

Case Study: Bike Sharing

David Nee

2023-04-18

Step 1: Install Packages

```
library(tidyverse)
```

```
## — Attaching core tidyverse packages — tidyverse 2.0.0 —
## ✓ dplyr      1.1.1    ✓ readr      2.1.4
## ✓ forcats   1.0.0    ✓ stringr   1.5.0
## ✓ ggplot2    3.4.2    ✓ tibble    3.2.1
## ✓ lubridate  1.9.2    ✓ tidyr     1.3.0
## ✓ purrr     1.0.1
## — 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
```

```
library(lubridate)
library(skimr)
library(dplyr)
library(readxl)
library(writexl)
library(openxlsx)
library(janitor)
```

```
##
## Attaching package: 'janitor'
##
## The following objects are masked from 'package:stats':
##
##   chisq.test, fisher.test
```

```
library(tinytex)
```

Step 2: Import and rename data frames

For the purposes of making easier code later on when combining them, the names of the tables will be simplified. The tables themselves are being imported using the `read_excel` function.

```

jan_trip22 <- read_excel("C:\\Users\\needa\\Desktop\\case study\\202201-divvy-tripdata.xlsx")

feb_trip22 <- read_excel("C:\\Users\\needa\\Desktop\\case study\\202202-divvy-tripdata.xlsx")

mar_trip22 <- read_excel("C:\\Users\\needa\\Desktop\\case study\\202203-divvy-tripdata.xlsx")

apr_trip22 <- read_excel("C:\\Users\\needa\\Desktop\\case study\\202204-divvy-tripdata.xlsx")

may_trip22 <- read_excel("C:\\Users\\needa\\Desktop\\case study\\202205-divvy-tripdata.xlsx")

jun_trip22 <- read_excel("C:\\Users\\needa\\Desktop\\case study\\202206-divvy-tripdata.xlsx")

jul_trip22 <- read_excel("C:\\Users\\needa\\Desktop\\case study\\202207-divvy-tripdata.xlsx")

aug_trip22 <- read_excel("C:\\Users\\needa\\Desktop\\case study\\202208-divvy-tripdata.xlsx")

sep_trip22 <- read_excel("C:\\Users\\needa\\Desktop\\case study\\202209-divvy-tripdata.xlsx")

oct_trip22 <- read_excel("C:\\Users\\needa\\Desktop\\case study\\202210-divvy-tripdata.xlsx")

nov_trip22 <- read_excel("C:\\Users\\needa\\Desktop\\case study\\202211-divvy-tripdata.xlsx")

dec_trip22 <- read_excel("C:\\Users\\needa\\Desktop\\case study\\202212-divvy-tripdata.xlsx")

```

Step 3: Compare column names to ensure consistent naming

Although initially excel was used to examine the tables, and to practice designing additional columns, the project was primarily in R, and so those steps will be redone here for consistency.

```
colnames(jan_trip22)
```

```

## [1] "ride_id"          "rideable_type"    "started_at"
## [4] "ended_at"         "start_station_name" "start_station_id"
## [7] "end_station_name" "end_station_id"   "start_lat"
## [10] "start_lng"        "end_lat"          "end_lng"
## [13] "member_casual"

```

```
colnames(feb_trip22)
```

```

## [1] "ride_id"          "rideable_type"    "started_at"
## [4] "ended_at"         "start_station_name" "start_station_id"
## [7] "end_station_name" "end_station_id"   "start_lat"
## [10] "start_lng"        "end_lat"          "end_lng"
## [13] "member_casual"

```

```
colnames(mar_trip22)
```

```

## [1] "ride_id"          "rideable_type"    "started_at"
## [4] "ended_at"         "start_station_name" "start_station_id"
## [7] "end_station_name" "end_station_id"   "start_lat"
## [10] "start_lng"        "end_lat"          "end_lng"
## [13] "member_casual"

```

```
colnames(apr_trip22)
```

```

## [1] "ride_id"          "rideable_type"    "started_at"
## [4] "ended_at"         "start_station_name" "start_station_id"
## [7] "end_station_name" "end_station_id"   "start_lat"
## [10] "start_lng"        "end_lat"          "end_lng"
## [13] "member_casual"

