# Project 2: Data Transformation

Group Members: Alina Vikhnevich, Olivia Azevedo, Alyssa Gurkas, Musrat Jahan

2025-03-09

## Contents



#### 0.0.1 Loading libraries:

```
library(kableExtra)
library(RSocrata)
library(tidyverse)
library(viridis)
library(readr)
library(readxl)
library(janitor)
```

```
library(lubridate)
library(ggplot2)
library(scales)
library(stringr)
library(forcats)
```

# 1 Introduction

### 1.0.1 Project Overview

The goal of this project is to tidy, transform, and analyze three different datasets using R, leveraging the `tidyverse`, `tidyr`, and `dplyr` packages. These datasets, originally in an untidy "wide" format, require cleaning, restructuring, and standardization before analysis can be performed.

By the end of this project, we will:

- Convert three untidy datasets into a structured format for analysis.
- Perform data wrangling using `tidyr` and `dplyr` to clean and reshape the data.
- Conduct exploratory data analysis (EDA) to uncover insights and trends.
- Document the transformation process and provide meaningful conclusions.

This project is a collaborative effort, and each dataset presents a unique challenge in terms of data cleaning, structuring, and interpretation. The final results will be published as an R Markdown report, demonstrating the power of data transformation techniques.

### 1.0.2 Overview of Datasets

Each dataset represents a different domain and requires a unique transformation approach. Below is a summary of the datasets used in this project:

1. **Dataset #1: Emissions Data**

- **Description**: This dataset provides information on pollutant emissions over multiple years. The data includes various emission sources and their impact over time.
- **Data Issues**: The dataset is in wide format, with emissions spread across multiple columns by year.
- **Transformation Steps**: We will convert it into a long format, making it easier to analyze trends over time.

2. **Dataset #2: New York State Department of Environmental Conservation's Application Review & Tracking System from 2020-2025 (DART)**

- **Description**: This dataset contains public about environment permits issued by New York State's Department of Environmental Conservation. This report explores water permits, regulated under the National Pollutant Discharge Elimination System (NPDES).
- **Data Issues**: The dataset is not normalized, and some entries are duplicated.
- **Transformation Steps**: We will standardize date formats, remove redundant data, and ensure consistency across permit records.

3. **Dataset #3: Cheese Nutritional Data**

- **Description**: This dataset provides nutritional information on various types of cheese.
- **Data Issues**: The dataset is structured as a wide table, making it difficult to compare across different cheese types.
- **Transformation Steps**: We will reshape the data into a long format, making it easier to compare nutritional values across different cheese varieties.

#### 1.0.3 Relevance of These Datasets

Each dataset requires different data transformation techniques, making them ideal for practicing `tidyr` and `dplyr` functions. The common themes across these datasets include:

- Converting wide-format data into long format.
- Standardizing date and time fields.
- Handling missing values and duplicates.
- Preparing the data for downstream statistical analysis and visualization.

## 1.1 By applying tidy data principles, we ensure that each dataset is structured, organized, and ready for analysis. The insights gained from this project can be used for policy recommendations, water quality analysis, and compliance to the Clean Water Act.

# 2 Data Preparation and Cleaning

# 3 Emissions Data:

```
df <- read.csv("https://raw.githubusercontent.com/justin-2028/Total-Emissions-Per-Countr

colnames(df) <- gsub("^X", "", colnames(df))

print(head(df))
```

```
##           Area             Item                              Element       Unit
## 1 Afghanistan    Crop Residues           Direct emissions (N2O) kilotonnes
## 2 Afghanistan    Crop Residues         Indirect emissions (N2O) kilotonnes
## 3 Afghanistan    Crop Residues                  Emissions (N2O) kilotonnes
## 4 Afghanistan    Crop Residues Emissions (CO2eq) from N2O (AR5) kilotonnes
## 5 Afghanistan    Crop Residues          Emissions (CO2eq) (AR5) kilotonnes
## 6 Afghanistan Rice Cultivation                  Emissions (CH4) kilotonnes
##       2000     2001     2002     2003     2004     2005     2006     2007
## 1    0.520   0.5267   0.8200   0.9988   0.8225   1.1821   1.0277   1.2426
## 2    0.117   0.1185   0.1845   0.2247   0.1851   0.2660   0.2312   0.2796
## 3    0.637   0.6452   1.0045   1.2235   1.0075   1.4481   1.2589   1.5222
## 4  168.807 170.9884 266.1975 324.2195 266.9995 383.7498 333.6093 403.3749
## 5  168.807 170.9884 266.1975 324.2195 266.9995 383.7498 333.6093 403.3749
## 6   18.200  16.9400  18.9000  20.3000  27.3000  22.4000  22.4000  23.8000
##       2008     2009     2010     2011     2012     2013     2014     2015
## 1   0.8869   1.3920   1.2742   1.0321   1.3726   1.4018   1.4584   1.2424
## 2   0.1996   0.3132   0.2867   0.2322   0.3088   0.3154   0.3281   0.2795
## 3   1.0865   1.7051   1.5609   1.2643   1.6815   1.7173   1.7865   1.5220
## 4 287.9099 451.8647 413.6467 335.0379 445.5958 455.0727 473.4174 403.3181
## 5 287.9099 451.8647 413.6467 335.0379 445.5958 455.0727 473.4174 403.3181
## 6  26.6000  28.0000  29.1200  29.4000  28.7000  28.7000  30.8000  22.9600
##       2016     2017     2018     2019     2020
## 1   1.1940   1.0617   0.8988   1.2176   1.3170
## 2   0.2687   0.2389   0.2022   0.2740   0.2963
## 3   1.4627   1.3005   1.1011   1.4916   1.6133
## 4 387.6130 344.6447 291.7838 395.2689 427.5284
## 5 387.6130 344.6447 291.7838 395.2689 427.5284
## 6  16.6600  15.3233  16.4555  17.8542  20.6577
```

