Lesson 3: Cleaning data

Background for this activity

In this activity, you'll review a scenario, and focus on cleaning real data in R. You will learn more about data cleaning functions and perform basic calculations to gain initial insights into your data.

Throughout this activity, you will also have the opportunity to practice writing your own code by making changes to the code chunks yourself. If you encounter an error or get stuck, you can always check the Lesson2_Clean_Solutions .rmd file in the Solutions folder under Week 3 for the complete, correct code.

The scenario

In this scenario, you are a junior data analyst working for a hotel booking company. You have been asked to clean a .csv file that was created after querying a database to combine two different tables from different hotels. In order to learn more about this data, you are going to need to use functions to preview the data's structure, including its columns and rows. You will also need to use basic cleaning functions to prepare this data for analysis.

Step 1: Load packages

In order to start cleaning your data, you will need to by install the required packages. If you have already installed and loaded `tidyverse`, `skimr`, and `janitor` in this session, feel free to skip the code chunks in this step.

```r
install.packages("tidyverse")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.3'
## (as 'lib' is unspecified)
```

```r
install.packages("skimr")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.3'
## (as 'lib' is unspecified)
```

```r
install.packages("janitor")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.3'
## (as 'lib' is unspecified)
```

Once a package is installed, you can load it by running the `library()` function with the package name inside the parentheses:

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.2     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- 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 error
```

```r
library(skimr)
library(janitor)
```

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

Step 2: Import data

The data you have been asked to clean is currently an external .csv file. In order to view and clean it in `R`, you will need to import it. The `tidyverse` library `readr` package has a number of functions for "reading in" or importing data, including .csv files.

In the chunk below, you will use the `read_csv()` function to import data from a .csv file in the project folder called "hotel_bookings.csv" and save it as a data frame called `bookings_df`:

If this line causes an error, copy in the line setwd("projects/Course 7/Week 3") before it.

```r
bookings_df <- read_csv("hotel_bookings.csv")
```

```
## Rows: 119390 Columns: 32
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (13): hotel, arrival_date_month, meal, country, market_segment, distrib...
## dbl  (18): is_canceled, lead_time, arrival_date_year, arrival_date_week_numb...
## date  (1): reservation_status_date
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Step 3: Getting to know your data

Before you start cleaning your data, take some time to explore it. You can use several functions that you are already familiar with to preview your data, including the `head()` function in the code chunk below:

```r
head(bookings_df)
```

```
## # A tibble: 6 x 32
##   hotel        is_canceled lead_time arrival_date_year arrival_date_month
##   <chr>              <dbl>     <dbl>             <dbl> <chr>
## 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
## 6 Resort Hotel           0        14              2015 July
## # i 27 more variables: arrival_date_week_number <dbl>,
## #   arrival_date_day_of_month <dbl>, stays_in_weekend_nights <dbl>,
## #   stays_in_week_nights <dbl>, adults <dbl>, children <dbl>, babies <dbl>,
## #   meal <chr>, country <chr>, market_segment <chr>,
## #   distribution_channel <chr>, is_repeated_guest <dbl>,
## #   previous_cancellations <dbl>, previous_bookings_not_canceled <dbl>,
## #   reserved_room_type <chr>, assigned_room_type <chr>, ...
```

