Analysis of Global COVID-19 Pandemic Data

Estimated time needed: **90** minutes

Overview:

There are 10 tasks in this final project. All tasks will be graded by your peers who are also completing this assignment within the same session.

You need to submit the following the screenshot for the code and output for each task for review.

If you need to refresh your memories about specific coding details, you may refer to previous hands-on labs for code examples.

In [2]:
```
# This lab requires 'httr' and 'rvest'packages, which are already pre-loaded into this l
# However, if you are working on your local RStudio, please uncomment the below codes an

install.packages("httr")
install.packages("rvest")
install.packages("curl")
```

```
also installing the dependency ‘curl’

Updating HTML index of packages in '.Library'
Making 'packages.html' ... done
Warning message:
“dependency ‘lifecycle’ is not available”also installing the dependency ‘rlang’

Warning message in install.packages("rvest"):
“installation of package ‘rvest’ had non-zero exit status”Updating HTML index of package
s in '.Library'
Making 'packages.html' ... done
Updating HTML index of packages in '.Library'
Making 'packages.html' ... done
```

In [3]:
```
library(httr)
library(rvest)
library(curl)
```

```
Loading required package: xml2
Using libcurl 7.68.0 with OpenSSL/1.1.1t

Attaching package: ‘curl’

The following object is masked from ‘package:httr’:

    handle_reset
```

Note: if you can import above libraries, please use install.packages() to install them first.

TASK 1: Get a `COVID-19 pandemic` Wiki page using HTTP request

First, let's write a function to use HTTP request to get a public COVID-19 Wiki page.

Before you write the function, you can open this public page from this

URL https://en.wikipedia.org/w/index.php?title=Template:COVID-19_testing_by_country using a web browser.

The goal of task 1 is to get the html page using HTTP request (`httr` library)

In [126…
```
get_wiki_covid19_page <- "https://en.wikipedia.org/w/index.php?title=Template:COVID-19_t
```

Call the `get_wiki_covid19_page` function to get a http response with the target html page

In [125…
```
# Call the get_wiki_covid19_page function and print the response
response <- GET(get_wiki_covid19_page)
response
```

```
Response [https://en.wikipedia.org/w/index.php?title=Template:COVID-19_testing_by_countr
y]
  Date: 2023-11-20 01:22
  Status: 200
  Content-Type: text/html; charset=UTF-8
  Size: 447 kB
<!DOCTYPE html>
<html class="client-nojs vector-feature-language-in-header-enabled vector-fea...
<head>
<meta charset="UTF-8">
<title>Template:COVID-19 testing by country - Wikipedia</title>
<script>(function(){var className="client-js vector-feature-language-in-heade...
"wgDefaultDateFormat":"dmy","wgMonthNames":["","January","February","March","...
"CS1 Russian-language sources (ru)","CS1 Bosnian-language sources (bs)","CS1 ...
"CS1 Malagasy-language sources (mg)","CS1 Malay-language sources (ms)","CS1 R...
"wgIsProbablyEditable":false,"wgRelevantPageIsProbablyEditable":false,"wgRest...
...
```

TASK 2: Extract COVID-19 testing data table from the wiki HTML page

On the COVID-19 testing wiki page, you should see a data table `<table>` node contains COVID-19 testing data by country on the page:

COVID-19 testing statistics by country

[hide]

Country or region	Date[a]	Tested	Units[b]	Confirmed (cases)	Confirmed / tested, %	Tested / population, %	Confirmed / population, %	Ref.
Afghanistan	17 Dec 2020	154,767	samples	49,621	32.1	0.40	0.13	[1]
Albania	18 Feb 2021	428,654	samples	96,838	22.6	15.0	3.4	[2]
Algeria	2 Nov 2020	230,553	samples	58,574	25.4	0.53	0.13	[3][4]
Andorra	15 Mar 2021	162,071	samples	11,285	7.0	209	14.6	[5]
Angola	12 Mar 2021	399,228	samples	20,981	5.3	1.3	0.067	[6]
Antigua and Barbuda	6 Mar 2021	15,268	samples	832	5.4	15.9	0.86	[7]
Argentina	25 Mar 2021	8,517,821	samples	2,278,115	26.7	18.8	5.0	[8]
Armenia	25 Mar 2021	822,634	samples	187,441	22.8	27.9	6.4	[9]
Australia	25 Mar 2021	15,334,583	samples	29,228	0.19	61.1	0.12	[10]
Austria	25 Mar 2021	21,147,134	samples	523,461	2.5	238	5.9	[11]
Azerbaijan	24 Mar 2021	2,799,101	samples	249,492	8.9	28.3	2.5	[12]
Bahamas	23 Mar 2021	73,979	samples	8,953	12.1	19.2	2.3	[13]
Bahrain	24 Mar 2021	3,464,973	samples	138,283	4.0	221	8.8	[14]
Bangladesh	5 Mar 2021	4,119,031	samples	549,184	13.3	2.5	0.33	[15]
Barbados	24 Mar 2021	137,322	samples	3,593	2.6	48.2	1.3	[16]
Belarus	25 Mar 2021	5,272,490	samples	314,993	6.0	55.5	3.3	[17]
Belgium	25 Mar 2021	10,772,328	samples	854,608	7.9	93.5	7.4	[18]
Belize	24 Mar 2021	95,541	samples	12,410	13.0	23.4	3.0	[19]
Benin	23 Mar 2021	520,466		6,501	1.2	4.4	0.055	[20]
Bhutan	26 Mar 2021	586,497	samples	870	0.15	79.1	0.12	[21]
Bolivia	23 Mar 2021	856,948	cases	266,086	31.1	7.5	2.3	[22]

Note the numbers you actually see on your page may be different from above because it is still an on-going pandemic when creating this notebook.

The goal of task 2 is to extract above data table and convert it into a data frame

Now use the `read_html` function in rvest library to get the root html node from response

In [6]:
```
# Get the root html node from the http response in task 1
covid19_root_node <- read_html( "https://en.wikipedia.org/w/index.php?title=Template:COV
covid19_root_node
```

```
{html_document}
<html class="client-nojs vector-feature-language-in-header-enabled vector-feature-langua
ge-in-main-page-header-disabled vector-feature-sticky-header-disabled vector-feature-pag
e-tools-pinned-disabled vector-feature-toc-pinned-clientpref-1 vector-feature-main-menu-
pinned-disabled vector-feature-limited-width-clientpref-1 vector-feature-limited-width-c
ontent-enabled vector-feature-zebra-design-disabled vector-feature-custom-font-size-clie
ntpref-0 vector-feature-client-preferences-disabled vector-feature-typography-survey-dis
abled vector-toc-available" lang="en" dir="ltr">
[1] <head>\n<meta http-equiv="Content-Type" content="text/html; charset=UTF-8 ...
[2] <body class="skin-vector skin-vector-search-vue mediawiki ltr sitedir-ltr ...
```

Get the tables in the HTML root node using `html_nodes` function.

