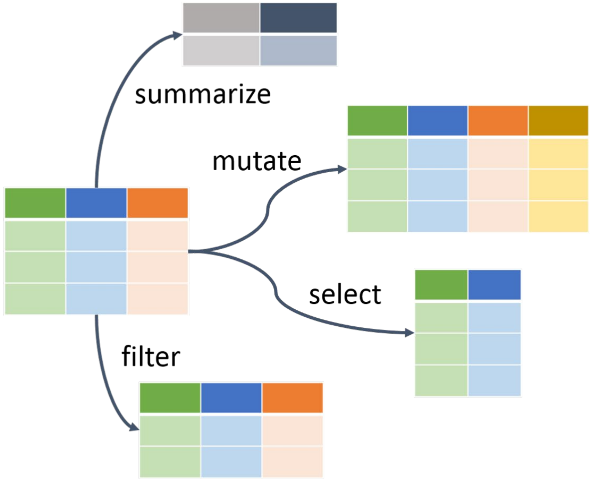
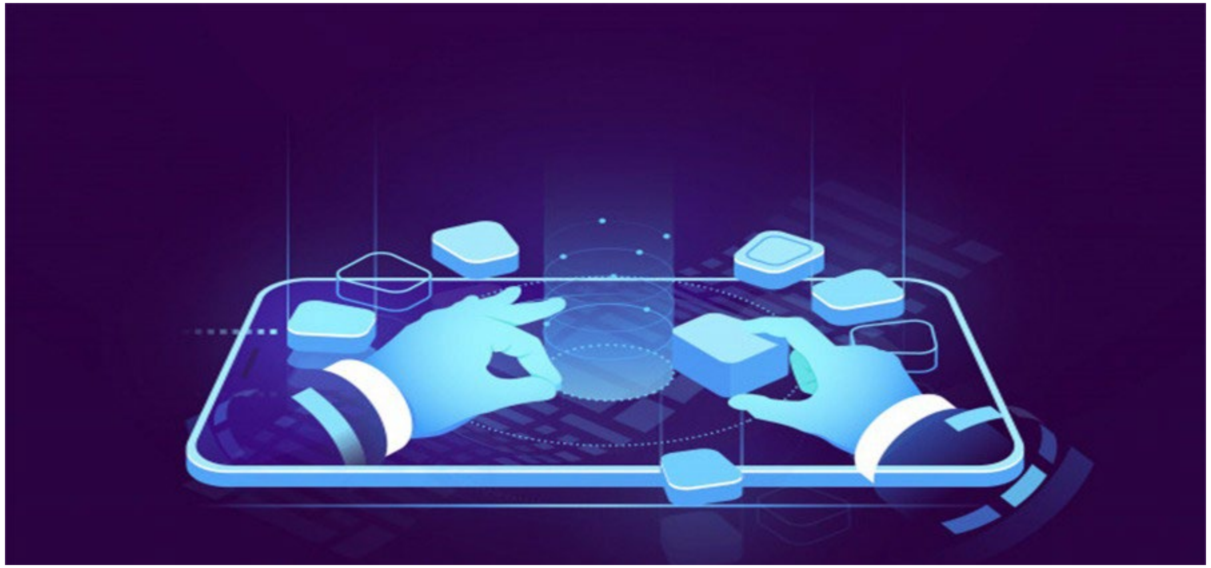
Data Manipulation I

