DATA Wrangling 40 minute

June 7, 2024

[1]:
```r
# Check whether you need to install the `tidyverse` library
require("tidyverse")
library(tidyverse)
```

```
Loading required package: tidyverse
Warning message:
"replacing previous import ‘lifecycle::last_warnings’ by ‘rlang::last_warnings’
when loading ‘tibble’"Warning message:
"replacing previous import ‘ellipsis::check_dots_unnamed’ by
‘rlang::check_dots_unnamed’ when loading ‘tibble’"Warning message:
"replacing previous import ‘ellipsis::check_dots_used’ by
‘rlang::check_dots_used’ when loading ‘tibble’"Warning message:
"replacing previous import ‘ellipsis::check_dots_empty’ by
‘rlang::check_dots_empty’ when loading ‘tibble’"  Attaching packages
                      tidyverse 1.3.0
 ggplot2 3.3.0       purrr   0.3.4
 tibble  3.0.1       dplyr   0.8.5
 tidyr   1.0.2       stringr 1.4.0
 readr   1.3.1       forcats 0.5.0
  Conflicts                          tidyverse_conflicts()
 dplyr::filter() masks stats::filter()
 dplyr::lag()    masks stats::lag()
```

[2]:
```r
# Download raw_bike_sharing_systems.csv
url <- "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/
 ↪IBMDeveloperSkillsNetwork-RP0321EN-SkillsNetwork/labs/datasets/
 ↪raw_bike_sharing_systems.csv"
download.file(url, destfile = "raw_bike_sharing_systems.csv")

# Download raw_cities_weather_forecast.csv
url <- "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/
 ↪IBMDeveloperSkillsNetwork-RP0321EN-SkillsNetwork/labs/datasets/
 ↪raw_cities_weather_forecast.csv"
download.file(url, destfile = "raw_cities_weather_forecast.csv")

# Download raw_worldcities.csv
```

```r
url <- "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/
 ↪IBMDeveloperSkillsNetwork-RP0321EN-SkillsNetwork/labs/datasets/
 ↪raw_worldcities.csv"
download.file(url, destfile = "raw_worldcities.csv")

# Download raw_seoul_bike_sharing.csv
url <- "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/
 ↪IBMDeveloperSkillsNetwork-RP0321EN-SkillsNetwork/labs/datasets/
 ↪raw_seoul_bike_sharing.csv"
download.file(url, destfile = "raw_seoul_bike_sharing.csv")
```