In [7]:
```
# Get the table node from the root html node
covid19_table_node <- html_node(covid19_root_node, "table")
covid19_table_node
```

```
{html_node}
<table class="box-Update plainlinks ombox ombox-content ambox-Update" role="presentatio
n">
[1] <tbody><tr>\n<td class="mbox-image"><span typeof="mw:File"><span><img alt ...
```

Read the specific table from the multiple tables in the `table_node` using the `html_table` function and convert it into dataframe using `as.data.frame`

_Hint:- Please read the `table_node` with index 2(ex:- table*node[2]).*

In [127…
```
# Read the table node and convert it into a data frame, and print the data frame for rev
covid19_data_frame <- html_table(covid19_table_node)
head(covid19_data_frame)
```

A data.frame: 1 × 2

X1	X2

	<lgl>	<chr>
1	NA	This template needs to be updated. Please help update this template to reflect recent events or newly available information. Relevant discussion may be found on the talk page.

In [10]:
```
library(rvest)

# URL of the page containing the table
url <- "https://en.wikipedia.org/w/index.php?title=Template:COVID-19_testing_by_country"
library(rvest)

# Read the HTML content from the webpage
webpage <- read_html(url)

# Extract the table using the provided XPath
table_nodes <- html_nodes(webpage, xpath = "/html/body/div[2]/div/div[3]/main/div[3]/div

# Check if the table node is found
if (length(table_nodes) > 0) {
  # Convert the table node into a data frame
  covid19_data_frame <- html_table(table_nodes[[1]], fill = TRUE)

  # Display the extracted data frame
  head(covid19_data_frame)
} else {
  print("No table found with the specified XPath.")
}
```

A data.frame: 6 × 9

	Country or region	Date[a]	Tested	Units[b]	Confirmed(cases)	Confirmed /tested,%	Tested /population,%	Confirme
	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>	
1	Afghanistan	17 Dec 2020	154,767	samples	49,621	32.1	0.40	
2	Albania	18 Feb 2021	428,654	samples	96,838	22.6	15.0	
3	Algeria	2 Nov 2020	230,553	samples	58,574	25.4	0.53	
4	Andorra	23 Feb 2022	300,307	samples	37,958	12.6	387	
5	Angola	2 Feb 2021	399,228	samples	20,981	5.3	1.3	
6	Antigua and Barbuda	6 Mar 2021	15,268	samples	832	5.4	15.9	

In [86]:
```
str(covid19_data_frame)
```

```
function (object, ...)
UseMethod("str")
```

A data.frame: 173 × 9

Country or region	Date[a]	Tested	Units[b]	Confirmed(cases)	Confirmed /tested,%
<chr>	<chr>	<chr>	<chr>	<chr>	<chr>
Afghanistan	17 Dec 2020	154,767	samples	49,621	32.1
Albania	18 Feb 2021	428,654	samples	96,838	22.6
Algeria	2 Nov 2020	230,553	samples	58,574	25.4

Andorra	23 Feb 2022	300,307	samples	37,958	12.6
Angola	2 Feb 2021	399,228	samples	20,981	5.3
Antigua and Barbuda	6 Mar 2021	15,268	samples	832	5.4
Argentina	16 Apr 2022	35,716,069	samples	9,060,495	25.4
Armenia	29 May 2022	3,099,602	samples	422,963	13.6
Australia	9 Sep 2022	78,548,492	samples	10,112,229	12.9
Austria	1 Feb 2023	205,817,752	samples	5,789,991	2.8
Azerbaijan	11 May 2022	6,838,458	samples	792,638	11.6
Bahamas	28 Nov 2022	259,366	samples	37,483	14.5
Bahrain	3 Dec 2022	10,578,766	samples	696,614	6.6
Bangladesh	24 Jul 2021	7,417,714	samples	1,151,644	15.5
Barbados	14 Oct 2022	770,100	samples	103,014	13.4
Belarus	9 May 2022	13,217,569	samples	982,809	7.4
Belgium	24 Jan 2023	36,548,544	samples	4,691,499	12.8
Belize	8 Jun 2022	572,900	samples	60,694	10.6
Benin	4 May 2021	595,112	samples	7,884	1.3
Bhutan	28 Feb 2022	1,736,168	samples	12,702	0.73
Bolivia	5 Jun 2022	4,358,669	cases	910,228	20.9
Bosnia and Herzegovina	27 Sep 2022	1,872,934	samples	399,887	21.4
Botswana	11 Jan 2022	2,026,898		232,432	11.5
Brazil	19 Feb 2021	23,561,497	samples	10,081,676	42.8
Brunei	2 Aug 2021	153,804	samples	338	0.22
Bulgaria	2 Feb 2023	10,993,239	samples	1,295,524	11.8
Burkina Faso	4 Mar 2021	158,777	samples	12,123	7.6
Burundi	5 Jan 2021	90,019		884	0.98
Cambodia	1 Aug 2021	1,812,706		77,914	4.3
Cameroon	18 Feb 2021	942,685	samples	32,681	3.5
⋮	⋮	⋮	⋮	⋮	⋮
Singapore	3 Aug 2021	16,206,203	samples	65,315	0.40
Slovakia	2 Feb 2023	7,391,882	samples	1,861,034	25.2
Slovenia	2 Feb 2023	2,826,117	samples	1,322,282	46.8
South Africa	24 May 2021	11,378,282	cases	1,637,848	14.4
South Korea	1 Mar 2021	6,592,010	samples	90,029	1.4
South Sudan	26 May 2021	164,472		10,688	6.5
Spain	1 Jul 2021	54,128,524	samples	3,821,305	7.1
Sri Lanka	30 Mar 2021	2,384,745	samples	93,128	3.9
Sudan	7 Jan 2021	158,804	samples	23,316	14.7
Sweden	24 May 2021	9,996,795	samples	1,074,751	10.8
Switzerland[l]	7 Nov 2022	23,283,909	samples	4,276,836	18.4