**3.0.0.1 Make longer** All the year columns were changed to one column under year. The dataset was made longer. This makes it tidy.

```r
df_longer <- df |>
  pivot_longer(
    cols = starts_with("2"),
    names_to = "year",
    values_to = "total emissions",
    values_drop_na = TRUE
  )
print(head(df))
```

```
##          Area           Item                           Element       Unit
## 1 Afghanistan    Crop Residues          Direct emissions (N2O) kilotonnes
## 2 Afghanistan    Crop Residues        Indirect emissions (N2O) kilotonnes
## 3 Afghanistan    Crop Residues                 Emissions (N2O) kilotonnes
## 4 Afghanistan    Crop Residues Emissions (CO2eq) from N2O (AR5) kilotonnes
## 5 Afghanistan    Crop Residues          Emissions (CO2eq) (AR5) kilotonnes
## 6 Afghanistan Rice Cultivation                 Emissions (CH4) kilotonnes
##      2000     2001     2002     2003     2004     2005     2006     2007
## 1   0.520   0.5267   0.8200   0.9988   0.8225   1.1821   1.0277   1.2426
## 2   0.117   0.1185   0.1845   0.2247   0.1851   0.2660   0.2312   0.2796
## 3   0.637   0.6452   1.0045   1.2235   1.0075   1.4481   1.2589   1.5222
## 4 168.807 170.9884 266.1975 324.2195 266.9995 383.7498 333.6093 403.3749
## 5 168.807 170.9884 266.1975 324.2195 266.9995 383.7498 333.6093 403.3749
## 6  18.200  16.9400  18.9000  20.3000  27.3000  22.4000  22.4000  23.8000
##        2008     2009     2010     2011     2012     2013     2014     2015
## 1    0.8869   1.3920   1.2742   1.0321   1.3726   1.4018   1.4584   1.2424
## 2    0.1996   0.3132   0.2867   0.2322   0.3088   0.3154   0.3281   0.2795
## 3    1.0865   1.7051   1.5609   1.2643   1.6815   1.7173   1.7865   1.5220
## 4  287.9099 451.8647 413.6467 335.0379 445.5958 455.0727 473.4174 403.3181
## 5  287.9099 451.8647 413.6467 335.0379 445.5958 455.0727 473.4174 403.3181
## 6   26.6000  28.0000  29.1200  29.4000  28.7000  28.7000  30.8000  22.9600
##        2016     2017     2018     2019     2020
## 1    1.1940   1.0617   0.8988   1.2176   1.3170
## 2    0.2687   0.2389   0.2022   0.2740   0.2963
## 3    1.4627   1.3005   1.1011   1.4916   1.6133
## 4  387.6130 344.6447 291.7838 395.2689 427.5284
## 5  387.6130 344.6447 291.7838 395.2689 427.5284
## 6   16.6600  15.3233  16.4555  17.8542  20.6577
```

```r
yearly_emissions_by_area <- aggregate(df_longer$'total emissions', by = list(df_longer$y

yearly_emissions_by_area
```

```
#rename columns
yearly_emissions_by_area <-
yearly_emissions_by_area %>%
  rename(
    year = Group.1,
    country = Group.2,
    emissions = x
    )

yearly_emissions_by_area

print(head(df))
```

#### 3.0.0.2 Total emissions per country for each year

#### 3.0.0.3 Analyze overall total emissions per country for each year

Too many different countries

```
ggplot(yearly_emissions_by_area, aes(x = year, y = emissions,
                                      fill = country)) +
  geom_tile()
```


```r
print(head(df))
```

```
##          Area          Item                          Element       Unit
## 1 Afghanistan    Crop Residues           Direct emissions (N2O) kilotonnes
## 2 Afghanistan    Crop Residues         Indirect emissions (N2O) kilotonnes
## 3 Afghanistan    Crop Residues                  Emissions (N2O) kilotonnes
## 4 Afghanistan    Crop Residues Emissions (CO2eq) from N2O (AR5) kilotonnes
## 5 Afghanistan    Crop Residues          Emissions (CO2eq) (AR5) kilotonnes
## 6 Afghanistan Rice Cultivation                  Emissions (CH4) kilotonnes
##        2000     2001     2002     2003     2004     2005     2006     2007
## 1    0.520   0.5267   0.8200   0.9988   0.8225   1.1821   1.0277   1.2426
## 2    0.117   0.1185   0.1845   0.2247   0.1851   0.2660   0.2312   0.2796
## 3    0.637   0.6452   1.0045   1.2235   1.0075   1.4481   1.2589   1.5222
## 4  168.807 170.9884 266.1975 324.2195 266.9995 383.7498 333.6093 403.3749
## 5  168.807 170.9884 266.1975 324.2195 266.9995 383.7498 333.6093 403.3749
## 6   18.200  16.9400  18.9000  20.3000  27.3000  22.4000  22.4000  23.8000
##         2008     2009     2010     2011     2012     2013     2014     2015
## 1     0.8869   1.3920   1.2742   1.0321   1.3726   1.4018   1.4584   1.2424
## 2     0.1996   0.3132   0.2867   0.2322   0.3088   0.3154   0.3281   0.2795
## 3     1.0865   1.7051   1.5609   1.2643   1.6815   1.7173   1.7865   1.5220
## 4   287.9099 451.8647 413.6467 335.0379 445.5958 455.0727 473.4174 403.3181
```

```
## 5 287.9099 451.8647 413.6467 335.0379 445.5958 455.0727 473.4174 403.3181
## 6  26.6000  28.0000  29.1200  29.4000  28.7000  28.7000  30.8000  22.9600
##        2016     2017     2018     2019     2020
## 1   1.1940   1.0617   0.8988   1.2176   1.3170
## 2   0.2687   0.2389   0.2022   0.2740   0.2963
## 3   1.4627   1.3005   1.1011   1.4916   1.6133
## 4 387.6130 344.6447 291.7838 395.2689 427.5284
## 5 387.6130 344.6447 291.7838 395.2689 427.5284
## 6  16.6600  15.3233  16.4555  17.8542  20.6577
```

