

MVLU COLLEGE

DATA ANALYSIS WITH SAS/ SPSS/ R

AIM: 11. Reshaping data using pivot_longer() and pivot_wider() (R).

The first screenshot shows the RStudio interface with the following code in the Source pane:

```
> install.packages("tidyr")
```

Warnings and messages indicate that Rtools is required for building R packages but is not currently installed. The package 'tidyr' is successfully unpacked and MD5 sums are checked. The downloaded binary packages are in:

```
c:\Users\itlab\AppData\Local\Temp\Rtmpg3qkBI\downloaded_packages
```

The code then attempts to load the 'tidyr' package:

```
> library(tidyr)
> library(dplyr)
```

Messages indicate that the following objects are masked from 'package:stats' (filter, lag) and 'package:base' (intersect, setdiff, setequal, union).

The code then attempts to read a CSV file:

```
> spotify <- read.csv("C:/Users/itlab/OneDrive/Documents/S105/DATASET")
```

An error occurs: "Error: '\u' used without hex digits in character string (<input>:1:25)".

The code is then corrected to use double quotes:

```
> spotify <- read.csv("C:/Users/itlab/OneDrive/Documents/S105/DATASET")
```

Another error occurs: "Error in file(file, "rt") : cannot open the connection".

The second screenshot shows the RStudio interface with the following code in the Source pane:

```
> head(spotify)
```

An error occurs: "Error: object 'spotify' not found".

The code is then corrected to use the full file path:

```
> spotify <- read.csv("C:/Users/itlab/OneDrive/Documents/S105/DATASET/spotify_data_clean.csv")
```

A warning message is displayed: "Warning message: In scan(file = file, what = what, sep = sep, quote = quote, dec = dec, : EOF within quoted string".

The code then displays the head of the 'spotify' dataset:

```
> head(spotify)
```

The output shows the following columns: track_id, track_name, track_number, track_popularity, explicit, artist_name, artist_popularity, artist_followers, artist_genres, album_id, album_name, album_release_date, album_total_tracks, album_type, track_duration_min.

The output shows the following rows:

track_id	track_name	track_number	track_popularity	explicit	artist_name	artist_popularity	artist_followers	artist_genres	album_id
1 3e355yeko1m1f5r6m2l	Trippy Mane (ft. Project Pat)	4	0	TRUE	Diplo	77	2812821	moombahton	5QRFNGNBEMGPBKf2XTz52
2 1oqw6G2ziwUq1pp2708	OMG!	1	0	TRUE	Yelawolf	64	2363438	country hip hop, southern hip hop	4SUmnmv0XTjRCLdjcZgG2
3 7mdkjzo1YlF1rx9et8pGmU	Hard 2 Find	1	4	TRUE	Riff Raff	48	193302	N/A	3E3Z6AL8gUyWALV89L7g0p
4 67rW0Z17o83qEp05YwWesW	Still Get Like That (ft. Project Pat & Starrah)	8	30	TRUE	Diplo	77	2813710	moombahton	5QRFNGNBEMGPBKf2XTz52
5 15xpttFR8rjspp0iUuzjf	ride me like a harley	2	0	TRUE	RumelIs	48	8682	dark r&b	06FDipSHYMZAZoyuYtc7kd
6 4ccpc2Y5eq0v9WK1050	BLEED	1	2	FALSE	Minzie	46	7218	dark r&b	2NqV9p3ZQw0EdL89enix8

The output also shows the following columns: album_name, album_release_date, album_total_tracks, album_type, track_duration_min.

The output shows the following rows:

album_name	album_release_date	album_total_tracks	album_type	track_duration_min
d00mscrv11, Vol. 1	2025-10-31	9	album	1.55
OMG!	2025-10-31	1	single	3.07
Hard 2 Find	2025-10-31	1	single	2.55
d00mscrv11, Vol. 1	2025-10-31	9	album	1.69
come closer / ride me like a harley	2025-10-30	2	single	2.39
BLEED	2025-10-30	3	single	2.76

The code then displays the head of the 'spotify' dataset using pivot_wider():

```
> df <- spotify %>%
+ select(track_name, track_popularity, artist_popularity)
> df <- spotify %>%
+ select(track_name, track_popularity, artist_popularity)
>
```

MVLU COLLEGE

DATA ANALYSIS WITH SAS/ SPSS/ R

The image displays two screenshots of the RStudio interface, demonstrating data manipulation using the dplyr package. Both screenshots show the same R script being executed, with the output visible in the console and the Environment pane.

Script Content (Visible in both screenshots):

```
> head(df)
  track_name track_popularity artist_popularity
1 Trippy Mane (ft. Project Pat) 0 77
2 OMG! 0 64
3 Hard 2 Find 4 48
4 Still Get Like That (ft. Project Pat & Starrah) 30 77
5 ride me like a harley 0 48
6 BLEED 2 46

> df_long <- df %>%
+   pivot_longer(
+     cols = c(track_popularity, artist_popularity),
+     names_to = "subject",
+     values_to = "score",
+   )
> df_long
# A tibble: 9,146 x 3
  track_name subject score
  <chr> <chr> <chr>
1 Trippy Mane (ft. Project Pat) track_popularity 0
2 Trippy Mane (ft. Project Pat) artist_popularity 77
3 OMG! track_popularity 0
4 OMG! artist_popularity 64
5 Hard 2 Find track_popularity 4
6 Hard 2 Find artist_popularity 48
7 Still Get Like That (ft. Project Pat & Starrah) track_popularity 30
8 Still Get Like That (ft. Project Pat & Starrah) artist_popularity 77
9 ride me like a harley track_popularity 0
10 ride me like a harley artist_popularity 48
# 9,136 more rows
# i use 'print(n = ...)' to see more rows

> df_wide <- df_long %>%
+   pivot_wider(
+     names_from = subject,
+     values_from = score,
+   )

Warning message:
values from 'score' are not uniquely identified; output will contain list-cols.
• use 'values_fn = list' to suppress this warning.
• use 'values_fn = {summary_fun}' to summarise duplicates.
• use the following dplyr code to identify duplicates.
{data} |>
dplyr::summarise(n = dplyr::n(), .by = c(track_name, subject)) |>
dplyr::filter(n > 1)
```

