

# Data Management and Transformation

Juan Carlos Villaseñor-Derbez (JC)

## Exercise 1: Reading and transforming data

Your boss asked you what sounds like a simple query:

How much money have tuna purse seiners made since 2000 when fishing for bigeye tuna (*Thunnus obesus*) in the Eastern Pacific Ocean?

Let's make some assumptions that will help us answer this question:

1. We will interpret “making money” as revenue, not profits
2. The market price of tuna since 2000 has remained relatively stable, at around US\$2/Kg (See Sibert et al. (2012))
3. We will focus on tuna production in the Eastern Pacific Ocean as reported by the IATTC

### Part A) Obtaining data from the wild

How to find the data:

1. Go to [iattc.org](http://iattc.org)
2. In the top menu, **hover** over DATA
3. Click on “Public domain”
4. You will be taken to a page titled “Public domain data for download”
5. We will use “EPO total estimated catch by year, flag, gear, species”
6. Click on **CatchByFlagGear.zip** to the right of the table to prompt a download
7. Save the zip file to inside your EVR628 project at: `data/raw/`<sup>1</sup>
8. Using your finder / explorer, navigate to `EVR628/data/raw/` and unzip the **CatchByFlagGear.zip** file<sup>2</sup>
9. You will get a new folder called **CatchByFlagGear**
10. Read the PDF file enclosed, which contains the documentation

---

<sup>1</sup>If your web browser didn't allow you to specify the download folder, your file is likely in the “Downloads” folder. Navigate there and copy it to the `data/raw/` folder.

<sup>2</sup>Windows users: You might have to click a button called “Extract” in the top of your explorer window.

## Part B) Reading data

### 0. Put your post-it up

1. Start a new script called `tuna_analysis` and save it to your `scripts/03_analysis` folder
2. Add a comment header and outline, and load the `tidyverse` package at the top of your script
3. Use the `read_csv()` function to load the new data and assign it to an object called `tuna_data`
4. What are the existing column names?<sup>3</sup>
5. Remove your post-it when you are done

```
# Load packages
library(tidyverse)
library(janitor)

# Load the data
tuna_data <- read_csv("data/raw/CatchByFlagGear/CatchByFlagGear1918-2023.csv")

Rows: 13595 Columns: 5
-- Column specification -----
Delimiter: ","
chr (3): BanderaFlag, ArteGear, EspeciesSpecies
dbl (2): AnoYear, t

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
# Check colnames after cleaning names
colnames(tuna_data)

[1] "AnoYear"          "BanderaFlag"       "ArteGear"         "EspeciesSpecies"
[5] "t"
```

## Part C) Renaming columns with `clean_names()` and `rename()`

### 0. Put your post-it up

1. In your console, install the `janitor` package using `install.packages("janitor")`
2. Load `janitor` at the top of your script, and then read the documentation for the `clean_names()` function
3. Modify your code for the `tuna_data` object so that you pipe into `clean_names()` after reading the data
4. What are the new column names?
5. Extend the pipeline above so that we can use the `rename()` function. Rename the columns so that we only retain the English portion of the name.

---

<sup>3</sup>Hint: use the `colnames()` function

Let's also rename `t` as `catch`

#### 6. Remove your post-it when you are done

```
# Load packages
library(tidyverse)
library(janitor)

# Load data
tuna_data <- read_csv("data/raw/CatchByFlagGear/CatchByFlagGear1918-2023.csv") |>
  # Clean column names
  clean_names() |>
  # Rename some columns
  rename(year = ano_year,
         flag = bandera_flag,
         gear = arte_gear,
         species = especies_species,
         catch = t)
```

Rows: 13595 Columns: 5

-- Column specification -----

Delimiter: ","

chr (3): BanderaFlag, ArteGear, EspeciesSpecies

dbl (2): AnoYear, t

i Use `spec()` to retrieve the full column specification for this data.

i Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

```
# Check column names
colnames(tuna_data)
```

```
[1] "year"     "flag"      "gear"      "species"   "catch"
```

Congratulations, you now have a tidy data set with which we can work! The next steps are to keep the data we care about, calculate revenues, and then calculate summaries. Let's do that.

