Function	Usage	Example/Result
`setwd("path")`	Sets the working directory to the specified path.	`setwd(" /Documents/Project")`
`read.csv("file.csv", header=TRUE, sep=",")`	Reads a CSV file into R as a data frame.	`fin <- read.csv("data.csv")`
`head(data, n)`	Displays the first `n` rows of a data frame.	`head(fin, 10)` returns the first 10 rows of `fin`.
`tail(data, n)`	Displays the last `n` rows of a data frame.	`tail(fin, 10)` returns the last 10 rows of `fin`.
`summary(data)`	Generates summary statistics for each column in a data frame.	`summary(fin)` shows min, max, mean, etc., for each column in `fin`.
`str(data)`	Displays the structure of a data frame, including column types and sample data.	`str(fin)` outputs the data types and example values of each column in `fin`.
`factor(data$col)`	Converts a column to a factor (categorical variable).	`fin$ID <- factor(fin$ID)` changes the `ID` column to factor type.
`as.numeric(data$col)`	Converts a column to a numeric type, if possible.	`fin$Expenses <- as.numeric(fin$Expenses)` converts `Expenses` to numeric.
`gsub("pattern", "replacement", data$col)`	Replaces patterns in strings within a column.	`fin$Revenue <- gsub("$$", "", fin$Revenue)` removes dollar signs from `Revenue`.
`is.na(data$col)`	Identifies missing (NA) values in a column. Returns a logical vector.	`is.na(fin$Expenses)` returns `TRUE` for NAs in `Expenses`.
`complete.cases(data)`	Returns a logical vector indicating rows with no missing values.	`fin[complete.cases(fin), ]` keeps only rows with no missing values in `fin`.
`na.strings = c("")`	Converts specified strings to NA when reading a file.	`read.csv("data.csv", na.strings = c(""))` treats empty fields as `NA`.
`which(condition)`	Returns indices of rows meeting a condition.	`which(fin$Revenue == 9746272)` returns indices where `Revenue` equals 9746272.
`is.na(data$col) <- value`	Sets specified NA values in a column.	`fin[is.na(fin$State) & fin$City == "New York", "State"] <- "NY"` replaces missing `State` values with "NY" for rows where `City` is "New York".
`median(data$col, na.rm=TRUE)`	Calculates the median, excluding NA values.	`median(fin$Employees, na.rm=TRUE)` calculates the median of `Employees`, ignoring NAs.
`library(dplyr)`	Loads the `dplyr` package for data manipulation.	`library(dplyr)` loads dplyr functions like `filter` and `mutate`.
`filter(data, condition)`	Filters rows in a data frame based on a condition.	`filter(fin, Employees > 100)` selects rows where `Employees` is greater than 100.
`mutate(data, new_col = expression)`	Adds a new column or modifies an existing column.	`mutate(fin, Profit = Revenue - Expenses)` adds a `Profit` column to `fin`.
`group_by(data, col)`	Groups data by a specific column for aggregation.	`group_by(fin, Industry)` groups `fin` by the `Industry` column.
`summarise(data, new_col = fun(col))`	Summarizes data, e.g., calculating mean or median by groups.	`summarise(fin, avg_profit = mean(Profit))` calculates average profit per group.
`replace_na(data, list(col = value))`	Replaces NA values in a column with specified value.	`replace_na(list(Revenue = 0))` fills missing `Revenue` values with 0.
`separate(data, col, into, sep)`	Splits a column into multiple columns by a delimiter.	`separate(data, score, c("home", "away"), sep = "-")` splits `score` into `home` and `away`.

`rename_all(data, fun)`	Applies a function to all column names, e.g., lowercase.	`rename_all(fin, tolower)` makes all column names in `fin` lowercase.
`ggplot(data) + aes(x, y)`	Sets up a ggplot with specified aesthetics.	`ggplot(fin) + aes(x = Revenue, y = Expenses)` creates a plot with `Revenue` on x and `Expenses` on y.
`geom_point()`	Adds points to a ggplot for scatter plots.	`+ geom_point()` adds a scatter plot layer to `ggplot`.
`geom_smooth()`	Adds a trend line to a ggplot scatter plot.	`+ geom_smooth()` adds a smooth trend line.
`geom_boxplot()`	Creates a boxplot in a ggplot.	`+ geom_boxplot()` creates a boxplot layer for `ggplot`.
`theme_minimal()`	Sets a minimalistic theme for ggplots.	`+ theme_minimal()` applies a clean, minimalistic style to the plot.
`coord_flip()`	Flips the x and y coordinates in ggplot for horizontal plots.	`+ coord_flip()` flips x and y axes.
`fct_reorder(col, var)`	Reorders factor levels based on a variable in ggplot.	`fct_reorder(venue, attendance)` orders venues by `attendance`.