Environment Pane (Visible in both screenshots):

Object	Class	Size
df	data.frame	4573 obs. of 3 variables
df_long	data.frame	9146 obs. of 3 variables
df_wide	data.frame	4047 obs. of 3 variables
spotify	data.frame	4573 obs. of 15 variables

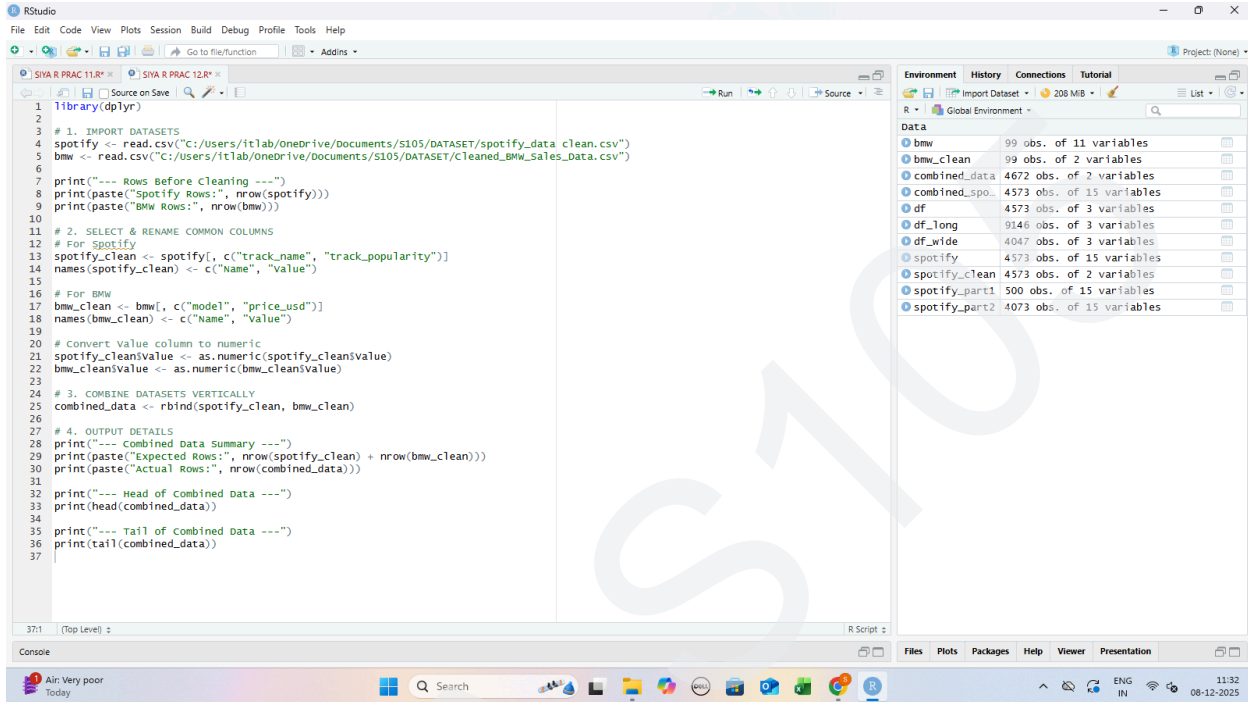
Output Differences:

- Top Screenshot:** The output of `df_wide` is a tibble with 4,047 rows and 3 columns: `track_name`, `track_popularity`, and `artist_popularity`. The first 10 rows are shown.
- Bottom Screenshot:** The output of `df_wide` is a tibble with 4,047 rows and 3 columns: `track_name`, `track_popularity`, and `artist_popularity`. The first 10 rows are shown, with some values in the `track_popularity` and `artist_popularity` columns being `list` objects (e.g., `<chr [1]>`).

MVLU COLLEGE

DATA ANALYSIS WITH SAS/ SPSS/ R

**AIM: 12. Combining datasets vertically (concatenation) using rbind()
(R).Write code toCombining datasets vertically (concatenation) using
rbind() in R studio.**