John Rios

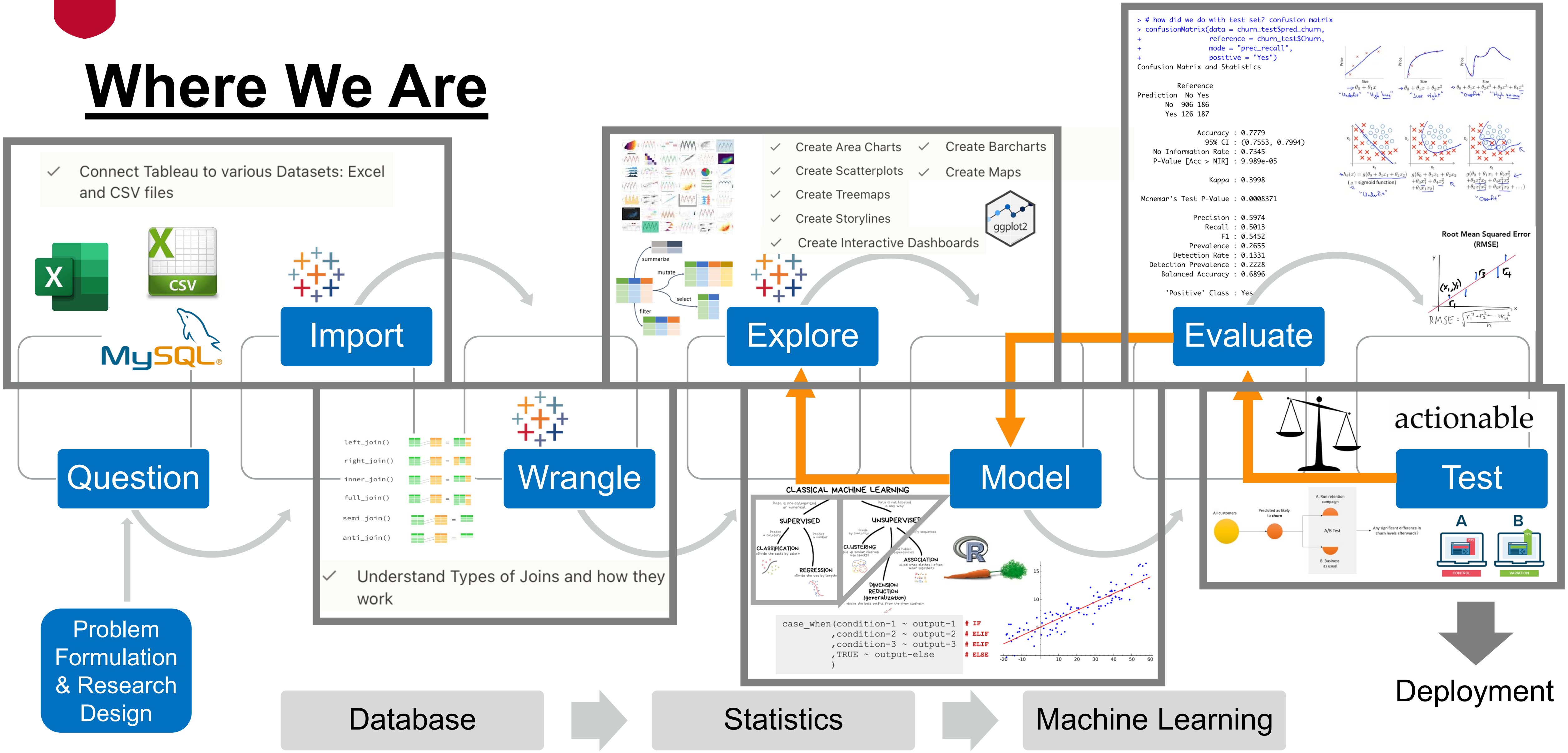
Business Intelligence and Analytics

Terry College of Business
UNIVERSITY OF GEORGIA





Where We Are



✓ Connect Tableau to various Datasets: Excel and CSV files

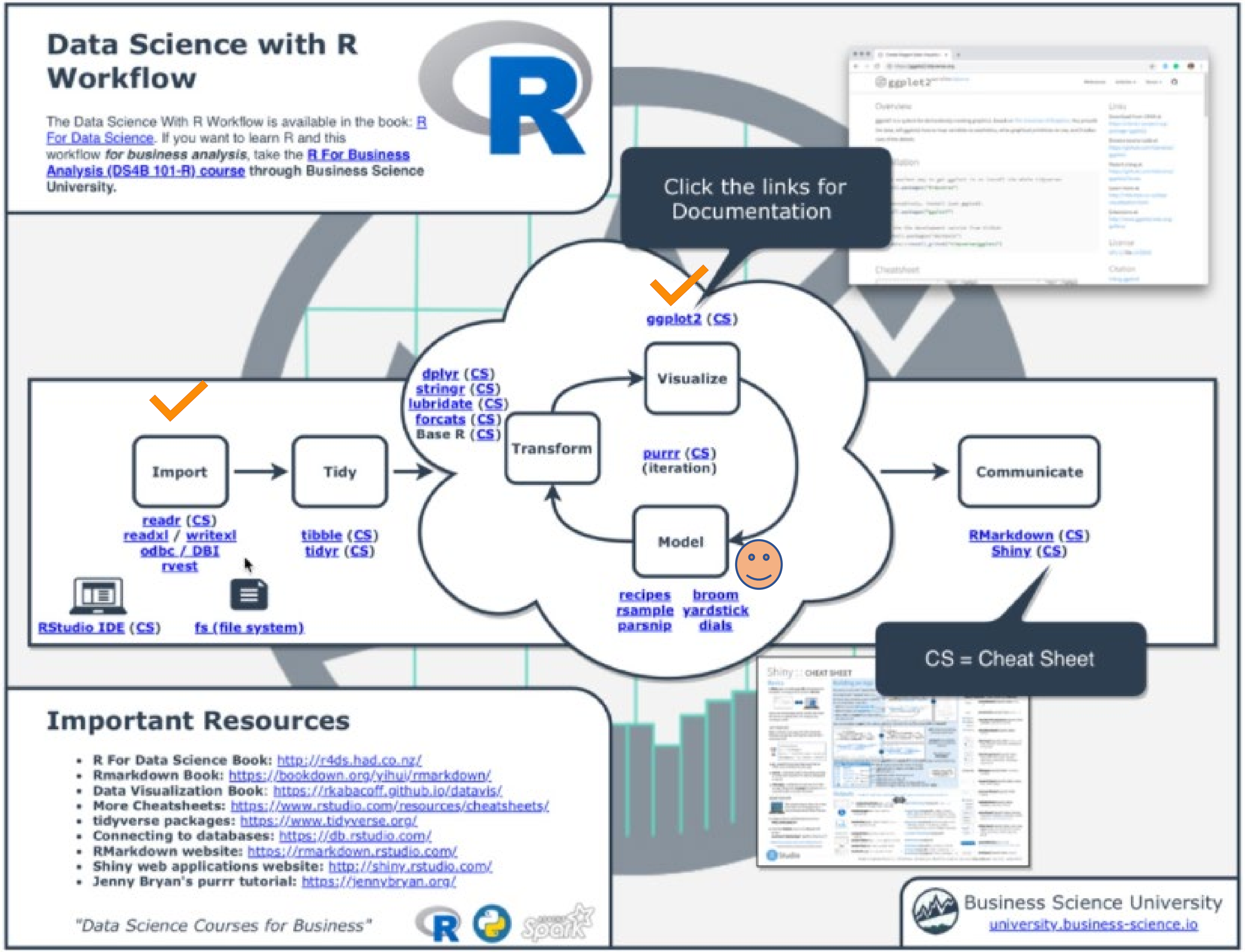
✓ Create Area Charts
✓ Create Scatterplots
✓ Create Treemaps
✓ Create Storylines
✓ Create Interactive Dashboards
✓ Create Barcharts
✓ Create Maps

✓ Understand Types of Joins and how they work

```
> # how did we do with test set? confusion matrix
> confusionMatrix(data = churn_test$pred_churn,
+                 reference = churn_test$Churn,
+                 mode = "prec_recall",
+                 positive = "Yes")
Confusion Matrix and Statistics

          Reference
Prediction  No Yes
       No  906 186
       Yes 126 187

               Accuracy : 0.7779
                 95% CI : (0.7553, 0.7994)
    No Information Rate : 0.7345
    P-Value [Acc > NIR] : 9.989e-05

                  Kappa : 0.3998

 Mcnemar's Test P-Value : 0.0008371

              Precision : 0.5974
                 Recall : 0.5013
                     F1 : 0.5452
             Prevalence : 0.2655
         Detection Rate : 0.1331
   Detection Prevalence : 0.2228
      Balanced Accuracy : 0.6896

       'Positive' Class : Yes
```

```
case_when(condition-1 ~ output-1     # IF
         ,condition-2 ~ output-2     # ELIF
         ,condition-3 ~ output-3     # ELIF
         ,TRUE ~ output-else         # ELSE
         )
```

```
left_join()
right_join()
inner_join()
full_join()
semi_join()
anti_join()
```



Data Science with R Workflow

The Data Science With R Workflow is available in the book: R For Data Science. If you want to learn R and this workflow ***for business analysis***, take the **R For Business Analysis (DS4B 101-R) course** through **Business Science University.**

Click the links for Documentation

Import → Tidy → Transform → Visualize → Model → Transform; → Communicate

Import: readr (CS)
readxl / writexl
odbc / DBI
rvest

Tidy: tibble (CS)
tidyr (CS)

Transform: dplyr (CS)
stringr (CS)
lubridate (CS)
forcats (CS)
Base R (CS)

Visualize: ggplot2 (CS)

purrr (CS)
(iteration)

Model: recipes
rsample
parsnip
broom
yardstick
dials

Communicate: RMarkdown (CS)
Shiny (CS)

RStudio IDE (CS)

fs (file system)

CS = Cheat Sheet

Important Resources

- **R For Data Science Book:** http://r4ds.had.co.nz/
- **Rmarkdown Book:** https://bookdown.org/yihui/rmarkdown/
- **Data Visualization Book:** https://rkabacoff.github.io/datavis/
- **More Cheatsheets:** https://www.rstudio.com/resources/cheatsheets/
- **tidyverse packages:** https://www.tidyverse.org/
- **Connecting to databases:** https://db.rstudio.com/
- **RMarkdown website:** https://rmarkdown.rstudio.com/
- **Shiny web applications website:** http://shiny.rstudio.com/
- **Jenny Bryan's purrr tutorial:** https://jennybryan.org/

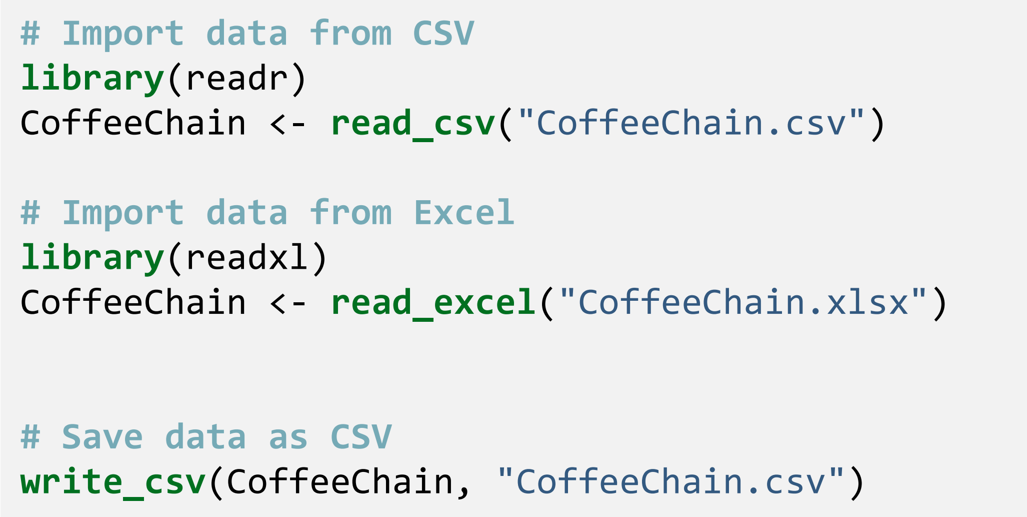
"Data Science Courses for Business"