```

```
colnames(may_trip22)
```

```
## [1] "ride_id"          "rideable_type"    "started_at"
## [4] "ended_at"         "start_station_name" "start_station_id"
## [7] "end_station_name" "end_station_id"   "start_lat"
## [10] "start_lng"        "end_lat"          "end_lng"
## [13] "member_casual"
```

```
colnames(jun_trip22)
```

```
## [1] "ride_id"          "rideable_type"    "started_at"
## [4] "ended_at"         "start_station_name" "start_station_id"
## [7] "end_station_name" "end_station_id"   "start_lat"
## [10] "start_lng"        "end_lat"          "end_lng"
## [13] "member_casual"
```

```
colnames(jul_trip22)
```

```
## [1] "ride_id"          "rideable_type"    "started_at"
## [4] "ended_at"         "start_station_name" "start_station_id"
## [7] "end_station_name" "end_station_id"   "start_lat"
## [10] "start_lng"        "end_lat"          "end_lng"
## [13] "member_casual"
```

```
colnames(aug_trip22)
```

```
## [1] "ride_id"          "rideable_type"    "started_at"
## [4] "ended_at"         "start_station_name" "start_station_id"
## [7] "end_station_name" "end_station_id"   "start_lat"
## [10] "start_lng"        "end_lat"          "end_lng"
## [13] "member_casual"
```

```
colnames(sep_trip22)
```

```
## [1] "ride_id"          "rideable_type"    "started_at"
## [4] "ended_at"         "start_station_name" "start_station_id"
## [7] "end_station_name" "end_station_id"   "start_lat"
## [10] "start_lng"        "end_lat"          "end_lng"
## [13] "member_casual"
```

```
colnames(oct_trip22)
```

```
## [1] "ride_id"          "rideable_type"    "started_at"
## [4] "ended_at"         "start_station_name" "start_station_id"
## [7] "end_station_name" "end_station_id"   "start_lat"
## [10] "start_lng"        "end_lat"          "end_lng"
## [13] "member_casual"
```

```
colnames(nov_trip22)
```

```
## [1] "ride_id"          "rideable_type"    "started_at"
## [4] "ended_at"         "start_station_name" "start_station_id"
## [7] "end_station_name" "end_station_id"   "start_lat"
## [10] "start_lng"        "end_lat"          "end_lng"
## [13] "member_casual"
```

```
colnames(dec_trip22)
```

```
## [1] "ride_id"          "rideable_type"    "started_at"
## [4] "ended_at"         "start_station_name" "start_station_id"
## [7] "end_station_name" "end_station_id"   "start_lat"
## [10] "start_lng"        "end_lat"          "end_lng"
## [13] "member_casual"
```