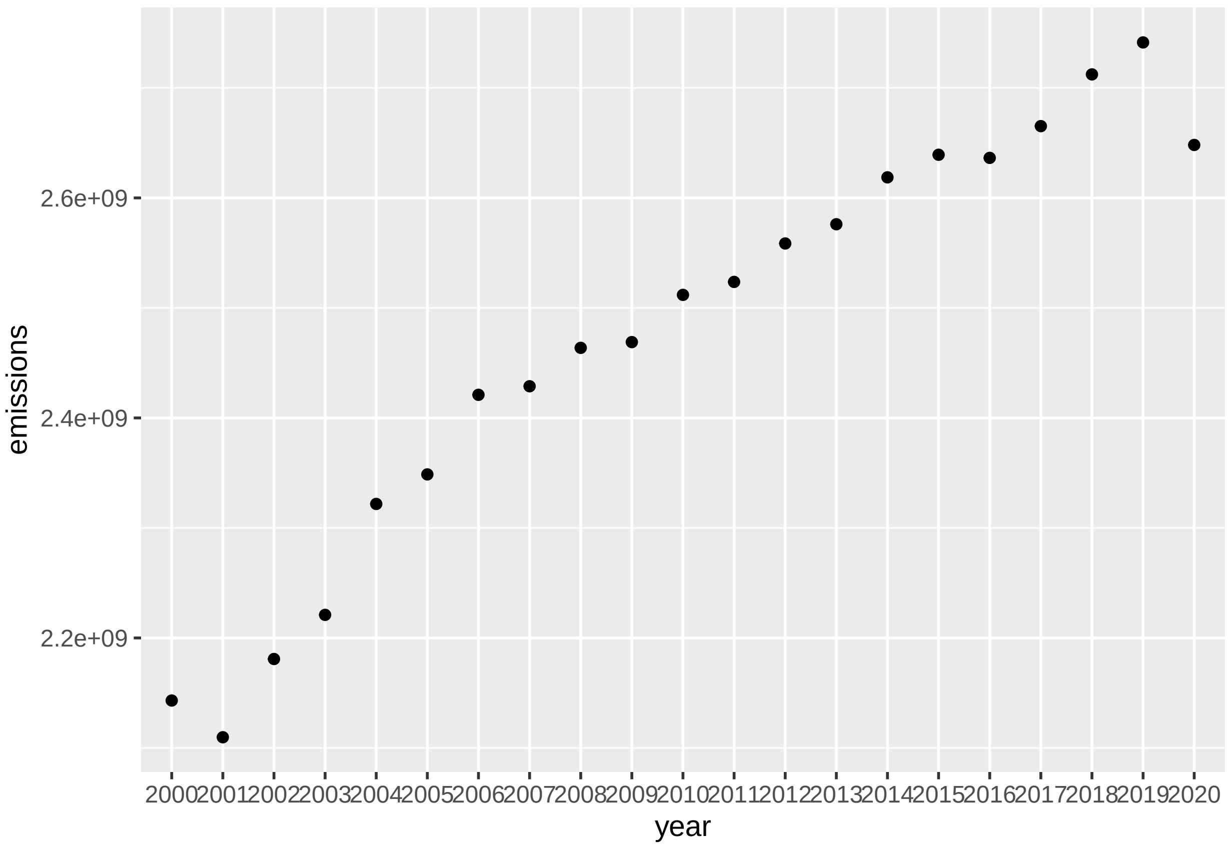
**3.0.0.4 Analyze total emissions over time** As you can see from the graph, total emissions have gone up steadily from 2000 to 2019, but in 2020, it decreased a significant amount. This might be due to more awareness about climate change and global warming.

```r
yearly_emissions <- aggregate(df_longer$'total emissions', by=list(df_longer$year), FUN
yearly_emissions
```

```
##    Group.1          x
## 1     2000 2143119367
## 2     2001 2109760240
## 3     2002 2180860383
## 4     2003 2220985897
## 5     2004 2321819994
## 6     2005 2348671376
## 7     2006 2421011446
## 8     2007 2428782054
## 9     2008 2463686673
## 10    2009 2468939225
## 11    2010 2511864950
## 12    2011 2523633556
## 13    2012 2558572795
## 14    2013 2576071615
## 15    2014 2618685489
## 16    2015 2639203455
## 17    2016 2636367158
## 18    2017 2665248135
## 19    2018 2712258358
## 20    2019 2741323659
## 21    2020 2648131930
```

```r
#rename columns
yearly_emissions <-
yearly_emissions %>%
```

```
  rename(
    year = Group.1,
    emissions = x
    )

ggplot(yearly_emissions, aes(x = year, y = emissions)) +
  geom_point()
```