Business Science University
university.business-science.io

https://www.business-science.io/r-cheatsheet.html

Import Reminder

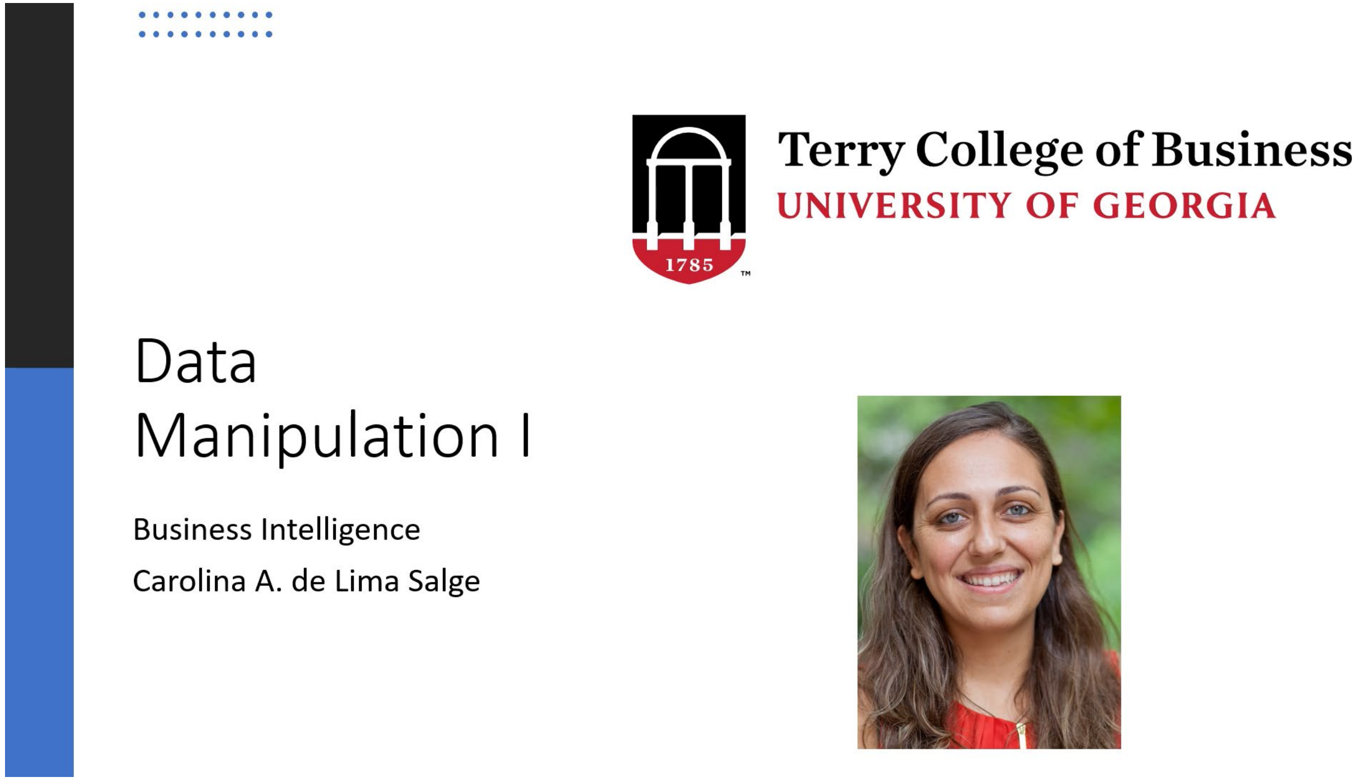
Used the **readr** package to import CSV files & the **readxl** package to import Excel files.



```r
# Import data from CSV
library(readr)
CoffeeChain <- read_csv("CoffeeChain.csv")

# Import data from Excel
library(readxl)
CoffeeChain <- read_excel("CoffeeChain.xlsx")


# Save data as CSV
write_csv(CoffeeChain, "CoffeeChain.csv")
```

Variable Types

- Numeric
- String
- Date-times
- Binary
- Factors



Terry College of Business
UNIVERSITY OF GEORGIA

Data
Manipulation I

Business Intelligence

Carolina A. de Lima Salge

Tibbles

- A rethinking of data frames, which are essentially data tables.
- One of the unifying features of Tidyverse.
- Tibbles are data frames, but they tweak some older behaviors to make life a little easier.

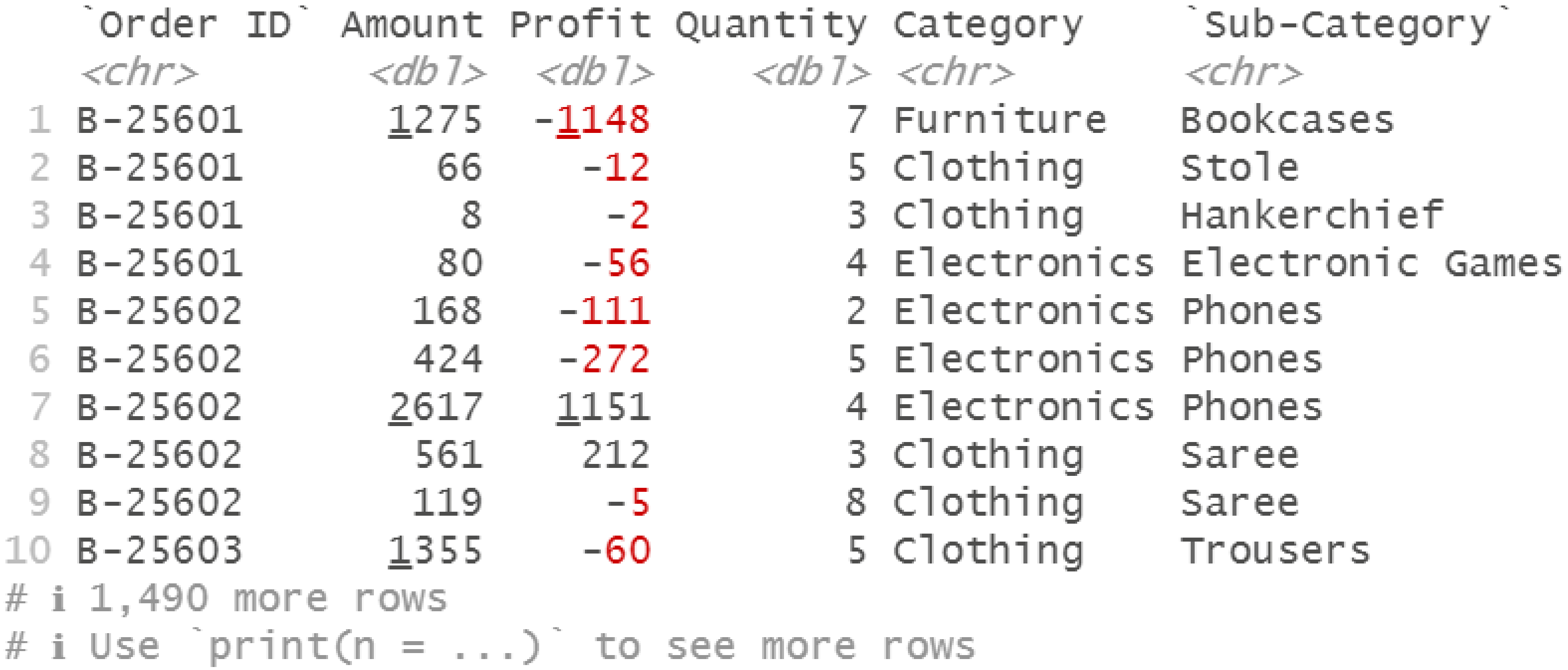
Tibbles

```
# Load tidyverse package
library(tidyverse)

# Import data as a tibble - using read_csv() function
orderDetails <- read_csv("Order Details.csv")
# Print the tibble
orderDetails

# Import the same data as a data frame - using read.csv() function
orderDetails2 <- read.csv("Order Details.csv")
# Print the data frame
orderDetails2
```