Step 4: check data types of columns to ensure consistency in data types

This is done both to ensure consistent values between the different frames, as well as to make sure the data types used are something that is workable for the data type involved.

```
str(jan_trip22)
```

```
## tibble [103,770 × 13] (S3: tbl_df/tbl/data.frame)
## $ ride_id      : chr [1:103770] "C2F7DD78E82EC875" "A6CF8980A652D272" "BD0F91DFF741C66D" "CBB80ED419105406" ...
## $ rideable_type : chr [1:103770] "electric_bike" "electric_bike" "classic_bike" "classic_bike" ...
## $ started_at   : POSIXct[1:103770], format: "2022-01-13 11:59:47" "2022-01-10 08:41:56" ...
## $ ended_at     : POSIXct[1:103770], format: "2022-01-13 12:02:44" "2022-01-10 08:46:17" ...
## $ start_station_name: chr [1:103770] "Glenwood Ave & Touhy Ave" "Glenwood Ave & Touhy Ave" "Sheffield Ave & Fullerton Ave" "Clark St & Bryn Mawr Ave" ...
## $ start_station_id : chr [1:103770] "525" "525" "TA1306000016" "KA1504000151" ...
## $ end_station_name : chr [1:103770] "Clark St & Touhy Ave" "Clark St & Touhy Ave" "Greenview Ave & Fullerton Ave" "Paulina St & Montrose Ave" ...
## $ end_station_id   : chr [1:103770] "RP-007" "RP-007" "TA1307000001" "TA1309000021" ...
## $ start_lat        : num [1:103770] 42 42 41.9 42 41.9 ...
## $ start_lng        : num [1:103770] -87.7 -87.7 -87.7 -87.7 -87.6 ...
## $ end_lat          : num [1:103770] 42 42 41.9 42 41.9 ...
## $ end_lng          : num [1:103770] -87.7 -87.7 -87.7 -87.7 -87.6 ...
## $ member_casual    : chr [1:103770] "casual" "casual" "member" "casual" ...
```

```
str(feb_trip22)
```

```
## tibble [115,609 × 13] (S3: tbl_df/tbl/data.frame)
## $ ride_id      : chr [1:115609] "E1E065E7ED285C02" "1602DCDC5B30FFE3" "BE7DD2AF4B55C4AF" "A1789BDF844412BE" ...
## $ rideable_type : chr [1:115609] "classic_bike" "classic_bike" "classic_bike" "classic_bike" ...
## $ started_at   : POSIXct[1:115609], format: "2022-02-19 18:08:41" "2022-02-20 17:41:30" ...
## $ ended_at     : POSIXct[1:115609], format: "2022-02-19 18:23:56" "2022-02-20 17:45:56" ...
## $ start_station_name: chr [1:115609] "State St & Randolph St" "Halsted St & Wrightwood Ave" "State St & Randolph St" "Southport Ave & Waveland Ave" ...
## $ start_station_id : chr [1:115609] "TA1305000029" "TA1309000061" "TA1305000029" "13235" ...
## $ end_station_name : chr [1:115609] "Clark St & Lincoln Ave" "Southport Ave & Wrightwood Ave" "Canal St & Adams St" "Broadway & Sheridan Rd" ...
## $ end_station_id   : chr [1:115609] "13179" "TA1307000113" "13011" "13323" ...
## $ start_lat        : num [1:115609] 41.9 41.9 41.9 41.9 41.9 ...
## $ start_lng        : num [1:115609] -87.6 -87.6 -87.6 -87.7 -87.6 ...
## $ end_lat          : num [1:115609] 41.9 41.9 41.9 42 41.9 ...
## $ end_lng          : num [1:115609] -87.6 -87.7 -87.6 -87.6 -87.6 ...
## $ member_casual    : chr [1:115609] "member" "member" "member" "member" ...
```

```
str(mar_trip22)
```

```
## tibble [284,042 × 13] (S3: tbl_df/tbl/data.frame)
## $ ride_id      : chr [1:284042] "47EC0A7F82E65D52" "8494861979B0F477" "EFE527AF80B66109" "9F446FD9DEE3F389" ...
## $ rideable_type : chr [1:284042] "classic_bike" "electric_bike" "classic_bike" "classic_bike" ...
## $ started_at   : POSIXct[1:284042], format: "2022-03-21 13:45:01" "2022-03-16 09:37:16" ...
## $ ended_at     : POSIXct[1:284042], format: "2022-03-21 13:51:18" "2022-03-16 09:43:34" ...
## $ start_station_name: chr [1:284042] "Wabash Ave & Wacker Pl" "Michigan Ave & Oak St" "Broadway & Berwyn Ave" "Wabash Ave & Wacker Pl" ...
## $ start_station_id : chr [1:284042] "TA1307000131" "13042" "13109" "TA1307000131" ...
## $ end_station_name : chr [1:284042] "Kingsbury St & Kinzie St" "Orleans St & Chestnut St (NEXT Apts)" "Broadway & Ridge Ave" "Franklin St & Jackson Blvd" ...
## $ end_station_id   : chr [1:284042] "KA1503000043" "620" "15578" "TA1305000025" ...
## $ start_lat        : num [1:284042] 41.9 41.9 42 41.9 41.9 ...
## $ start_lng        : num [1:284042] -87.6 -87.6 -87.7 -87.6 -87.6 ...
## $ end_lat          : num [1:284042] 41.9 41.9 42 41.9 41.9 ...
## $ end_lng          : num [1:284042] -87.6 -87.6 -87.7 -87.6 -87.7 ...
## $ member_casual    : chr [1:284042] "member" "member" "member" "member" ...
```

```
str(apr_trip22)
```

```
## tibble [371,249 × 13] (S3: tbl_df/tbl/data.frame)
## $ ride_id      : chr [1:371249] "3564070EEFD12711" "0B820C7FCF22F489" "89EEEE32293F07FF" "84D4751AEB31888D" ...
## $ rideable_type : chr [1:371249] "electric_bike" "classic_bike" "classic_bike" "classic_bike" ...
## $ started_at   : POSIXct[1:371249], format: "2022-04-06 17:42:48" "2022-04-24 19:23:07" ...
## $ ended_at     : POSIXct[1:371249], format: "2022-04-06 17:54:36" "2022-04-24 19:43:17" ...
## $ start_station_name: chr [1:371249] "Paulina St & Howard St" "Wentworth Ave & Cermak Rd" "Halsted St & Polk St" "Wentworth Ave & Cermak Rd" ...
## $ start_station_id : chr [1:371249] "515" "13075" "TA1307000121" "13075" ...
## $ end_station_name : chr [1:371249] "University Library (NU)" "Green St & Madison St" "Green St & Madison St" "Delano Ct & Roosevelt Rd" ...
## $ end_station_id   : chr [1:371249] "605" "TA1307000120" "TA1307000120" "KA1706005007" ...
## $ start_lat        : num [1:371249] 42 41.9 41.9 41.9 41.9 ...
## $ start_lng        : num [1:371249] -87.7 -87.6 -87.6 -87.6 -87.6 ...
## $ end_lat          : num [1:371249] 42.1 41.9 41.9 41.9 41.9 ...
## $ end_lng          : num [1:371249] -87.7 -87.6 -87.6 -87.6 -87.6 ...
## $ member_casual    : chr [1:371249] "member" "member" "member" "casual" ...
```