You can also summarize or preview the data with the `str()` and `glimpse()` functions to get a better understanding of the data by running the code chunks below:

```r
str(bookings_df)
```

```
## spc_tbl_ [119,390 x 32] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ hotel                         : chr [1:119390] "Resort Hotel" "Resort Hotel" "Resort Hotel" "Reso
##  $ is_canceled                   : num [1:119390] 0 0 0 0 0 0 0 0 1 1 ...
##  $ lead_time                     : num [1:119390] 342 737 7 13 14 14 0 9 85 75 ...
##  $ arrival_date_year             : num [1:119390] 2015 2015 2015 2015 2015 ...
##  $ arrival_date_month            : chr [1:119390] "July" "July" "July" "July" ...
##  $ arrival_date_week_number      : num [1:119390] 27 27 27 27 27 27 27 27 27 27 ...
##  $ arrival_date_day_of_month     : num [1:119390] 1 1 1 1 1 1 1 1 1 1 ...
##  $ stays_in_weekend_nights       : num [1:119390] 0 0 0 0 0 0 0 0 0 0 ...
##  $ stays_in_week_nights          : num [1:119390] 0 0 1 1 2 2 2 2 3 3 ...
##  $ adults                        : num [1:119390] 2 2 1 1 2 2 2 2 2 2 ...
##  $ children                      : num [1:119390] 0 0 0 0 0 0 0 0 0 0 ...
##  $ babies                        : num [1:119390] 0 0 0 0 0 0 0 0 0 0 ...
##  $ meal                          : chr [1:119390] "BB" "BB" "BB" "BB" ...
##  $ country                       : chr [1:119390] "PRT" "PRT" "GBR" "GBR" ...
##  $ market_segment                : chr [1:119390] "Direct" "Direct" "Direct" "Corporate" ...
##  $ distribution_channel          : chr [1:119390] "Direct" "Direct" "Direct" "Corporate" ...
##  $ is_repeated_guest             : num [1:119390] 0 0 0 0 0 0 0 0 0 0 ...
##  $ previous_cancellations        : num [1:119390] 0 0 0 0 0 0 0 0 0 0 ...
##  $ previous_bookings_not_canceled: num [1:119390] 0 0 0 0 0 0 0 0 0 0 ...
##  $ reserved_room_type            : chr [1:119390] "C" "C" "A" "A" ...
##  $ assigned_room_type            : chr [1:119390] "C" "C" "C" "A" ...
##  $ booking_changes               : num [1:119390] 3 4 0 0 0 0 0 0 0 0 ...
##  $ deposit_type                  : chr [1:119390] "No Deposit" "No Deposit" "No Deposit" "No Deposit
##  $ agent                         : chr [1:119390] "NULL" "NULL" "NULL" "304" ...
##  $ company                       : chr [1:119390] "NULL" "NULL" "NULL" "NULL" ...
##  $ days_in_waiting_list          : num [1:119390] 0 0 0 0 0 0 0 0 0 0 ...
##  $ customer_type                 : chr [1:119390] "Transient" "Transient" "Transient" "Transient" ..
##  $ adr                           : num [1:119390] 0 0 75 75 98 ...
##  $ required_car_parking_spaces   : num [1:119390] 0 0 0 0 0 0 0 0 0 0 ...
##  $ total_of_special_requests     : num [1:119390] 0 0 0 0 1 1 0 1 1 0 ...
##  $ reservation_status            : chr [1:119390] "Check-Out" "Check-Out" "Check-Out" "Check-Out" ..
##  $ reservation_status_date       : Date[1:119390], format: "2015-07-01" "2015-07-01" ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   hotel = col_character(),
##   ..   is_canceled = col_double(),
##   ..   lead_time = col_double(),
##   ..   arrival_date_year = col_double(),
##   ..   arrival_date_month = col_character(),
##   ..   arrival_date_week_number = col_double(),
##   ..   arrival_date_day_of_month = col_double(),
##   ..   stays_in_weekend_nights = col_double(),
##   ..   stays_in_week_nights = col_double(),
##   ..   adults = col_double(),
##   ..   children = col_double(),
##   ..   babies = col_double(),
##   ..   meal = col_character(),
##   ..   country = col_character(),
##   ..   market_segment = col_character(),
##   ..   distribution_channel = col_character(),
##   ..   is_repeated_guest = col_double(),
##   ..   previous_cancellations = col_double(),
##   ..   previous_bookings_not_canceled = col_double(),
##   ..   reserved_room_type = col_character(),
##   ..   assigned_room_type = col_character(),
##   ..   booking_changes = col_double(),
##   ..   deposit_type = col_character(),
##   ..   agent = col_character(),
##   ..   company = col_character(),
##   ..   days_in_waiting_list = col_double(),
##   ..   customer_type = col_character(),
##   ..   adr = col_double(),
##   ..   required_car_parking_spaces = col_double(),
##   ..   total_of_special_requests = col_double(),
##   ..   reservation_status = col_character(),
##   ..   reservation_status_date = col_date(format = "")
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```r
glimpse(bookings_df)
```

```
## Rows: 119,390
## Columns: 32
## $ hotel                          <chr> "Resort Hotel", "Resort Hotel", "Resort~
## $ is_canceled                    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, ~
## $ lead_time                      <dbl> 342, 737, 7, 13, 14, 14, 0, 9, 85, 75, ~
## $ arrival_date_year              <dbl> 2015, 2015, 2015, 2015, 2015, 2015, 201~
## $ arrival_date_month             <chr> "July", "July", "July", "July", "July",~
## $ arrival_date_week_number       <dbl> 27, 27, 27, 27, 27, 27, 27, 27, 27, 27,~
## $ arrival_date_day_of_month      <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ stays_in_weekend_nights        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ stays_in_week_nights           <dbl> 0, 0, 1, 1, 2, 2, 2, 2, 3, 3, 4, 4, 4, ~
## $ adults                         <dbl> 2, 2, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, ~
## $ children                       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ babies                         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ meal                           <chr> "BB", "BB", "BB", "BB", "BB", "BB", "BB~
## $ country                        <chr> "PRT", "PRT", "GBR", "GBR", "GBR", "GBR~
## $ market_segment                 <chr> "Direct", "Direct", "Direct", "Corporat~
## $ distribution_channel           <chr> "Direct", "Direct", "Direct", "Corporat~
## $ is_repeated_guest              <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ previous_cancellations         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ previous_bookings_not_canceled <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ reserved_room_type             <chr> "C", "C", "A", "A", "A", "A", "C", "C",~
## $ assigned_room_type             <chr> "C", "C", "C", "A", "A", "A", "C", "C",~
## $ booking_changes                <dbl> 3, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ deposit_type                   <chr> "No Deposit", "No Deposit", "No Deposit~
## $ agent                          <chr> "NULL", "NULL", "NULL", "304", "240", "~
## $ company                        <chr> "NULL", "NULL", "NULL", "NULL", "NULL",~
## $ days_in_waiting_list           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ customer_type                  <chr> "Transient", "Transient", "Transient", ~
## $ adr                            <dbl> 0.00, 0.00, 75.00, 75.00, 98.00, 98.00,~
## $ required_car_parking_spaces    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ total_of_special_requests      <dbl> 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 3, ~
## $ reservation_status             <chr> "Check-Out", "Check-Out", "Check-Out", ~
## $ reservation_status_date        <date> 2015-07-01, 2015-07-01, 2015-07-02, 20~
```