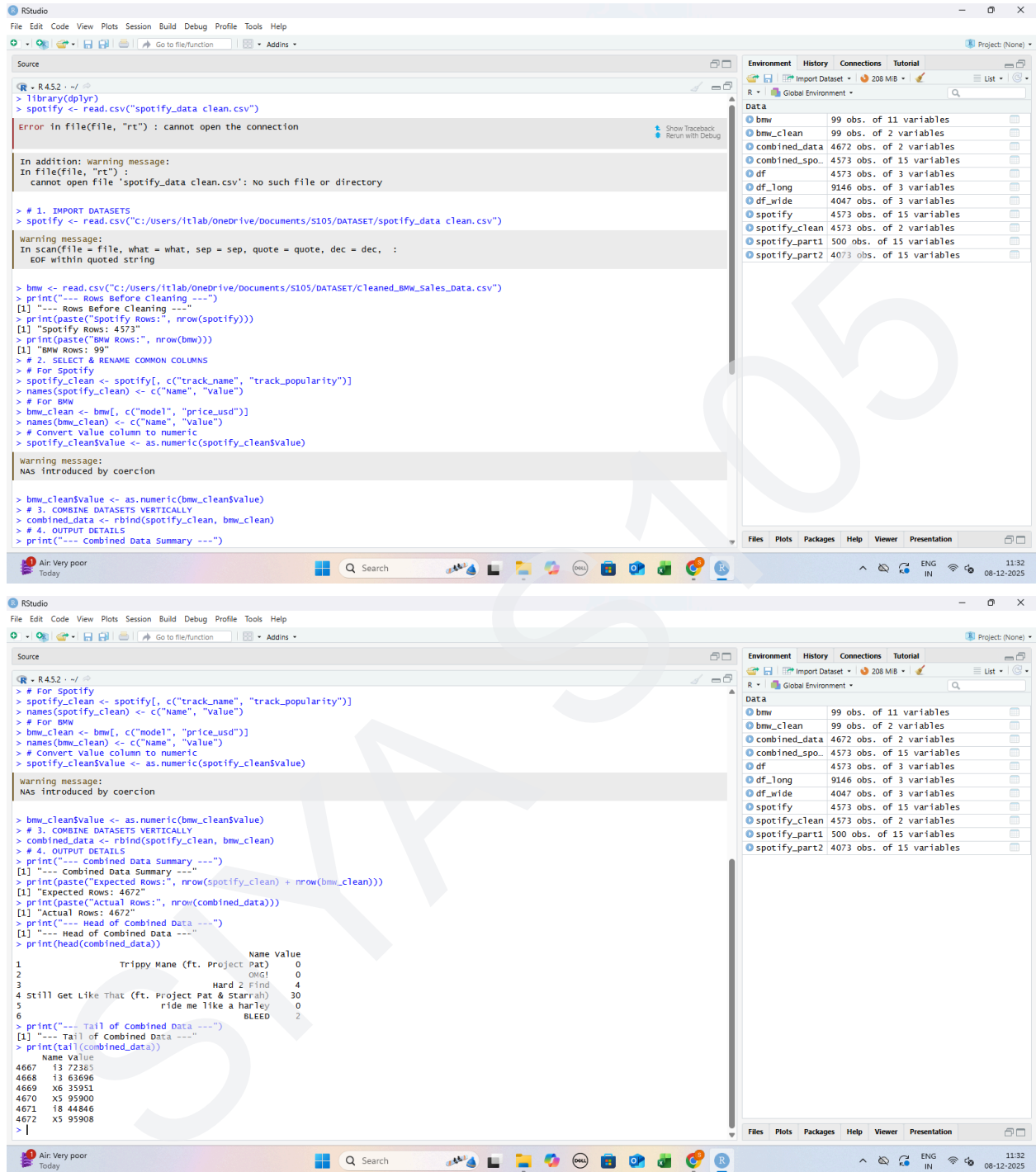
```
1 library(dplyr)
2
3 # 1. IMPORT DATASETS
4 spotify <- read.csv("c:/users/itlab/oneDrive/Documents/S105/DATASET/spotify_data_clean.csv")
5 bmw <- read.csv("c:/users/itlab/oneDrive/Documents/S105/DATASET/Cleaned_Bmw_Sales_Data.csv")
6
7 print("--- Rows Before Cleaning ---")
8 print(paste("Spotify Rows:", nrow(spotify)))
9 print(paste("BMW Rows:", nrow(bmw)))
10
11 # 2. SELECT & RENAME COMMON COLUMNS
12 # For Spotify
13 spotify_clean <- spotify[, c("track_name", "track_popularity")]
14 names(spotify_clean) <- c("Name", "Value")
15
16 # For BMW
17 bmw_clean <- bmw[, c("model", "price_usd")]
18 names(bmw_clean) <- c("Name", "Value")
19
20 # Convert value column to numeric
21 spotify_clean$value <- as.numeric(spotify_clean$value)
22 bmw_clean$value <- as.numeric(bmw_clean$value)
23
24 # 3. COMBINE DATASETS VERTICALLY
25 combined_data <- rbind(spotify_clean, bmw_clean)
26
27 # 4. OUTPUT DETAILS
28 print("--- Combined Data Summary ---")
29 print(paste("Expected Rows:", nrow(spotify_clean) + nrow(bmw_clean)))
30 print(paste("Actual Rows:", nrow(combined_data)))
31
32 print("--- Head of Combined Data ---")
33 print(head(combined_data))
34
35 print("--- Tail of Combined Data ---")
36 print(tail(combined_data))
37
```

The screenshot shows the RStudio interface with the following data in the Environment pane:

Object	Obs.	Vars.
bmw	99	11
bmw_clean	99	2
combined_data	4672	2
combined_spo.	4573	15
df	4573	3
df_long	9146	3
df_wide	4047	3
spotify	4573	15
spotify_clean	4573	2
spotify_part1	500	15
spotify_part2	4073	15

MVLU COLLEGE

DATA ANALYSIS WITH SAS/ SPSS/ R



The image displays two screenshots of the RStudio interface, showing the process of reading and cleaning data files.