```
str(may_trip22)
```

```
## tibble [634,858 × 13] (S3: tbl_df/tbl/data.frame)
## $ ride_id      : chr [1:634858] "EC2DE40644C6B0F4" "1C31AD03897EE385" "1542FBEC830415CF" "6FF59852924528F8" ...
## $ rideable_type : chr [1:634858] "classic_bike" "classic_bike" "classic_bike" "classic_bike" ...
## $ started_at   : POSIXct[1:634858], format: "2022-05-23 23:06:58" "2022-05-11 08:53:28" ...
## $ ended_at     : POSIXct[1:634858], format: "2022-05-23 23:40:19" "2022-05-11 09:31:22" ...
## $ start_station_name: chr [1:634858] "Wabash Ave & Grand Ave" "DuSable Lake Shore Dr & Monroe St" "Clinton St & Madison St" "Clinton St & Madison St" ...
## $ start_station_id : chr [1:634858] "TA1307000117" "13300" "TA1305000032" "TA1305000032" ...
## $ end_station_name : chr [1:634858] "Halsted St & Roscoe St" "Field Blvd & South Water St" "Wood St & Milwaukee Ave" "Clark St & Randolph St" ...
## $ end_station_id   : chr [1:634858] "TA1309000025" "15534" "13221" "TA1305000030" ...
## $ start_lat        : num [1:634858] 41.9 41.9 41.9 41.9 41.9 ...
## $ start_lng        : num [1:634858] -87.6 -87.6 -87.6 -87.6 -87.6 ...
## $ end_lat          : num [1:634858] 41.9 41.9 41.9 41.9 41.9 ...
## $ end_lng          : num [1:634858] -87.6 -87.6 -87.7 -87.6 -87.7 ...
## $ member_casual    : chr [1:634858] "member" "member" "member" "member" ...
```

```
str(jun_trip22)
```

```
## tibble [769,204 × 13] (S3: tbl_df/tbl/data.frame)
## $ ride_id      : chr [1:769204] "600CFD130D0FD2A4" "F5E6B5C1682C6464" "B6EB6D27BAD771D2" "C9C320375DE1D5C6" ...
## $ rideable_type : chr [1:769204] "electric_bike" "electric_bike" "electric_bike" "electric_bike" ...
## $ started_at   : POSIXct[1:769204], format: "2022-06-30 17:27:53" "2022-06-30 18:39:52" ...
## $ ended_at     : POSIXct[1:769204], format: "2022-06-30 17:35:15" "2022-06-30 18:47:28" ...
## $ start_station_name: chr [1:769204] NA NA NA NA ...
## $ start_station_id : chr [1:769204] NA NA NA NA ...
## $ end_station_name : chr [1:769204] NA NA NA NA ...
## $ end_station_id   : chr [1:769204] NA NA NA NA ...
## $ start_lat        : num [1:769204] 41.9 41.9 41.9 41.8 41.9 ...
## $ start_lng        : num [1:769204] -87.6 -87.6 -87.7 -87.7 -87.6 ...
## $ end_lat          : num [1:769204] 41.9 41.9 41.9 41.8 41.9 ...
## $ end_lng          : num [1:769204] -87.6 -87.6 -87.6 -87.7 -87.6 ...
## $ member_casual    : chr [1:769204] "casual" "casual" "casual" "casual" ...
```

```
str(jul_trip22)
```

```
## tibble [823,488 × 13] (S3: tbl_df/tbl/data.frame)
## $ ride_id      : chr [1:823488] "954144C2F67B1932" "292E027607D218B6" "57765852588AD6E0" "B5B6BE44314590E6" ...
## $ rideable_type : chr [1:823488] "classic_bike" "classic_bike" "classic_bike" "classic_bike" ...
## $ started_at   : POSIXct[1:823488], format: "2022-07-05 08:12:47" "2022-07-26 12:53:38" ...
## $ ended_at     : POSIXct[1:823488], format: "2022-07-05 08:24:32" "2022-07-26 12:55:31" ...
## $ start_station_name: chr [1:823488] "Ashland Ave & Blackhawk St" "Buckingham Fountain (Temp)" "Buckingham Fountain (Temp)" "Buckingham Fountain (Temp)" ...
## $ start_station_id : chr [1:823488] "13224" "15541" "15541" "15541" ...
## $ end_station_name : chr [1:823488] "Kingsbury St & Kinzie St" "Michigan Ave & 8th St" "Michigan Ave & 8th St" "Woodlawn Ave & 55th St" ...
## $ end_station_id   : chr [1:823488] "KA1503000043" "623" "623" "TA1307000164" ...
## $ start_lat        : num [1:823488] 41.9 41.9 41.9 41.9 41.9 ...
## $ start_lng        : num [1:823488] -87.7 -87.6 -87.6 -87.6 -87.6 ...
## $ end_lat          : num [1:823488] 41.9 41.9 41.9 41.8 41.9 ...
## $ end_lng          : num [1:823488] -87.6 -87.6 -87.6 -87.6 -87.7 ...
## $ member_casual    : chr [1:823488] "member" "casual" "casual" "casual" ...
```

```
str(aug_trip22)
```

```
## tibble [785,932 × 13] (S3: tbl_df/tbl/data.frame)
## $ ride_id      : chr [1:785932] "550CF7EFEAE0C618" "DAD198F405F9C5F5" "E6F2BC47B65CB7FD" "F597830181C2E13C" ...
## $ rideable_type : chr [1:785932] "electric_bike" "electric_bike" "electric_bike" "electric_bike" ...
## $ started_at   : POSIXct[1:785932], format: "2022-08-07 21:34:15" "2022-08-08 14:39:21" ...
## $ ended_at     : POSIXct[1:785932], format: "2022-08-07 21:41:46" "2022-08-08 14:53:23" ...
## $ start_station_name: chr [1:785932] NA NA NA NA ...
## $ start_station_id : chr [1:785932] NA NA NA NA ...
## $ end_station_name : chr [1:785932] NA NA NA NA ...
## $ end_station_id   : chr [1:785932] NA NA NA NA ...
## $ start_lat        : num [1:785932] 41.9 41.9 42 41.9 41.9 ...
## $ start_lng        : num [1:785932] -87.7 -87.6 -87.7 -87.7 -87.7 ...
## $ end_lat          : num [1:785932] 41.9 41.9 42 42 41.8 ...
## $ end_lng          : num [1:785932] -87.7 -87.6 -87.7 -87.7 -87.7 ...
## $ member_casual    : chr [1:785932] "casual" "casual" "casual" "casual" ...
```