You can also use `colnames()` to check the names of the columns in your data set. Run the code chunk below to find out the column names in this data set:

```r
colnames(bookings_df)
```

```
##  [1] "hotel"                          "is_canceled"
##  [3] "lead_time"                      "arrival_date_year"
##  [5] "arrival_date_month"             "arrival_date_week_number"
##  [7] "arrival_date_day_of_month"      "stays_in_weekend_nights"
##  [9] "stays_in_week_nights"           "adults"
## [11] "children"                       "babies"
## [13] "meal"                           "country"
## [15] "market_segment"                 "distribution_channel"
## [17] "is_repeated_guest"              "previous_cancellations"
## [19] "previous_bookings_not_canceled" "reserved_room_type"
## [21] "assigned_room_type"             "booking_changes"
## [23] "deposit_type"                   "agent"
## [25] "company"                        "days_in_waiting_list"
## [27] "customer_type"                  "adr"
## [29] "required_car_parking_spaces"    "total_of_special_requests"
## [31] "reservation_status"             "reservation_status_date"
```

Some packages contain more advanced functions for summarizing and exploring your data. One example is the `skimr` package, which has a number of functions for this purpose. For example, the `skim_without_charts()` function provides a detailed summary of the data. Try running the code below:

```r
skim_without_charts(bookings_df)
```

Table 1: Data summary

Name	bookings_df
Number of rows	119390
Number of columns	32
Column type frequency:	
character	13
Date	1
numeric	18
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
hotel	0	1	10	12	0	2	0
arrival_date_month	0	1	3	9	0	12	0
meal	0	1	2	9	0	5	0
country	0	1	2	4	0	178	0
market_segment	0	1	6	13	0	8	0
distribution_channel	0	1	3	9	0	5	0

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
reserved_room_type	0	1	1	1	0	10	0
assigned_room_type	0	1	1	1	0	12	0
deposit_type	0	1	10	10	0	3	0
agent	0	1	1	4	0	334	0
company	0	1	1	4	0	353	0
customer_type	0	1	5	15	0	4	0
reservation_status	0	1	7	9	0	3	0

Variable type: Date

skim_variable	n_missing	complete_rate	min	max	median	n_unique
reservation_status_date	0	1	2014-10-17	2017-09-14	2016-08-07	926

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
is_canceled	0	1	0.37	0.48	0.00	0.00	0.00	1	1
lead_time	0	1	104.01	106.86	0.00	18.00	69.00	160	737
arrival_date_year	0	1	2016.16	0.71	2015.00	2016.00	2016.00	2017	2017
arrival_date_week_number	0	1	27.17	13.61	1.00	16.00	28.00	38	53
arrival_date_day_of_month	0	1	15.80	8.78	1.00	8.00	16.00	23	31
stays_in_weekend_nights	0	1	0.93	1.00	0.00	0.00	1.00	2	19
stays_in_week_nights	0	1	2.50	1.91	0.00	1.00	2.00	3	50
adults	0	1	1.86	0.58	0.00	2.00	2.00	2	55
children	4	1	0.10	0.40	0.00	0.00	0.00	0	10
babies	0	1	0.01	0.10	0.00	0.00	0.00	0	10
is_repeated_guest	0	1	0.03	0.18	0.00	0.00	0.00	0	1
previous_cancellations	0	1	0.09	0.84	0.00	0.00	0.00	0	26
previous_bookings_not_canceled	0	1	0.14	1.50	0.00	0.00	0.00	0	72
booking_changes	0	1	0.22	0.65	0.00	0.00	0.00	0	21
days_in_waiting_list	0	1	2.32	17.59	0.00	0.00	0.00	0	391
adr	0	1	101.83	50.54	-6.38	69.29	94.58	126	5400
required_car_parking_spaces	0	1	0.06	0.25	0.00	0.00	0.00	0	8
total_of_special_requests	0	1	0.57	0.79	0.00	0.00	0.00	1	5