Top Screenshot: The R console shows an error message: "Error in file(file, 'rt') : cannot open the connection". The warning message indicates: "cannot open file 'spotify_data clean.csv': no such file or directory". The code being executed is:

```
> library(dplyr)
> spotify <- read.csv("spotify_data clean.csv")

# 1. IMPORT DATASETS
> spotify <- read.csv("c:/users/itlab/OneDrive/Documents/S105/DATASET/spotify_data clean.csv")

# 2. SELECT & RENAME COMMON COLUMNS
> # For Spotify
> spotify_clean <- spotify[, c("track_name", "track_popularity")]
> names(spotify_clean) <- c("Name", "value")
> # For BMW
> bmw_clean <- bmw[, c("model", "price_usd")]
> names(bmw_clean) <- c("Name", "value")
> # Convert value column to numeric
> spotify_clean$value <- as.numeric(spotify_clean$value)

# 3. COMBINE DATASETS VERTICALLY
> combined_data <- rbind(spotify_clean, bmw_clean)

# 4. OUTPUT DETAILS
> print("---- Combined Data Summary ----")
```

The Environment pane on the right shows the following objects:

Object	Size
bmw	99 obs. of 11 variables
bmw_clean	99 obs. of 2 variables
combined_data	4672 obs. of 2 variables
combined_spo	4573 obs. of 15 variables
df	4573 obs. of 3 variables
df_long	9146 obs. of 3 variables
df_wide	4047 obs. of 3 variables
spotify	4573 obs. of 15 variables
spotify_clean	4573 obs. of 2 variables
spotify_part1	500 obs. of 15 variables
spotify_part2	4073 obs. of 15 variables

Bottom Screenshot: The R console shows the successful execution of the code. The warning message indicates: "NAS introduced by coercion". The code being executed is:

```
> bmw_clean$value <- as.numeric(bmw_clean$value)

# 3. COMBINE DATASETS VERTICALLY
> combined_data <- rbind(spotify_clean, bmw_clean)

# 4. OUTPUT DETAILS
> print("---- Combined Data Summary ----")
[1] "---- Combined Data Summary ----"
> print(paste("Expected Rows:", nrow(spotify_clean) + nrow(bmw_clean)))
[1] "Expected Rows: 4672"
> print(paste("Actual Rows:", nrow(combined_data)))
[1] "Actual Rows: 4672"
> print("---- Head of Combined Data ----")
[1] "---- Head of Combined Data ----"
> print(head(combined_data))
  Name Value
1 Trippy Mane (ft. Project Pat) 0
2 Omg! 0
3 Hard 2 Prind 4
4 still Get Like That (ft. Project Pat & Starrah) 30
5 ride me like a harley 0
6 BLEED 2

> print("---- Tail of Combined data ----")
[1] "---- Tail of Combined data ----"
> print(tail(combined_data))
  Name Value
4667 13 72365
4668 13 63696
4669 x6 35951
4670 x5 95900
4671 18 44846
4672 x5 95908
```

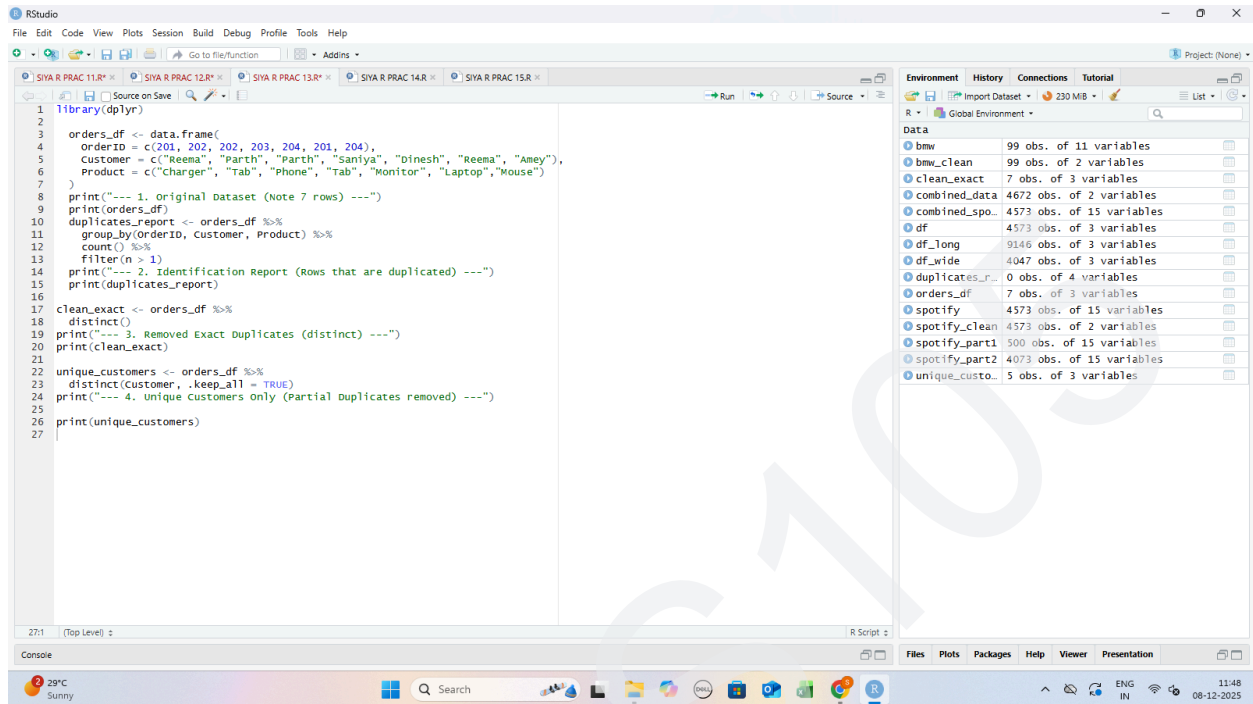
The Environment pane on the right shows the following objects:

Object	Size
bmw	99 obs. of 11 variables
bmw_clean	99 obs. of 2 variables
combined_data	4672 obs. of 2 variables
combined_spo	4573 obs. of 15 variables
df	4573 obs. of 3 variables
df_long	9146 obs. of 3 variables
df_wide	4047 obs. of 3 variables
spotify	4573 obs. of 15 variables
spotify_clean	4573 obs. of 2 variables
spotify_part1	500 obs. of 15 variables
spotify_part2	4073 obs. of 15 variables

MVLU COLLEGE

DATA ANALYSIS WITH SAS/ SPSS/ R

AIM:13 Identifying and handling duplicates using distinct() (R).



The screenshot displays the RStudio interface with a script editor on the left and an environment pane on the right. The script editor contains R code that demonstrates how to identify and handle duplicates in a dataset using the `distinct()` function from the `dplyr` package. The code includes comments and print statements to guide the user through the process.

```
1 library(dplyr)
2
3 orders_df <- data.frame(
4   orderID = c(201, 202, 202, 203, 204, 201, 204),
5   Customer = c("Reema", "Parth", "Parth", "Saniya", "Dinesh", "Reema", "Amey"),
6   Product = c("charger", "Tab", "Phone", "Tab", "Monitor", "Laptop", "Mouse")
7 )
8 print("---- 1. Original Dataset (Note 7 rows) ----")
9 print(orders_df)
10 duplicates_report <- orders_df %>%
11   group_by(orderID, Customer, Product) %>%
12   count() %>%
13   filter(n > 1)
14 print("---- 2. Identification Report (Rows that are duplicated) ----")
15 print(duplicates_report)
16
17 clean_exact <- orders_df %>%
18   distinct()
19 print("---- 3. Removed Exact Duplicates (distinct) ----")
20 print(clean_exact)
21
22 unique_customers <- orders_df %>%
23   distinct(Customer, .keep_all = TRUE)
24 print("---- 4. Unique Customers Only (Partial Duplicates removed) ----")
25
26 print(unique_customers)
27
```

The environment pane on the right shows the following objects:

Object	Variables
bmw	99 obs. of 11 variables
bmw_clean	99 obs. of 2 variables
clean_exact	7 obs. of 3 variables
combined_data	4672 obs. of 2 variables
combined_spo...	4573 obs. of 15 variables
df	4573 obs. of 3 variables
df_long	9146 obs. of 3 variables
df_wide	4047 obs. of 3 variables
duplicates_r...	0 obs. of 4 variables
orders_df	7 obs. of 3 variables
spotify	4573 obs. of 15 variables
spotify_clean	4573 obs. of 2 variables
spotify_part1	500 obs. of 15 variables
spotify_part2	4073 obs. of 15 variables
unique_custo...	5 obs. of 3 variables

MVLU COLLEGE

DATA ANALYSIS WITH SAS/ SPSS/ R

```
RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function Addins
Source
R - R452 - ~/
> library(dplyr)
> orders_df <- data.frame(
+   orderID = c(201, 202, 202, 203, 204, 201, 204),
+   customer = c("Reema", "Parth", "Parth", "Saniya", "Dinesh", "Reema", "Amei"),
+   Product = c("charger", "Tab", "Phone", "Tab", "Monitor", "Laptop", "Mouse")
+ )
> print("--- 1. original Dataset (Note 7 rows) ---")
[1] "--- 1. original Dataset (Note 7 rows) ---"
> print(orders_df)
# A tibble: 7 x 3
  orderID customer Product
  <dbl>   <chr>    <chr>
1     201   Reema  charger
2     202   Parth    Tab
3     202   Parth    Phone
4     203   Saniya   Tab
5     204   Dinesh   Monitor
6     201   Reema    Laptop
7     204   Amei     Mouse
> duplicates_report <- orders_df %>%
+   group_by(orderID, customer, Product) %>%
+   count() %>% # counts occurrences
+   filter(n > 1) # keeps only rows that appear more than once
> duplicates_report <- orders_df %>%
+   group_by(orderID, customer, Product) %>%
+   count() %>%
+   filter(n > 1)
> print("--- 2. Identification Report (Rows that are duplicated) ---")
[1] "--- 2. Identification Report (Rows that are duplicated) ---"
> print(duplicates_report)
# A tibble: 0 x 4
# Groups:   orderID, customer, Product [0]
# 4 variables: orderID <dbl>, customer <chr>, Product <chr>, n <int>
> clean_exact <- orders_df %>%
+   distinct()
> print("--- 3. Removed Exact Duplicates (distinct) ---")
[1] "--- 3. Removed Exact Duplicates (distinct) ---"
> print(clean_exact)
# A tibble: 6 x 3
  orderID customer Product
  <dbl>   <chr>    <chr>
1     201   Reema  charger
2     202   Parth    Tab
3     202   Parth    Phone
4     203   Saniya   Tab
5     204   Dinesh   Monitor
6     201   Reema    Laptop
7     204   Amei     Mouse
```