[3]:
```r
dataset_list <- c('raw_bike_sharing_systems.csv', 'raw_seoul_bike_sharing.csv',␣
 ↪'raw_cities_weather_forecast.csv', 'raw_worldcities.csv')
for (i in dataset_list){
    csv <- read_csv(i)
    glimpse(csv)
    }
```

```
Parsed with column specification:
cols(
  COUNTRY = col_character(),
  City = col_character(),
  Name = col_character(),
  SYSTEM = col_character(),
  OPERATOR = col_character(),
  LAUNCHED = col_character(),
  DISCONTINUED = col_character(),
  STATIONS = col_character(),
  BICYCLES = col_character(),
  DAILY_RIDERSHIP = col_character()
)
```

```
Rows: 480
Columns: 10
$ COUNTRY         <chr> "Albania", "Argentina", "Argentina", "Argentina", "Ar…
$ City            <chr> "Tirana", "Mendoza", "San Lorenzo, Santa Fe", "Buenos…
$ Name            <chr> "Ecovolis", "Metrobici", "Biciudad", "Ecobici", "Mi B…
$ SYSTEM          <chr> NA, NA, "Biciudad", "Serttel Brasil", NA, "PBSC & 8D"…
$ OPERATOR        <chr> NA, NA, NA, "Bike In Baires Consortium.[10]", NA, "Mo…
$ LAUNCHED        <chr> "March 2011", "2014", "27 November 2016", "2010", "2 …
$ DISCONTINUED    <chr> NA, NA, NA, NA, NA, "30 November 2019[13]", NA, "July…
$ STATIONS        <chr> "8", "2", "8", "400", "47", "53", "150", "dockless", …
$ BICYCLES        <chr> "200", "40", "80", "4000", "480", "676", "2000", "125…
$ DAILY_RIDERSHIP <chr> NA, NA, NA, "21917", NA, NA, NA, NA, NA, NA, NA, "280…
```

```
Parsed with column specification:
cols(
  Date = col_character(),
  RENTED_BIKE_COUNT = col_double(),
  Hour = col_double(),
  TEMPERATURE = col_double(),
  HUMIDITY = col_double(),
  WIND_SPEED = col_double(),
  Visibility = col_double(),
  DEW_POINT_TEMPERATURE = col_double(),
  SOLAR_RADIATION = col_double(),
  RAINFALL = col_double(),
  Snowfall = col_double(),
  SEASONS = col_character(),
  HOLIDAY = col_character(),
  FUNCTIONING_DAY = col_character()
)

Rows: 8,760
Columns: 14
$ Date                  <chr> "01/12/2017", "01/12/2017", "01/12/2017", "01/1…
$ RENTED_BIKE_COUNT     <dbl> 254, 204, 173, 107, 78, 100, 181, 460, 930, 490…
$ Hour                  <dbl> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 1…
$ TEMPERATURE           <dbl> -5.2, -5.5, -6.0, -6.2, -6.0, -6.4, -6.6, -7.4,…
$ HUMIDITY              <dbl> 37, 38, 39, 40, 36, 37, 35, 38, 37, 27, 24, 21,…
$ WIND_SPEED            <dbl> 2.2, 0.8, 1.0, 0.9, 2.3, 1.5, 1.3, 0.9, 1.1, 0.…
$ Visibility            <dbl> 2000, 2000, 2000, 2000, 2000, 2000, 2000, 2000,…
$ DEW_POINT_TEMPERATURE <dbl> -17.6, -17.6, -17.7, -17.6, -18.6, -18.7, -19.5…
$ SOLAR_RADIATION       <dbl> 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00,…
$ RAINFALL              <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,…
$ Snowfall              <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,…
$ SEASONS               <chr> "Winter", "Winter", "Winter", "Winter", "Winter…
$ HOLIDAY               <chr> "No Holiday", "No Holiday", "No Holiday", "No H…
$ FUNCTIONING_DAY       <chr> "Yes", "Yes", "Yes", "Yes", "Yes", "Yes", "Yes"…

Parsed with column specification:
cols(
  city = col_character(),
  weather = col_character(),
  visibility = col_double(),
  temp = col_double(),
  temp_min = col_double(),
  temp_max = col_double(),
  pressure = col_double(),
  humidity = col_double(),
  wind_speed = col_double(),
  wind_deg = col_double(),
  season = col_character(),
  forecast_datetime = col_datetime(format = "")
)

Rows: 160
```

```
Columns: 12
$ city             <chr> "Seoul", "Seoul", "Seoul", "Seoul", "Seoul", "Seoul…
$ weather          <chr> "Clear", "Clear", "Clouds", "Clouds", "Clouds", "Ra…
$ visibility       <dbl> 10000, 10000, 10000, 10000, 10000, 10000, 10000, 10…
$ temp             <dbl> 12.32, 11.48, 9.99, 7.87, 10.09, 9.74, 11.39, 9.75,…
$ temp_min         <dbl> 10.91, 9.81, 8.82, 7.87, 10.09, 9.74, 11.39, 9.75, …
$ temp_max         <dbl> 12.32, 11.48, 9.99, 7.87, 10.09, 9.74, 11.39, 9.75,…
$ pressure         <dbl> 1015, 1016, 1015, 1014, 1014, 1014, 1012, 1012, 101…
$ humidity         <dbl> 50, 48, 46, 46, 37, 48, 44, 57, 51, 62, 69, 65, 45,…
$ wind_speed       <dbl> 2.18, 1.25, 0.94, 0.83, 1.96, 3.24, 5.65, 5.63, 4.9…
$ wind_deg         <dbl> 248, 142, 130, 31, 309, 267, 275, 267, 280, 284, 29…
$ season           <chr> "Spring", "Spring", "Spring", "Spring", "Spring", "…
$ forecast_datetime <dttm> 2021-04-16 12:00:00, 2021-04-16 15:00:00, 2021-04-…

Parsed with column specification:
cols(
  City = col_character(),
  CITY_ASCII = col_character(),
  LAT = col_double(),
  LNG = col_double(),
  COUNTRY = col_character(),
  ISO2 = col_character(),
  ISO3 = col_character(),
  ADMIN_NAME = col_character(),
  CAPITAL = col_character(),
  POPULATION = col_double(),
  ID = col_double()
)

Rows: 26,569
Columns: 11
$ City       <chr> "Tokyo", "Jakarta", "Delhi", "Mumbai", "Manila", "Shanghai…
$ CITY_ASCII <chr> "Tokyo", "Jakarta", "Delhi", "Mumbai", "Manila", "Shanghai…
$ LAT        <dbl> 35.6897, -6.2146, 28.6600, 18.9667, 14.5958, 31.1667, -23.…
$ LNG        <dbl> 139.6922, 106.8451, 77.2300, 72.8333, 120.9772, 121.4667, …
$ COUNTRY    <chr> "Japan", "Indonesia", "India", "India", "Philippines", "Ch…
$ ISO2       <chr> "JP", "ID", "IN", "IN", "PH", "CN", "BR", "KR", "MX", "CN"…
$ ISO3       <chr> "JPN", "IDN", "IND", "IND", "PHL", "CHN", "BRA", "KOR", "M…
$ ADMIN_NAME <chr> "Tōkyō", "Jakarta", "Delhi", "Mahārāshtra", "Manila", "Sha…
$ CAPITAL    <chr> "primary", "primary", "admin", "admin", "primary", "admin"…
$ POPULATION <dbl> 37977000, 34540000, 29617000, 23355000, 23088000, 22120000…
$ ID         <dbl> 1392685764, 1360771077, 1356872604, 1356226629, 1608618140…
```

