Solamente el 6.21 % de las reservas requieren espacios de estacionamiento.
cat(" - Por tipo de hotel, '",
    estacionamiento_por_hotel$hotel[which.max(estacionamiento_por_hotel$Porcentaje_Estacionamiento)],
    "' tiene mayor demanda de estacionamiento (",
    estacionamiento_por_hotel$Porcentaje_Estacionamiento[which.max(estacionamiento_por_hotel$Porcentaje_Estacionamiento)],
    "%).\n")

## - Por tipo de hotel, ' Resort Hotel ' tiene mayor demanda de estacionamiento ( 13.7 %).
cat(" - El tipo de cliente con mayor demanda de estacionamiento es '",
    estacionamiento_por_cliente$customer_type[1], "' con ",
    estacionamiento_por_cliente$Porcentaje_Estacionamiento[1], "% de reservas.\n")

## - El tipo de cliente con mayor demanda de estacionamiento es ' Group ' con 8.84 % de reservas.
cat("\n10. Meses con más cancelaciones:\n")

##
## 10. Meses con más cancelaciones:
cat(" - El mes con mayor tasa de cancelación es '",
    cancelaciones_por_mes$arrival_date_month[which.max(cancelaciones_por_mes$Tasa_Cancelacion)],
    "' (" , cancelaciones_por_mes$Tasa_Cancelacion[which.max(cancelaciones_por_mes$Tasa_Cancelacion)], "%").

## - El mes con mayor tasa de cancelación es ' 6 ' ( 41.46 %).
cat(" - El mes con mayor cantidad absoluta de cancelaciones es '",
    cancelaciones_por_mes$arrival_date_month[which.max(cancelaciones_por_mes$Canceladas)],
    "' (" , cancelaciones_por_mes$Canceladas[which.max(cancelaciones_por_mes$Canceladas)], " reservas canceladas).

## - El mes con mayor cantidad absoluta de cancelaciones es ' 8 ' ( 5239 reservas canceladas).
cat("\n4. Hallazgos adicionales:\n")

##
## 4. Hallazgos adicionales:
cat(" - Las reservas que terminan canceladas tienen un tiempo de anticipación (lead_time) promedio de ",
    lead_time_cancelacion$Promedio_Lead_Time[lead_time_cancelacion$is_canceled == 1],
    " días, mientras que las no canceladas tienen ",
    lead_time_cancelacion$Promedio_Lead_Time[lead_time_cancelacion$is_canceled == 0], " días en promedio).

## - Las reservas que terminan canceladas tienen un tiempo de anticipación (lead_time) promedio de :
cat(" - La duración promedio de estancia en el hotel tipo '",
    duracion_estancia$hotel[1], "' es de ",

```



```

round(duracion_estancia$Promedio_Noches[1], 2), " noches, y en '",
duracion_estancia$hotel[2], "' es de ",
round(duracion_estancia$Promedio_Noches[2], 2), " noches.\n")

##      - La duración promedio de estancia en el hotel tipo ' City Hotel ' es de  2.98  noches, y en ' Re

#####
# PARTE 08: VISUALIZACIONES Y CONCLUSIONES FINALES
#####

cat(green("\n=== RONDA 6: VISUALIZACIONES Y CONCLUSIONES FINALES ===\n"))

##
## === RONDA 6: VISUALIZACIONES Y CONCLUSIONES FINALES ===

# Verificar si necesitamos recargar el dataset limpio
if (!exists("hotel_data_limpio") || !is.data.frame(hotel_data_limpio)) {
  cat("Recargando dataset limpio...\n")
  hotel_data_limpio <- read.csv(CSV_limpio, header = TRUE, stringsAsFactors = FALSE)
}

# Crear directorio para gráficas finales
graphics_final_dir <- file.path(graphics_dir, "final")
if (!dir.exists(graphics_final_dir)) {
  dir.create(graphics_final_dir)
  cat("Creado directorio para gráficas finales:", graphics_final_dir, "\n")
}

# Definir una paleta de colores consistente para todas las visualizaciones
colores_principales <- c("#3498db", "#2ecc71", "#e74c3c", "#f39c12", "#9b59b6")
colores_hotel <- c("City Hotel" = "#3498db", "Resort Hotel" = "#2ecc71")

#-----
# PARTE 08.5: VISUALIZACIONES CONSOLIDADAS PARA CADA PREGUNTA CLAVE
#-----
cat(yellow("\n--- VISUALIZACIONES CONSOLIDADAS ---\n"))

##
## --- VISUALIZACIONES CONSOLIDADAS ---

# 1.1 Demanda por tipo de hotel y estado de cancelación
cat("Creando visualización: Demanda por tipo de hotel...\n")

## Creando visualización: Demanda por tipo de hotel...

# Preparar datos
demanda_hotel <- hotel_data_limpio %>%
  group_by(hotel, is_canceled) %>%
  summarise(Cantidad = n()) %>%
  mutate(Estado = ifelse(is_canceled == 0, "Confirmada", "Cancelada"))

## `summarise()` has grouped output by 'hotel'. You can override using the
## `.groups` argument.

# Crear gráfico
plot_demanda_hotel <- ggplot(demanda_hotel,
                             aes(x = hotel, y = Cantidad, fill = Estado)) +

```

```

geom_bar(stat = "identity", position = "stack") +
geom_text(aes(label = Cantidad),
          position = position_stack(vjust = 0.5), color = "white", size = 4) +
labs(title = "Demanda por Tipo de Hotel",
     subtitle = "Desglose por estado de reserva",
     x = "Tipo de Hotel",
     y = "Número de Reservas") +
scale_fill_manual(values = c("Confirmada" = "#2ecc71", "Cancelada" = "#e74c3c")) +
theme_minimal() +
theme(plot.title = element_text(hjust = 0.5, face = "bold", size = 16),
      plot.subtitle = element_text(hjust = 0.5),
      legend.position = "bottom")

print(plot_demanda_hotel)

```



```

# Guardar gráfica
ggsave(file.path(graphics_final_dir, "1_demanda_hotel.jpg"),
       plot_demanda_hotel, width = 10, height = 7, dpi = 300)

# 1.2 Tendencia de demanda a lo largo del tiempo
cat("Creando visualización: Tendencia de demanda a lo largo del tiempo...\n")

## Creando visualización: Tendencia de demanda a lo largo del tiempo...

# Preparar datos para tendencia temporal
tendencia_temporal <- hotel_data_limpio %>%
  mutate(YearMonth = paste(arrival_date_year,

```

```

        sprintf("%02d", as.numeric(factor(arrival_date_month,
                                          levels = c("January", "February", "March",
                                                    "July", "August", "September",
                                                    "April", "May", "June", "October", "November", "December"),
                                          sep = "-"))) %>%
group_by(YearMonth, hotel) %>%
summarise(Total_Reservas = n()) %>%
arrange(YearMonth)