```
str(sep_trip22)
```

```
## tibble [701,339 × 13] (S3: tbl_df/tbl/data.frame)
## $ ride_id      : chr [1:701339] "5156990AC19CA285" "E12D4A16BF51C274" "A02B53CD7DB72DD7" "C82E05FEE872DF11" ...
## $ rideable_type : chr [1:701339] "electric_bike" "electric_bike" "electric_bike" "electric_bike" ...
## $ started_at   : POSIXct[1:701339], format: "2022-09-01 08:36:22" "2022-09-01 17:11:29" ...
## $ ended_at     : POSIXct[1:701339], format: "2022-09-01 08:39:05" "2022-09-01 17:14:45" ...
## $ start_station_name: chr [1:701339] NA NA NA NA ...
## $ start_station_id : chr [1:701339] NA NA NA NA ...
## $ end_station_name : chr [1:701339] "California Ave & Milwaukee Ave" NA NA NA ...
## $ end_station_id   : num [1:701339] 13084 NA NA NA NA ...
## $ start_lat        : num [1:701339] 41.9 41.9 41.9 41.9 41.9 ...
## $ start_lng        : num [1:701339] -87.7 -87.6 -87.6 -87.7 -87.7 ...
## $ end_lat          : num [1:701339] 41.9 41.9 41.9 41.9 41.9 ...
## $ end_lng          : num [1:701339] -87.7 -87.6 -87.6 -87.7 -87.7 ...
## $ member_casual    : chr [1:701339] "casual" "casual" "casual" "casual" ...
```

```
str(oct_trip22)
```

```
## tibble [558,685 × 13] (S3: tbl_df/tbl/data.frame)
## $ ride_id      : chr [1:558685] "A50255C1E17942AB" "DB692A70BD2DD4E3" "3C02727AAF60F873" "47E653FDC2D99236" ...
## $ rideable_type : chr [1:558685] "classic_bike" "electric_bike" "electric_bike" "electric_bike" ...
## $ started_at   : POSIXct[1:558685], format: "2022-10-14 17:13:30" "2022-10-01 16:29:26" ...
## $ ended_at     : POSIXct[1:558685], format: "2022-10-14 17:19:39" "2022-10-01 16:49:06" ...
## $ start_station_name: chr [1:558685] "Noble St & Milwaukee Ave" "Damen Ave & Charleston St" "Hoeyne Ave & Balmoral Ave" "Rush St & Cedar St" ...
## $ start_station_id : chr [1:558685] "13290" "13288" "655" "KA1504000133" ...
## $ end_station_name : chr [1:558685] "Larrabee St & Division St" "Damen Ave & Cullerton St" "Western Ave & Leland Ave" "Orleans St & Chestnut St (NEXT Apts)" ...
## $ end_station_id   : chr [1:558685] "KA1504000079" "13089" "TA1307000140" "620" ...
## $ start_lat        : num [1:558685] 41.9 41.9 42 41.9 41.9 ...
## $ start_lng        : num [1:558685] -87.7 -87.7 -87.7 -87.6 -87.6 ...
## $ end_lat          : num [1:558685] 41.9 41.9 42 41.9 41.9 ...
## $ end_lng          : num [1:558685] -87.6 -87.7 -87.7 -87.6 -87.6 ...
## $ member_casual    : chr [1:558685] "member" "casual" "member" "member" ...
```

```
str(nov_trip22)
```

```
## tibble [337,735 × 13] (S3: tbl_df/tbl/data.frame)
## $ ride_id      : chr [1:337735] "BCC66FC6FAB27CC7" "772AB67E902C180F" "585EAD07FDEC0152" "91C4E7ED3C262FF9" ...
## $ rideable_type : chr [1:337735] "electric_bike" "classic_bike" "classic_bike" "classic_bike" ...
## $ started_at   : POSIXct[1:337735], format: "2022-11-10 06:21:55" "2022-11-04 07:31:55" ...
## $ ended_at     : POSIXct[1:337735], format: "2022-11-10 06:31:27" "2022-11-04 07:46:25" ...
## $ start_station_name: chr [1:337735] "Canal St & Adams St" "Canal St & Adams St" "Indiana Ave & Roosevelt Rd" "Indiana Ave & Roosevelt Rd" ...
## $ start_station_id : chr [1:337735] "13011" "13011" "SL-005" "SL-005" ...
## $ end_station_name : chr [1:337735] "St. Clair St & Erie St" "St. Clair St & Erie St" "St. Clair St & Erie St" "St. Clair St & Erie St" ...
## $ end_station_id   : chr [1:337735] "13016" "13016" "13016" "13016" ...
## $ start_lat        : num [1:337735] 41.9 41.9 41.9 41.9 41.9 ...
## $ start_lng        : num [1:337735] -87.6 -87.6 -87.6 -87.6 -87.6 ...
## $ end_lat          : num [1:337735] 41.9 41.9 41.9 41.9 41.9 ...
## $ end_lng          : num [1:337735] -87.6 -87.6 -87.6 -87.6 -87.6 ...
## $ member_casual    : chr [1:337735] "member" "member" "member" "member" ...
```

```
str(dec_trip22)
```

```
## tibble [181,806 × 13] (S3: tbl_df/tbl/data.frame)
## $ ride_id      : chr [1:181806] "65DBD2F447EC51C2" "0C201AA7EA0EA1AD" "E0B148CCB358A49D" "54C5775D2B7C9188" ...
## $ rideable_type : chr [1:181806] "electric_bike" "classic_bike" "electric_bike" "classic_bike" ...
## $ started_at   : POSIXct[1:181806], format: "2022-12-05 10:47:18" "2022-12-18 06:42:33" ...
## $ ended_at     : POSIXct[1:181806], format: "2022-12-05 10:56:34" "2022-12-18 07:08:44" ...
## $ start_station_name: chr [1:181806] "Clifton Ave & Armitage Ave" "Broadway & Belmont Ave" "Sangamon St & Lake St" "Shields Ave & 31st St" ...
## $ start_station_id : chr [1:181806] "TA1307000163" "13277" "TA1306000015" "KA1503000038" ...
## $ end_station_name : chr [1:181806] "Sedgwick St & Webster Ave" "Sedgwick St & Webster Ave" "St. Clair St & Erie St" "Damen Ave & Madison St" ...
## $ end_station_id   : chr [1:181806] "13191" "13191" "13016" "13134" ...
## $ start_lat        : num [1:181806] 41.9 41.9 41.9 41.8 41.9 ...
## $ start_lng        : num [1:181806] -87.7 -87.6 -87.7 -87.6 -87.7 ...
## $ end_lat          : num [1:181806] 41.9 41.9 41.9 41.9 41.9 ...
## $ end_lng          : num [1:181806] -87.6 -87.6 -87.6 -87.7 -87.7 ...
## $ member_casual    : chr [1:181806] "member" "casual" "member" "member" ...
```