[4]:
```r
for (dataset_name in dataset_list){
    # Read dataset
    dataset <- read_csv(dataset_name)
    # Standardized its columns:
    colnames(dataset) <- toupper(colnames(dataset))
    # Convert all column names to uppercase
    colnames(dataset) <- str_replace_all(colnames(dataset), " ", "_")
    # Replace any white space separators by underscores, using the␣
  ↪str_replace_all function

    # Save the dataset
    write.csv(dataset, dataset_name, row.names=FALSE)
}
```

```
Parsed with column specification:
cols(
  COUNTRY = col_character(),
  City = col_character(),
  Name = col_character(),
  SYSTEM = col_character(),
  OPERATOR = col_character(),
  LAUNCHED = col_character(),
  DISCONTINUED = col_character(),
  STATIONS = col_character(),
  BICYCLES = col_character(),
  DAILY_RIDERSHIP = col_character()
)
Parsed with column specification:
cols(
  Date = col_character(),
  RENTED_BIKE_COUNT = col_double(),
  Hour = col_double(),
  TEMPERATURE = col_double(),
  HUMIDITY = col_double(),
  WIND_SPEED = col_double(),
  Visibility = col_double(),
  DEW_POINT_TEMPERATURE = col_double(),
  SOLAR_RADIATION = col_double(),
  RAINFALL = col_double(),
  Snowfall = col_double(),
  SEASONS = col_character(),
  HOLIDAY = col_character(),
  FUNCTIONING_DAY = col_character()
)
Parsed with column specification:
cols(
  city = col_character(),
  weather = col_character(),
  visibility = col_double(),
  temp = col_double(),
  temp_min = col_double(),
  temp_max = col_double(),
  pressure = col_double(),
  humidity = col_double(),
  wind_speed = col_double(),
  wind_deg = col_double(),
  season = col_character(),
  forecast_datetime = col_datetime(format = "")
)
Parsed with column specification:
cols(
  City = col_character(),
  CITY_ASCII = col_character(),
  LAT = col_double(),
  LNG = col_double(),
  COUNTRY = col_character(),
  ISO2 = col_character(),
  ISO3 = col_character(),
  ADMIN_NAME = col_character(),
  CAPITAL = col_character(),
  POPULATION = col_double(),
  ID = col_double()
)
```