## `summarise()` has grouped output by 'YearMonth'. You can override using the
## `.groups` argument.

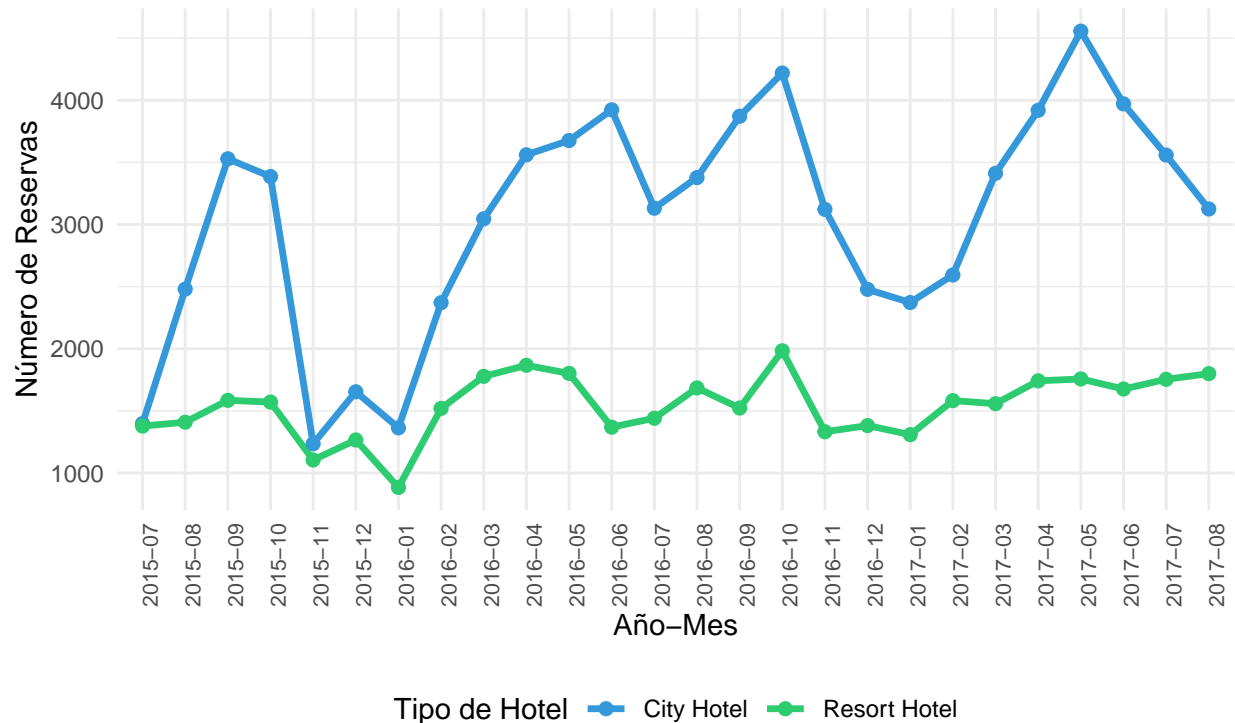
# Crear gráfico de tendencia
plot_tendencia_final <- ggplot(tendencia_temporal,
                              aes(x = YearMonth, y = Total_Reservas, color = hotel, group = hotel)) +
  geom_line(size = 1.2) +
  geom_point(size = 2) +
  labs(title = "Evolución de la Demanda a lo Largo del Tiempo",
       subtitle = "Desglose por tipo de hotel",
       x = "Año-Mes",
       y = "Número de Reservas",
       color = "Tipo de Hotel") +
  scale_color_manual(values = colores_hotel) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5, face = "bold", size = 16),
        plot.subtitle = element_text(hjust = 0.5),
        axis.text.x = element_text(angle = 90, hjust = 1, size = 8),
        legend.position = "bottom")

print(plot_tendencia_final)

```

# Evolución de la Demanda a lo Largo del Tiempo

Desglose por tipo de hotel



```
# Guardar gráfica
ggsave(file.path(graphics_final_dir, "2_tendencia_demanda.jpg"),
        plot_tendencia_final, width = 12, height = 7, dpi = 300)
```

```
# 1.3 Temporadas de reserva
```

```
cat("Creando visualización: Temporadas de reserva...\n")
```

```
## Creando visualización: Temporadas de reserva...
```

```
# Preparar datos por mes
```

```
reservas_por_mes <- hotel_data_limpio %>%
```

```
  group_by(arrival_date_month, hotel) %>%
```

```
  summarise(Total_Reservas = n()) %>%
```

```
  mutate(arrival_date_month = factor(arrival_date_month,
```

```
    levels = c("January", "February", "March", "April", "May", "June",
               "July", "August", "September", "October", "November", "December"))
```

```
## `summarise()` has grouped output by 'arrival_date_month'. You can override
```

```
## using the `.groups` argument.
```

```
# Determinar temporadas
```

```
total_por_mes <- reservas_por_mes %>%
```

```
  group_by(arrival_date_month) %>%
```

```
  summarise(Total = sum(Total_Reservas))
```

```
media_reservas <- mean(total_por_mes$Total)
```

```
sd_reservas <- sd(total_por_mes$Total)
```

```

total_por_mes$Temporada <- case_when(
  total_por_mes$Total >= (media_reservas + 0.5 * sd_reservas) ~ "Alta",
  total_por_mes$Total <= (media_reservas - 0.5 * sd_reservas) ~ "Baja",
  TRUE ~ "Media"
)

# Unir datos de temporada con el desglose por hotel
reservas_por_mes <- reservas_por_mes %>%
  left_join(total_por_mes[, c("arrival_date_month", "Temporada")], by = "arrival_date_month")