Taiwan[m]	3 Feb 2023	30,275,725	samples	8,622,129	28.48
Tanzania	18 Nov 2020	3,880		509	13.1
Thailand	4 Mar 2021	1,579,597	cases	26,162	1.7
Togo	6 Jan 2023	807,269		39,358	4.9
Trinidad and Tobago	3 Jan 2022	512,730	cases	92,997	18.1
Tunisia	23 Aug 2021	2,893,625	samples	703,732	24.3
Turkey	2 Jul 2021	61,236,294	samples	5,435,831	8.9
Uganda	11 Feb 2021	852,444	samples	39,979	4.7
Ukraine	24 Nov 2021	15,648,456	samples	3,367,461	21.5
United Arab Emirates	1 Feb 2023	198,685,717	samples	1,049,537	0.53
United Kingdom	19 May 2022	522,526,476	samples	22,232,377	4.3
United States	29 Jul 2022	929,349,291	samples	90,749,469	9.8
Uruguay	16 Apr 2022	6,089,116	samples	895,592	14.7
Uzbekistan	7 Sep 2020	2,630,000	samples	43,975	1.7
Venezuela	30 Mar 2021	3,179,074	samples	159,149	5.0
Vietnam	28 Aug 2022	45,772,571	samples	11,403,302	24.9
Zambia	10 Mar 2022	3,301,860	samples	314,850	9.5
Zimbabwe	15 Oct 2022	2,529,087	samples	257,893	10.2

.mw-parser-output .reflist{font-size:90%;margin-bottom:0.5em;list-style-type:decimal}.mw-parser-output .reflist .references{font-size:100%;margin-bottom:0;list-style-type:inherit}.mw-parser-output .reflist-columns-2{column-width:30em}.mw-parser-output .reflist-columns-3{column-width:25em}.mw-parser-output .reflist-columns{margin-top:0.3em}.mw-parser-output .reflist-columns ol{margin-top:0}.mw-parser-output .reflist-columns li{page-break-inside:avoid;break-inside:avoid-column}.mw-parser-output .reflist-upper-alpha{list-style-type:upper-alpha}.mw-parser-output .reflist-upper-roman{list-style-type:upper-roman}.mw-parser-output .reflist-lower-alpha{list-style-type:lower-

upper-roman{list-style-type:upper-roman}.mw-parser-output .reflist-lower-alpha{list-style-type:lower-alpha}.mw-parser-output .reflist-lower-greek{list-style-type:lower-greek}.mw-parser-output .reflist-lower-roman{list-style-type:lower-roman} ^ Local time. ^ For some countries it is unclear whether they report samples or cases. One person tested twice is recorded as one case and two samples. ^ Excluding Taiwan. ^ Excluding Northern Cyprus. ^ Excluding Greenland and the Faroe Islands. ^ Excluding Overseas France. ^ Testing data from 4 May to 12 May is missing because of the transition to the new reporting system SI-DEP. ^ Excluding Abkhazia and South Ossetia. ^ Data for residents only. ^ Excluding Transnistria. ^ Northern Cyprus is not recognized as a sovereign state by any country except Turkey. ^ Includes data for Liechtenstein. ^ Not a United Nations member.

TASK 3: Pre-process and export the extracted data frame

The goal of task 3 is to pre-process the extracted data frame from the previous step, and export it as a csv file

Let's get a summary of the data frame

As you can see from the summary, the columns names are little bit different to understand and some column data types are not correct. For example, the `Tested` column shows as `character`.

As such, the data frame read from HTML table will need some pre-processing such as removing irrelvant columns, renaming columns, and convert columns into proper data types.

In [16]:
```
# Print the summary of the data frame
summary(covid19_data_frame)
```

```
 Country or region    Date[a]             Tested            Units[b]
 Length:173         Length:173         Length:173         Length:173
 Class :character   Class :character   Class :character   Class :character
 Mode  :character   Mode  :character   Mode  :character   Mode  :character
 Confirmed(cases)   Confirmed/tested,% Tested/population,%
 Length:173         Length:173          Length:173
 Class :character   Class :character    Class :character
 Mode  :character   Mode  :character    Mode  :character
 Confirmed/population,%     Ref.
 Length:173             Length:173
 Class :character       Class :character
 Mode  :character       Mode  :character
```

In [23]:
```
str(covid19_data_frame)
```

```
'data.frame':   173 obs. of  9 variables:
 $ Country or region      : chr  "Afghanistan" "Albania" "Algeria" "Andorra" ...
 $ Date[a]                : chr  "17 Dec 2020" "18 Feb 2021" "2 Nov 2020" "23 Feb 2022"
...
 $ Tested                 : chr  "154,767" "428,654" "230,553" "300,307" ...
 $ Units[b]               : chr  "samples" "samples" "samples" "samples" ...
 $ Confirmed(cases)       : chr  "49,621" "96,838" "58,574" "37,958" ...
 $ Confirmed/tested,%    : chr  "32.1" "22.6" "25.4" "12.6" ...
 $ Tested/population,%   : chr  "0.40" "15.0" "0.53" "387" ...
 $ Confirmed/population,%: chr  "0.13" "3.4" "0.13" "49.0" ...
 $ Ref.                   : chr  "[1]" "[2]" "[3][4]" "[5]" ...
```

Call the `preprocess_covid_data_frame` function

We have prepared a pre-processing function for you to conver the data frame but you can also try to write one by yourself

In [28]:
```
# ## preprocess_covid_data_frame <- function(data_frame)
# {

#     shape <- dim(data_frame)

#     # Remove the World row
#     data_frame<-data_frame[!(data_frame$`Country.or.region`=="World"),]
#     # Remove the last row
#     data_frame <- data_frame[1:172, ]

#     # We dont need the Units and Ref columns, so can be removed
#     data_frame["Ref."] <- NULL
#     data_frame["Units.b."] <- NULL

#     # Renaming the columns
#     names(data_frame) <- c("country", "date", "tested", "confirmed", "confirmed.tested

#     # Convert column data types
#     data_frame$country <- as.factor(data_frame$country)
#     data_frame$date <- as.factor(data_frame$date)
#     data_frame$tested <- as.numeric(gsub(",","",data_frame$tested))
#     data_frame$confirmed <- as.numeric(gsub(",","",data_frame$confirmed))
#     data_frame$'confirmed.tested.ratio' <- as.numeric(gsub(",","",data_frame$`confirme
#     data_frame$'tested.population.ratio' <- as.numeric(gsub(",","",data_frame$`tested.
#     data_frame$'confirmed.population.ratio' <- as.numeric(gsub(",","",data_frame$`conf
#     data_frame$ref <- NULL
#     return(data_frame)
# }
```