[5]:
```
for (dataset_name in dataset_list){
    # Print a summary for each data set to check whether the column names were␣
 ↪correctly converted
    dataset <- read_csv(dataset_name)
  print(colnames(dataset))
}
```

```
Parsed with column specification:
cols(
  COUNTRY = col_character(),
  CITY = col_character(),
  NAME = col_character(),
  SYSTEM = col_character(),
  OPERATOR = col_character(),
  LAUNCHED = col_character(),
  DISCONTINUED = col_character(),
  STATIONS = col_character(),
  BICYCLES = col_character(),
  DAILY_RIDERSHIP = col_character()
)

 [1] "COUNTRY"         "CITY"            "NAME"            "SYSTEM"
 [5] "OPERATOR"        "LAUNCHED"        "DISCONTINUED"    "STATIONS"
 [9] "BICYCLES"        "DAILY_RIDERSHIP"

Parsed with column specification:
cols(
  DATE = col_character(),
  RENTED_BIKE_COUNT = col_double(),
  HOUR = col_double(),
  TEMPERATURE = col_double(),
  HUMIDITY = col_double(),
  WIND_SPEED = col_double(),
  VISIBILITY = col_double(),
  DEW_POINT_TEMPERATURE = col_double(),
  SOLAR_RADIATION = col_double(),
  RAINFALL = col_double(),
  SNOWFALL = col_double(),
  SEASONS = col_character(),
  HOLIDAY = col_character(),
  FUNCTIONING_DAY = col_character()
)

 [1] "DATE"                  "RENTED_BIKE_COUNT"     "HOUR"
 [4] "TEMPERATURE"           "HUMIDITY"              "WIND_SPEED"
 [7] "VISIBILITY"            "DEW_POINT_TEMPERATURE" "SOLAR_RADIATION"
[10] "RAINFALL"              "SNOWFALL"              "SEASONS"
[13] "HOLIDAY"               "FUNCTIONING_DAY"

Parsed with column specification:
cols(
  CITY = col_character(),
  WEATHER = col_character(),
  VISIBILITY = col_double(),
  TEMP = col_double(),
  TEMP_MIN = col_double(),
  TEMP_MAX = col_double(),
  PRESSURE = col_double(),
  HUMIDITY = col_double(),
  WIND_SPEED = col_double(),
  WIND_DEG = col_double(),
  SEASON = col_character(),
  FORECAST_DATETIME = col_datetime(format = "")
)

 [1] "CITY"              "WEATHER"           "VISIBILITY"
 [4] "TEMP"              "TEMP_MIN"          "TEMP_MAX"
 [7] "PRESSURE"          "HUMIDITY"          "WIND_SPEED"
[10] "WIND_DEG"          "SEASON"            "FORECAST_DATETIME"

Parsed with column specification:
cols(
  CITY = col_character(),
  CITY_ASCII = col_character(),
  LAT = col_double(),
  LNG = col_double(),
  COUNTRY = col_character(),
  ISO2 = col_character(),
  ISO3 = col_character(),
  ADMIN_NAME = col_character(),
  CAPITAL = col_character(),
  POPULATION = col_double(),
  ID = col_double()
)
```

```
 [1] "CITY"       "CITY_ASCII" "LAT"        "LNG"        "COUNTRY"
 [6] "ISO2"       "ISO3"       "ADMIN_NAME" "CAPITAL"    "POPULATION"
[11] "ID"
```

[6]:
```
###Process the web-scraped bike sharing system dataset
# First load the dataset
bike_sharing_df <- read_csv("raw_bike_sharing_systems.csv")
```

```
Parsed with column specification:
cols(
  COUNTRY = col_character(),
  CITY = col_character(),
  NAME = col_character(),
  SYSTEM = col_character(),
  OPERATOR = col_character(),
  LAUNCHED = col_character(),
  DISCONTINUED = col_character(),
  STATIONS = col_character(),
  BICYCLES = col_character(),
  DAILY_RIDERSHIP = col_character()
)
```

[7]:
```
# Print its head
head(bike_sharing_df)
```

A tibble: 6 × 10

COUNTRY	CITY	NAME	SYSTEM	OPERATOR
<chr>	<chr>	<chr>	<chr>	<chr>
Albania	Tirana	Ecovolis	NA	NA
Argentina	Mendoza	Metrobici	NA	NA
Argentina	San Lorenzo, Santa Fe	Biciudad	Biciudad	NA
Argentina	Buenos Aires	Ecobici	Serttel Brasil	Bike In Baires C
Argentina	Rosario	Mi Bici Tu Bici[11]	NA	NA
Australia	Melbourne[12]	Melbourne Bike Share	PBSC & 8D	Motivate

[8]:
```
# Select the four columns
sub_bike_sharing_df <- bike_sharing_df %>% select(COUNTRY, CITY, SYSTEM,␣
↪BICYCLES)
```