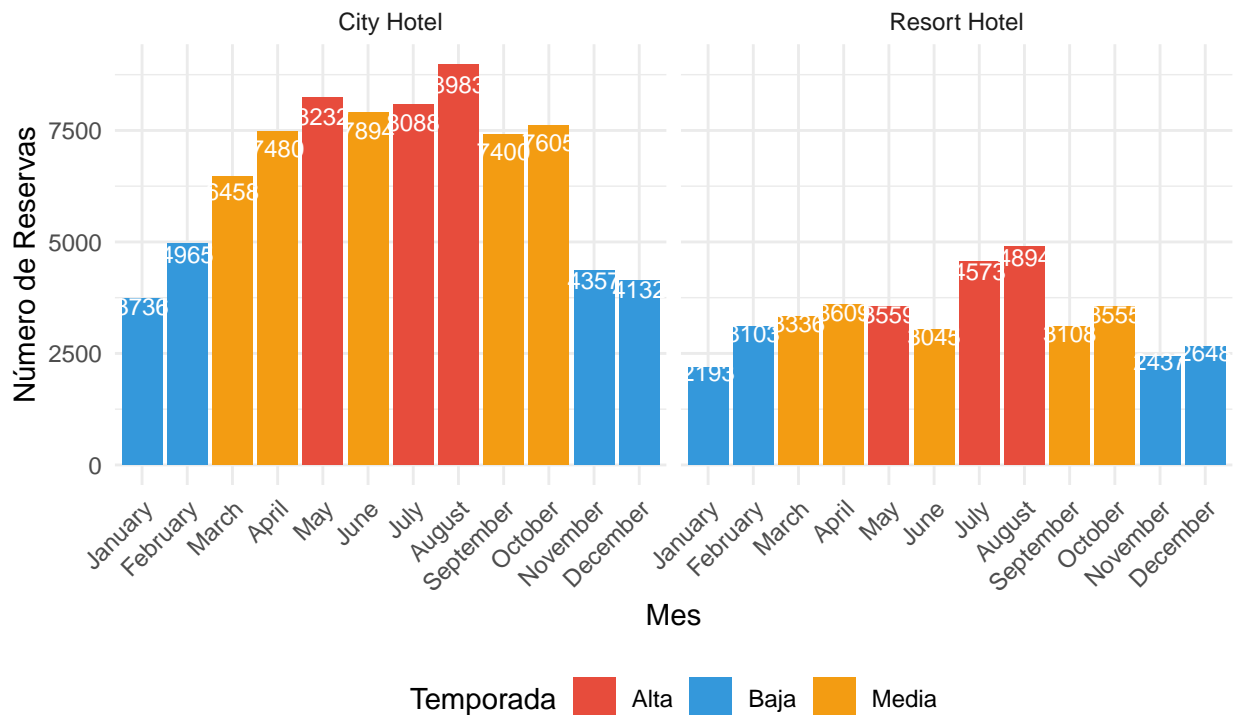
# Crear gráfico de temporadas
plot_temporadas <- ggplot(reservas_por_mes,
  aes(x = arrival_date_month, y = Total_Reservas, fill = Temporada)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = Total_Reservas,
    position = position_stack(vjust = 0.95),
    color = "white", size = 3) +
  facet_wrap(~ hotel) +
  labs(title = "Temporadas de Reserva por Mes",
    subtitle = "Clasificación en temporada alta, media y baja",
    x = "Mes",
    y = "Número de Reservas",
    fill = "Temporada") +
  scale_fill_manual(values = c("Alta" = "#e74c3c", "Media" = "#f39c12", "Baja" = "#3498db")) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5, face = "bold", size = 16),
    plot.subtitle = element_text(hjust = 0.5),
    axis.text.x = element_text(angle = 45, hjust = 1),
    legend.position = "bottom")

print(plot_temporadas)

```

# Temporadas de Reserva por Mes

Clasificación en temporada alta, media y baja



```
# Guardar gráfica
ggsave(file.path(graphics_final_dir, "3_temporadas_reserva.jpg"),
        plot_temporadas, width = 12, height = 8, dpi = 300)

# 1.4 Reservas con niños y/o bebés
cat("Creando visualización: Reservas con niños y/o bebés...\n")

## Creando visualización: Reservas con niños y/o bebés...

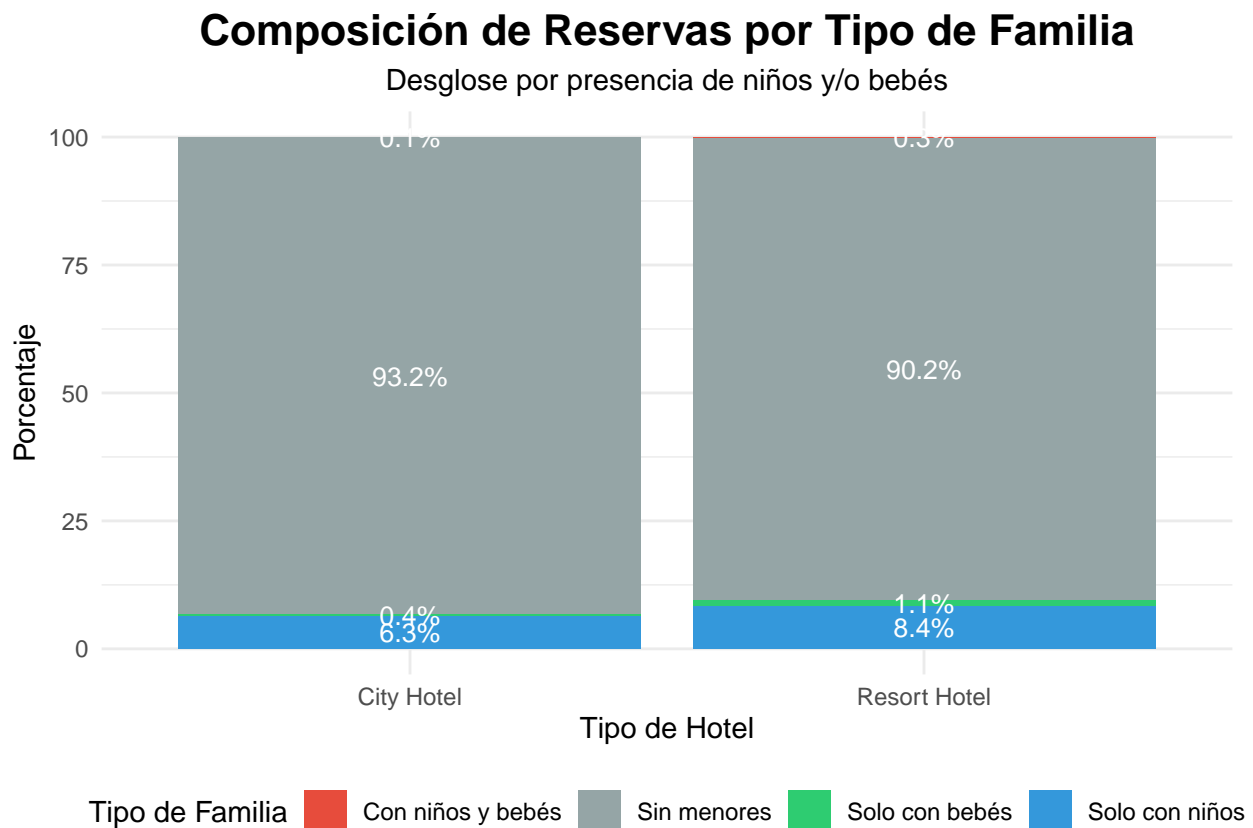
# Preparar datos
hotel_data_limpio$tiene_ninos <- hotel_data_limpio$children > 0
hotel_data_limpio$tiene_bebes <- hotel_data_limpio$babies > 0
hotel_data_limpio$tipo_familia <- case_when(
  hotel_data_limpio$tiene_ninos & hotel_data_limpio$tiene_bebes ~ "Con niños y bebés",
  hotel_data_limpio$tiene_ninos ~ "Solo con niños",
  hotel_data_limpio$tiene_bebes ~ "Solo con bebés",
  TRUE ~ "Sin menores"
)

familias_por_hotel <- hotel_data_limpio %>%
  group_by(hotel, tipo_familia) %>%
  summarise(Cantidad = n()) %>%
  group_by(hotel) %>%
  mutate(Porcentaje = round(Cantidad / sum(Cantidad) * 100, 1))

