**Data Manipulation with *tidyvere* and *dplyr***

**Md. Neaz Ali**

**M.Sc in Statistics**

**Department of Statistics**

**Islamic University, Kushtia, - 7003, Bangladesh**

# **1. Introduction to tidyverse and dplyr**

* **tidyverse**: A collection of R packages designed for data science. Includes:
  + ggplot2 → data visualization
  + dplyr → data manipulation
  + tidyr → data tidying
  + readr → data import
  + stringr, forcats, etc.
* **dplyr**: Core package for data manipulation. Provides **verbs** (functions) that are simple and intuitive.

|  |
| --- |
| # Install tidyverse  install.packages("tidyverse")  # Load  library(dplyr) |

2. Core dplyr Verbs

2.1 filter() → Select rows (like WHERE in SQL)

|  |
| --- |
| # Syntax  filter(data, condition)  # Example: keep rows with Age > 25  df %>% filter(Age > 25) |

2.2 select() → Select columns

|  |
| --- |
| # Syntax  select(data, col1, col2, ...)  # Example: keep only Name and Score  df %>% select(Name, Score) |

2.3 arrange() → Sort rows

|  |
| --- |
| # Syntax  arrange(data, column)  # Example: sort by Score ascending  df %>% arrange(Score)  # Sort by Score descending  df %>% arrange(desc(Score)) |

2.4 mutate() → Create/modify columns

|  |
| --- |
| # Syntax  mutate(data, new\_col = expression)  # Example: create new column Pass/Fail  df %>% mutate(Passed = ifelse(Score >= 80, "Yes", "No")) |

2.5 summarise() → Summary statistics

|  |
| --- |
| # Syntax  summarise(data, summary\_name = function(column))  # Example: calculate mean score  df %>% summarise(Avg\_Score = mean(Score)) |

2.6 group\_by() + summarise() → Grouped operations

|  |
| --- |
| df %>%  group\_by(Passed) %>%  summarise(Avg\_Score = mean(Score), Count = n()) |

3. Data Joins (Merging Tables)

Just like SQL joins:

* inner\_join(x, y, by) → only matching rows
* left\_join(x, y, by) → keep all rows from left
* right\_join(x, y, by) → keep all rows from right
* full\_join(x, y, by) → keep all rows from both

|  |
| --- |
| students <- data.frame(ID = 1:3, Name = c("A", "B", "C"))  scores <- data.frame(ID = c(1,2,4), Score = c(90, 85, 70))  students %>% left\_join(scores, by = "ID") |

**4. Example Dataset**

We’ll use a simple dataset for practice:

|  |
| --- |
| students <- data.frame(  ID = 1:6,  Name = c("Ali", "Sara", "John", "Mina", "Ravi", "Lily"),  Age = c(20, 22, 21, 23, 22, 20),  Marks = c(85, 90, 70, 60, 95, 88),  Dept = c("CS", "Math", "CS", "Physics", "Math", "CS")  ) |

5. Practice Problems

Basic

1. Select only the Name and Marks columns.
2. Filter students who scored **above 80**.
3. Filter students in the **CS department**.
4. Arrange students by Marks in **descending order**.
5. Select students with Age > 21 and Marks > 80.

Intermediate

1. Create a new column Grade:
   1. "A" if Marks ≥ 85, else "B".
2. Add 5 bonus points to all students’ Marks.
3. Summarize the **average Marks** of all students.
4. Find the **maximum Marks** in the dataset.
5. Count how many students are in each department.

Grouped operations

1. Compute the **average Marks per department**.
2. Find the **minimum Age** per department.
3. Count how many students passed (Marks ≥ 70) vs failed.
4. Group students by Grade (A/B) and find the **mean Age**.

Joins

1. Create another dataframe sports:

|  |
| --- |
| sports <- data.frame(ID = c(1,3,5,6), Sport = c("Football", "Tennis", "Cricket", "Hockey")) |

Perform a **left join** with students.

1. Use an **inner join** to keep only students with sports data.
2. Use a **full join** to see all students and all sports.

Challenging

1. Find the **top 3 students** with highest Marks.
2. Compute **department-wise pass percentage** (Marks ≥ 70).
3. Create a summary table: for each Dept → show Avg Marks, Max Marks, No. of Students.

6. Suggested Homework (Optional)

1. Import a real dataset (e.g., mtcars or iris).
2. Apply **all the verbs (filter, select, arrange, mutate, summarise, group\_by)**.
3. Perform at least **one join** with another dataset.