In [98]:
```r
# Ensure column names are in sync with the provided dataset
preprocess_covid_data_frame <- function(data_frame)
    {
col_names <- c("Country or region", "Date[a]", "Tested", "Units[b]", "Confirmed(cases)",
  names(data_frame) <- col_names

  # Remove the World row
  data_frame <- data_frame[!(data_frame$`Country or region` == "World"), ]
  # Remove the last row
  data_frame <- data_frame[1:(nrow(data_frame) - 1), ]


  # Renaming the columns
  names(data_frame) <- c("country", "date", "tested", "Units", "confirmed", "confirmed.t

  # Convert column data types
  data_frame$country <- as.factor(data_frame$country)
  data_frame$date <- as.Date(data_frame$date, format = "%d %b %Y") # Assuming Date[a] fo
  data_frame$tested <- as.numeric(gsub(",", "", data_frame$tested))
  data_frame$confirmed <- as.numeric(gsub(",", "", data_frame$confirmed))
  data_frame$confirmed.tested.ratio <- as.numeric(data_frame$confirmed.tested.ratio)
  data_frame$tested.population.ratio <- as.numeric(data_frame$tested.population.ratio)
  data_frame$confirmed.population.ratio <- as.numeric(data_frame$confirmed.population.ra

# Remove problematic characters and replace them with NA in 'confirmed' and 'confirmed.t
  data_frame$confirmed <- as.numeric(gsub("[^0-9.]", "", data_frame$confirmed))
  data_frame$confirmed.tested.ratio <- as.numeric(gsub("[^0-9.]", "", data_frame$confirm

  # Handle NA values or empty strings after cleaning
  data_frame$confirmed[is.na(data_frame$confirmed)] <- NA
  data_frame$confirmed.tested.ratio[is.na(data_frame$confirmed.tested.ratio)] <- NA

  # Remaining preprocessing steps
  data_frame$country <- as.factor(data_frame$country)
  data_frame$date <- as.Date(data_frame$date, format = "%Y-%m-%d")
  data_frame$tested <- as.numeric(gsub(",", "", data_frame$tested))
  data_frame$tested.population.ratio <- as.numeric(data_frame$tested.population.ratio)
  data_frame$confirmed.population.ratio <- as.numeric(data_frame$confirmed.population.ra


  # Remove unnecessary columns
  data_frame <- data_frame[, !(names(data_frame) %in% c("Ref.", "Units"))]

  data_frame$Ref. <- NULL


  return(data_frame)
}

# Call the updated function with your dataset
wiki_covid19_data_frame <- preprocess_covid_data_frame(covid19_data_frame)
```

```
Warning message in preprocess_covid_data_frame(covid19_data_frame):
“NAs introduced by coercion”
```

In [102…
```r
# call `preprocess_covid_data_frame` function and assign it to a new data frame
wiki_covid19_data_frame <- preprocess_covid_data_frame(covid19_data_frame)

wiki_covid19_data_frame <- wiki_covid19_data_frame[, -ncol(wiki_covid19_data_frame)]


head(wiki_covid19_data_frame)
```

```
Warning message in preprocess_covid_data_frame(covid19_data_frame):
“NAs introduced by coercion”
```

A data.frame: 6 × 7

	country	date	tested	confirmed	confirmed.tested.ratio	tested.population.ratio	confirmed.population.ra
	<fct>	<date>	<dbl>	<dbl>	<dbl>	<dbl>	<dl
1	Afghanistan	2020-12-17	154767	49621	32.1	0.40	0.1
2	Albania	2021-02-18	428654	96838	22.6	15.00	3.4
3	Algeria	2020-11-02	230553	58574	25.4	0.53	0.1
4	Andorra	2022-02-23	300307	37958	12.6	387.00	49.0
5	Angola	2021-02-02	399228	20981	5.3	1.30	0.0
6	Antigua and Barbuda	2021-03-06	15268	832	5.4	15.90	0.8

In [103…
```
str(wiki_covid19_data_frame)
```

```
'data.frame':   172 obs. of  7 variables:
 $ country                  : Factor w/ 172 levels "Afghanistan",..: 1 2 3 4 5 6 7 8 9
10 ...
 $ date                     : Date, format: "2020-12-17" "2021-02-18" ...
 $ tested                   : num  154767 428654 230553 300307 399228 ...
 $ confirmed                : num  49621 96838 58574 37958 20981 ...
 $ confirmed.tested.ratio   : num  32.1 22.6 25.4 12.6 5.3 5.4 25.4 13.6 12.9 2.8 ...
 $ tested.population.ratio  : num  0.4 15 0.53 387 1.3 15.9 78.3 105 313 NA ...
 $ confirmed.population.ratio: num  0.13 3.4 0.13 49 0.067 0.86 20 14.3 40.3 65 ...
```

In [52]:
```

```

<NA>

<NA> · 832 · 338 · 884 · 161 · 45 · 494 · 136 · 0 · 961 · 995 · 509

Get the summary of the processed data frame again

In [ ]:
```
# Print the summary of the processed data frame again
```

After pre-processing, you can see the columns and columns names are simplified, and columns types are converted into correct types.

The data frame has following columns:

- **country** - The name of the country
- **date** - Reported date
- **tested** - Total tested cases by the reported date
- **confirmed** - Total confirmed cases by the reported date
- **confirmed.tested.ratio** - The ratio of confirmed cases to the tested cases
- **tested.population.ratio** - The ratio of tested cases to the population of the country
- **confirmed.population.ratio** - The ratio of confirmed cases to the population of the country

OK, we can call `write.csv()` function to save the csv file into a file.

In [104…
```
# Export the data frame to a csv file
write.csv(wiki_covid19_data_frame, file = "covid.csv", row.names = FALSE)
```

Note for IBM Waston Studio, there is no traditional "hard disk" associated with a R workspace.

Even if you call `write.csv()` method to save the data frame as a csv file, it won't be shown in IBM Cloud Object Storage asset UI automatically.