Environment History Connections Tutorial
R - Global Environment
Data
bmw 99 obs. of 11 variables
bmw_clean 99 obs. of 2 variables
clean_exact 7 obs. of 3 variables
combined_data 4672 obs. of 2 variables
combined_spo... 4573 obs. of 15 variables
df 4573 obs. of 3 variables
df_long 9146 obs. of 3 variables
df_wide 4047 obs. of 3 variables
duplicates_r... 0 obs. of 4 variables
orders_df 7 obs. of 3 variables
spotify 4573 obs. of 15 variables
spotify_clean 4573 obs. of 2 variables
spotify_part1 500 obs. of 15 variables
spotify_part2 4073 obs. of 15 variables
unique_customo... 5 obs. of 3 variables

Files Plots Packages Help Viewer Presentation
23°C Sunny 11:48 08-12-2025

```
RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function Addins
Source
R - R452 - ~/
> library(dplyr)
> orders_df <- data.frame(
+   orderID = c(201, 202, 202, 203, 204, 201, 204),
+   customer = c("Reema", "Parth", "Parth", "Saniya", "Dinesh", "Reema", "Amei"),
+   Product = c("charger", "Tab", "Phone", "Tab", "Monitor", "Laptop", "Mouse")
+ )
> print("--- 1. original Dataset (Note 7 rows) ---")
[1] "--- 1. original Dataset (Note 7 rows) ---"
> print(orders_df)
# A tibble: 7 x 3
  orderID customer Product
  <dbl>   <chr>    <chr>
1     201   Reema  charger
2     202   Parth    Tab
3     202   Parth    Phone
4     203   Saniya   Tab
5     204   Dinesh   Monitor
6     201   Reema    Laptop
7     204   Amei     Mouse
> duplicates_report <- orders_df %>%
+   group_by(orderID, customer, Product) %>%
+   count() %>% # counts occurrences
+   filter(n > 1) # keeps only rows that appear more than once
> duplicates_report <- orders_df %>%
+   group_by(orderID, customer, Product) %>%
+   count() %>%
+   filter(n > 1)
> print("--- 2. Identification Report (Rows that are duplicated) ---")
[1] "--- 2. Identification Report (Rows that are duplicated) ---"
> print(duplicates_report)
# A tibble: 0 x 4
# Groups:   orderID, customer, Product [0]
# 4 variables: orderID <dbl>, customer <chr>, Product <chr>, n <int>
> clean_exact <- orders_df %>%
+   distinct()
> print("--- 3. Removed Exact Duplicates (distinct) ---")
[1] "--- 3. Removed Exact Duplicates (distinct) ---"
> print(clean_exact)
# A tibble: 6 x 3
  orderID customer Product
  <dbl>   <chr>    <chr>
1     201   Reema  charger
2     202   Parth    Tab
3     202   Parth    Phone
4     203   Saniya   Tab
5     204   Dinesh   Monitor
6     201   Reema    Laptop
7     204   Amei     Mouse
> unique_customers <- orders_df %>%
+   distinct(customer, .keep_all = TRUE)
> print("--- 4. Unique Customers only (Partial Duplicates removed) ---")
[1] "--- 4. Unique Customers only (Partial Duplicates removed) ---"
> print(unique_customers)
# A tibble: 6 x 3
  orderID customer Product
  <dbl>   <chr>    <chr>
1     201   Reema  charger
2     202   Parth    Tab
3     203   Saniya   Tab
4     204   Dinesh   Monitor
5     201   Reema    Laptop
6     204   Amei     Mouse
```

Environment History Connections Tutorial
R - Global Environment
Data
bmw 99 obs. of 11 variables
bmw_clean 99 obs. of 2 variables
clean_exact 7 obs. of 3 variables
combined_data 4672 obs. of 2 variables
combined_spo... 4573 obs. of 15 variables
df 4573 obs. of 3 variables
df_long 9146 obs. of 3 variables
df_wide 4047 obs. of 3 variables
duplicates_r... 0 obs. of 4 variables
orders_df 7 obs. of 3 variables
spotify 4573 obs. of 15 variables
spotify_clean 4573 obs. of 2 variables
spotify_part1 500 obs. of 15 variables
spotify_part2 4073 obs. of 15 variables
unique_customo... 5 obs. of 3 variables

Files Plots Packages Help Viewer Presentation
23°C Sunny 11:48 08-12-2025

MVLU COLLEGE

DATA ANALYSIS WITH SAS/ SPSS/ R

AIM:14 Extracting date components using lubridate:: functions (R).

The first screenshot shows the RStudio interface with the following code in the script editor:

```
1 install.packages("lubridate")
2 library(lubridate)
3 library(dplyr)
4
5 movies_df <- data.frame(
6   Movie_ID = 1:5,
7   Release_Date = c("2020-09-04", "2021-12-17", "2022-05-06", "2023-07-21", "2024-11-08")
8 )
9
10 print("--- Original Dataset ---")
11 print(movies_df)
12
13 processed_movies <- movies_df %>%
14   mutate(
15     Actual_Date = ymd(Release_Date),
16     Year = year(Actual_Date),
17     Month_No = month(Actual_Date),
18     Month_Name = month(Actual_Date, label = TRUE, abbr = FALSE),
19     Day = day(Actual_Date),
20     Weekday_Name = wday(Actual_Date, label = TRUE, abbr = FALSE),
21     Quarter = quarter(Actual_Date),
22     Day_of_Year = yday(Actual_Date)
23   )
24
25 print("--- Movies with Extracted Date Components ---")
26 print(processed_movies)
27
28 current_time <- now()
29
30 print("--- System Date-Time Extraction ---")
31 print(paste("Current Year:", year(current_time)))
32 print(paste("Current Month:", month(current_time)))
33 print(paste("Current Day:", day(current_time)))
34 print(paste("Current Hour:", hour(current_time)))
35 print(paste("Current Minute:", minute(current_time)))
36
```