**3.0.0.5 Total emissions per country** Some of the top countries that contributed to emissions are China, USA, Brazil, India, Indonesia, and Democratic Republic of the Congo.

```
emissions_by_area <- aggregate(df_longer$'total emissions', by = list(df_longer$Area), F

#rename columns
emissions_by_area <-
emissions_by_area %>%
  rename(
    country = Group.1,
    emissions = x
    )
```

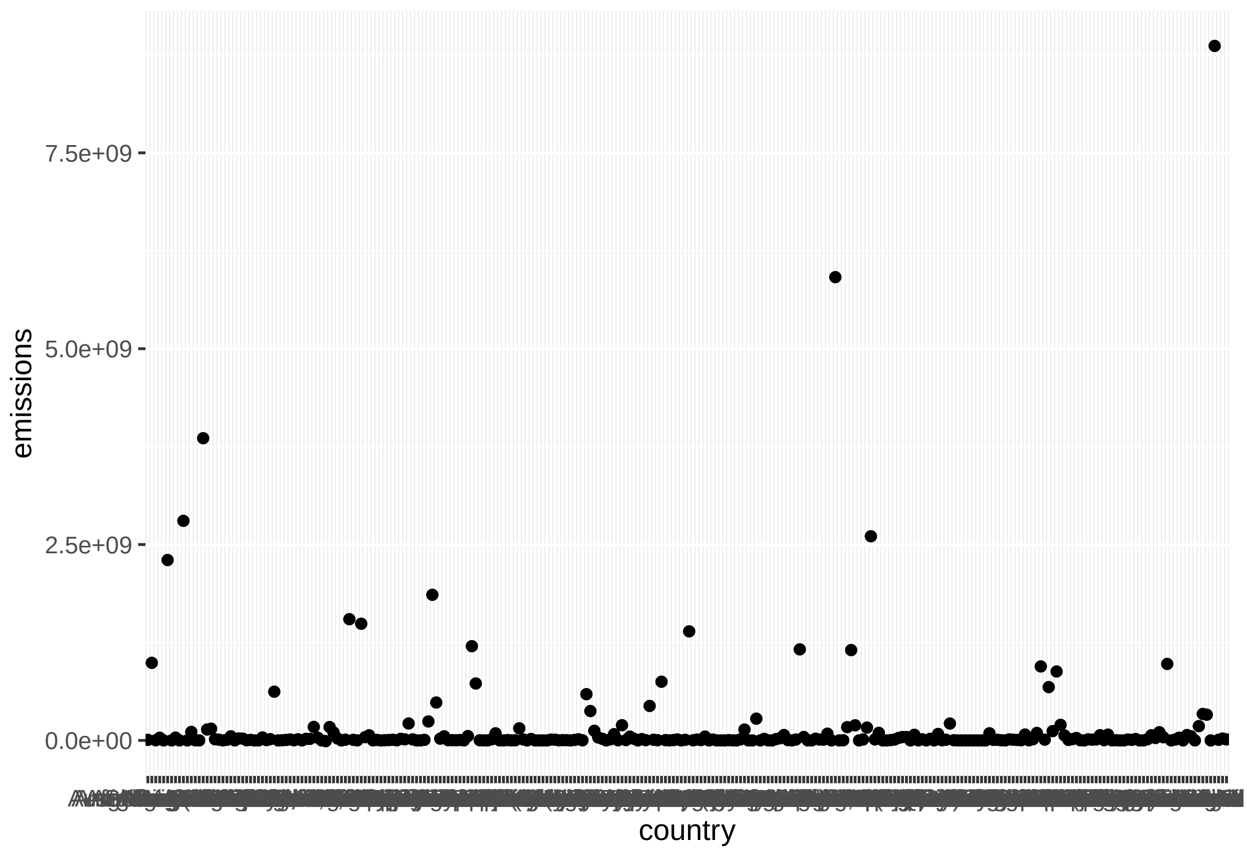
```r
top <- emissions_by_area[order(-emissions_by_area$emissions),]

print(head(df))
```

```
##          Area            Item                          Element      Unit
## 1 Afghanistan    Crop Residues          Direct emissions (N2O) kilotonnes
## 2 Afghanistan    Crop Residues        Indirect emissions (N2O) kilotonnes
## 3 Afghanistan    Crop Residues                 Emissions (N2O) kilotonnes
## 4 Afghanistan    Crop Residues Emissions (CO2eq) from N2O (AR5) kilotonnes
## 5 Afghanistan    Crop Residues          Emissions (CO2eq) (AR5) kilotonnes
## 6 Afghanistan Rice Cultivation                 Emissions (CH4) kilotonnes
##      2000     2001     2002     2003     2004     2005     2006     2007
## 1   0.520   0.5267   0.8200   0.9988   0.8225   1.1821   1.0277   1.2426
## 2   0.117   0.1185   0.1845   0.2247   0.1851   0.2660   0.2312   0.2796
## 3   0.637   0.6452   1.0045   1.2235   1.0075   1.4481   1.2589   1.5222
## 4 168.807 170.9884 266.1975 324.2195 266.9995 383.7498 333.6093 403.3749
## 5 168.807 170.9884 266.1975 324.2195 266.9995 383.7498 333.6093 403.3749
## 6  18.200  16.9400  18.9000  20.3000  27.3000  22.4000  22.4000  23.8000
##       2008     2009     2010     2011     2012     2013     2014     2015
## 1   0.8869   1.3920   1.2742   1.0321   1.3726   1.4018   1.4584   1.2424
## 2   0.1996   0.3132   0.2867   0.2322   0.3088   0.3154   0.3281   0.2795
## 3   1.0865   1.7051   1.5609   1.2643   1.6815   1.7173   1.7865   1.5220
## 4 287.9099 451.8647 413.6467 335.0379 445.5958 455.0727 473.4174 403.3181
## 5 287.9099 451.8647 413.6467 335.0379 445.5958 455.0727 473.4174 403.3181
## 6  26.6000  28.0000  29.1200  29.4000  28.7000  28.7000  30.8000  22.9600
##       2016     2017     2018     2019     2020
## 1   1.1940   1.0617   0.8988   1.2176   1.3170
## 2   0.2687   0.2389   0.2022   0.2740   0.2963
## 3   1.4627   1.3005   1.1011   1.4916   1.6133
## 4 387.6130 344.6447 291.7838 395.2689 427.5284
## 5 387.6130 344.6447 291.7838 395.2689 427.5284
## 6  16.6600  15.3233  16.4555  17.8542  20.6577
```