We notice that all the values are consistent, except for the column `end_station_id` in `sep_trip22`, which is `num` when it should be `chr`.

Step 5: Data transformation, to make data types in columns consistent

Adjustment to `sep_trip22` to make it consistent.

```
sep_trip22$end_station_id <- as.character(sep_trip22$end_station_id)
```

After this, we can check `sep_trip22` again, to ensure the column `end_station_id` has been changed to the `chr` data type.

```
str(sep_trip22)
```

```
## tibble [701,339 × 13] (S3: tbl_df/tbl/data.frame)
## $ ride_id      : chr [1:701339] "5156990AC19CA285" "E12D4A16BF51C274" "A02B53CD7DB72DD7" "C82E05FEE872DF11" ...
## $ rideable_type : chr [1:701339] "electric_bike" "electric_bike" "electric_bike" "electric_bike" ...
## $ started_at   : POSIXct[1:701339], format: "2022-09-01 08:36:22" "2022-09-01 17:11:29" ...
## $ ended_at     : POSIXct[1:701339], format: "2022-09-01 08:39:05" "2022-09-01 17:14:45" ...
## $ start_station_name: chr [1:701339] NA NA NA NA ...
## $ start_station_id : chr [1:701339] NA NA NA NA ...
## $ end_station_name : chr [1:701339] "California Ave & Milwaukee Ave" NA NA NA ...
## $ end_station_id   : chr [1:701339] "13084" NA NA NA ...
## $ start_lat       : num [1:701339] 41.9 41.9 41.9 41.9 41.9 ...
## $ start_lng       : num [1:701339] -87.7 -87.6 -87.6 -87.7 -87.7 ...
## $ end_lat         : num [1:701339] 41.9 41.9 41.9 41.9 41.9 ...
## $ end_lng         : num [1:701339] -87.7 -87.6 -87.6 -87.7 -87.7 ...
## $ member_casual   : chr [1:701339] "casual" "casual" "casual" "casual" ...
```

We can see that the value types are all now consistent between data frames, so we can move on.

Step 6: Combine to a single data frame for easier analysis

The frames are combined together to make a single frame for the entire year. Part of the reason R was chosen for this project is that the size of the combined table would be larger than excel is capable of working with.

```
full_trip22 <- bind_rows(jan_trip22, feb_trip22, mar_trip22, apr_trip22, may_trip22, jun_trip22, jul_trip22, aug_trip22, sep_trip22,
oct_trip22, nov_trip22, dec_trip22)
```

Step 7: View data types of combined table to see if further adjustments are needed

Since combined tables can change data types in the process, this can be used to ensure that the data types are of usable data.

```
str(full_trip22)
```

```
## tibble [5,667,717 × 13] (S3: tbl_df/tbl/data.frame)
## $ ride_id      : chr [1:5667717] "C2F7DD78E82EC875" "A6CF8980A652D272" "BD0F91DFF741C66D" "CBB80ED419105406" ...
## $ rideable_type : chr [1:5667717] "electric_bike" "electric_bike" "classic_bike" "classic_bike" ...
## $ started_at   : POSIXct[1:5667717], format: "2022-01-13 11:59:47" "2022-01-10 08:41:56" ...
## $ ended_at     : POSIXct[1:5667717], format: "2022-01-13 12:02:44" "2022-01-10 08:46:17" ...
## $ start_station_name: chr [1:5667717] "Glenwood Ave & Touhy Ave" "Glenwood Ave & Touhy Ave" "Sheffield Ave & Fullerton Ave" "Clark St & Bryn Mawr Ave" ...
## $ start_station_id : chr [1:5667717] "525" "525" "TA1306000016" "KA1504000151" ...
## $ end_station_name : chr [1:5667717] "Clark St & Touhy Ave" "Clark St & Touhy Ave" "Greenview Ave & Fullerton Ave" "Paulina St & Montrose Ave" ...
## $ end_station_id   : chr [1:5667717] "RP-007" "RP-007" "TA1307000001" "TA1309000021" ...
## $ start_lat       : num [1:5667717] 42 42 41.9 42 41.9 ...
## $ start_lng       : num [1:5667717] -87.7 -87.7 -87.7 -87.7 -87.6 ...
## $ end_lat         : num [1:5667717] 42 42 41.9 42 41.9 ...
## $ end_lng         : num [1:5667717] -87.7 -87.7 -87.7 -87.7 -87.6 ...
## $ member_casual   : chr [1:5667717] "casual" "casual" "member" "casual" ...
```