Quick Exercise

Open R Studio, create an R Markdown file, then copy and paste the code from the previous slide into a chunk of R code and execute line by line (Ctrl + Enter / Cmd + Enter)

Do you see something like the figure below? If yes, is it a tibble or a data frame?



```
   `Order ID` Amount Profit Quantity Category    `Sub-Category`
   <chr>       <dbl>  <dbl>    <dbl> <chr>       <chr>
 1 B-25601      1275  -1148        7 Furniture   Bookcases
 2 B-25601        66    -12        5 Clothing    Stole
 3 B-25601         8     -2        3 Clothing    Hankerchief
 4 B-25601        80    -56        4 Electronics Electronic Games
 5 B-25602       168   -111        2 Electronics Phones
 6 B-25602       424   -272        5 Electronics Phones
 7 B-25602      2617   1151        4 Electronics Phones
 8 B-25602       561    212        3 Clothing    Saree
 9 B-25602       119     -5        8 Clothing    Saree
10 B-25603      1355    -60        5 Clothing    Trousers
# ℹ 1,490 more rows
# ℹ Use `print(n = ...)` to see more rows
```

Tibbles

```
# To create a tibble from individual
vectors, use tibble()
library(tibble)
gender <- c("m","f","f")
age <- c(5,8,3)
# Create tibble
tb <- tibble(`1 gender`=gender, `2 age`=age)
tb

# From dataframe to tibble
iris_tibble <- as_tibble(iris)
# From tibble to dataframe
iris_df  <- as.data.frame(iris_tibble)
```

```
# The parenthesis around the code will
also print the tibble
(tb <- tibble(x = 1:5,
       y = 1,
       z = x ^ 2 + y))
# A tibble: 5 x 3
      x     y     z
  <int> <dbl> <dbl>
1     1     1     2
2     2     1     5
3     3     1    10
4     4     1    17
5     5     1    26
```

Tibbles

```
# To subset a tibble use $ or [[
tb$x
[1] 1 2 3 4 5
tb[["x"]]
[1] 1 2 3 4 5
tb[[1]]
[1] 1 2 3 4 5

# tb$x == tb[["x"]] == tb[[1]]
# $ and [[ reference a column in a table (table$column; table[[column]])
# [[ can also be used to reference a specific cell

# Subset the table to show the second element of the first column (table[[cell2, column1]])
tb[[2,"x"]]
[1] 2
tb[[2,1]]
[1] 2
```

Dplyr

One of the packages in the Tidyverse that enables the transformation of data

- Look at a subset of the rows—**filter()**
- Reorder rows—**arrange()**
- Rename variables—**rename()**
- Create new variables—**mutate()**
- Collapse values down to a summary—**summarize()**

Data Import

```
library(tidyverse)

orderList <- read_csv("List of Orders.csv")
orderDetails <- read_csv("Order Details.csv")
salesTarget <- read_csv("Sales target.csv")
```

https://www.kaggle.com/benroshan/ecommerce-data?select=Order+Details.csv

Logical operations

Logical operator	Symbol
EQUAL	==
AND	&
OR	|
NOT	!

Missing values

- Missing values are indicated by NA (not available)
- Arithmetic expressions and functions containing missing values generate missing values

```
sum(c(1,NA,2))
```

- Use the na.rm=T option to exclude missing values for calculations

```
sum(c(1,NA,2),na.rm=TRUE)
```

The Pipe

The %>% focuses on the transformations, not what is being transformed, which makes the code easier to read.

- Only order details with products in Category == “Furniture”
- Take the `orderDetails` dataset **then** `filter()`

```
orderDetails %>%
     filter(., Category == "Furniture")
```

Filter Rows

A reference to orderDetails

```
orderDetails %>%
       filter(., Category == "Furniture")
# A tibble: 243 x 6
   `Order ID` Amount Profit Quantity Category  `Sub-Category`
   <chr>       <dbl>  <dbl>    <dbl> <chr>     <chr>
 1 B-25601      1275  -1148        7 Furniture Bookcases
 2 B-25603        24    -30        1 Furniture Chairs
 3 B-25608      1364  -1864        5 Furniture Tables
 4 B-25608       476      0        3 Furniture Chairs
 5 B-25610        30     -5        2 Furniture Furnishings
 6 B-25612       259    -55        2 Furniture Chairs
 7 B-25614       494     54        4 Furniture Bookcases
 8 B-25618       362    127        1 Furniture Bookcases
 9 B-25626      1103   -276        3 Furniture Chairs
10 B-25628        35     -8        2 Furniture Furnishings
# … with 233 more rows
```

Only order details with products in Category == “Furniture”

Filter Rows

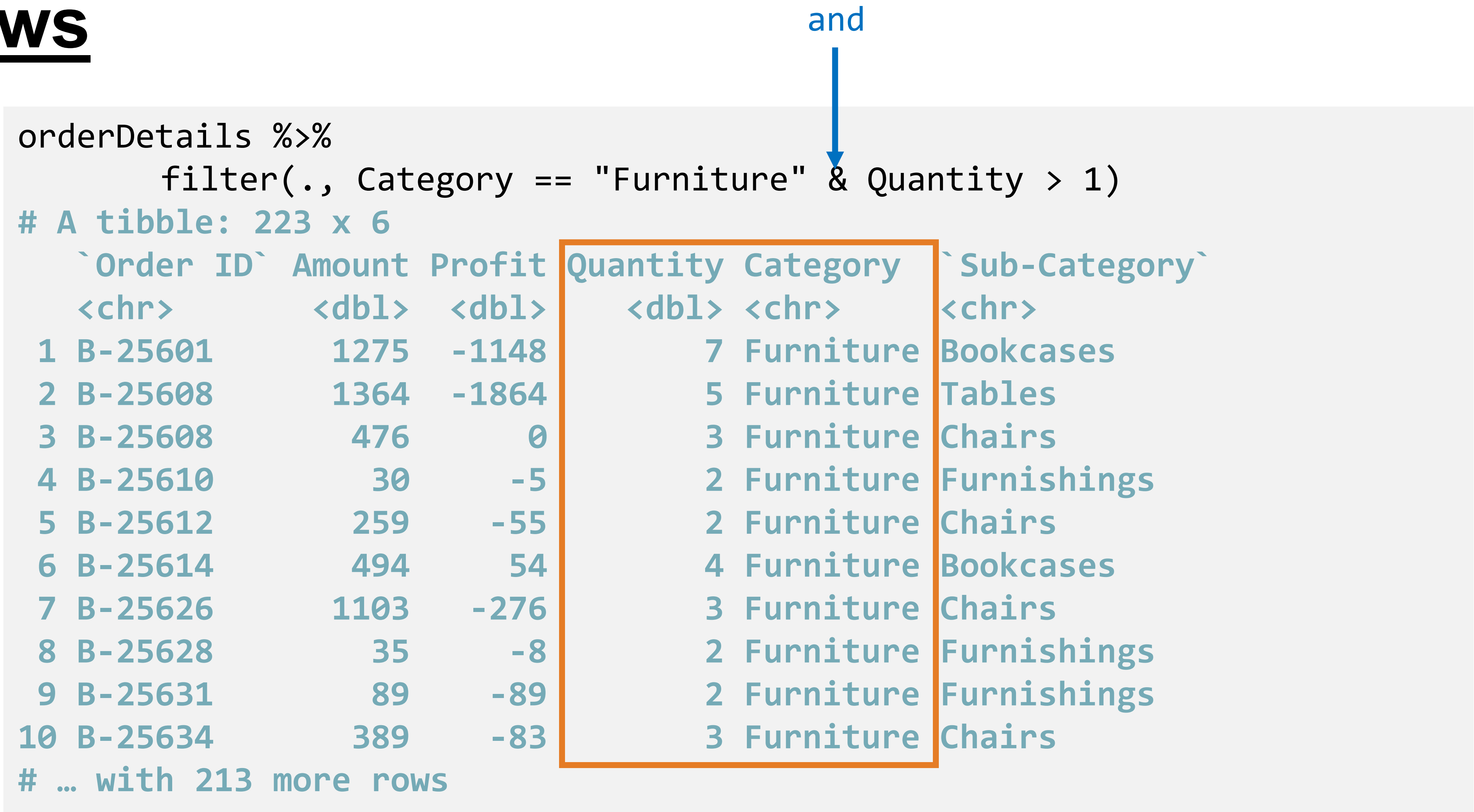
and

```
orderDetails %>%
       filter(., Category == "Furniture", Quantity > 1)
# A tibble: 223 x 6
   `Order ID` Amount Profit Quantity Category  `Sub-Category`
   <chr>       <dbl>  <dbl>    <dbl> <chr>     <chr>
 1 B-25601      1275  -1148        7 Furniture Bookcases
 2 B-25608      1364  -1864        5 Furniture Tables
 3 B-25608       476      0        3 Furniture Chairs
 4 B-25610        30     -5        2 Furniture Furnishings
 5 B-25612       259    -55        2 Furniture Chairs
 6 B-25614       494     54        4 Furniture Bookcases
 7 B-25626      1103   -276        3 Furniture Chairs
 8 B-25628        35     -8        2 Furniture Furnishings
 9 B-25631        89    -89        2 Furniture Furnishings
10 B-25634       389    -83        3 Furniture Chairs
# … with 213 more rows
```