#### 3.0.0.6 Analysis of Total emissions per country

too many countries, cant read

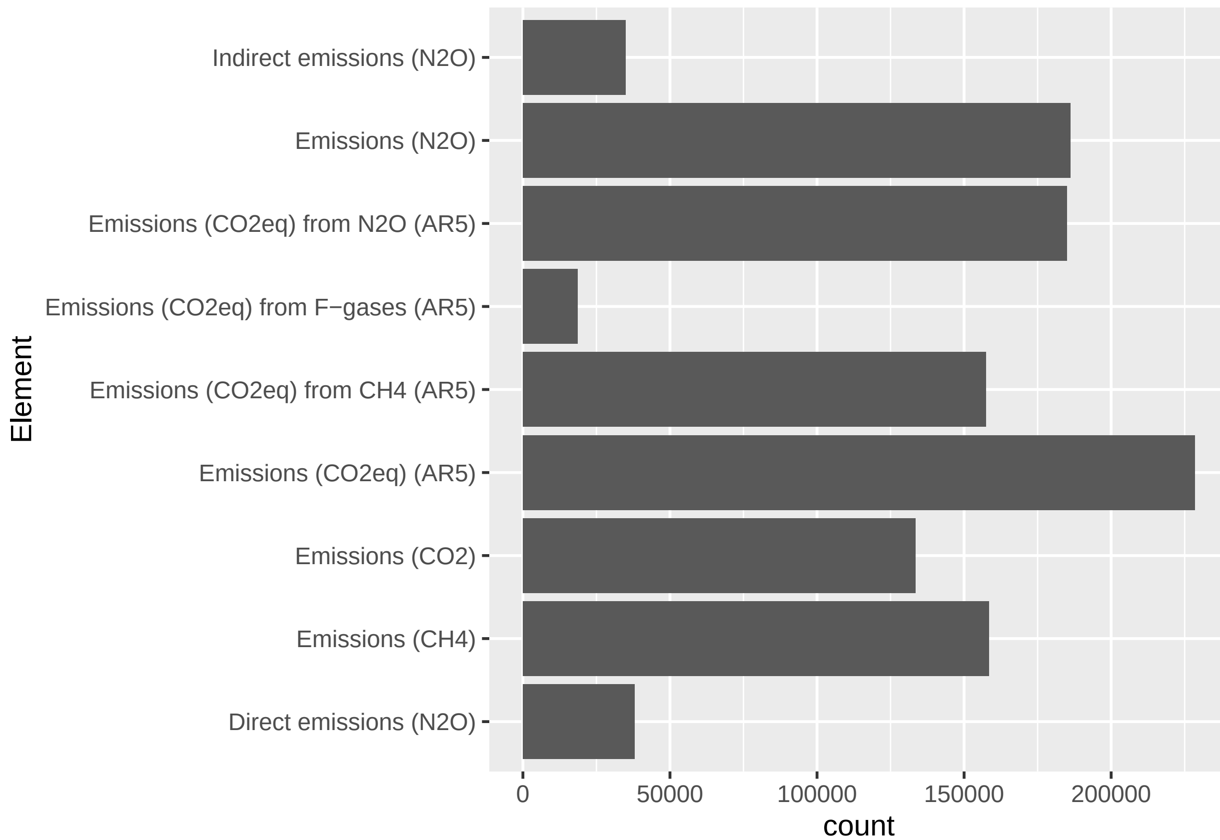
```r
ggplot(emissions_by_area, aes(x = country, y = emissions), label=NA)+ geom_point()
```




#### 3.0.0.7 Aanalyze by emission type

Emissions (CO2eq) (AR5) are highest. They are over 200,000 kilotonnes. The second highest place is tied with emissions (N20) and emissions (CO2eq) from N20 (AR5). Lowest emissions are (CO2eq) from F-gases, less than 25,000 kilotonnes.