However, you may still check if the `covid.csv` exists using following code snippet:

In [105…
```
# Get working directory
wd <- getwd()
# Get exported
file_path <- paste(wd, sep="", "/covid.csv")
# File path
print(file_path)
file.exists(file_path)
```

```
[1] "/resources/labs/authoride/IBMSkillsNetwork+RP0101EN/v2/M5_Final/covid.csv"
TRUE
```

Optional Step: If you have difficulties finishing above webscraping tasks, you may still continue with next tasks by downloading a provided csv file from here:

In [ ]:
```
## Download a sample csv file
# covid_csv_file <- download.file("https://cf-courses-data.s3.us.cloud-object-storage.ap
# covid_data_frame_csv <- read.csv("covid.csv", header=TRUE, sep=",")
```

TASK 4: Get a subset of the extracted data frame

The goal of task 4 is to get the 5th to 10th rows from the data frame with only `country` and `confirmed` columns selected

In [106…
```
# Read covid_data_frame_csv from the csv file
covid_data_frame_csv <- read.csv("covid.csv", header=TRUE, sep=",")
# Get the 5th to 10th rows, with two "country" "confirmed" columns
covid_data_frame_csv[ 5:10, c( "country", "confirmed") ]
```

A data.frame: 6 × 2

	country	confirmed
	<fct>	<int>
5	Angola	20981
6	Antigua and Barbuda	832
7	Argentina	9060495
8	Armenia	422963
9	Australia	10112229
10	Austria	5789991

TASK 5: Calculate worldwide COVID testing positive ratio

The goal of task 5 is to get the total confirmed and tested cases worldwide, and try to figure the overall positive ratio using `confirmed cases / tested cases`

In [107…
```
# Get the total confirmed cases worldwide
confirmed_cases <- covid_data_frame_csv[ , 4]
```

```
confirmed_cases
total_confirmed_cases <- sum(confirmed_cases)
total_confirmed_cases
```

49621 · 96838 · 58574 · 37958 · 20981 · 832 · 9060495 · 422963 · 10112229 · 5789991 · 792638 · 37483 · 696614 · 1151644 · 103014 · 982809 · 4691499 · 60694 · 7884 · 12702 · 910228 · 399887 · 232432 · 10081676 · 338 · 1295524 · 12123 · 884 · 77914 · 32681 · 4423053 · 4020 · 5123007 · 87655 · 6314769 · 561054 · 1267798 · 1112470 · 644160 · 4590529 · 3399947 · 15631 · 14821 · 626030 · 25961 · 480720 · 283947 · 161052 · 17113 · 613954 · 49253 · 278446 · 34237 · 68848 · 371135 · 29183646 · 25325 · 4469 · 732965 · 3733519 · 96708 · 5548487 · 10662 · 161 · 1230098 · 24878 · 8400 · 66129 · 33874 · 377859 · 1909948 · 203162 · 43585554 · 6812127 · 7232268 · 2448484 · 1700817 · 1792137 · 25651205 · 33285 · 151931 · 432773 · 739847 · 385144 · 107729 · 107410 · 624573 · 85253 · 45 · 144518 · 542649 · 32910 · 5396 · 501862 · 1170108 · 244182 · 19831 · 88086 · 1880734 · 174658 · 14449 · 36606 · 18103 · 494 · 3749860 · 516864 · 136053 · 98449 · 1272299 · 105866 · 440741 · 166229 · 984475 · 1692834 · 136 · 2136662 · 4740 · 155657 · 0 · 155689 · 103034 · 554778 · 114434 · 588728 · 574105 · 1029701 · 961 · 647950 · 4177786 · 4073980 · 5993861 · 1499976 · 473440 · 724250 · 18358459 · 98209 · 995 · 29550 · 9585 · 23427 · 753632 · 46509 · 2473599 · 65315 · 1861034 · 1322282 · 1637848 · 90029 · 10688 · 3821305 · 93128 · 23316 · 1074751 · 4276836 · 8622129 · 509 · 26162 · 39358 · 92997 · 703732 · 5435831 · 39979 · 3367461 · 1049537 · 22232377 · 90749469 · 895592 · 43975 · 159149 · 11403302 · 314850 · 257893

431434555

In [108…
```
# Get the total tested cases worldwide
tested_cases <- covid_data_frame_csv[ , 3]
tested_cases
total_tested_cases <- sum(tested_cases)
total_tested_cases
```

154767 · 428654 · 230553 · 300307 · 399228 · 15268 · 35716069 · 3099602 · 78548492 · 205817752 · 6838458 · 259366 · 10578766 · 7417714 · 770100 · 13217569 · 36548544 · 572900 · 595112 · 1736168 · 4358669 · 1872934 · 2026898 · 23561497 · 153804 · 10993239 · 158777 · 90019 · 1812706 · 942685 · 66343123 · 99027 · 48154268 · 1.6e+08 · 36875818 · 2575363 · 5481285 · 14301394 · 27820163 · 22544928 · 67682707 · 305941 · 209803 · 3574665 · 124838 · 1627189 · 3137519 · 1847861 · 403773 · 3637908 · 415110 · 2981185 · 774000 · 667953 · 9042453 · 272417258 · 958807 · 43217 · 4888787 · 65247345 · 1305749 · 101576831 · 164573 · 28684 · 6800560 · 494898 · 145231 · 648569 · 223475 · 1133782 · 11394556 · 1988652 · 866177937 · 76062770 · 52269202 · 19090652 · 12990476 · 41373364 · 269127054 · 429177 · 1184973 · 8487288 · 7407053 · 11575012 · 1322806 · 611357 · 7754247 · 695415 · 114030 · 3630095 · 4599186 · 431221 · 128246 · 2578215 · 9046584 · 4248188 · 119608 · 624784 · 23705425 · 2216560 · 322504 · 1211456 · 268093 · 289552 · 10503678 · 3213594 · 3354200 · 394388 · 14217563 · 688570 · 4047680 · 1062663 · 5804358 · 14526293 · 41962 · 7757935 · 79321 · 1544008 · 16914 · 881870 · 7096998 · 9811888 · 509959 · 9173593 · 3078533 · 7475016 · 47490 · 2609819 · 36073768 · 34402980 · 36064311 · 27515490 · 4061988 · 5405393 · 295542733 · 2885812 · 30231 · 212132 · 113504 · 192613 · 41849069 · 624502 · 12185475 · 16206203 · 7391882 · 2826117 · 11378282 · 6592010 · 164472 · 54128524 · 2384745 · 158804 · 9996795 · 23283909 · 30275725 · 3880 · 1579597 · 807269 · 512730 · 2893625 · 61236294 · 852444 · 15648456 · 198685717 · 522526476 · 929349291 · 6089116 · 2630000 · 3179074 · 45772571 · 3301860 · 2529087