Only order details with products in Category == “Furniture” & Quantity > 1

Filter Rows

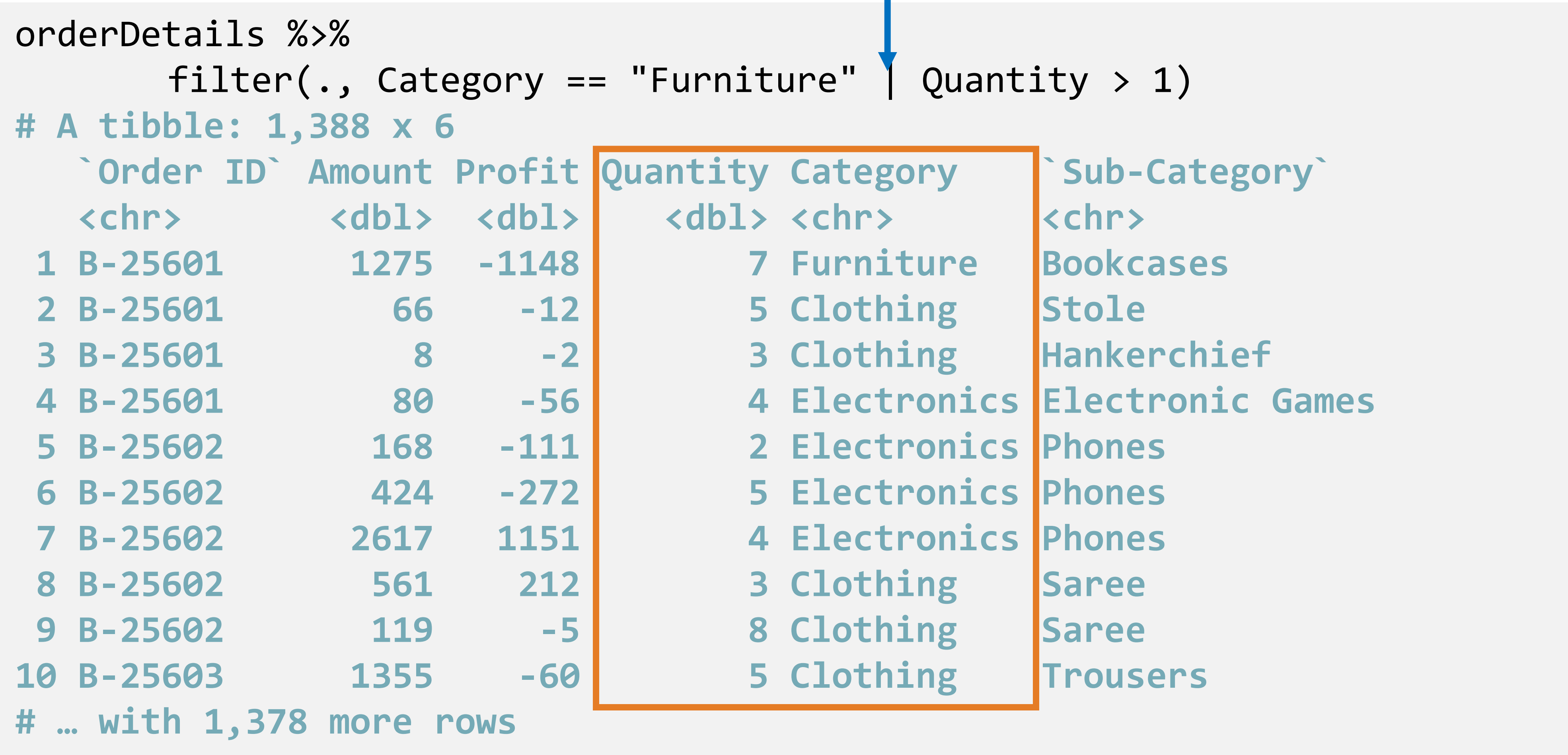
Only order details with products in Category == “Furniture” & Quantity > 1



```
orderDetails %>%
       filter(., Category == "Furniture" & Quantity > 1)
# A tibble: 223 x 6
   `Order ID` Amount Profit Quantity Category  `Sub-Category`
   <chr>       <dbl>  <dbl>    <dbl> <chr>     <chr>
 1 B-25601      1275  -1148        7 Furniture Bookcases
 2 B-25608      1364  -1864        5 Furniture Tables
 3 B-25608       476      0        3 Furniture Chairs
 4 B-25610        30     -5        2 Furniture Furnishings
 5 B-25612       259    -55        2 Furniture Chairs
 6 B-25614       494     54        4 Furniture Bookcases
 7 B-25626      1103   -276        3 Furniture Chairs
 8 B-25628        35     -8        2 Furniture Furnishings
 9 B-25631        89    -89        2 Furniture Furnishings
10 B-25634       389    -83        3 Furniture Chairs
# … with 213 more rows
```

Filter Rows

or



```
orderDetails %>%
       filter(., Category == "Furniture" | Quantity > 1)
# A tibble: 1,388 x 6
   `Order ID` Amount Profit Quantity Category    `Sub-Category`
   <chr>       <dbl>  <dbl>    <dbl> <chr>       <chr>
 1 B-25601      1275  -1148        7 Furniture   Bookcases
 2 B-25601        66    -12        5 Clothing    Stole
 3 B-25601         8     -2        3 Clothing    Hankerchief
 4 B-25601        80    -56        4 Electronics Electronic Games
 5 B-25602       168   -111        2 Electronics Phones
 6 B-25602       424   -272        5 Electronics Phones
 7 B-25602      2617   1151        4 Electronics Phones
 8 B-25602       561    212        3 Clothing    Saree
 9 B-25602       119     -5        8 Clothing    Saree
10 B-25603      1355    -60        5 Clothing    Trousers
# … with 1,378 more rows
```