Step 4: Cleaning your data

Based on the functions you have used so far, how would you describe your data in a brief to your stakeholder? Now, let's say you are primarily interested in the following variables: 'hotel', 'is_canceled', and 'lead_time'. Create a new data frame with just those columns, calling it `trimmed_df` by adding the variable names to this code chunk:

```r
trimmed_df <- bookings_df %>%
  select( 'hotel', 'is_canceled', 'lead_time' )
```

Remember to check the solutions doc if you are having trouble filling out any of these code chunks.

You might notice that some of the column names aren't very intuitive, so you will want to rename them to make them easier to understand. You might want to create the same exact data frame as above, but rename the variable 'hotel' to be named 'hotel_type' to be crystal clear on what the data is about

Fill in the space to the left of the '=' symbol with the new variable name:

```
trimmed_df %>%
  select(hotel, is_canceled, lead_time) %>%
  rename( hotel_type = hotel)
```

```
## # A tibble: 119,390 x 3
##    hotel_type   is_canceled lead_time
##    <chr>              <dbl>     <dbl>
##  1 Resort Hotel           0       342
##  2 Resort Hotel           0       737
##  3 Resort Hotel           0         7
##  4 Resort Hotel           0        13
##  5 Resort Hotel           0        14
##  6 Resort Hotel           0        14
##  7 Resort Hotel           0         0
##  8 Resort Hotel           0         9
##  9 Resort Hotel           1        85
## 10 Resort Hotel           1        75
## # i 119,380 more rows
```

Another common task is to either split or combine data in different columns. In this example, you can combine the arrival month and year into one column using the unite() function:

```
example_df <- bookings_df %>%
  select(arrival_date_year, arrival_date_month) %>%
  unite(arrival_month_year, c("arrival_date_month", "arrival_date_year"), sep = " ")
```

Step 5: Another way of doing things

You can also use the`mutate()` function to make changes to your columns. Let's say you wanted to create a new column that summed up all the adults, children, and babies on a reservation for the total number of people. Modify the code chunk below to create that new column:

```
example_df <- bookings_df %>%
  mutate(guests = adults, + children, + babies)

head(example_df)
```

```
## # A tibble: 6 x 35
##   hotel        is_canceled lead_time arrival_date_year arrival_date_month
##   <chr>              <dbl>     <dbl>             <dbl> <chr>
## 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
## 6 Resort Hotel           0        14              2015 July
## # i 30 more variables: arrival_date_week_number <dbl>,
## #   arrival_date_day_of_month <dbl>, stays_in_weekend_nights <dbl>,
## #   stays_in_week_nights <dbl>, adults <dbl>, children <dbl>, babies <dbl>,
## #   meal <chr>, country <chr>, market_segment <chr>,
## #   distribution_channel <chr>, is_repeated_guest <dbl>,
## #   previous_cancellations <dbl>, previous_bookings_not_canceled <dbl>,
## #   reserved_room_type <chr>, assigned_room_type <chr>, ...
```

Great. Now it's time to calculate some summary statistics! Calculate the total number of canceled bookings and the average lead time for booking - you'll want to start your code after the %>% symbol. Make a column called 'number_canceled' to represent the total number of canceled bookings. Then, make a column called 'average_lead_time' to represent the average lead time. Use the `summarize()` function to do this in the code chunk below:

```
example_df <- bookings_df %>%
  summarize(number_canceled = sum(is_canceled),
            average_lead_time = mean(lead_time))

head(example_df)
```

```
## # A tibble: 1 x 2
##   number_canceled average_lead_time
##             <dbl>             <dbl>
## 1           44224              104.
```

If you are having trouble completing any of the code chunks in these activities, remember that you can reference the RMarkdown documents in the 'Solutions' for help.

Activity Wrap Up

Now you have some experience cleaning and analyzing data in `R`; you used basic cleaning functions like `rename()` and performed basic calculations on real data. You can continue to practice these skills by modifying the code chunks in the rmd file, or use this code as a starting point in your own project console. One of the reasons `R` is such a powerful tool for data analysis is because you can perform so many different tasks in one place. With the functions you have been learning in this course, you can import data, create and view data frames, and even clean data without leaving your console.

Make sure to mark this activity as complete in Coursera.