The Environment pane on the right shows the following data objects:

Object	Size
bmw	99 obs. of 11 variables
bmw_clean	99 obs. of 2 variables
clean_exact	7 obs. of 3 variables
combined_data	4672 obs. of 2 variables
combined_spo	4573 obs. of 15 variables
df	4573 obs. of 3 variables
df_long	9146 obs. of 3 variables
df_wide	4047 obs. of 3 variables
duplicates_r	0 obs. of 4 variables
movies_df	5 obs. of 2 variables
orders_df	7 obs. of 3 variables
processed_mo	5 obs. of 10 variables
spotify	4573 obs. of 15 variables
spotify_clean	4573 obs. of 2 variables
spotify_part1	500 obs. of 15 variables
spotify_part2	4073 obs. of 15 variables
unique_custo	5 obs. of 3 variables

The second screenshot shows the RStudio console output for the same code:

```
> install.packages("lubridate")
Warning message:
Rtools is required to build R packages but is not currently installed. Please download and install the appropriate version of Rtools before proceeding:
https://cran.rstudio.com/bin/windows/rtools/
Installing package into 'C:/Users/itlab/AppData/Local/R/win-library/4.5'
(as 'lib' is unspecified)

also installing the dependency 'timechange'

trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.5/timechange_0.3.0.zip'
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.5/lubridate_1.9.4.zip'
package 'timechange' successfully unpacked and MD5 sums checked
package 'lubridate' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
C:/Users/itlab/AppData/Local/Temp/Rtmpg3qkBI/downloaded_packages
> library(lubridate)

Attaching package: 'lubridate'

The following objects are masked from 'package:base':
    date, intersect, setdiff, union

> library(dplyr)
> movies_df <- data.frame(
+   Movie_ID = 1:5,
+   Release_Date = c("2020-09-04", "2021-12-17", "2022-05-06", "2023-07-21", "2024-11-08")
+ )
> movies_df
  Movie_ID Release_Date
1         1 2020-09-04
2         2 2021-12-17
3         3 2022-05-06
4         4 2023-07-21
5         5 2024-11-08
```


MVLU COLLEGE

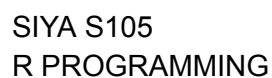
DATA ANALYSIS WITH SAS/ SPSS/ R

The image displays two screenshots of the RStudio interface, showing R code execution and the environment pane.

Top Screenshot: The Source pane shows R code for processing a dataset. The code includes comments and functions to extract date components (Year, Month, Day, Weekday, Quarter, Day of Year) from a date variable. The Environment pane on the right lists variables in the Global Environment, including 'Data' (99 obs. of 11 variables), 'bmw_clean' (99 obs. of 2 variables), 'clean_exact' (7 obs. of 3 variables), 'combined_data' (4672 obs. of 2 variables), 'combined_spo' (4573 obs. of 15 variables), 'df' (4573 obs. of 3 variables), 'df_long' (9146 obs. of 3 variables), 'df_wide' (4047 obs. of 3 variables), 'duplicates_r' (0 obs. of 4 variables), 'movies_df' (5 obs. of 2 variables), 'orders_df' (7 obs. of 3 variables), 'processed_mo' (5 obs. of 10 variables), 'spotify' (4573 obs. of 15 variables), 'spotify_clean' (4573 obs. of 2 variables), 'spotify_part1' (500 obs. of 15 variables), 'spotify_part2' (4073 obs. of 15 variables), and 'unique_custo' (5 obs. of 3 variables). The current time is 2025-12-08 11:53:58 IST.

Bottom Screenshot: The Source pane shows the same R code as the top screenshot, but with additional code for system date-time extraction. The Environment pane on the right lists the same variables as the top screenshot, but with 'processed_mo' (5 obs. of 10 variables) and 'spotify' (4573 obs. of 15 variables) added. The current time is 2025-12-08 11:53:58 IST.

AIM:15 Generating basic summaries using str() or summary() (R).



MVLU COLLEGE

DATA ANALYSIS WITH SAS/ SPSS/ R

The image displays two screenshots of the RStudio interface, showing R code being executed to analyze a dataset named 'marks_df'.