Only order details with products in Category == “Furniture” or Quantity > 1

Filter Rows

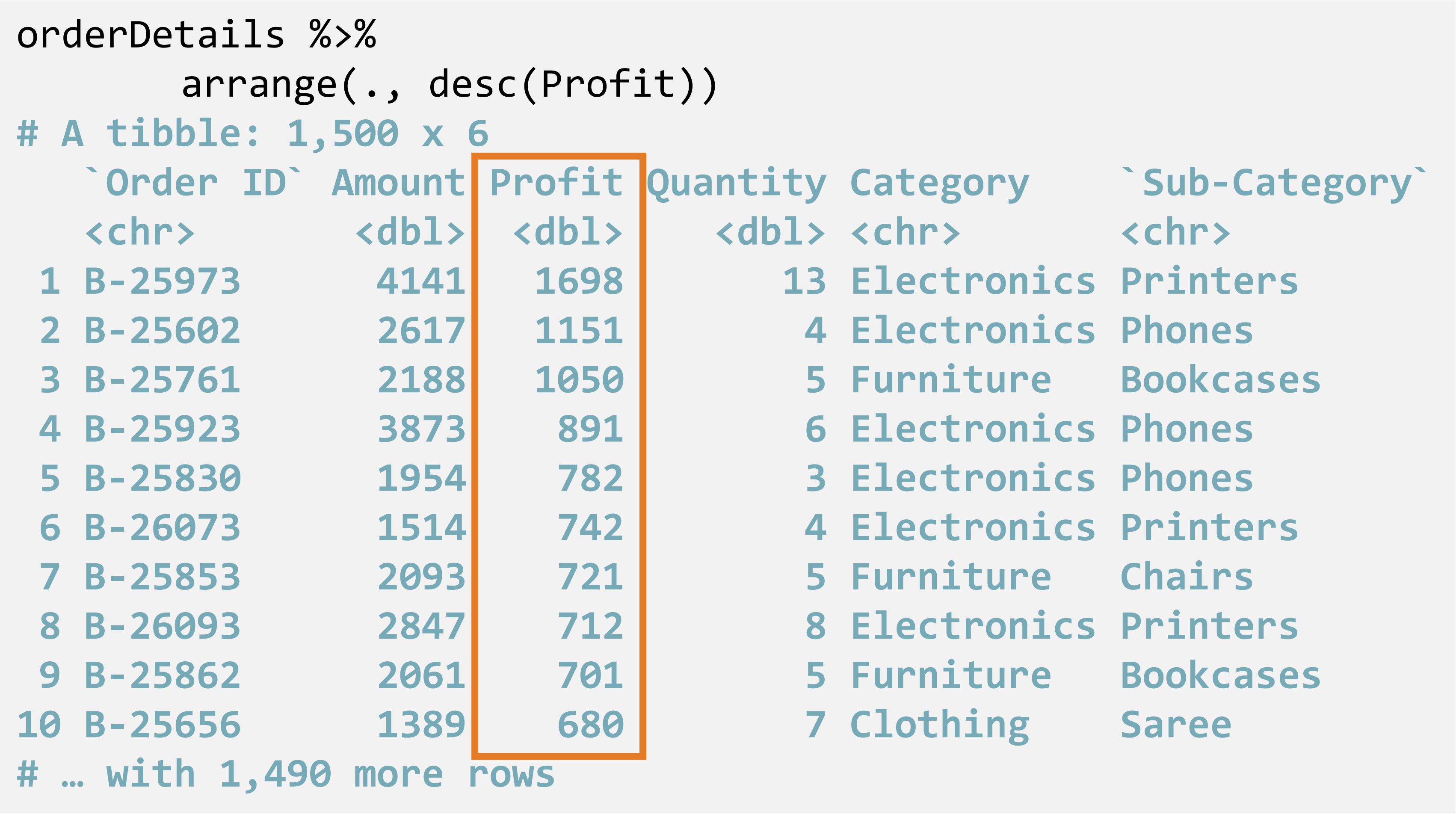
Only order details with products in Category in “Furniture” or “Clothing”

```
orderDetails %>%
  filter(., Category %in% c("Furniture", "Clothing"))

# A tibble: 1,192 x 6
   `Order ID` Amount Profit Quantity Category  `Sub-Category`
   <chr>       <dbl>  <dbl>    <dbl> <chr>     <chr>
 1 B-25601      1275  -1148        7 Furniture Bookcases
 2 B-25601        66    -12        5 Clothing  Stole
 3 B-25601         8     -2        3 Clothing  Hankerchief
 4 B-25602       561    212        3 Clothing  Saree
 5 B-25602       119     -5        8 Clothing  Saree
 6 B-25603      1355    -60        5 Clothing  Trousers
 7 B-25603        24    -30        1 Furniture Chairs
 8 B-25603       193   -166        3 Clothing  Saree
 9 B-25603       180      5        3 Clothing  Trousers
10 B-25603       116     16        4 Clothing  Stole
# … with 1,182 more rows
```

Practice Question

Mike is trying to create a new table called **posBAprofit** that filters the orderDetails table to show observations where profit is above zero but below the average. How could you create this table in R?

Reorder (Arrange) Rows

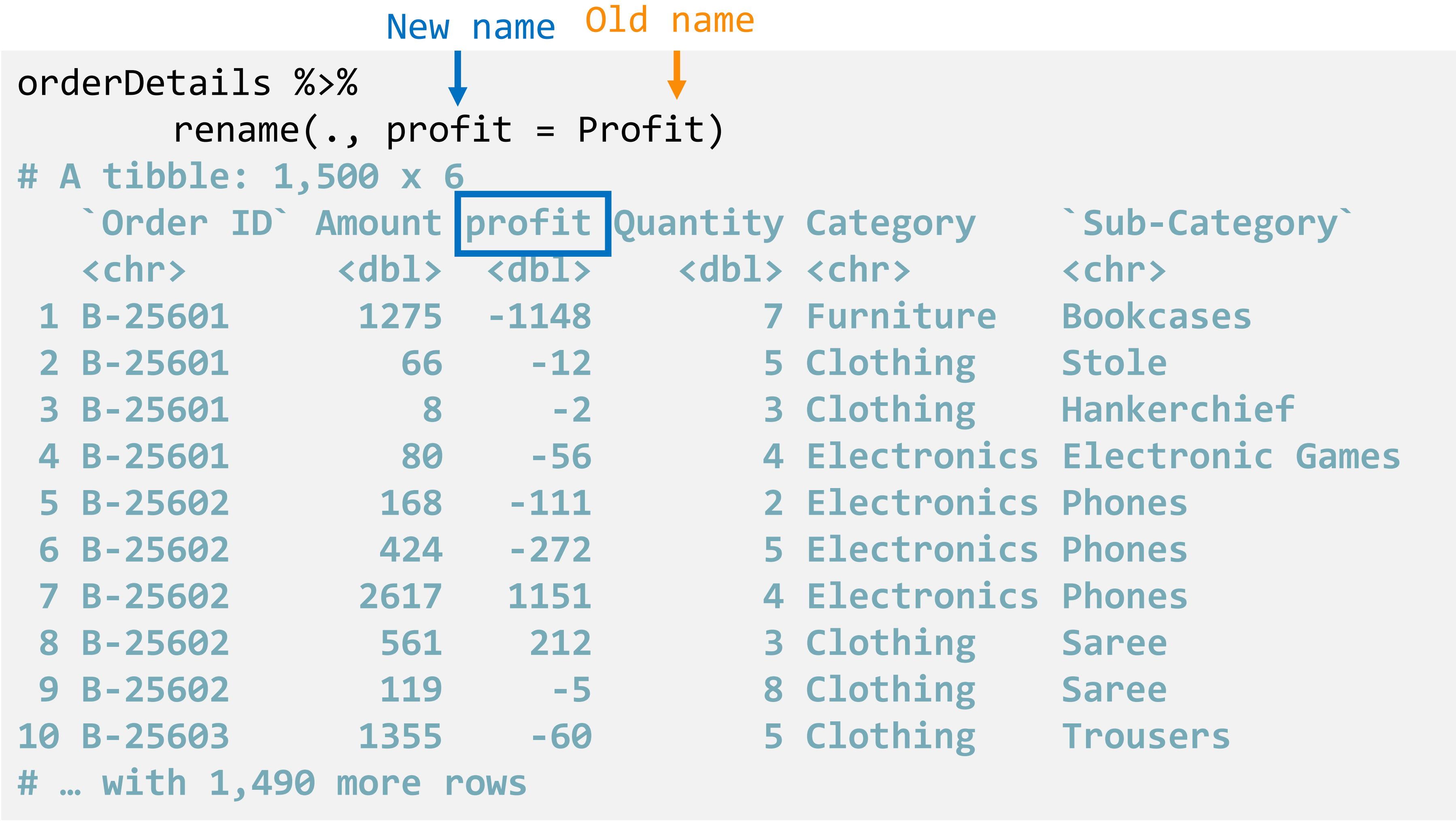


```
orderDetails %>%
       arrange(., desc(Profit))
# A tibble: 1,500 x 6
   `Order ID` Amount Profit Quantity Category    `Sub-Category`
   <chr>       <dbl>  <dbl>    <dbl> <chr>       <chr>
 1 B-25973      4141   1698       13 Electronics Printers
 2 B-25602      2617   1151        4 Electronics Phones
 3 B-25761      2188   1050        5 Furniture   Bookcases
 4 B-25923      3873    891        6 Electronics Phones
 5 B-25830      1954    782        3 Electronics Phones
 6 B-26073      1514    742        4 Electronics Printers
 7 B-25853      2093    721        5 Furniture   Chairs
 8 B-26093      2847    712        8 Electronics Printers
 9 B-25862      2061    701        5 Furniture   Bookcases
10 B-25656      1389    680        7 Clothing    Saree
# … with 1,490 more rows
```