### Part D) Filtering rows with `filter()`

#### 0. Put your post-it up

1. What are the unique species represented in the data?
2. What are the fishing gears represented in the data?
3. Does the documentation say what these codes are?<sup>4</sup>
4. What is the species code for bigeye tuna?
5. What is the gear code for purse seine?
6. Create a new object called `ps_tuna_data` that takes the `tuna_data` and filters it to retain data for:

<sup>4</sup>Hint: There is a cryptic link to the [reference codes](#)

- a. bigeye tuna
- b. caught by tuna purse seiners
- c. since 2000

**7. Remove your post-it when you are done**

```
# Check unique values for the species column (using a pipe)
tuna_data$species |> unique()

[1] "SKJ"  "YFT"  "SWO"  "BZX"  "ALB"  "PBF"  "BET"  "BLM"  "BUM"  "MLS"  "CGX"  "MZZ"
[13] "BKJ"  "BIL"  "SKH"  "TUN"  "DOX"  "SFA"  "SSP"  "SRX"  "BXQ"

# Check unique values for the gear colum, without a pipe
unique(tuna_data$gear)

[1] "LP"   "PS"   "UNK"  "HAR"  "LTL"  "RG"   "LL"   "GN"   "OTR"  "LHP"  "TX"

# Create a new data set called ps_tuna_data after filtering
ps_tuna_data <- tuna_data |>
  filter(species == "BET",
         gear == "PS",
         year >= 2000)
```

### Part E) Creating new columns with `mutate()`

**0. Put your post-it up**

1. Modify the `ps_tuna_data` pipeline to create a new column called `revenue` that calculates the revenue generated by selling the catch<sup>5</sup>
2. Make sure you calculate revenues in Millions of USD
- 3. Remove your post-it when you are done**

```
# Create a new data set called ps_tuna_data after filtering
ps_tuna_data <- tuna_data |>
  filter(species == "BET", # Retain BET values only
         gear == "PS", # Retain PS values only
         year >= 2000) |> # Retain data from 2000 onwards
  mutate(revenue = catch * 1000 * 2 / 1e6) # Calculate revenue
```

### Part F) Calculating group summaries with `group_by()` and `summarize()`

**0. Put your post-it up**

1. The data right now report catch at the year-by-flag level. Modify the `ps_tuna_data` pipeline so that we have total catch and revenue by year.<sup>6</sup>
- 2. Remove your post-it when you are done**

---

<sup>5</sup>Hint: If I catch 10 kilos and the price per kilo is US\$2, then I make US\$20 because  $10 * 2 = 20$

<sup>6</sup>Hint: You will need to use the `group_by`, `summarize()`, and `sum()` functions.

```

ps_tuna_data <- tuna_data |>
  filter(species == "BET", # Retain BET values only
         gear == "PS",      # Retain PS values only
         year >= 2000) |> # Retain data from 2000 onwards
  mutate(revenue = catch * 1000 * 2 / 1e6) |> # Calculate revenue
  group_by(year) |>          # Specify that I am grouping by year
# Tell summarize that I want to collapse the catch column by summing all its values
  summarize(catch = sum(catch),
            revenue = sum(revenue)) # Same, but for revenues

```

**!** Important

During class we only calculated total revenue. The above code calculates total revenue AND total catch.

## Part G) Visualize the data and answer the question

Remember, the question was:

How much money have tuna purse seiners made since 2000 when fishing for bigeye tuna (*Thunnus obesus*)?

The question is ambiguous because one could answer “They have made X M USD since 2000” or “Every year since 2000, they have made Y M USD per year.” So let’s get both:

0. **Put your post-it up**
1. What is the total revenue?<sup>7</sup>
2. What is the average annual revenue?<sup>8</sup>
3. Build a time-series showing revenues by year. Make sure to correctly label the axis and include a title and caption describing the figure and the data source, respectively.
4. **Remove your post-it when you are done.**

```
# Get total revenue
sum(ps_tuna_data$revenue)
```

[1] 3070.97

```
# Get mean annual revenue
mean(ps_tuna_data$revenue)
```

[1] 127.9571

```
# Build plot
ggplot(data = ps_tuna_data,           # Specify my data
```