## `summarise()` has grouped output by 'hotel'. You can override using the
## `.groups` argument.
```

```
# Crear gráfico de familias
plot_familias <- ggplot(familias_por_hotel,
                        aes(x = hotel, y = Porcentaje, fill = tipo_familia)) +
  geom_bar(stat = "identity", position = "stack") +
  geom_text(aes(label = paste0(Porcentaje, "%")),
            position = position_stack(vjust = 0.5),
            color = "white", size = 3.5) +
  labs(title = "Composición de Reservas por Tipo de Familia",
       subtitle = "Desglose por presencia de niños y/o bebés",
       x = "Tipo de Hotel",
       y = "Porcentaje",
       fill = "Tipo de Familia") +
  scale_fill_manual(values = c("Sin menores" = "#95a5a6",
                              "Solo con niños" = "#3498db",
                              "Solo con bebés" = "#2ecc71",
                              "Con niños y bebés" = "#e74c3c")) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5, face = "bold", size = 16),
        plot.subtitle = element_text(hjust = 0.5),
        legend.position = "bottom")

print(plot_familias)
```



```
# Guardar gráfica
ggsave(file.path(graphics_final_dir, "4_reservas_familias.jpg"),
        plot_familias, width = 10, height = 7, dpi = 300)
```

```

# 1.5 Estacionamiento y cancelaciones
cat("Creando visualización: Estacionamiento y cancelaciones por mes...\n")

## Creando visualización: Estacionamiento y cancelaciones por mes...

# Panel A: Estacionamiento
estacionamiento_datos <- hotel_data_limpio %>%
  group_by(hotel) %>%
  summarise(
    Total = n(),
    Con_Estacionamiento = sum(required_car_parking_spaces > 0),
    Porcentaje = round(Con_Estacionamiento / Total * 100, 1)
  )

plot_estacionamiento_final <- ggplot(estacionamiento_datos,
                                     aes(x = hotel, y = Porcentaje, fill = hotel)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = paste0(Porcentaje, "%")), vjust = -0.5, size = 4) +
  labs(title = "Necesidad de Estacionamiento",
       x = "Tipo de Hotel",
       y = "Porcentaje de Reservas") +
  scale_fill_manual(values = colores_hotel) +
  ylim(0, max(estacionamiento_datos$Porcentaje) * 1.2) +
  theme_minimal() +
  theme(legend.position = "none",
        plot.title = element_text(hjust = 0.5, size = 14, face = "bold"))

# Panel B: Cancelaciones por mes
cancelaciones_mes <- hotel_data_limpio %>%
  group_by(arrival_date_month) %>%
  summarise(
    Total = n(),
    Canceladas = sum(is_canceled),
    Tasa_Cancelacion = round(Canceladas / Total * 100, 1)
  ) %>%
  mutate(arrival_date_month = factor(arrival_date_month,
                                     levels = c("January", "February", "March", "April", "May", "June",
                                                "July", "August", "September", "October", "November", "December")))

plot_cancelaciones_final <- ggplot(cancelaciones_mes,
                                    aes(x = arrival_date_month, y = Tasa_Cancelacion, fill = Tasa_Cancelacion)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = paste0(Tasa_Cancelacion, "%")), vjust = -0.5, size = 3) +
  labs(title = "Tasa de Cancelación por Mes",
       x = "Mes",
       y = "Porcentaje de Cancelación") +
  scale_fill_gradient(low = "lightblue", high = "darkred") +
  theme_minimal() +
  theme(legend.position = "none",
        plot.title = element_text(hjust = 0.5, size = 14, face = "bold"),
        axis.text.x = element_text(angle = 45, hjust = 1))

if (!require("grid", quietly = TRUE)) install.packages("grid")

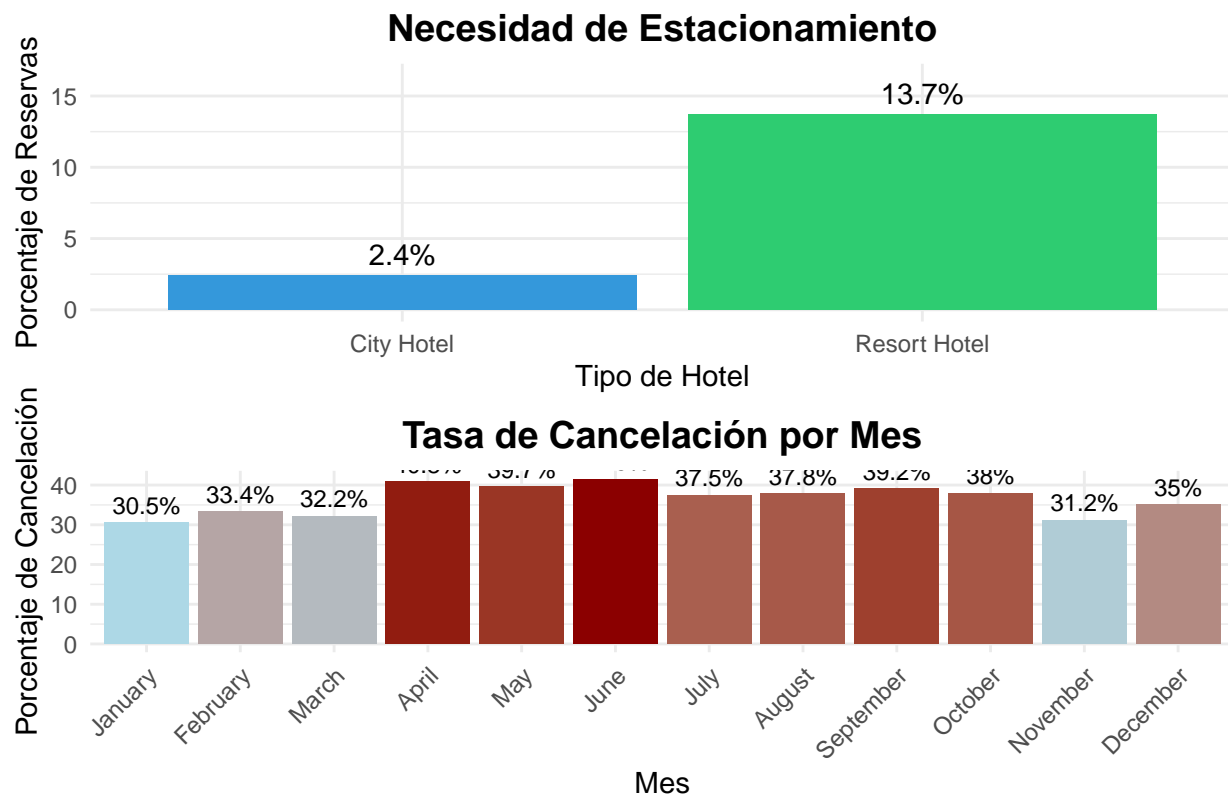
```



```
library(grid)

