Data Wrangling – Part 2

`filter()` Function

Filter Function - `filter()`: when filtering think about filtering ROWS. Filter uses Boolean logic.

Command Illustration

```
new_dataframe_name <- dataframe_name %>%
     filter(boolean expression using the name of a column)
```

For the illustration examples, assume the dataframe is the following:

Illustration_Data

Name	Age	total_Income	var_1	var_2	var_3	zipcode	honesty	cat_total
Val	18	18000	apple	carrots	elephant	60001	agree	0
Derek	25	25000	grapes	carrots	tiger	60073	disagree	1
Whitney	30	30000	bananas	carrots	lion	60109	disagree	2
Sasha	40	40000	peaches	carrots	rabbit	60111	disagree	1
Daniella	45	45000	bananas	carrots	shark	60155	agree	1

Ways to filter:

1. ***Simple Conditions (one boolean expression) -*** For example: Greater than (`>`), less than (`<`), or equal to (`==`)

Example 1: Select rows where "Age" is greater than 30

```
example_1 <- Illustration_Data %>%
  filter(Age > 30)
```

Output will be a dataframe that looks like:

Name	Age	total_Income	var_1	var_2	var_3	zipcode	honesty	cat_total
Sasha	40	40000	peaches	carrots	rabbit	60111	disagree	1
Daniella	45	45000	bananas	carrots	shark	60155	agree	1

Example 2: Select rows with "agree" in the "honesty" column

```
example_2 <- Illustration_Data %>%
  filter(honesty == "agree")
```

Output will be a dataframe that looks like:

Name	Age	total_Income	var_1	var_2	var_3	zipcode	honesty	cat_total
Val	18	18000	apple	carrots	elephant	60001	agree	0
Daniella	45	45000	bananas	carrots	shark	60155	agree	1

2. ***Multiple Conditions -*** you can combine conditions using logical operators like `&` (AND) and `|` (OR).

Example 3: Select rows where "Age" is greater than 30 and "total_Income" is less than 50000

```
example_3 <- Illustration_Data %>%
  filter(Age > 30 & total_Income < 50000)
```

Output will be a dataframe that looks like:

Name	Age	total_Income	var_1	var_2	var_3	zipcode	honesty	cat_total
Sasha	40	40000	peaches	carrots	rabbit	60111	disagree	1
Daniella	45	45000	bananas	carrots	shark	60155	agree	1

Example 4: Select rows where "Age" is greater than 20 and "total_Income" is less than 30000 and zipcode is equal to 60073

```
example_4 <- Illustration_Data %>%
  filter(Age > 20 & total_Income < 30000 & zipcode == 60073)
```

Output will be a dataframe that looks like:

Name	Age	total_Income	var_1	var_2	var_3	zipcode	honesty	cat_total
Derek	25	25000	grapes	carrots	tiger	60073	disagree	1

Example 5: Select rows where "Age" is greater than 30 or "total_Income" is greater than 20000

```
example_5 <- Illustration_Data %>%
  filter(Age > 30 | total_Income > 20000)
```

Output will be a dataframe that looks like:

Name	Age	total_Income	var_1	var_2	var_3	zipcode	honesty	cat_total
Derek	25	25000	grapes	carrots	tiger	60073	disagree	1
Whitney	30	30000	bananas	carrots	lion	60109	disagree	2
Sasha	40	40000	peaches	carrots	rabbit	60111	disagree	1
Daniella	45	45000	bananas	carrots	shark	60155	agree	1

3. ***Exclusion -*** to exclude certain rows, you can use the `!=` operator (not equal to).

Example 6: Exclude rows with "zipcode" equal to 60111

```
example_6 <- Illustration_Data %>%
  filter(zipcode != 60111)
```

Output will be a dataframe that looks like:

Name	Age	total_Income	var_1	var_2	var_3	zipcode	honesty	cat_total
Val	18	18000	apple	carrots	elephant	60001	agree	0
Derek	25	25000	grapes	carrots	tiger	60073	disagree	1
Whitney	30	30000	bananas	carrots	lion	60109	disagree	2
Daniella	45	45000	bananas	carrots	shark	60155	agree	1

4. ***Filter rows based on vector of conditions -*** The `%in%` operator is useful for filtering rows with values in a specified vector.

Example 7: Select rows where "var_1" is either "bananas" or "grapes"

```
example_7 <- Illustration_Data %>%
  filter(var_1 %in% c("bananas", "grapes"))
```

Output will be a dataframe that looks like:

Name	Age	total_Income	var_1	var_2	var_3	zipcode	honesty	cat_total
Derek	25	25000	grapes	carrots	tiger	60073	disagree	1
Whitney	30	30000	bananas	carrots	lion	60109	disagree	2
Daniella	45	45000	bananas	carrots	shark	60155	agree	1