Selecting Variables

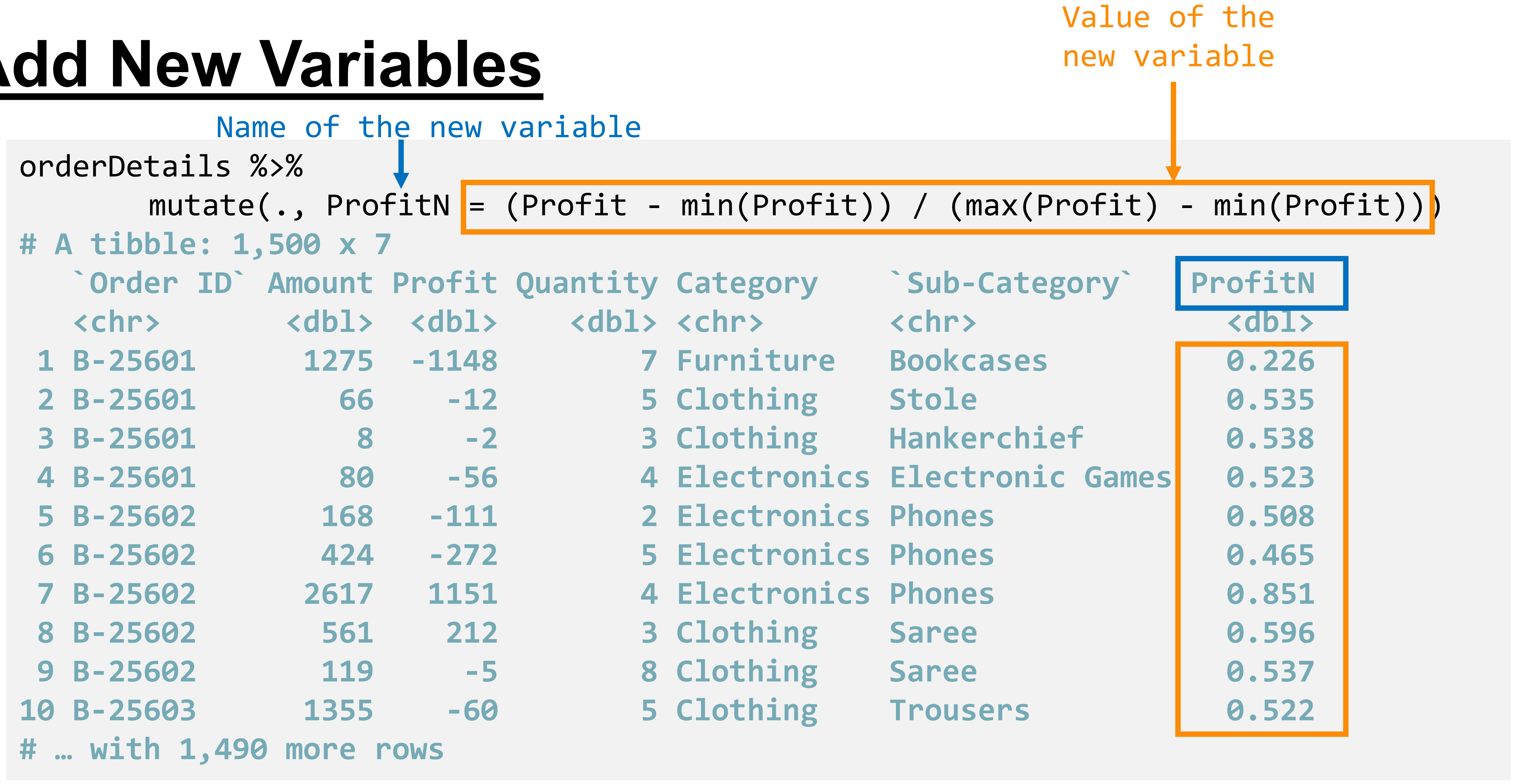
```
orderDetails %>%
       select(., c(Amount:Category))
# A tibble: 1,500 × 4
   Amount Profit Quantity Category
    <dbl>  <dbl>    <dbl> <chr>
 1   1275  -1148        7 Furniture
 2     66    -12        5 Clothing
 3      8     -2        3 Clothing
 4     80    -56        4 Electronics
 5    168   -111        2 Electronics
 6    424   -272        5 Electronics
 7   2617   1151        4 Electronics
 8    561    212        3 Clothing
 9    119     -5        8 Clothing
10   1355    -60        5 Clothing
# … with 1,490 more rows
```

Rename Variables



```
orderDetails %>%
       rename(., profit = Profit)
# A tibble: 1,500 x 6
   `Order ID` Amount profit Quantity Category    `Sub-Category`
   <chr>       <dbl>  <dbl>    <dbl> <chr>       <chr>
 1 B-25601      1275  -1148        7 Furniture   Bookcases
 2 B-25601        66    -12        5 Clothing    Stole
 3 B-25601         8     -2        3 Clothing    Hankerchief
 4 B-25601        80    -56        4 Electronics Electronic Games
 5 B-25602       168   -111        2 Electronics Phones
 6 B-25602       424   -272        5 Electronics Phones
 7 B-25602      2617   1151        4 Electronics Phones
 8 B-25602       561    212        3 Clothing    Saree
 9 B-25602       119     -5        8 Clothing    Saree
10 B-25603      1355    -60        5 Clothing    Trousers
# … with 1,490 more rows
```

Add New Variables



Value of the new variable

Name of the new variable

```
orderDetails %>%
       mutate(., ProfitN = (Profit - min(Profit)) / (max(Profit) - min(Profit)))
# A tibble: 1,500 x 7
   `Order ID` Amount Profit Quantity Category    `Sub-Category`   ProfitN
   <chr>       <dbl>  <dbl>    <dbl> <chr>       <chr>              <dbl>
 1 B-25601      1275  -1148        7 Furniture   Bookcases          0.226
 2 B-25601        66    -12        5 Clothing    Stole              0.535
 3 B-25601         8     -2        3 Clothing    Hankerchief        0.538
 4 B-25601        80    -56        4 Electronics Electronic Games   0.523
 5 B-25602       168   -111        2 Electronics Phones             0.508
 6 B-25602       424   -272        5 Electronics Phones             0.465
 7 B-25602      2617   1151        4 Electronics Phones             0.851
 8 B-25602       561    212        3 Clothing    Saree              0.596
 9 B-25602       119     -5        8 Clothing    Saree              0.537
10 B-25603      1355    -60        5 Clothing    Trousers           0.522
# … with 1,490 more rows
```