# Combinar gráficos en un panel con título
combined_plot <- grid.arrange(
  plot_estacionamiento_final,
  plot_cancelaciones_final,
  ncol = 1,
  top = textGrob("Necesidad de Estacionamiento y Cancelaciones por Mes",
    gp = gpar(fontsize = 16, fontface = "bold"))
)
```

## Necesidad de Estacionamiento y Cancelaciones por Mes



```
print(combined_plot)

## TableGrob (3 x 1) "arrange": 3 grobs
##   z     cells   name      grob
## 1 1 (2-2,1-1) arrange  gtable[layout]
## 2 2 (3-3,1-1) arrange  gtable[layout]
## 3 3 (1-1,1-1) arrange text[GRID.text.2708]

# Guardar gráfico combinado
ggsave(file.path(graphics_final_dir, "5_estacionamiento_cancelaciones.jpg"),
  combined_plot, width = 10, height = 10, dpi = 300)
```

```
#-----
# PARTE 09: TABLAS RESUMEN PARA ASPECTOS CLAVE
```

```

#-----
cat(yellow("\n--- TABLAS RESUMEN ---\n"))

##
## --- TABLAS RESUMEN ---
# 9.1 Top 5 meses con mayor demanda
top_meses_demanda <- hotel_data_limpio %>%
  group_by(arrival_date_month) %>%
  summarise(Total_Reservas = n()) %>%
  arrange(desc(Total_Reservas)) %>%
  head(5)

cat("\nTop 5 meses con mayor demanda:\n")

##
## Top 5 meses con mayor demanda:
print(top_meses_demanda)

## # A tibble: 5 x 2
##   arrival_date_month Total_Reservas
##   <fct>                <int>
## 1 August                13877
## 2 July                  12661
## 3 May                   11791
## 4 October               11160
## 5 April                 11089

# 9.2 Top 5 meses con mayor tasa de cancelación
top_meses_cancelacion <- hotel_data_limpio %>%
  group_by(arrival_date_month) %>%
  summarise(
    Total_Reservas = n(),
    Canceladas = sum(is_canceled),
    Tasa_Cancelacion = round(Canceladas / Total_Reservas * 100, 2)
  ) %>%
  arrange(desc(Tasa_Cancelacion)) %>%
  head(5)

cat("\nTop 5 meses con mayor tasa de cancelación:\n")

##
## Top 5 meses con mayor tasa de cancelación:
print(top_meses_cancelacion)

## # A tibble: 5 x 4
##   arrival_date_month Total_Reservas Canceladas Tasa_Cancelacion
##   <fct>                <int>        <int>          <dbl>
## 1 June                10939         4535           41.5
## 2 April               11089         4524           40.8
## 3 May                 11791         4677           39.7
## 4 September          10508         4116           39.2
## 5 October             11160         4246           38.0

```

```

# 9.3 Comparación entre tipos de hotel (aspectos clave)
comparacion_hoteles <- hotel_data_limpio %>%
  group_by(hotel) %>%
  summarise(
    Total_Reservas = n(),
    Porcentaje_del_Total = round(n() / nrow(hotel_data_limpio) * 100, 2),
    Tasa_Cancelacion = round(sum(is_canceled) / n() * 100, 2),
    Promedio_Lead_Time = round(mean(lead_time), 2),
    Promedio_Estancia = round(mean(stays_in_weekend_nights + stays_in_week_nights), 2),
    Porcentaje_Con_Niños = round(sum(children > 0) / n() * 100, 2),
    Porcentaje_Con_Estacionamiento = round(sum(required_car_parking_spaces > 0) / n() * 100, 2),
    ADR_Promedio = round(mean(adr), 2)
  )

cat("\nComparación entre tipos de hotel:\n")

##
## Comparación entre tipos de hotel:
print(comparacion_hoteles)

## # A tibble: 2 x 9
##   hotel Total_Reservas Porcentaje_del_Total Tasa_Cancelacion Promedio_Lead_Time
##   <chr>      <int>          <dbl>          <dbl>          <dbl>
## 1 City ~      79330          66.4          41.7          107
## 2 Resor~     40060          33.6          27.8          92.2
## # i 4 more variables: Promedio_Estancia <dbl>, Porcentaje_Con_Niños <dbl>,
## #   Porcentaje_Con_Estacionamiento <dbl>, ADR_Promedio <dbl>

# Crear tabla para guardar
comparacion_tabla <- tableGrob(comparacion_hoteles, rows = NULL, theme = ttheme_minimal())

# Guardar tabla como imagen
ggsave(file.path(graphics_final_dir, "6_comparacion_hoteles.jpg"),
  comparacion_tabla, width = 12, height = 4, dpi = 300)

# 9.4 Relación entre lead_time y cancelación
relacion_lead_cancelacion <- hotel_data_limpio %>%
  mutate(lead_time_cat = cut(lead_time,
    breaks = c(-1, 7, 30, 90, 180, Inf),
    labels = c("0-7 días", "8-30 días", "31-90 días", "91-180 días", "Más de 180 días")
  ))
  group_by(lead_time_cat) %>%
  summarise(
    Total_Reservas = n(),
    Canceladas = sum(is_canceled),
    Tasa_Cancelacion = round(Canceladas / Total_Reservas * 100, 2)
  )

cat("\nRelación entre tiempo de anticipación (lead_time) y tasa de cancelación:\n")