[9]:
```
sub_bike_sharing_df %>%
    summarize_all(class) %>%
    gather(variable, class)
```

A tibble: 4 × 2

variable <chr>	class <chr>
COUNTRY	character
CITY	character
SYSTEM	character
BICYCLES	character

[10]:
```
# grepl searches a string for non-digital characters, and returns TRUE or FALSE
# if it finds any non-digital characters, then the bicyle column is not purely␣
↪numeric
find_character <- function(strings) grepl("[^0-9]", strings)
```

[11]:
```
sub_bike_sharing_df %>%
    select(BICYCLES) %>%
    filter(find_character(BICYCLES)) %>%
    slice(0:10)
```

A spec_tbl_df: 10 × 1

BICYCLES <chr>
4115[22]
310[59]
500[72]
[75]
180[76]
600[77]
[78]
initially 800 (later 2500)
100 (220)
370[114]

[12]:
```
# Define a 'reference link' character class,
# `[A-z0-9]` means at least one character
# `\\[` and `\\]` means the character is wrapped by [], such as for [12] or␣
↪[abc]
ref_pattern <- "\\[[A-z0-9]+\\]"
find_reference_pattern <- function(strings) grepl(ref_pattern, strings)
```

[13]:
```
# Check whether the COUNTRY column has any reference links
sub_bike_sharing_df %>%
    select(COUNTRY) %>%
    filter(find_reference_pattern(COUNTRY)) %>%
    slice(0:10)
```

A spec_tbl_df: 0 × 1

COUNTRY <chr>

[14]:
```r
# Check whether the CITY column has any reference links
sub_bike_sharing_df %>%
    select(CITY) %>%
    filter(find_reference_pattern(CITY)) %>%
    slice(0:10)
```

A spec_tbl_df: 10 × 1

CITY <chr>
Melbourne[12]
Brisbane[14][15]
Lower Austria[18]
Namur[19]
Brussels[21]
Salvador[23]
Belo Horizonte[24]
João Pessoa[25]
(Pedro de) Toledo[26]
Rio de Janeiro[27]

[15]:
```r
# Check whether the System column has any reference links
sub_bike_sharing_df %>%
    select(SYSTEM) %>%
    filter(find_reference_pattern(SYSTEM)) %>%
    slice(0:10)
```

A spec_tbl_df: 7 × 1

SYSTEM <chr>
EasyBike[58]
4 Gen.[61]
3 Gen. SmooveKey[113]
3 Gen. Smoove[141][142][143][139]
3 Gen. Smoove[179]
3 Gen. Smoove[181]
3 Gen. Smoove[183]

[16]:
```r
##TASK: Remove undesired reference links using regular expressions
#Create a function to remove reference links
remove_ref <- function(strings) {
  ref_pattern <- "\\[[A-z0-9]+\\]" # Define a pattern matching a reference link␣
 ↪such as [1]
  result <- stringr::str_replace_all(strings,ref_pattern,"")  # Replace all␣
 ↪matched substrings with a white space
  result <-  trimws(result)
    return(result)
}
```

```
[17]: # Use the function to remove the reference links
      sub_bike_sharing_df %>% #use mutate and remove_ref fcn to remove ref in CITY␣
       ↪and SYSTEM
        mutate(SYSTEM=remove_ref(SYSTEM),
               CITY=remove_ref(CITY))
```