5396881644

In [109…
```
# Get the positive ratio (confirmed / tested)
positive_ratio <- total_confirmed_cases/total_tested_cases
positive_ratio
```

0.0799414520197323

TASK 6: Get a country list which reported their testing data

The goal of task 6 is to get a catalog or sorted list of countries who have reported their COVID-19 testing data

In [110…
```
# Get the `country` column
country_column <- covid_data_frame_csv[ , 1]
country_column
```

Afghanistan · Albania · Algeria · Andorra · Angola · Antigua and Barbuda · Argentina · Armenia · Australia · Austria · Azerbaijan · Bahamas · Bahrain · Bangladesh · Barbados · Belarus · Belgium · Belize · Benin · Bhutan · Bolivia · Bosnia and Herzegovina · Botswana · Brazil · Brunei · Bulgaria · Burkina Faso · Burundi · Cambodia · Cameroon · Canada · Chad · Chile · China[c] · Colombia · Costa Rica · Croatia · Cuba · Cyprus[d] · Czechia · Denmark[e] · Djibouti · Dominica · Dominican Republic · DR Congo · Ecuador · Egypt · El Salvador · Equatorial Guinea · Estonia · Eswatini · Ethiopia · Faroe Islands · Fiji · Finland · France[f][g] · Gabon · Gambia · Georgia[h] · Germany · Ghana · Greece · Greenland · Grenada · Guatemala · Guinea · Guinea-Bissau · Guyana · Haiti · Honduras · Hungary · Iceland · India · Indonesia · Iran · Iraq · Ireland · Israel · Italy · Ivory Coast · Jamaica · Japan · Jordan · Kazakhstan · Kenya · Kosovo · Kuwait · Kyrgyzstan · Laos · Latvia · Lebanon · Lesotho · Liberia · Libya · Lithuania · Luxembourg[i] · Madagascar · Malawi · Malaysia · Maldives · Mali · Malta · Mauritania · Mauritius · Mexico · Moldova[j] · Mongolia · Montenegro · Morocco · Mozambique · Myanmar · Namibia · Nepal · Netherlands · New Caledonia · New Zealand · Niger · Nigeria · North Korea · North Macedonia · Northern Cyprus[k] · Norway · Oman · Pakistan · Palestine · Panama · Papua New Guinea · Paraguay · Peru · Philippines · Poland · Portugal · Qatar · Romania · Russia · Rwanda · Saint Kitts and Nevis · Saint Lucia · Saint Vincent · San Marino · Saudi Arabia · Senegal · Serbia · Singapore · Slovakia · Slovenia · South Africa · South Korea · South Sudan · Spain · Sri Lanka · Sudan · Sweden · Switzerland[l] · Taiwan[m] · Tanzania · Thailand · Togo · Trinidad and Tobago · Tunisia · Turkey · Uganda · Ukraine · United Arab Emirates · United Kingdom · United States · Uruguay · Uzbekistan · Venezuela · Vietnam · Zambia · Zimbabwe

▶ **Levels**:

In [111…
```
class(country_column)
```

'factor'

In [112…
```
# Conver the country column into character so that you can easily sort them
as.character ( country_column)
```

'Afghanistan' · 'Albania' · 'Algeria' · 'Andorra' · 'Angola' · 'Antigua and Barbuda' · 'Argentina' · 'Armenia' · 'Australia' · 'Austria' · 'Azerbaijan' · 'Bahamas' · 'Bahrain' · 'Bangladesh' · 'Barbados' · 'Belarus' · 'Belgium' · 'Belize' · 'Benin' · 'Bhutan' · 'Bolivia' · 'Bosnia and Herzegovina' · 'Botswana' · 'Brazil' · 'Brunei' · 'Bulgaria' · 'Burkina Faso' · 'Burundi' · 'Cambodia' · 'Cameroon' · 'Canada' · 'Chad' · 'Chile' · 'China[c]' · 'Colombia' · 'Costa Rica' · 'Croatia' · 'Cuba' · 'Cyprus[d]' · 'Czechia' · 'Denmark[e]' · 'Djibouti' · 'Dominica' · 'Dominican Republic' · 'DR Congo' · 'Ecuador' · 'Egypt' · 'El Salvador' · 'Equatorial Guinea' · 'Estonia' · 'Eswatini' · 'Ethiopia' · 'Faroe Islands' · 'Fiji' · 'Finland' · 'France[f][g]' · 'Gabon' · 'Gambia' · 'Georgia[h]' · 'Germany' · 'Ghana' · 'Greece' · 'Greenland' · 'Grenada' · 'Guatemala' · 'Guinea' · 'Guinea-Bissau' · 'Guyana' · 'Haiti' · 'Honduras' · 'Hungary' · 'Iceland' · 'India' · 'Indonesia' · 'Iran' · 'Iraq' · 'Ireland' · 'Israel' · 'Italy' · 'Ivory Coast' · 'Jamaica' · 'Japan' · 'Jordan' · 'Kazakhstan' · 'Kenya' · 'Kosovo' · 'Kuwait' · 'Kyrgyzstan' · 'Laos' · 'Latvia' · 'Lebanon' · 'Lesotho' · 'Liberia' · 'Libya' · 'Lithuania' · 'Luxembourg[i]' · 'Madagascar' · 'Malawi' · 'Malaysia' · 'Maldives' · 'Mali' · 'Malta' · 'Mauritania' · 'Mauritius' · 'Mexico' · 'Moldova[j]' · 'Mongolia' ·