Step 8: Add day of week column

One issue with the data as presented is that it would be difficult to use all the time data in a single column for effective visualization. And so we will break it into several different ones.

```
full_trip22$day_of_week <- weekdays(full_trip22$started_at)
```

step 9: Add column for trip duration

Trip duration will be one of the most important values later. It can be determined by taking the time difference between the start and end times.

```
full_trip22$trip_duration <- as.numeric(difftime(full_trip22$ended_at, full_trip22$started_at, units = "mins"))
```


Step 10: Add a new column for the month the bikeshare service was used

Since the time of year can be better serviced dividing by month, rather than mere date, another new column was added.

```
full_trip22$month <- month(full_trip22$started_at, label = TRUE)
```

Step 11: Use the month columns to create a delineation by season

First, we will define our own function that can be used to turn the name of a month into a season. Although season is not strictly speaking necessary, it may be helpful to make more aesthetic visualizations.

```
get_season <- function(month) {  
  season <- character(length(month)) # Initialize an empty vector to store seasons  
  season[month %in% c("Dec", "Jan", "Feb")] <- "Winter"  
  season[month %in% c("Mar", "Apr", "May")] <- "Spring"  
  season[month %in% c("Jun", "Jul", "Aug")] <- "Summer"  
  season[month %in% c("Sep", "Oct", "Nov")] <- "Fall"  
  return(season)  
}
```

Then we use the function to add a column for season.

```
full_trip22 <- full_trip22 %>% mutate(season = get_season(month))
```

Step 12: Add column for hour

For the combined data frame, a column for hour and time of day will be added, so as to determine general time the bikes are used.

```
full_trip22$hour <- hour(full_trip22$started_at)
```

Step 13: Use hour column to create time of day column

Similar to the idea of season, this can help for making visualizations more aesthetic or intuitive.

```
full_trip22$time_of_day <- ifelse(full_trip22$hour >= 5 & full_trip22$hour < 12, "Morning",  
                                ifelse(full_trip22$hour >= 12 & full_trip22$hour < 17, "Afternoon",  
                                ifelse(full_trip22$hour >= 17 & full_trip22$hour < 21,  
                                "Evening", "Night")))
```

Step 14: Check for duplicated columns

Duplicated columns will be checked for and erased, so as to make sure it doesn't make data lopsided with additional entries. Since the printout for this would be long, this will have message and echo set to false, which prevents seeing it.

```
duplicated(full_trip22)
```

It seems there are no duplicated columns, so no adjustment is necessary.

Step 15: Search for missing values

Missing values can disrupt data analysis by giving misleading answers. So in order to tell how to clean a data frame, one should determine how many there are ahead of time.

```
num_null_rows <- sum(rowSums(is.na(full_trip22)) > 0)  
  
cat("Number of rows with NULL values: ", num_null_rows)
```

```
## Number of rows with NULL values: 1590263
```

The value of num_null_rows is 1590328, which is a full 28% of the total data frame. Since this would be too large an amount to delete, without risking losing good data, these values will be replaced with UNKNOWN instead. Since the null values are for start and end station, it is also possible that they are null due to some locations not beginning at a particular station, or not recording it, rather than being bad data.

```
full_trip22 <- data.frame(lapply(full_trip22, function(x) ifelse(is.na(x), "UNKNOWN", x)))
```

Step 16: Search for string inconsistencies

This step is for the purposes of looking for typos or capitalization errors, or other inconsistencies that can disrupt the data. This is because two categories that may be supposed to be the same might be registered as separate entities later on during analysis. This step will only be applied to the columns that have values that will be delineated by specific categories. Since the result is long, the output will not be shown.

```
unique(full_trip22$rideable_type)
unique(full_trip22$member_casual)
unique_start_station <- unique(full_trip22$start_station_name)
unique_start_station <- sort(unique_start_station)
print(unique_start_station)
unique_end_station <- unique(full_trip22$end_station_name)
unique_end_station <- sort(unique_end_station)
print(unique_end_station)
```

There doesn't appear to be any inconsistencies when comparing the data.