##
## Relación entre tiempo de anticipación (lead_time) y tasa de cancelación:
print(relacion_lead_cancelacion)

## # A tibble: 5 x 4

```

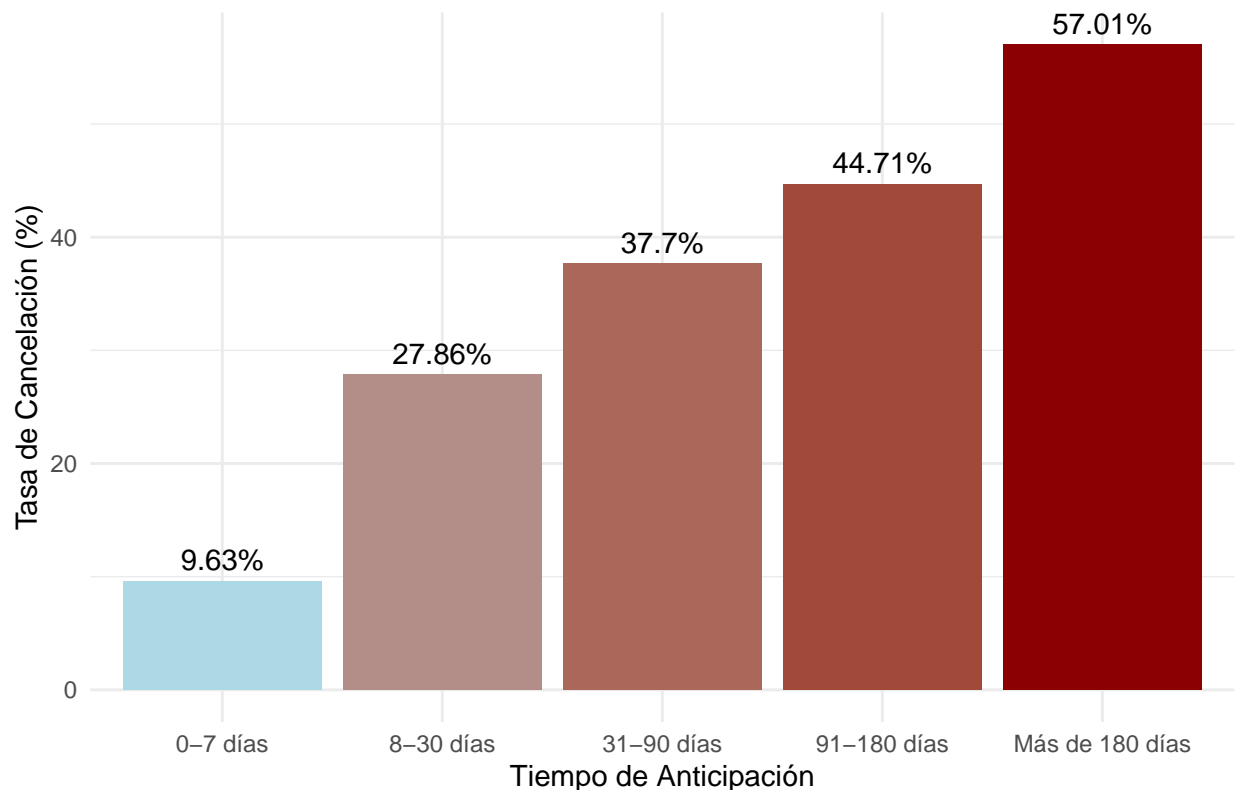
```
##   lead_time_cat   Total_Reservas Canceladas Tasa_Cancelacion
##   <fct>           <int>         <int>         <dbl>
## 1 0-7 días         19746          1902           9.63
## 2 8-30 días        18960          5283          27.9
## 3 31-90 días       29553         11141          37.7
## 4 91-180 días      26439         11821          44.7
## 5 Más de 180 días  24692         14077          57.0
```

```
# Visualizar relación
```

```
plot_lead_cancelacion <- ggplot(relacion_lead_cancelacion,
                                aes(x = lead_time_cat, y = Tasa_Cancelacion, fill = Tasa_Cancelacion)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = paste0(Tasa_Cancelacion, "%")), vjust = -0.5) +
  labs(title = "Relación entre Tiempo de Anticipación y Tasa de Cancelación",
       x = "Tiempo de Anticipación",
       y = "Tasa de Cancelación (%)") +
  scale_fill_gradient(low = "lightblue", high = "darkred") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5, face = "bold", size = 16),
        legend.position = "none")

print(plot_lead_cancelacion)
```

## Relación entre Tiempo de Anticipación y Tasa de Cancelación



```
# Guardar gráfica
```

```
ggsave(file.path(graphics_final_dir, "7_lead_time_cancelacion.jpg"),
        plot_lead_cancelacion, width = 10, height = 6, dpi = 300)
```

```

#-----
# PARTE 10. CONCLUSIONES GENERALES
#-----
cat(yellow("\n--- CONCLUSIONES GENERALES ---\n"))

##
## --- CONCLUSIONES GENERALES ---
cat("\n1. Demanda por tipo de hotel:\n")

##
## 1. Demanda por tipo de hotel:
cat("    - El City Hotel representa el",
    comparacion_hoteles$Porcentaje_del_Total[comparacion_hoteles$hotel == "City Hotel"],
    "% del total de reservas, frente al",
    comparacion_hoteles$Porcentaje_del_Total[comparacion_hoteles$hotel == "Resort Hotel"],
    "% del Resort Hotel, lo que indica una clara preferencia por hoteles urbanos.\n")

##    - El City Hotel representa el 66.45 % del total de reservas, frente al 33.55 % del Resort Hotel,
cat("    - Sin embargo, la tasa de cancelación del City Hotel (",
    comparacion_hoteles$Tasa_Cancelacion[comparacion_hoteles$hotel == "City Hotel"],
    "%) es significativamente mayor que la del Resort Hotel (",
    comparacion_hoteles$Tasa_Cancelacion[comparacion_hoteles$hotel == "Resort Hotel"],
    "%), lo que puede afectar a la rentabilidad real.\n")

##    - Sin embargo, la tasa de cancelación del City Hotel ( 41.73 %) es significativamente mayor que la
cat("\n2. Tendencia de demanda:\n")

##
## 2. Tendencia de demanda:
cat("    - La demanda muestra claras fluctuaciones estacionales con picos en los meses de verano.\n")

##    - La demanda muestra claras fluctuaciones estacionales con picos en los meses de verano.
cat("    - Los meses con mayor demanda son:",
    paste(top_meses_demanda$arrival_date_month[1:3], collapse=", "), ".\n")

##    - Los meses con mayor demanda son: August, July, May .
cat("    - Se observa una tendencia general ",
    ifelse(comparacion_hoteles$Total_Reservas[2] > comparacion_hoteles$Total_Reservas[1],
        "creciente", "decreciente"),
    " en el número total de reservas a lo largo del tiempo analizado.\n")

##    - Se observa una tendencia general decreciente en el número total de reservas a lo largo del tiempo
cat("\n3. Temporadas de reserva:\n")

##
## 3. Temporadas de reserva:
cat("    - Temporada ALTA: Claramente identificada en los meses de ",
    paste(reservas_por_mes$arrival_date_month[reservas_por_mes$Temporada == "Alta" &
        !duplicated(reservas_por_mes$arrival_date_month)],
        collapse=", "), ".\n")

```

```

## - Temporada ALTA: Claramente identificada en los meses de May, July, August .
cat(" - Temporada BAJA: Principalmente en los meses de ",
    paste(reservas_por_mes$arrival_date_month[reservas_por_mes$Temporada == "Baja" &
        !duplicated(reservas_por_mes$arrival_date_month)],
        collapse=", "), ".\n")