'Montenegro' · 'Morocco' · 'Mozambique' · 'Myanmar' · 'Namibia' · 'Nepal' · 'Netherlands' · 'New Caledonia' · 'New Zealand' · 'Niger' · 'Nigeria' · 'North Korea' · 'North Macedonia' · 'Northern Cyprus[k]' · 'Norway' · 'Oman' · 'Pakistan' · 'Palestine' · 'Panama' · 'Papua New Guinea' · 'Paraguay' · 'Peru' · 'Philippines' · 'Poland' · 'Portugal' · 'Qatar' · 'Romania' · 'Russia' · 'Rwanda' · 'Saint Kitts and Nevis' · 'Saint Lucia' · 'Saint Vincent' · 'San Marino' · 'Saudi Arabia' · 'Senegal' · 'Serbia' · 'Singapore' · 'Slovakia' · 'Slovenia' · 'South Africa' · 'South Korea' · 'South Sudan' · 'Spain' · 'Sri Lanka' · 'Sudan' · 'Sweden' · 'Switzerland[l]' · 'Taiwan[m]' · 'Tanzania' · 'Thailand' · 'Togo' · 'Trinidad and Tobago' · 'Tunisia' · 'Turkey' · 'Uganda' · 'Ukraine' · 'United Arab Emirates' · 'United Kingdom' · 'United States' · 'Uruguay' · 'Uzbekistan' · 'Venezuela' · 'Vietnam' · 'Zambia' · 'Zimbabwe'

In [113…

```
# Sort the countries AtoZ
sort(country_column)
```

Afghanistan · Albania · Algeria · Andorra · Angola · Antigua and Barbuda · Argentina · Armenia · Australia · Austria · Azerbaijan · Bahamas · Bahrain · Bangladesh · Barbados · Belarus · Belgium · Belize · Benin · Bhutan · Bolivia · Bosnia and Herzegovina · Botswana · Brazil · Brunei · Bulgaria · Burkina Faso · Burundi · Cambodia · Cameroon · Canada · Chad · Chile · China[c] · Colombia · Costa Rica · Croatia · Cuba · Cyprus[d] · Czechia · Denmark[e] · Djibouti · Dominica · Dominican Republic · DR Congo · Ecuador · Egypt · El Salvador · Equatorial Guinea · Estonia · Eswatini · Ethiopia · Faroe Islands · Fiji · Finland · France[f][g] · Gabon · Gambia · Georgia[h] · Germany · Ghana · Greece · Greenland · Grenada · Guatemala · Guinea · Guinea-Bissau · Guyana · Haiti · Honduras · Hungary · Iceland · India · Indonesia · Iran · Iraq · Ireland · Israel · Italy · Ivory Coast · Jamaica · Japan · Jordan · Kazakhstan · Kenya · Kosovo · Kuwait · Kyrgyzstan · Laos · Latvia · Lebanon · Lesotho · Liberia · Libya · Lithuania · Luxembourg[i] · Madagascar · Malawi · Malaysia · Maldives · Mali · Malta · Mauritania · Mauritius · Mexico · Moldova[j] · Mongolia · Montenegro · Morocco · Mozambique · Myanmar · Namibia · Nepal · Netherlands · New Caledonia · New Zealand · Niger · Nigeria · North Korea · North Macedonia · Northern Cyprus[k] · Norway · Oman · Pakistan · Palestine · Panama · Papua New Guinea · Paraguay · Peru · Philippines · Poland · Portugal · Qatar · Romania · Russia · Rwanda · Saint Kitts and Nevis · Saint Lucia · Saint Vincent · San Marino · Saudi Arabia · Senegal · Serbia · Singapore · Slovakia · Slovenia · South Africa · South Korea · South Sudan · Spain · Sri Lanka · Sudan · Sweden · Switzerland[l] · Taiwan[m] · Tanzania · Thailand · Togo · Trinidad and Tobago · Tunisia · Turkey · Uganda · Ukraine · United Arab Emirates · United Kingdom · United States · Uruguay · Uzbekistan · Venezuela · Vietnam · Zambia · Zimbabwe

▶ **Levels**:

In [114…

```
# Sort the countries ZtoA
 Country_ZtoA <- sort(country_column, decreasing = TRUE)
 Country_ZtoA
```

Zimbabwe · Zambia · Vietnam · Venezuela · Uzbekistan · Uruguay · United States · United Kingdom · United Arab Emirates · Ukraine · Uganda · Turkey · Tunisia · Trinidad and Tobago · Togo · Thailand · Tanzania · Taiwan[m] · Switzerland[l] · Sweden · Sudan · Sri Lanka · Spain · South Sudan · South Korea · South Africa · Slovenia · Slovakia · Singapore · Serbia · Senegal · Saudi Arabia · San Marino · Saint Vincent · Saint Lucia · Saint Kitts and Nevis · Rwanda · Russia · Romania · Qatar · Portugal · Poland · Philippines · Peru · Paraguay · Papua New Guinea · Panama · Palestine · Pakistan · Oman · Norway · Northern Cyprus[k] · North Macedonia · North Korea · Nigeria · Niger · New Zealand · New Caledonia · Netherlands · Nepal · Namibia · Myanmar · Mozambique · Morocco · Montenegro · Mongolia · Moldova[j] · Mexico · Mauritius · Mauritania · Malta · Mali · Maldives · Malaysia · Malawi · Madagascar · Luxembourg[i] · Lithuania · Libya · Liberia · Lesotho · Lebanon · Latvia · Laos · Kyrgyzstan · Kuwait · Kosovo · Kenya · Kazakhstan · Jordan · Japan · Jamaica · Ivory Coast · Italy · Israel · Ireland · Iraq · Iran · Indonesia · India ·

Iceland · Hungary · Honduras · Haiti · Guyana · Guinea-Bissau · Guinea · Guatemala · Grenada · Greenland · Greece · Ghana · Germany · Georgia[h] · Gambia · Gabon · France[f][g] · Finland · Fiji · Faroe Islands · Ethiopia · Eswatini · Estonia · Equatorial Guinea · El Salvador · Egypt · Ecuador · DR Congo · Dominican Republic · Dominica · Djibouti · Denmark[e] · Czechia · Cyprus[d] · Cuba · Croatia · Costa Rica · Colombia · China[c] · Chile · Chad · Canada · Cameroon · Cambodia · Burundi · Burkina Faso · Bulgaria · Brunei · Brazil · Botswana · Bosnia and Herzegovina · Bolivia · Bhutan · Benin · Belize · Belgium · Belarus · Barbados · Bangladesh · Bahrain · Bahamas · Azerbaijan · Austria · Australia · Armenia · Argentina · Antigua and Barbuda · Angola · Andorra · Algeria · Albania · Afghanistan

▶ **Levels**:

In [115…

```
# Print the sorted ZtoA list
print( Country_ZtoA)
```

```
  [1] Zimbabwe              Zambia                 Vietnam
  [4] Venezuela             Uzbekistan             Uruguay
  [7] United States         United Kingdom         United Arab Emirates
 [10] Ukraine               Uganda                 Turkey
 [13] Tunisia               Trinidad and Tobago    Togo
 [16] Thailand              Tanzania               Taiwan[m]
 [19] Switzerland[l]        Sweden                 Sudan
 [22] Sri Lanka             Spain                  South Sudan
 [25] South Korea           South Africa           Slovenia
 [28] Slovakia              Singapore              Serbia
 [31] Senegal               Saudi Arabia           San Marino
 [34] Saint Vincent         Saint Lucia            Saint Kitts and Nevis
 [37] Rwanda                Russia                 Romania
 [40] Qatar                 Portugal               Poland
 [43] Philippines           Peru                   Paraguay
 [46] Papua New Guinea      Panama                 Palestine
 [49] Pakistan              Oman                   Norway
 [52] Northern Cyprus[k]    North Macedonia        North Korea
 [55] Nigeria               Niger                  New Zealand
 [58] New Caledonia         Netherlands            Nepal
 [61] Namibia               Myanmar                Mozambique
 [64] Morocco               Montenegro             Mongolia
 [67] Moldova[j]            Mexico                 Mauritius
 [70] Mauritania            Malta                  Mali
 [73] Maldives              Malaysia               Malawi
 [76] Madagascar            Luxembourg[i]          Lithuania
 [79] Libya                 Liberia                Lesotho
 [82] Lebanon               Latvia                 Laos
 [85] Kyrgyzstan            Kuwait                 Kosovo
 [88] Kenya                 Kazakhstan             Jordan
 [91] Japan                 Jamaica                Ivory Coast
 [94] Italy                 Israel                 Ireland
 [97] Iraq                  Iran                   Indonesia
[100] India                 Iceland                Hungary
[103] Honduras              Haiti                  Guyana
[106] Guinea-Bissau         Guinea                 Guatemala
[109] Grenada               Greenland              Greece
[112] Ghana                 Germany                Georgia[h]
[115] Gambia                Gabon                  France[f][g]
[118] Finland               Fiji                   Faroe Islands
[121] Ethiopia              Eswatini               Estonia
[124] Equatorial Guinea     El Salvador            Egypt
[127] Ecuador               DR Congo               Dominican Republic
[130] Dominica              Djibouti               Denmark[e]
[133] Czechia               Cyprus[d]              Cuba
[136] Croatia               Costa Rica             Colombia
[139] China[c]              Chile                  Chad
[142] Canada                Cameroon               Cambodia
```

```
[145] Burundi                Burkina Faso           Bulgaria
[148] Brunei                 Brazil                 Botswana
[151] Bosnia and Herzegovina Bolivia                Bhutan
[154] Benin                  Belize                 Belgium
[157] Belarus                Barbados               Bangladesh
[160] Bahrain                Bahamas                Azerbaijan
[163] Austria                Australia              Armenia
[166] Argentina              Antigua and Barbuda    Angola
[169] Andorra                Algeria                Albania
[172] Afghanistan
172 Levels: Afghanistan Albania Algeria Andorra Angola ... Zimbabwe
```

TASK 7: Identify countries names with a specific pattern

The goal of task 7 is using a regular expression to find any countires start with `United`

In [116…
```
# Use a regular expression `United.+` to find matches
matches <- regexpr("United.+", covid_data_frame_csv[ ,"country"])
countires_start_with_United<- regmatches(covid_data_frame_csv[ ,"country"], matches)
countires_start_with_United
```

'United Arab Emirates' · 'United Kingdom' · 'United States'

In [117…
```
# Print the matched country names
print(countires_start_with_United)
```

```
[1] "United Arab Emirates" "United Kingdom"       "United States"
```

TASK 8: Pick two countries you are interested, and then review their testing data

The goal of task 8 is to compare the COVID-19 test data between two countires, you will need to select two rows from the dataframe, and select `country` , `confirmed` , `confirmed-population-ratio` columns

In [118…
```
# Select a subset (should be only one row) of data frame based on a selected country nam
wiki_covid19_data_frame[65, c( "country", "confirmed", "confirmed.population.ratio") ]
```

A data.frame: 1 × 3

	country <fct>	confirmed <dbl>	confirmed.population.ratio <dbl>
65	Guatemala	1230098	7.1

In [120…
```
# Select a subset (should be only one row) of data frame based on a selected country nam
wiki_covid19_data_frame[ 73, c("country", "confirmed", "confirmed.population.ratio") ]
```

A data.frame: 1 × 3

	country <fct>	confirmed <dbl>	confirmed.population.ratio <dbl>
73	India	43585554	31.7

TASK 9: Compare which one of the selected countries has a larger ratio of confirmed cases to population

The goal of task 9 is to find out which country you have selected before has larger ratio of confirmed cases to population, which may indicate that country has higher COVID-19 infection risk

In [123…

```
if (43585554 > 1230098) {
print( "India has larger ratio of confirmed cases to population")
} else {
print( "Guatemala has larger ratio of confirmed cases to population")
}
```

```
[1] "India has larger ratio of confirmed cases to population"
```

TASK 10: Find countries with confirmed to population ratio rate less than a threshold

The goal of task 10 is to find out which countries have the confirmed to population ratio less than 1%, it may indicate the risk of those countries are relatively low

In [124…

```
# Get a subset of any countries with `confirmed.population.ratio` less than the threshol
threshold = "lessRisk"
if (threshold == "lessRisk"){
subset(wiki_covid19_data_frame, confirmed.population.ratio < .01)
} else {
subset(wiki_covid19_data_frame, confirmed.population.ratio > .01)
}
```

A data.frame: 5 × 7

	country	date	tested	confirmed	confirmed.tested.ratio	tested.population.ratio	confirmed.populatio
	<fct>	<date>	<dbl>	<dbl>	<dbl>	<dbl>	
28	Burundi	2021-01-05	90019	884	0.980	0.7600	0.
34	China[c]	2020-07-31	160000000	87655	0.055	11.1000	0.
89	Laos	2021-03-01	114030	45	0.039	1.6000	0.
119	North Korea	2020-11-25	16914	0	0.000	0.0660	0.
156	Tanzania	2020-11-18	3880	509	13.100	0.0065	0.