---

<sup>7</sup>Hint: Use \$ and sum()

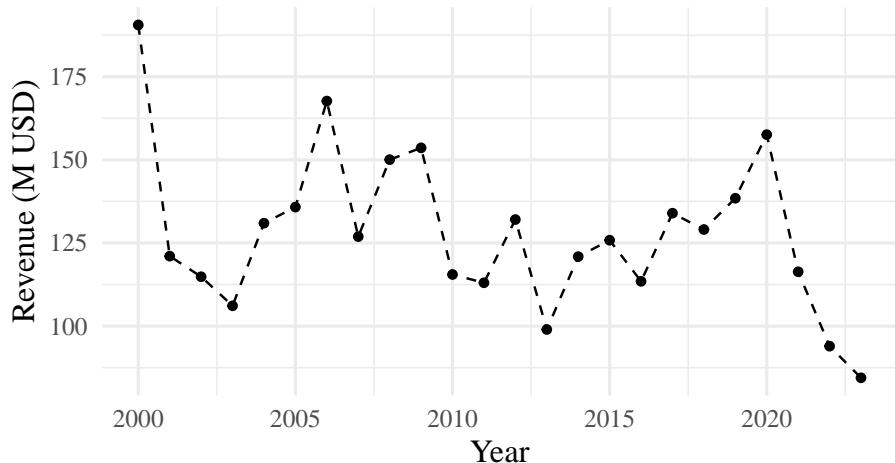
<sup>8</sup>Hint: Use \$ and mean()

```

mapping = aes(x = year, y = revenue)) + # And my aesthetics
geom_line(linetype = "dashed") +          # Add a dashed line
geom_point() +                           # With points on top
labs(x = "Year",                         # Add some labels
y = "Revenue (M USD)",                  # Add some labels
title = "Annual revenue from fishing bigeye tuna by purse seine vessels",
caption = "Data come from the IATTC") +
# Modify the theme
theme_minimal(base_size = 14,           # Font size 14
base_family = "Times")                 # Font family Times

```

Annual revenue from fishing bigeye tuna by purse seine vessels



Data come from the IATTC

## Extra exercises for you to practice

1. Make a figure showing total catch by species during 2023

```

# Build a new data.frame that has catch by species (in thousand tons)
catch_2023 <- tuna_data |>
  filter(year == 2023) |>
  group_by(species) |>
  summarize(total_catch = sum(catch) / 1e3)

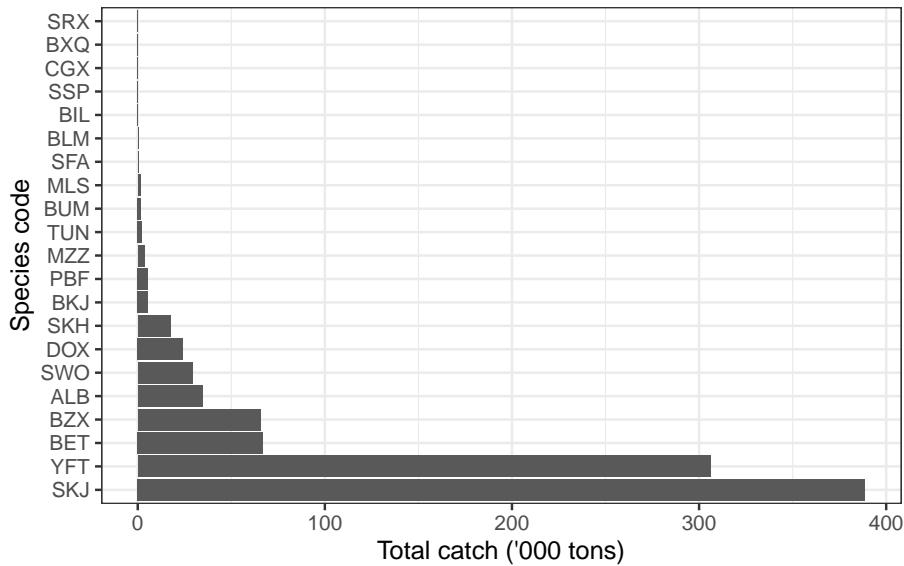
# Now build the figure
ggplot(data = catch_2023,
       aes(x = fct_reorder(species, total_catch, .desc = T), # I am using this fct_reorder to
            y = total_catch)) +
  geom_col()