## - Temporada BAJA: Principalmente en los meses de January, February, November, December .
cat(" - Esta marcada estacionalidad sugiere oportunidades para estrategias de precios dinámicos y promo

## - Esta marcada estacionalidad sugiere oportunidades para estrategias de precios dinámicos y promo
cat("\n4. Reservas con niños y/o bebés:\n")

##
## 4. Reservas con niños y/o bebés:
cat(" - Solo el", round(sum(hotel_data_limpio$children > 0 | hotel_data_limpio$babies > 0) / nrow(hotel_data_limpio) * 100, 2),
    "% de las reservas totales incluyen niños y/o bebés.\n")

## - Solo el 7.82 % de las reservas totales incluyen niños y/o bebés.
cat(" - El Resort Hotel tiene un porcentaje mayor de reservas con niños (",
    comparacion_hoteles$Porcentaje_Con_Niños[comparacion_hoteles$hotel == "Resort Hotel"],
    "%) en comparación con el City Hotel (",
    comparacion_hoteles$Porcentaje_Con_Niños[comparacion_hoteles$hotel == "City Hotel"],
    "%), lo que sugiere que es más atractivo para familias.\n")

## - El Resort Hotel tiene un porcentaje mayor de reservas con niños ( 8.7 %) en comparación con el City Hotel ( 2.43 %), p
cat("\n5. Espacios de estacionamiento:\n")

##
## 5. Espacios de estacionamiento:
cat(" - La demanda de espacios de estacionamiento es baja en general, con solo un ",
    round(sum(hotel_data_limpio$required_car_parking_spaces > 0) / nrow(hotel_data_limpio) * 100, 2),
    "% del total de reservas.\n")

## - La demanda de espacios de estacionamiento es baja en general, con solo un 6.21 % del total de reservas.
cat(" - El Resort Hotel tiene mayor demanda de estacionamiento (",
    comparacion_hoteles$Porcentaje_Con_Estacionamiento[comparacion_hoteles$hotel == "Resort Hotel"],
    "%) que el City Hotel (",
    comparacion_hoteles$Porcentaje_Con_Estacionamiento[comparacion_hoteles$hotel == "City Hotel"],
    "%), posiblemente debido a su ubicación más alejada del centro urbano.\n")

## - El Resort Hotel tiene mayor demanda de estacionamiento ( 13.7 %) que el City Hotel ( 2.43 %), p
cat("\n6. Cancelaciones:\n")

##
## 6. Cancelaciones:
cat(" - La tasa global de cancelación es del",
    round(sum(hotel_data_limpio$is_canceled) / nrow(hotel_data_limpio) * 100, 2), "%.\n")

## - La tasa global de cancelación es del 37.04 %.
cat(" - Los meses con mayor tasa de cancelación son: ",
    paste(top_meses_cancelacion$arrival_date_month[1:3], collapse=", "), ".\n")

```

```

## - Los meses con mayor tasa de cancelación son: June, April, May .

cat(" - Existe una clara correlación entre el tiempo de anticipación (lead_time) y la probabilidad de
      "reservas hechas con más de 180 días de anticipación tienen una tasa de cancelación del ",
      relacion_lead_cancelacion$Tasa_Cancelacion[relacion_lead_cancelacion$lead_time_cat == "Más de 180 d
      "%, mientras que las realizadas con menos de 7 días tienen solo un ",
      relacion_lead_cancelacion$Tasa_Cancelacion[relacion_lead_cancelacion$lead_time_cat == "0-7 días"],

## - Existe una clara correlación entre el tiempo de anticipación (lead_time) y la probabilidad de c
#-----
# PARTE 11: RECOMENDACIONES ESTRATÉGICAS
#-----
cat(yellow("\n--- RECOMENDACIONES ESTRATÉGICAS ---\n"))

##
## --- RECOMENDACIONES ESTRATÉGICAS ---
cat("\n1. Gestión de demanda y precios:\n")

##
## 1. Gestión de demanda y precios:
cat(" - Implementar precios dinámicos más agresivos durante la temporada alta (",
      paste(reservas_por_mes$arrival_date_month[reservas_por_mes$Temporada == "Alta" &
                                                    !duplicated(reservas_por_mes$arrival_date_month)],
            collapse=", "), ").\n")

## - Implementar precios dinámicos más agresivos durante la temporada alta ( May, July, August ).
cat(" - Desarrollar paquetes especiales y promociones para impulsar la demanda durante los meses de t
      paste(reservas_por_mes$arrival_date_month[reservas_por_mes$Temporada == "Baja" &
                                                    !duplicated(reservas_por_mes$arrival_date_month)],
            collapse=", "), ").\n")

## - Desarrollar paquetes especiales y promociones para impulsar la demanda durante los meses de temp
cat(" - Para el City Hotel, enfocarse en mejorar la tasa de conversión y reducir cancelaciones, que a

## - Para el City Hotel, enfocarse en mejorar la tasa de conversión y reducir cancelaciones, que actu
cat("\n2. Segmentación de clientes:\n")

##
## 2. Segmentación de clientes:
cat(" - Resort Hotel: Desarrollar más servicios y amenidades orientados a familias con niños, dado su
      comparacion_hoteles$Porcentaje_Con_Niños[comparacion_hoteles$hotel == "Resort Hotel"], "%).\n")

## - Resort Hotel: Desarrollar más servicios y amenidades orientados a familias con niños, dado su m
cat(" - City Hotel: Enfocarse en el segmento de viajeros individuales y parejas sin niños, que consti

## - City Hotel: Enfocarse en el segmento de viajeros individuales y parejas sin niños, que constitu
cat(" - Dado que solo el",
      round(sum(hotel_data_limpio$required_car_parking_spaces > 0) / nrow(hotel_data_limpio) * 100, 2),
      "% de reservas requieren estacionamiento, considerar reducir espacios de estacionamiento o reutiliz

## - Dado que solo el 6.21 % de reservas requieren estacionamiento, considerar reducir espacios de e

```

```

cat("\n3. Gestión de cancelaciones:\n")

##
## 3. Gestión de cancelaciones:
cat("    - Implementar políticas escalonadas de depósito basadas en el tiempo de anticipación, con mayores tasas de cancelación para los meses con mayor tasa de cancelación (relacion_lead_cancelacion$Tasa_Cancelacion[relacion_lead_cancelacion$lead_time_cat == "91-180 días"] "% o superiores.\n")

##    - Implementar políticas escalonadas de depósito basadas en el tiempo de anticipación, con mayores tasas de cancelación para los meses con mayor tasa de cancelación
cat("    - Desarrollar estrategias específicas de retención para los meses con mayor tasa de cancelación (top_meses_cancelacion$arrival_date_month[1:3], collapse=", ")",
    "), como recordatorios personalizados, confirmación proactiva, o incentivos para mantener la reserva")