Top Screenshot: The R console shows the following code and output:

```
> marks_df <- read.csv("C:/Users/itlab/OneDrive/Documents/S105/DATASET/College_Marks_Dataset.csv")
> print("--- Dataset Successfully Loaded ---")
[1] "--- Dataset Successfully Loaded ---"
> print(head(marks_df))
  Student_ID   Name      Class SSC_Marks HSC_Marks College_Marks Attendance_Percentage Grade
1    S1000 Student_0 Commerce      635      452        692          84.71 C
2    S1001 Student_1 Commerce      494      535        551          81.99 D
3    S1002 Student_2 Science      542      460        634          92.06 B
4    S1003 Student_3 Science      441      483        686          79.27 D
5    S1004 Student_4 Arts        427      544        569          91.99 A+
6    S1005 Student_5 Science      520      539        519          88.11 B

> print("--- OUTPUT OF str() ---")
[1] "--- OUTPUT OF str() ---"
> str(marks_df)
'data.frame':   1000 obs. of  8 variables:
 $ Student_ID   : chr  "S1000" "S1001" "S1002" "S1003" ...
 $ Name         : chr  "Student_0" "Student_1" "Student_2" "Student_3" ...
 $ Class        : chr  "Commerce" "Commerce" "Science" "Science" ...
 $ SSC_Marks    : int   635 494 542 441 427 520 504 509 499 411 ...
 $ HSC_Marks    : int   452 535 460 483 544 539 573 481 474 450 ...
 $ College_Marks: int   692 551 634 686 569 519 646 504 668 636 ...
 $ Attendance_Percentage: num  84.7 82 92.1 79.3 92 ...
 $ Grade       : chr   "C" "D" "B" "D" ...

> print("--- OUTPUT OF summary() ---")
[1] "--- OUTPUT OF summary() ---"
> summary(marks_df)
  Student_ID   Name      Class      SSC_Marks    HSC_Marks    College_Marks    Attendance_Percentage
Length:1000   Length:1000   Length:1000   Min. :400.0   Min. :450.0   Min. :500.0   Min. :60.03
Class :character Class :character Class :character 1st Qu.:437.0 1st Qu.:484.8 1st Qu.:552.0 1st Qu.:69.57
Mode :character Mode :character Mode :character Median :476.0 Median :523.5 Median :602.0 Median :80.57
Mean :476.2 Mean :524.0 Mean :603.1 Mean :79.95
3rd Qu.:516.0 3rd Qu.:564.0 3rd Qu.:655.2 3rd Qu.:89.95
Max. :550.0 Max. :600.0 Max. :700.0 Max. :99.95

Grade
Length:1000
Class :character
Mode :character

> marks_df[sapply(marks_df, is.character)] <-
+  lapply(marks_df[sapply(marks_df, is.character)], as.factor)
> print("--- OUTPUT OF summary() AFTER FACTOR CONVERSION ---")
[1] "--- OUTPUT OF summary() AFTER FACTOR CONVERSION ---"

> summary(marks_df)
  Student_ID   Name      Class      SSC_Marks    HSC_Marks    College_Marks    Attendance_Percentage Grade
Length:1000   Length:1000   Length:1000   Min. :400.0   Min. :450.0   Min. :500.0   Min. :60.03
Class :character Class :character Class :character 1st Qu.:437.0 1st Qu.:484.8 1st Qu.:552.0 1st Qu.:69.57
Mode :character Mode :character Mode :character Median :476.0 Median :523.5 Median :602.0 Median :80.57
Mean :476.2 Mean :524.0 Mean :603.1 Mean :79.95
3rd Qu.:516.0 3rd Qu.:564.0 3rd Qu.:655.2 3rd Qu.:89.95
Max. :550.0 Max. :600.0 Max. :700.0 Max. :99.95

Grade
Length:1000
Class :character
Mode :character
```

Bottom Screenshot: The R console shows the following code and output:

```
> marks_df[sapply(marks_df, is.character)] <-
+  lapply(marks_df[sapply(marks_df, is.character)], as.factor)
> print("--- OUTPUT OF summary() AFTER FACTOR CONVERSION ---")
[1] "--- OUTPUT OF summary() AFTER FACTOR CONVERSION ---"
> summary(marks_df)
  Student_ID   Name      Class      SSC_Marks    HSC_Marks    College_Marks    Attendance_Percentage Grade
Length:1000   Length:1000   Length:1000   Min. :400.0   Min. :450.0   Min. :500.0   Min. :60.03
Class :character Class :character Class :character 1st Qu.:437.0 1st Qu.:484.8 1st Qu.:552.0 1st Qu.:69.57
Mode :character Mode :character Mode :character Median :476.0 Median :523.5 Median :602.0 Median :80.57
Mean :476.2 Mean :524.0 Mean :603.1 Mean :79.95
3rd Qu.:516.0 3rd Qu.:564.0 3rd Qu.:655.2 3rd Qu.:89.95
Max. :550.0 Max. :600.0 Max. :700.0 Max. :99.95

Grade
Length:1000
Class :character
Mode :character

> marks_df[sapply(marks_df, is.character)] <-
+  lapply(marks_df[sapply(marks_df, is.character)], as.factor)
> print("--- OUTPUT OF summary() AFTER FACTOR CONVERSION ---")
[1] "--- OUTPUT OF summary() AFTER FACTOR CONVERSION ---"
> summary(marks_df)
  Student_ID   Name      Class      SSC_Marks    HSC_Marks    College_Marks    Attendance_Percentage Grade
Length:1000   Length:1000   Length:1000   Min. :400.0   Min. :450.0   Min. :500.0   Min. :60.03
Class :character Class :character Class :character 1st Qu.:437.0 1st Qu.:484.8 1st Qu.:552.0 1st Qu.:69.57
Mode :character Mode :character Mode :character Median :476.0 Median :523.5 Median :602.0 Median :80.57
Mean :476.2 Mean :524.0 Mean :603.1 Mean :79.95
3rd Qu.:516.0 3rd Qu.:564.0 3rd Qu.:655.2 3rd Qu.:89.95
Max. :550.0 Max. :600.0 Max. :700.0 Max. :99.95

Grade
Length:1000
Class :character
Mode :character

> if("Marks" %in% colnames(marks_df)){
+   avg_marks <- mean(marks_df$Marks, na.rm = TRUE)
+   max_marks <- max(marks_df$Marks, na.rm = TRUE)
+   min_marks <- min(marks_df$Marks, na.rm = TRUE)
+   print(paste("Average Marks:", avg_marks))
+   print(paste("Highest Marks:", max_marks))
+   print(paste("Lowest Marks:", min_marks))
+ }
> print("--- Practical Completed Successfully ---")
[1] "--- Practical Completed Successfully ---"
> }
```