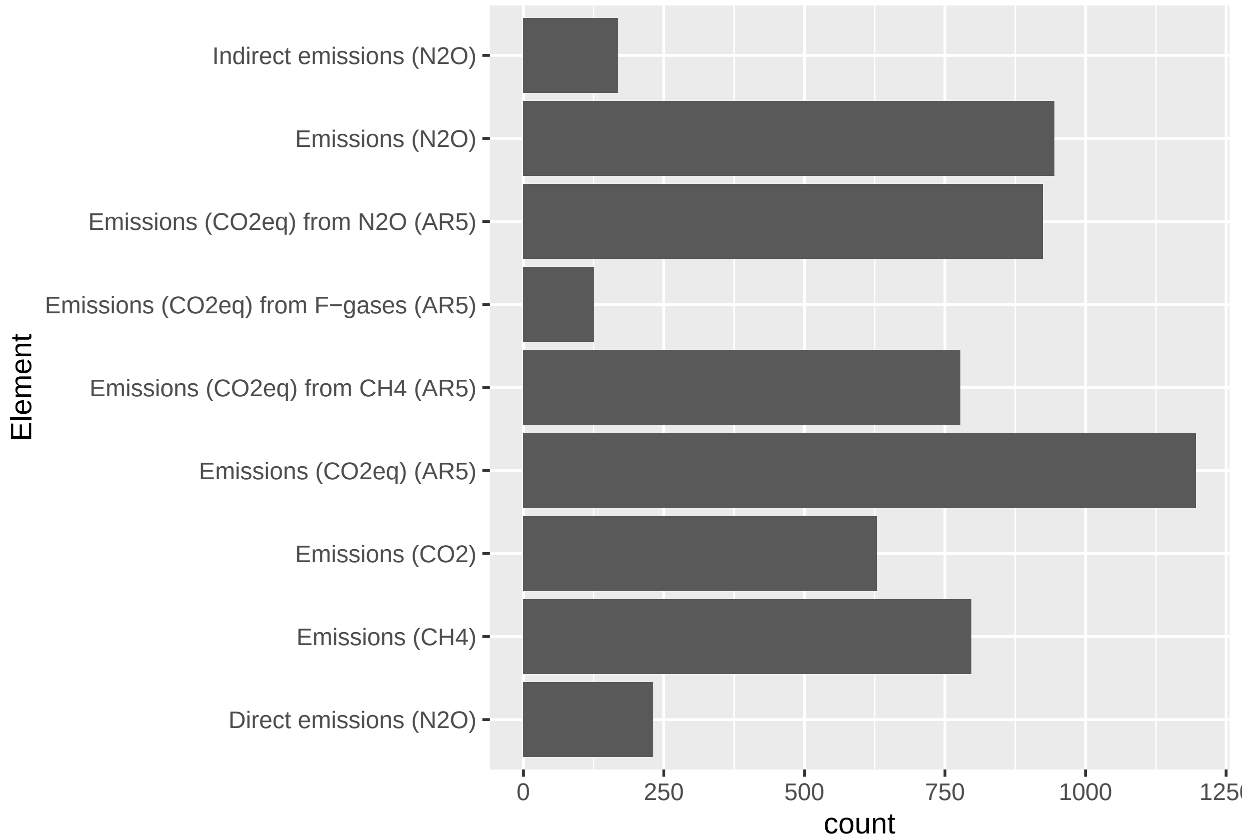
```r
ggplot(df_longer, aes(y=Element)) +
        geom_bar()
```




#### 3.0.0.8 USA emission types distribution

The distribution looks very similar to the distribution with the data from all the regions. For USA, the counts are smaller. Highest are emissions (CO2eq) (AR5), a little less than 1250 kilotonnes. Lowest emissions are (CO2eq) from F-gases, around 125 kilotonnes. The second highest place is from emissions (N20), around 950 kilotonnes.

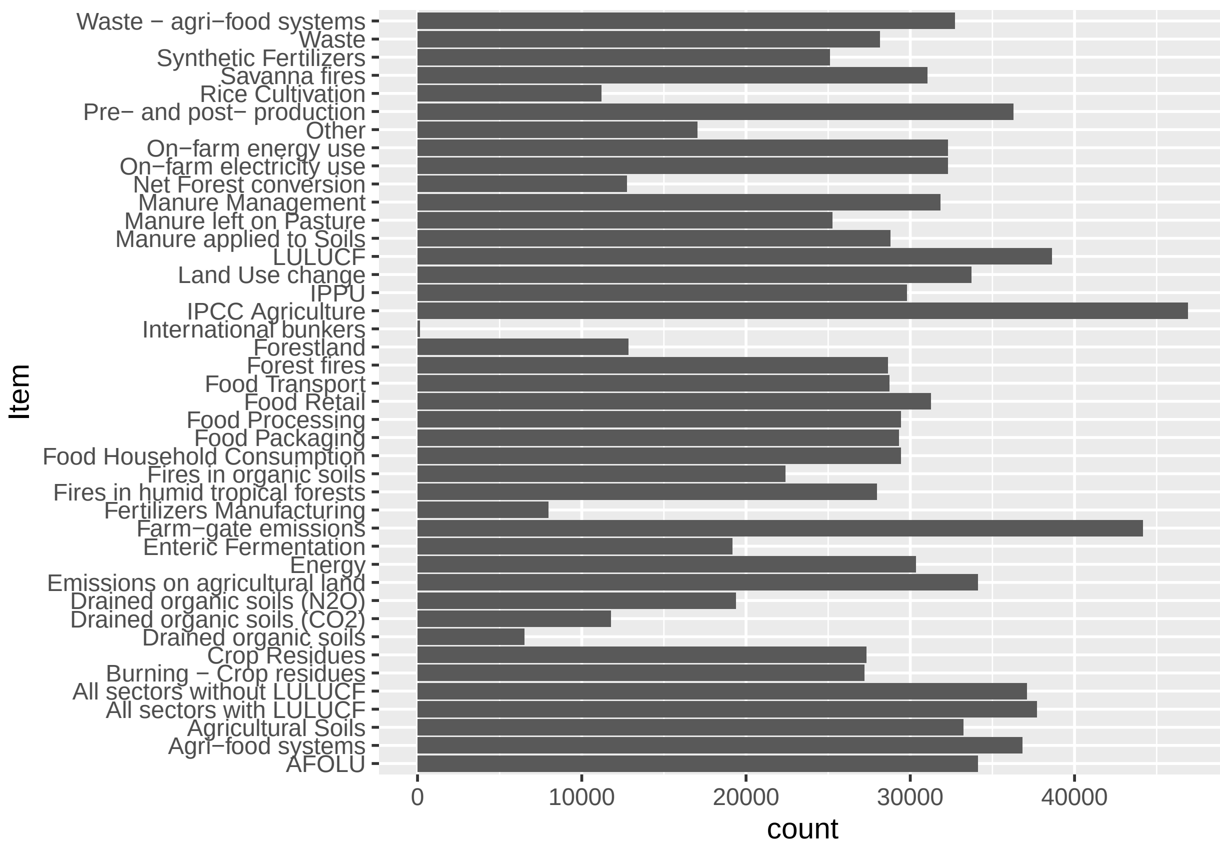
```
usa <- df_longer %>%
  filter(Area == "United States of America")

ggplot(usa, aes(y=Element)) +
         geom_bar()
```