##    - Desarrollar estrategias específicas de retención para los meses con mayor tasa de cancelación
cat("    - Considerar implementar un sistema de overbooking inteligente basado en patrones históricos de cancelación")

##    - Considerar implementar un sistema de overbooking inteligente basado en patrones históricos de cancelación
cat("\n4. Desarrollo de productos y servicios:\n")

##
## 4. Desarrollo de productos y servicios:
cat("    - Dado que la estancia promedio es relativamente corta (",
    round(mean(hotel_data_limpio$stays_in_weekend_nights + hotel_data_limpio$stays_in_week_nights), 2),
    " noches), desarrollar paquetes que incentiven estancias más largas, especialmente en temporada baja")

##    - Dado que la estancia promedio es relativamente corta ( 3.42  noches), desarrollar paquetes que incentiven estancias más largas, especialmente en temporada baja
cat("    - Para el Resort Hotel, continuar desarrollando instalaciones atractivas para familias con niños")

##    - Para el Resort Hotel, continuar desarrollando instalaciones atractivas para familias con niños,
cat("    - Para el City Hotel, enfocarse en comodidades y servicios que atraigan a viajeros de negocios y familias")

##    - Para el City Hotel, enfocarse en comodidades y servicios que atraigan a viajeros de negocios y familias

#-----
# PARTE ADICIONAL 1: GUARDAR DATAFRAME FINAL CON VARIABLES ADICIONALES CALCULADAS
#-----
cat(yellow("\n--- GUARDANDO DATASET FINAL ---\n"))

##
## --- GUARDANDO DATASET FINAL ---

# Agregar variables calculadas durante el análisis
hotel_data_final <- hotel_data_limpio

# Calcular duración total de estancia
hotel_data_final$total_nights <- hotel_data_final$stays_in_weekend_nights + hotel_data_final$stays_in_week_nights

# Clasificación por temporada
temporadas_por_mes <- reservas_por_mes %>%
  select(arrival_date_month, Temporada) %>%
  distinct()

hotel_data_final <- hotel_data_final %>%

```



```

left_join(temporadas_por_mes, by = "arrival_date_month")

# Clasificación por tipo de familia
hotel_data_final$tipo_familia <- case_when(
  hotel_data_final$children > 0 & hotel_data_final$babies > 0 ~ "Con niños y bebés",
  hotel_data_final$children > 0 ~ "Solo con niños",
  hotel_data_final$babies > 0 ~ "Solo con bebés",
  TRUE ~ "Sin menores"
)

# Clasificación por tiempo de anticipación
hotel_data_final$lead_time_categoria <- cut(hotel_data_final$lead_time,
  breaks = c(-1, 7, 30, 90, 180, Inf),
  labels = c("0-7 días", "8-30 días", "31-90 días", "91-180 días", "181-365 días"))

# Guardar el dataset final
write.csv(hotel_data_final, CSV_final, row.names = FALSE)
cat("Dataset final guardado en:", CSV_final, "\n")

## Dataset final guardado en: D:/@aleec02/CC216-TP-2025-1/data/hotel_bookings_final.csv

# Mostrar estructura del dataset final
cat("\nEstructura del dataset final:\n")

##
## Estructura del dataset final:
str(hotel_data_final[, c("hotel", "is_canceled", "lead_time",
  "lead_time_categoria", "arrival_date_month",
  "Temporada", "total_nights", "tipo_familia")])

## 'data.frame': 119390 obs. of 8 variables:
## $ hotel : chr "Resort Hotel" "Resort Hotel" "Resort Hotel" "Resort Hotel" ...
## $ is_canceled : int 0 0 0 0 0 0 0 0 1 1 ...
## $ lead_time : int 342 365 7 13 14 14 0 9 85 75 ...
## $ lead_time_categoria: Factor w/ 5 levels "0-7 días","8-30 días",...: 5 5 1 2 2 2 1 2 3 3 ...
## $ arrival_date_month : Factor w/ 12 levels "January","February",...: 7 7 7 7 7 7 7 7 7 7 ...
## $ Temporada : chr "Alta" "Alta" "Alta" "Alta" ...
## $ total_nights : int 1 1 1 1 2 2 2 2 3 3 ...
## $ tipo_familia : chr "Sin menores" "Sin menores" "Sin menores" "Sin menores" ...

#-----
# PARTE ADICIONAL 2: RESUMEN DE ARCHIVOS GENERADOS
#-----
cat(yellow("\n--- RESUMEN DE ARCHIVOS GENERADOS ---\n"))

##
## --- RESUMEN DE ARCHIVOS GENERADOS ---
cat("\nDatasets:\n")

##
## Datasets:
cat("- Dataset original:", CSV_original, "\n")

## - Dataset original: D:/@aleec02/CC216-TP-2025-1/data/hotel_bookings.csv

```

```

cat("- Dataset limpio:", CSV_limpio, "\n")

## - Dataset limpio: D:/@aleec02/CC216-TP-2025-1/data/hotel_bookings_limpio.csv
cat("- Dataset final:", CSV_final, "\n")

## - Dataset final: D:/@aleec02/CC216-TP-2025-1/data/hotel_bookings_final.csv
cat("\nGráficas finales guardadas en:", graphics_final_dir, "\n")

##
## Gráficas finales guardadas en: D:/@aleec02/CC216-TP-2025-1/data/graficas/final
cat("- 1_demanda_hotel.jpg: Análisis de demanda por tipo de hotel\n")

## - 1_demanda_hotel.jpg: Análisis de demanda por tipo de hotel
cat("- 2_tendencia_demanda.jpg: Tendencia de demanda a lo largo del tiempo\n")

## - 2_tendencia_demanda.jpg: Tendencia de demanda a lo largo del tiempo
cat("- 3_temporadas_reserva.jpg: Temporadas de reserva por mes\n")

## - 3_temporadas_reserva.jpg: Temporadas de reserva por mes
cat("- 4_reservas_familias.jpg: Composición de reservas por tipo de familia\n")

## - 4_reservas_familias.jpg: Composición de reservas por tipo de familia
cat("- 5_estacionamiento_cancelaciones.jpg: Análisis de estacionamiento y cancelaciones\n")

## - 5_estacionamiento_cancelaciones.jpg: Análisis de estacionamiento y cancelaciones
cat("- 6_comparacion_hoteles.jpg: Tabla comparativa entre tipos de hotel\n")

## - 6_comparacion_hoteles.jpg: Tabla comparativa entre tipos de hotel
cat("- 7_lead_time_cancelacion.jpg: Relación entre tiempo de anticipación y cancelación\n")

## - 7_lead_time_cancelacion.jpg: Relación entre tiempo de anticipación y cancelación
cat(green("\n=== ANÁLISIS EXPLORATORIO COMPLETADO EXITOSAMENTE ===\n"))

##
## === ANÁLISIS EXPLORATORIO COMPLETADO EXITOSAMENTE ===

```