```

```

  labs(x = "Species code",
       y = "Total catch ('000 tons)") +
  coord_flip() +
  theme_bw()

```



2. Which country caught the most tuna during 2020?

```

# There is more than one way to do this one
# Option 1

# Build a data that has total catch by country in 2020
countryCatch <- tunaData |>
  filter(year == 2020) |> # Retain only data from 2020
  group_by(flag) |> # Get total catch by flag (i.e. sum catch across all species)
  summarize(totalCatch = sum(catch))

# And then we can use brackets, dollar signs, and boolean operators to extract the flag
countryCatch$flag[countryCatch$totalCatch == max(countryCatch$totalCatch)]

[1] "ECU"

# Option2: The way I haven't shown you
tunaData |>
  filter(year == 2020) |>
  group_by(flag) |>
  summarize(totalCatch = sum(catch)) |> # Up until here, the pipeline is the same
  arrange(desc(totalCatch)) |> # Then I use the arrange function to sort the data in descending order
  head(1) |> # I then retain only the first row, which now _should_ contain the data I want

```

```

  pull(flag) # This is a "tidy" version of using a dollar sign to extract a column

[1] "ECU"

  3. For each species, identify the year in which catch was at it's maximum

# Option 1: With what you already know
tuna_data |>
  group_by(year, species) |>
  summarize(total_catch = sum(catch)) |> # We firrst calculate total catch by species and ye
  group_by(species) |> # Then we group by species
  filter(total_catch == max(total_catch)) |> # And use the filter function. Since the data a
  select(species, year_of_max_catch = year) |> # And we keep the columns we care about
  arrange(species)

`summarise()` has grouped output by 'year'. You can override using the
`.groups` argument.

# A tibble: 21 x 2
# Groups:   species [21]
  species year_of_max_catch
  <chr>        <dbl>
1 ALB            2014
2 BET            2000
3 BIL            2013
4 BKJ            2016
5 BLM            1973
6 BUM            1963
7 BXQ            2019
8 BZX            2023
9 CGX            1983
10 DOX           2009
# i 11 more rows

# Option 2: Using slice_max
tuna_data |>
  group_by(year, species) |>
  summarize(total_catch = sum(catch)) |>
  group_by(species) |>
  slice_max(total_catch) |>
  select(species, year_of_max_catch = year) |>
  arrange(species)

`summarise()` has grouped output by 'year'. You can override using the
`.groups` argument.

# A tibble: 21 x 2
# Groups:   species [21]

```

```

species year_of_max_catch
<chr> <dbl>
1 ALB 2014
2 BET 2000
3 BIL 2013
4 BKJ 2016
5 BLM 1973
6 BUM 1963
7 BXQ 2019
8 BZX 2023
9 CGX 1983
10 DOX 2009
# i 11 more rows

4. How many species have been caught by Mexican-flagged vessels since
2000?

# Option 1: A pipeline that ends in a vector, with unique and length
# Start from tuna_data
tuna_data |>
  filter(flag == "MEX") |> # Retain observations associated with Mexico
  pull(species) |> # Pull the species column away from the data.frame, at this point we have
  unique() |> # Get a unique list of species
  length() # Count the number of unique species

[1] 21

# Alternatively, retain the data.frame structure
tuna_data |>
  filter(flag == "MEX") |> # Retain observations associated with Mexico
  group_by(flag) |>
  summarize(n_species = n_distinct(species)) # The n_distinct() function is a tidy version of

# A tibble: 1 x 2
  flag   n_species
  <chr>     <int>
1 MEX        21

```

Sibert, John, Inna Senina, Patrick Lehodey, and John Hampton. 2012. “Shifting from Marine Reserves to Maritime Zoning for Conservation of Pacific Bigeye Tuna (*Thunnus obesus*).” *Proc. Natl. Acad. Sci. U. S. A.* 109 (October): 18221–25.