#### 3.0.0.9 Item Analysis

Highest item is IPCC Agriculture. Second highest is farm gate emissions. Lowest is international bunkers.

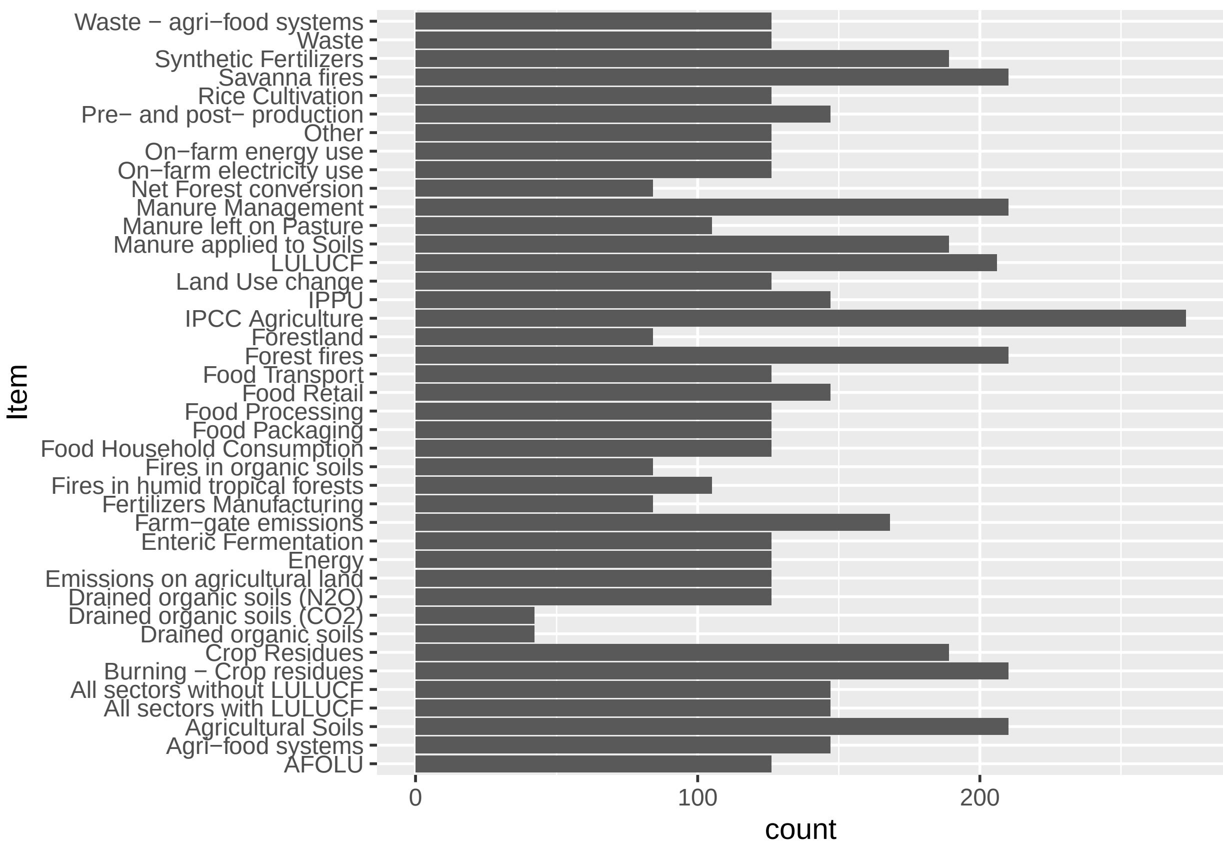
```
ggplot(df_longer, aes(y=Item)) +
         geom_bar()
```




#### 3.0.0.10 Item Analysis USA

Highest item is IPCC Agriculture, just like in the overall data. Lowest is drained organic soils (C02) and drained organic soils.

```r
ggplot(usa, aes(y=Item)) +
        geom_bar()
```



Waste − agri−food systems
Waste
Synthetic Fertilizers
Savanna fires
Rice Cultivation
Pre− and post− production
Other
On−farm energy use
On−farm electricity use
Net Forest conversion
Manure Management
Manure left on Pasture
Manure applied to Soils
LULUCF
Land Use change
IPPU
IPCC Agriculture
Forestland
Forest fires
Food Transport
Food Retail
Food Processing
Food Packaging
Food Household Consumption
Fires in organic soils
Fires in humid tropical forests
Fertilizers Manufacturing
Farm−gate emissions
Enteric Fermentation
Energy
Emissions on agricultural land
Drained organic soils (N2O)
Drained organic soils (CO2)
Drained organic soils
Crop Residues
Burning − Crop residues
All sectors without LULUCF
All sectors with LULUCF
Agricultural Soils
Agri−food systems
AFOLU
Item
0
100
200
count

# 4 NYSDEC Water Permit Data (DART):

**4.0.0.1 Loading NYSDEC DART Data** Data used in this section comes from New York State Department of Environmental Conservation's Application Review & Tracking System (DART on the Web).

