Task4

AUTHOR
Oleksandra Ustymenko

PUBLISHED
March 10, 2026

```r
library(tidyverse)
```

```
── Attaching core tidyverse packages ────────────────────────
tidyverse 2.0.0 ──
✔ dplyr     1.1.4     ✔ readr     2.1.5
✔ forcats   1.0.0     ✔ stringr   1.5.1
✔ ggplot2   3.5.1     ✔ tibble    3.2.1
✔ lubridate 1.9.3     ✔ tidyr     1.3.1
✔ purrr     1.0.2
── Conflicts ──────────────────────────────────────────
tidyverse_conflicts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
ℹ Use the conflicted package (<http://conflicted.r-lib.org/>)
to force all conflicts to become errors
```

```r
extract_name_day_type <- function(file_path) {
    file_path |>
        basename() |>
        tools::file_path_sans_ext() |>
        str_split_1("_")
}
```

Question 1

Question description

1. `explore_city(file_path)`

```r
explore_city <- function(file_path) {
  # Check if the file exists
  if (!file.exists(file_path)) {
    stop("File not found. Please check the path.")
  }

  # Load the data
  data <- read.csv(file_path)

  # Extract city name and day type from the file name
  city_name <- tools::file_path_sans_ext(basename(file_path))
  if (grepl("weekends", city_name, ignore.case = TRUE)) {
    day_type <- "weekends"
    city_name <- gsub("_weekends", "", city_name, ignore.case =
  } else if (grepl("weekdays", city_name, ignore.case = TRUE))
    day_type <- "weekdays"
    city_name <- gsub("_weekdays", "", city_name, ignore.case =
  } else {
    day_type <- "unknown"
  }

  # Calculate the number of rows
  n_rows <- nrow(data)

  # Calculate price statistics if the column 'realSum' exists
  if ("realSum" %in% colnames(data)) {
    min_price <- min(data$realSum, na.rm = TRUE)
    avg_price <- mean(data$realSum, na.rm = TRUE)
    max_price <- max(data$realSum, na.rm = TRUE)
  } else {
    min_price <- NA
    avg_price <- NA
    max_price <- NA
  }

  # Calculate average satisfaction if the column 'guest_satisfa
  if ("guest_satisfaction_overall" %in% colnames(data)) {
    avg_satisfaction <- mean(data$guest_satisfaction_overall, r
  } else {
    avg_satisfaction <- NA
  }

  # Return a list with all results
  return(data.frame(
    city_name = city_name,
    day_type = day_type,
    n_rows = n_rows,
    min_price = min_price,
    avg_price = avg_price,
    max_price = max_price,
    avg_satisfaction = avg_satisfaction
  ))
}
```

```r
explore_city("./data/airbnb/amsterdam_weekdays.csv")
```

```
  city_name day_type n_rows min_price avg_price max_price
avg_satisfaction
```

```
1 amsterdam weekdays   1103  128.8871  545.0205  7782.907
94.36265
```

```
prepare_dataset <- function(folder_path) {
  # Get a list of all CSV files in the folder
  files <- list.files(path = folder_path, pattern = "*.csv", fu

  # Initialize an empty list to store individual data frames
  data_list <- list()

  # Loop through each file to read it and add new columns
  for (file in files) {
    # Read the CSV file
    data <- read.csv(file)

    # Extract the city name and day type from the file name
    city_name <- tools::file_path_sans_ext(basename(file))
    if (grepl("weekends", city_name, ignore.case = TRUE)) {
      day_type <- "weekends"
      city_name <- gsub("_weekends", "", city_name, ignore.case
    } else if (grepl("weekdays", city_name, ignore.case = TRUE)
      day_type <- "weekdays"
      city_name <- gsub("_weekdays", "", city_name, ignore.case
    } else {
      day_type <- "unknown"
    }

    # Add the new columns to the data
    data$city <- city_name
    data$day_type <- day_type

    # Append the data frame to the list
    data_list[[length(data_list) + 1]] <- data
  }

  # Merge all the data frames into one
  merged_data <- do.call(rbind, data_list)

  # Save the merged data to a CSV file named "airbnb.csv"
  write.csv(merged_data, "./data/airbnb.csv", row.names = FALSE

  # Return the merged dataset
  return(merged_data)
}
```

```
df <- prepare_dataset("./data/airbnb")
```

```r
# Load necessary libraries
library(dplyr)
library(stringr)
library(tools)

# Function to extract city name and day type from the file name
extract_name_day_type <- function(file_path) {
  file_path |>
    basename() |>
    tools::file_path_sans_ext() |>
    str_split_1("_")
}

# Function to prepare the dataset
prepare_dataset <- function(folder_path) {
  # Read all CSV files in the given folder
  files <- list.files(folder_path, pattern = "*.csv", full.name

  # Read and merge all files into one dataset
  all_data <- lapply(files, function(file) {
    city_day_type <- extract_name_day_type(file)
    data <- read.csv(file)

    # Add columns for city and day type
    data$city <- city_day_type[1]
    data$day_type <- city_day_type[2]

    return(data)
  })

  # Merge all the data into one data frame
  merged_data <- bind_rows(all_data)

  # Save the merged dataset to a CSV file
  write.csv(merged_data, "airbnb.csv", row.names = FALSE)

  return(merged_data)
}
```

```r
# Function to perform t-test between two cities
compare_means <- function(city1_name, city2_name, merged_data)

  # Filter data for city 1 and city 2
  city1_data <- merged_data %>% filter(city == city1_name)
  city2_data <- merged_data %>% filter(city == city2_name)

  # Perform t-test on realSum prices between the two cities
  t_test_result <- t.test(city1_data$realSum, city2_data$realSu

  # Return the result of the t-test
  return(t_test_result)
}
```

```
# Load the merged dataset
merged_data <- read.csv("./airbnb.csv")  # Path to the folder w

# Perform t-test between Amsterdam and Barcelona
test_result <- compare_means("amsterdam", "barcelona", merged_d

# Print the result of the t-test
print(test_result)
```

```

	Welch Two Sample t-test

data:  city1_data$realSum and city2_data$realSum
t = 24.154, df = 3953, p-value < 2.2e-16
alternative hypothesis: true difference in means is not equal
to 0
95 percent confidence interval:
 256.6835 302.0347
sample estimates:
mean of x mean of y 
 573.1128  293.7537 
```

```
```