Step 17: Look for bad data

We can test for any negative or zero values for trip duration, which implies it is bad data.

```
zero_or_negative_count <- sum(full_trip22$trip_duration <= 0)
```

The returned count is 531, Since we are only looking for positive numbers, we will cut out any without valid values.

```
full_trip22 <- full_trip22[full_trip22$trip_duration > 0, ]
```

step 18: Check trip duration to see if there are any outliers

```
summary(full_trip22$trip_duration)
```

##	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
##	0.02	5.82	10.28	19.45	18.47	41387.25

We can see from this that most trips are about ten minutes, with some longer trips shifting the average to about 13. This is useful information to know, due to perhaps conflicting with intuitions that may suggest that people would use bikes primarily for longer periods.

Since max trip is listed as 41387 minutes, this should be taken as an outlier. While there is difficulty in determining exactly what value is high enough that it should be erased, due to trips outside the standard deviation being common, nearly all trips are under an hour in length, so we can probably safely use a limit of no more than twelve hours.

```
full_trip22 <- full_trip22 %>%
  filter(trip_duration <= 720)
```

Step 19: View summaries

```
summary(full_trip22$trip_duration)
```

##	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
##	0.0167	5.8167	10.2667	15.9065	18.4167	719.1167

```
summary(full_trip22)
```

```
## ride_id rideable_type started_at ended_at
## Length:5659769 Length:5659769 Min. :1.641e+09 Min. :1.641e+09
## Class :character Class :character 1st Qu.:1.654e+09 1st Qu.:1.654e+09
## Mode :character Mode :character Median :1.659e+09 Median :1.659e+09
## Mean :1.658e+09 Mean :1.658e+09
## 3rd Qu.:1.663e+09 3rd Qu.:1.663e+09
## Max. :1.673e+09 Max. :1.673e+09
## start_station_name start_station_id end_station_name end_station_id
## Length:5659769 Length:5659769 Length:5659769 Length:5659769
## Class :character Class :character Class :character Class :character
## Mode :character Mode :character Mode :character Mode :character
##
##
## start_lat start_lng end_lat end_lng
## Min. :41.64 Min. : -87.84 Length:5659769 Length:5659769
## 1st Qu.:41.88 1st Qu.: -87.66 Class :character Class :character
## Median :41.90 Median : -87.64 Mode :character Mode :character
## Mean :41.90 Mean : -87.65
## 3rd Qu.:41.93 3rd Qu.: -87.63
## Max. :45.64 Max. : -73.80
## member_casual day_of_week trip_duration month
## Length:5659769 Length:5659769 Min. : 0.0167 Min. : 1.000
## Class :character Class :character 1st Qu.: 5.8167 1st Qu.: 5.000
## Mode :character Mode :character Median :10.2667 Median : 7.000
## Mean :15.9065 Mean : 7.112
## 3rd Qu.:18.4167 3rd Qu.: 9.000
## Max. :719.1167 Max. :12.000
## season hour time_of_day
## Length:5659769 Min. : 0.00 Length:5659769
## Class :character 1st Qu.:11.00 Class :character
## Mode :character Median :15.00 Mode :character
## Mean :14.22
## 3rd Qu.:18.00
## Max. :23.00
```

```
skim(full_trip22)
```

Data summary

Name full_trip22
Number of rows 5659769
Number of columns 19

Column type frequency:



character 12
numeric 7

Group variables None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
ride_id	0	1	13	23	0	5659769	0
rideable_type	0	1	11	13	0	3	0
start_station_name	0	1	7	64	0	1675	0
start_station_id	0	1	2	44	0	1313	0
end_station_name	0	1	7	64	0	1693	0
end_station_id	0	1	2	44	0	1317	0
end_lat	0	1	1	16	0	1606	0
end_lng	0	1	1	17	0	1590	0
member_casual	0	1	6	6	0	2	0
day_of_week	0	1	6	9	0	7	0
season	0	1	4	6	0	4	0
time_of_day	0	1	5	9	0	4	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
started_at	0	1	11658302575.536646	234.901640995205	0.001653765789	0.001658503052	0.001663314033	0.001672531166	0.001672531166	
ended_at	0	1	11658303529.936646	188.141640995308	0.001653767030	0.001658504159	0.001663314801	0.001672542396	0.001672542396	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
start_lat	0	1	41.90	0.05	41.64	41.88	41.90	41.93	45.64	
start_lng	0	1	-87.65	0.03	-87.84	-87.66	-87.64	-87.63	-73.80	
trip_duration	0	1	15.91	22.15	0.02	5.82	10.27	18.42	719.12	
month	0	1	7.11	2.53	1.00	5.00	7.00	9.00	12.00	
hour	0	1	14.22	5.03	0.00	11.00	15.00	18.00	23.00	

Step 20: Cut down on amount of columns for easier import

Since tableau can struggle with larger file sizes, the data frame had column removed for easier import. Initially it was going to be divided into two separate frames, but it was left as one, due to the glitches occurring with tableau public in terms of file size export.

```
full_trip22export = subset(full_trip22, select = -c(ride_id, started_at, ended_at, start_station_name, start_station_id, end_station_name, end_station_id, start_lat, end_lat, start_lng, end_lng) )
```

Step 21: Export data table for visualization in Tableau

While R also has a variety of data visualization tools, this data will be exported for visualization in tableau, to give a more dynamic interaction.

```
write.xlsx(full_trip22export, file = "full_trip22export.xlsx", rowNames = TRUE)
```

Now it is ready for import.