A spec_tbl_df: 480 × 4

COUNTRY	CITY	SYSTEM
<chr>	<chr>	<chr>
Albania	Tirana	NA
Argentina	Mendoza	NA
Argentina	San Lorenzo, Santa Fe	Biciudad
Argentina	Buenos Aires	Serttel Brasil
Argentina	Rosario	NA
Australia	Melbourne	PBSC & 8D
Australia	Brisbane	3 Gen. Cyclocity
Australia	Melbourne	4 Gen. oBike
Australia	Sydney	4 Gen. oBike
Australia	Sydney	4 Gen. Ofo
Australia	Sydney	Reddy Go
Austria	Vienna	3 Gen. Cyclocity
Austria	Burgenland	3 Gen. nextbike
Austria	Lower Austria	3 Gen. nextbike
Austria	Salzburg	3 Gen. nextbike
Austria	Vienna	2 Gen.
Austria	Vorarlberg	3 Gen. nextbike
Bangladesh	Dhaka	JoBike
Belgium	Namur	3 Gen. Cyclocity
Belgium	Antwerp	3 Gen. Clear CC
Belgium	Brussels	3 Gen. Cyclocity
Brazil	Salvador	tembici
Brazil	Belo Horizonte	Mobilicidade
Brazil	Fortaleza	Mobilicidade
Brazil	João Pessoa	Mobilicidade
Brazil	(Pedro de) Toledo	Toopedalando
Brazil	Rio de Janeiro	tembici
Brazil	São Paulo	tembici
Brazil	Sorocaba	tembici
Bulgaria	Burgas	Mobilicidade
United States	Fullerton, California	Bike Nation
United States	Hoboken, New Jersey	3 Gen. nextbike
United States	Houston, Texas	3 Gen. B-Cycle
United States	Jersey City	8D
United States	Kailua, Hawaii	3 Gen. B-Cycle
United States	Kansas City, Missouri	3 Gen. B-Cycle
United States	Kona District, Hawaii	PBSC
United States	Lansing, MI	A2B Bikeshare
United States	Lincoln, Nebraska	3 Gen. B-Cycle
United States	Los Angeles	3 Gen. B-Cycle
United States	Madison, Wisconsin	3 Gen. B-Cycle
United States	Milwaukee, Wisconsin	3 Gen. B-Cycle
United States	Minneapolis, Minnesota and Saint Paul, Minnesota	PBSC & 8D
United States	Oklahoma City, Oklahoma	Spokies
United States	Omaha, Nebraska	3 Gen. B-Cycle
United States	Philadelphia, Pennsylvania	3 Gen. B-Cycle
United States	Phoenix, Arizona	3 Gen. CycleHop
United States	Pittsburgh, Pennsylvania	3 Gen. nextbike
United States	Portland, Oregon	NA
United States	Portland, Oregon	1 Gen. WhiteBike

[18]:
```
# Check whether all reference links are removed
sub_bike_sharing_df %>%
  select(COUNTRY, CITY, SYSTEM, BICYCLES) %>%
  filter(find_reference_pattern(COUNTRY) | find_reference_pattern(CITY) |␣
 ↪find_reference_pattern(CITY) |find_reference_pattern(BICYCLES) )
```

A spec_tbl_df: 188 × 4

COUNTRY <chr>	CITY <chr>	SYSTEM <chr>
Australia	Melbourne[12]	PBSC &
Australia	Brisbane[14][15]	3 Gen. C
Austria	Lower Austria[18]	3 Gen. n
Belgium	Namur[19]	3 Gen. C
Belgium	Brussels[21]	3 Gen. C
Brazil	Salvador[23]	tembici
Brazil	Belo Horizonte[24]	Mobilici
Brazil	João Pessoa[25]	Mobilici
Brazil	(Pedro de) Toledo[26]	Toopeda
Brazil	Rio de Janeiro[27]	tembici
Brazil	São Paulo[28]	tembici
Brazil	Sorocaba[29]	tembici
Canada	Victoria[30]	NA
Canada	Hamilton[31]	Social B
Canada	Kitchener, Ontario[32][33]	Commun
Canada	Montreal[34]	PBSC &
Canada	Toronto[35]	PBSC
China	Guangzhou[36][37]	NA
China	Taizhou, Jiangsu[38]	NA
China	Taizhou, Jiangsu[38]	NA
China	Chengdu (Jinniu District)[39]	Shangha
China	Chengdu (Gaoxin District [zh])[40]	NA
China	Hangzhou[41][42]	NA
China	Huaian[43]	NA
China	Kunshan[44]	Forever
China	Nantong[45]	Forever
China	Shanghai[46][47][48][49]	Forever
China	Shaoxing[50]	NA
China	Zhenjiang[51]	NA
Colombia	Medellin[52]	3 Gen (f
United States	Battle Creek[277]	3 Gen. B
United States	Black Rock City[278]	Yellow B
United States	Boston, Massachusetts[279]	PBSC &
United States	Charlotte, North Carolina[280]	3 Gen. B
United States	Chattanooga, Tennessee[281]	PBSC
United States	Cincinnati, Ohio[282]	3 Gen. B
United States	El Paso, Texas[283]	3 Gen. B
United States	Eugene, Oregon[284]	PeaceHe
United States	Fargo, ND[285][286]	3 Gen. B
United States	Fort Worth, Texas[287]	3 Gen. B
United States	Hoboken, New Jersey[288]	3 Gen. n
United States	Houston, Texas[289][290]	3 Gen. B
United States	Jersey City[292]	8D
United States	Kailua, Hawaii[293][294]	3 Gen. B
United States	Kansas City, Missouri[295]	3 Gen. B
United States	Lincoln, Nebraska[297]	3 Gen. B
United States	Madison, Wisconsin[298]	3 Gen. B
United States	Milwaukee, Wisconsin[299][300]	3 Gen. B
United States	Minneapolis, Minnesota and Saint Paul, Minnesota[301][302]	PBSC &
United States	Oklahoma City, Oklahoma[303]	Spokies

```
[19]: ##TASK: Extract the numeric value using regular expressions
      extract_num <- function(columns) {
        digitals_pattern <- "[^0-9]" #define a pattern matching digital substring
      # Find the first match using str_extract
          str_extract(columns, digitals_pattern)
      # Convert the result to numeric using the as.numeric() function
        columns <- as.numeric(columns)
      }
```

```
[20]: # Use the mutate() function on the BICYCLES column
      bike_sharing_df %>% #use mutate and to apply function to BICYLCLES
        mutate(BICYCLES=extract_num(BICYCLES))
```

```
Warning message in extract_num(BICYCLES):
"NAs introduced by coercion"
```