Add New Variables

Name of the new variable

```
orderDetails %>%
  mutate(., ProfitClass = case_when(Profit > 0 ~ "Positive",
    Profit < 0 ~ "Negative", Profit == 0 ~ "Zero"))
# A tibble: 1,500 x 7
   `Order ID` Amount Profit Quantity Category    `Sub-Category`    ProfitClass
   <chr>       <dbl>  <dbl>    <dbl> <chr>       <chr>             <chr>
 1 B-25601      1275  -1148        7 Furniture   Bookcases         Negative
 2 B-25601        66    -12        5 Clothing    Stole             Negative
 3 B-25601         8     -2        3 Clothing    Hankerchief       Negative
 4 B-25601        80    -56        4 Electronics Electronic Games Negative
 5 B-25602       168   -111        2 Electronics Phones            Negative
 6 B-25602       424   -272        5 Electronics Phones            Negative
 7 B-25602      2617   1151        4 Electronics Phones            Positive
 8 B-25602       561    212        3 Clothing    Saree             Positive
 9 B-25602       119     -5        8 Clothing    Saree             Negative
10 B-25603      1355    -60        5 Clothing    Trousers          Negative
# … with 1,490 more rows
```

Grouped Summaries

Calculate the average profit for each “Category” and “Sub-Category” combination. Then, sort the results descendingly to identify the most profitable groups.

Take the `orderDetails` dataset **then** `group_by()` **then** `summarise()` **then** `arrange()`

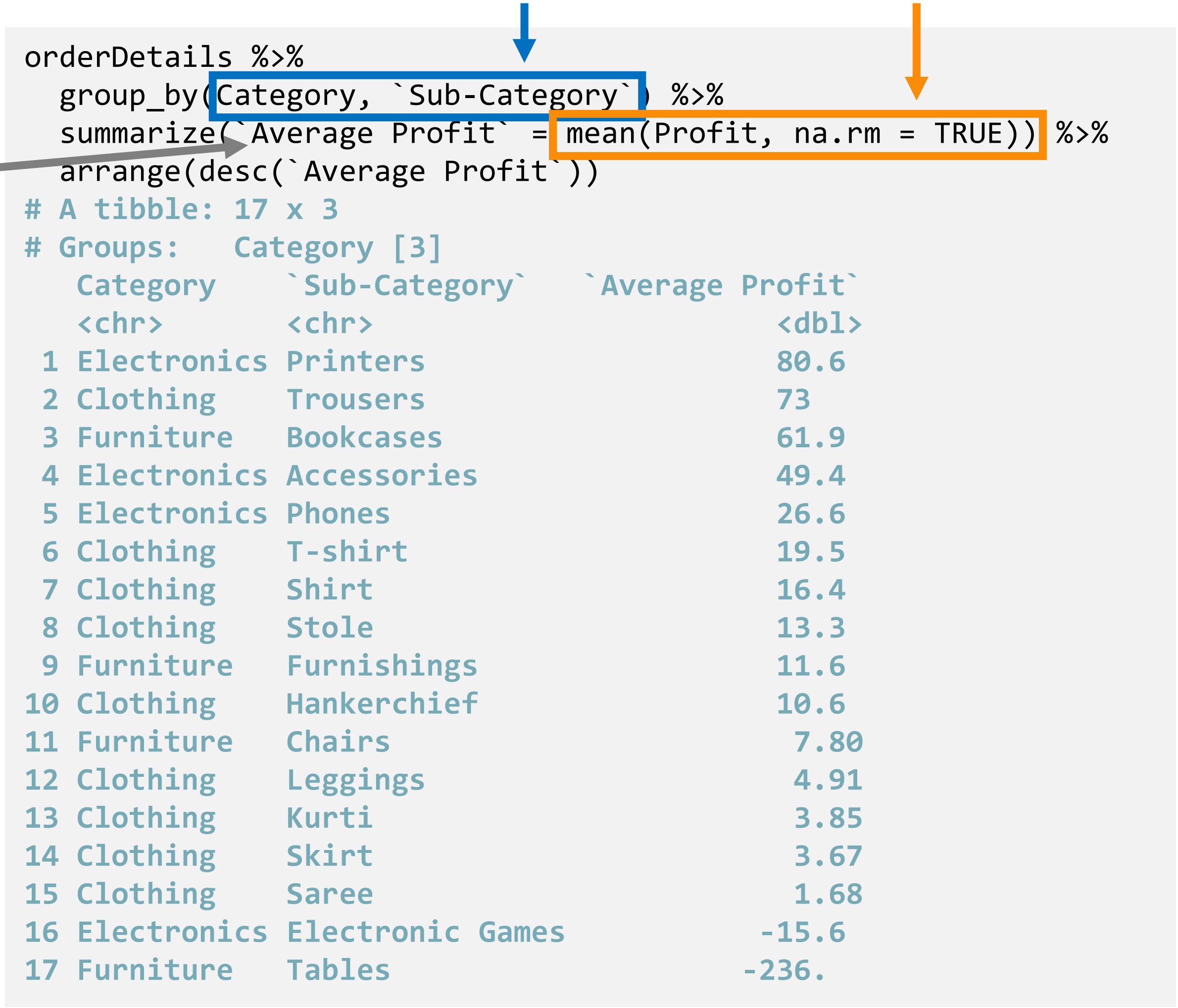
```
orderDetails %>%
  group_by(Category, `Sub-Category`) %>%
  summarize(`Average Profit` = mean(Profit, na.rm = TRUE)) %>%
  arrange(desc(`Average Profit`))
```

Grouped Summaries

Grouped variables

Summarized function and variable

Name of the new summarized variable



```
orderDetails %>%
  group_by(Category, `Sub-Category`) %>%
  summarize(`Average Profit` = mean(Profit, na.rm = TRUE)) %>%
  arrange(desc(`Average Profit`))
# A tibble: 17 x 3
# Groups:   Category [3]
   Category    `Sub-Category`    `Average Profit`
   <chr>       <chr>                        <dbl>
 1 Electronics Printers                     80.6
 2 Clothing    Trousers                     73
 3 Furniture   Bookcases                    61.9
 4 Electronics Accessories                  49.4
 5 Electronics Phones                       26.6
 6 Clothing    T-shirt                      19.5
 7 Clothing    Shirt                        16.4
 8 Clothing    Stole                        13.3
 9 Furniture   Furnishings                  11.6
10 Clothing    Hankerchief                  10.6
11 Furniture   Chairs                        7.80
12 Clothing    Leggings                      4.91
13 Clothing    Kurti                         3.85
14 Clothing    Skirt                         3.67
15 Clothing    Saree                         1.68
16 Electronics Electronic Games            -15.6
17 Furniture   Tables                     -236.
```

Useful Summary Functions

- Location
  - **mean(x)** and **median(x)**
- Spread
  - **sd(x)** and **IQR(x)**
- Rank
  - **min(x), quantile(x, 0.25)**, and **max(x)**
- Count
  - **n(x), sum(!is.na(x))**, and **n_distinct(x)**

dplyr helper functions

- Use with select()
  - starts_with()
    - matches column names that begin with certain characters
  - ends_with()
    - matches column names that end with specific characters
  - contains()
    - matches column names that contain a sequence of characters

Try Yourself!

Using the *orderList* dataset, create a table with the number of orders per state and sort it descendingly.