A spec_tbl_df: 480 × 10

COUNTRY	CITY	NAME
<chr>	<chr>	<chr>
Albania	Tirana	Ecovoli
Argentina	Mendoza	Metrob
Argentina	San Lorenzo, Santa Fe	Biciuda
Argentina	Buenos Aires	Ecobici
Argentina	Rosario	Mi Bici
Australia	Melbourne[12]	Melbou
Australia	Brisbane[14][15]	CityCy
Australia	Melbourne	oBike
Australia	Sydney	oBike
Australia	Sydney	Ofo
Australia	Sydney	Reddy
Austria	Vienna	Citybik
Austria	Burgenland	LEIHR
Austria	Lower Austria[18]	LEIHR
Austria	Salzburg	nextbik
Austria	Vienna	Viennal
Austria	Vorarlberg	NA
Bangladesh	Dhaka	JoBike
Belgium	Namur[19]	Libiavel
Belgium	Antwerp	Velo
Belgium	Brussels[21]	Villo!
Brazil	Salvador[23]	Bike Sa
Brazil	Belo Horizonte[24]	Bikebh
Brazil	Fortaleza	Biciclet
Brazil	João Pessoa[25]	SAMBA
Brazil	(Pedro de) Toledo[26]	Tooped
Brazil	Rio de Janeiro[27]	Bike Ri
Brazil	São Paulo[28]	Bikesan
Brazil	Sorocaba[29]	Integral
Bulgaria	Burgas	VeloBu
United States	Fullerton, California	OCTA
United States	Hoboken, New Jersey[288]	Hudson
United States	Houston, Texas[289][290]	Houston
United States	Jersey City[292]	Citi Bik
United States	Kailua, Hawaii[293][294]	Hawaii
United States	Kansas City, Missouri[295]	Kansas
United States	Kona District, Hawaii	NA
United States	Lansing, MI	Capital
United States	Lincoln, Nebraska[297]	BikeLN
United States	Los Angeles	Metro I
United States	Madison, Wisconsin[298]	Madison
United States	Milwaukee, Wisconsin[299][300]	Bublr B
United States	Minneapolis, Minnesota and Saint Paul, Minnesota[301][302]	Nice Ri
United States	Oklahoma City, Oklahoma[303]	Spokies
United States	Omaha, Nebraska[304]	Omaha
United States	Philadelphia, Pennsylvania[305][306]	Indego
United States	Phoenix, Arizona	Grid Bi
United States	Pittsburgh, Pennsylvania	Healthy
United States	Portland, Oregon	Biketow
United States	Portland, Oregon	Yellow

```
[21]: summary(bike_sharing_df$BICYCLES)
```

```
   Length     Class      Mode 
      480 character character 
```

```
[22]: write.csv(bike_sharing_df, "bike_sharing_systems.csv")
```