DART is a web-based application and tracking system that is designed for the general public. DART hosts information about NYSDEC's processing and issuance of environmental permits under the Uniform Procedures Act. The data is updated daily, and more information about the data can be found in the data dictionary.

In this section, data was previously filtered to only include DART entries from 2020-2025, and will be focused on waste water permits that discharge to surface water.

```
library(readr)
dart <- read_csv("https://raw.githubusercontent.com/AlinaVikhnevich/data_607/refs/heads/
```

**4.0.0.2 Defining Regex Patterns to Detect NPDES IDs** To identify wastewater permits, there are three regex patterns to identify:

1. NPDES Permit (meaning a regular permit).
2. General Permit
3. Individual Permit (these are permits that are processed under general permits).

For more information about permit types please see the question "What are the primary differences between a NODES individual permit and a NPDES general permit" under EPA's NPDES Permit Basics Site.

```
# p_type = permit type, in this exercise we are filtering for wastewater permits
p_type <- c("P/C/I SPDES - Surface Discharge",
            "Municipal SPDES - Surface Discharge",
            "Industrial SPDES - Surface Discharge")

# defining the regex patterns for the IDs we want to track
npdes_pattern <- "NY\\d{7}"
gp_pattern <- "GP\\d{7}"
individual_pattern <- "NY[A-Z]\\d{2}[A-Z]\\d{3}"
all_patterns <- paste(npdes_pattern,gp_pattern,individual_pattern, sep="|")
```

**4.0.0.3 Creating the NPDES Universe** Creating the permit universe pulling from NYSDEC's DART System and detecting the string patterns within DART to assign permit type: npdes, individual(i.e., a permit covered under a general permit), general, or multi (meaning the DART entry had multiple associated IDs).

```
universe <- dart |>
  filter(`permit_type` %in% p_type) |>
  mutate(
    npdes = str_count(`other_known_ids`, npdes_pattern), # the str_counts are taking co
    individual = str_count(`other_known_ids`, individual_pattern),
    gp =  str_count(`other_known_ids`, gp_pattern),
    sum_ids = rowSums(across(c(`npdes`, `individual`,`gp`))),
    npdes_id = str_extract_all(`other_known_ids`, all_patterns),
    date_received=as.Date(date_received, format = "%d-%m-%Y")
    ) |>
  mutate(applicant_id =cur_group_id(),.by = applicant) |> # creating applicant id
  mutate(facility_id = cur_group_id(),.by = c(facility,location,town_or_city)) |> # cre
  distinct() |> # removing duplicate rows
  mutate(
        dart_p_type = case_when(sum_ids  > 1 ~ "multi", # if entry is associated with
                              sum_ids & npdes == 1 ~ "npdes",
                              sum_ids & individual == 1 ~ "individual",
                              sum_ids & gp == 1 ~ "gp")) |>
  unnest_longer(npdes_id, keep_empty = FALSE) |>
  filter(!is.na(npdes_id))
```

Note: The code above filters entries that did not have a NPDES ID listed in the "Other Known IDs" column, however, were listed as NPDES permits in the Permit Type Column. However, out of 35,642 entries, only 69 were missing NPDES IDs.

#### 4.0.0.4 Table 1: Permit Level Data

This table shows the most recent permit information

```
tbl1_permit_lvl <- universe  |>
  group_by(npdes_id) |>
  slice(which.max(date_received)) |>
  select(npdes_id,facility_id,application_id,applicant,applicant_id,permit_type,
         status,date_received,upa_class,seqr_class,seqr_determination,
         lead_agency,coastal_zone_status, final_disposition,permit_effective_date,
         permit_expration_date,dec_contact,shpa_status,enivronmental_justice)
```

#### 4.0.0.5 Table 2: Permit Action Level Data

This table shows the permit history. each observation in this table represents a permit action.

```
tbl2_permit_act_lvl <- universe |>
    mutate(action_id = paste(npdes_id,date_received, sep = "_")) |>
    distinct() |>
```

```r
  mutate(dup_flag = duplicated(action_id),
         transfer_flag=str_detect(toupper(short_description),"TRANSFER")) |>
  select(action_id,facility,facility_id,npdes_id,application_id,applicant,
         application_type,date_received,status,short_description,
         enb_publication_date,written_comments_due,dup_flag,transfer_flag)

tbl2_permit_act_lvl$short_description <- tolower(tbl2_permit_act_lvl$short_description)
```

#### 4.0.0.6 Table 3: Facility Level Data

This table shows the facility information. Each observation in this table represents a facility associated with NPDES permits.

```r
tbl3_facility_lvl <- universe |>
  select(facility_id,facility,
         location,town_or_city) |>
  distinct() |>
  arrange(facility_id)
```

#### 4.0.0.7 Table 4: NPDES Permit Applicant Table

This table shows the applicant information. Each observation in this table represents a permit applicant for NPDES permits.

```r
tbl4_app_lvl <- universe |>
  group_by(applicant_id) |>
  slice(which.max(date_received)) |>
  select(applicant_id,applicant,application_id)
```

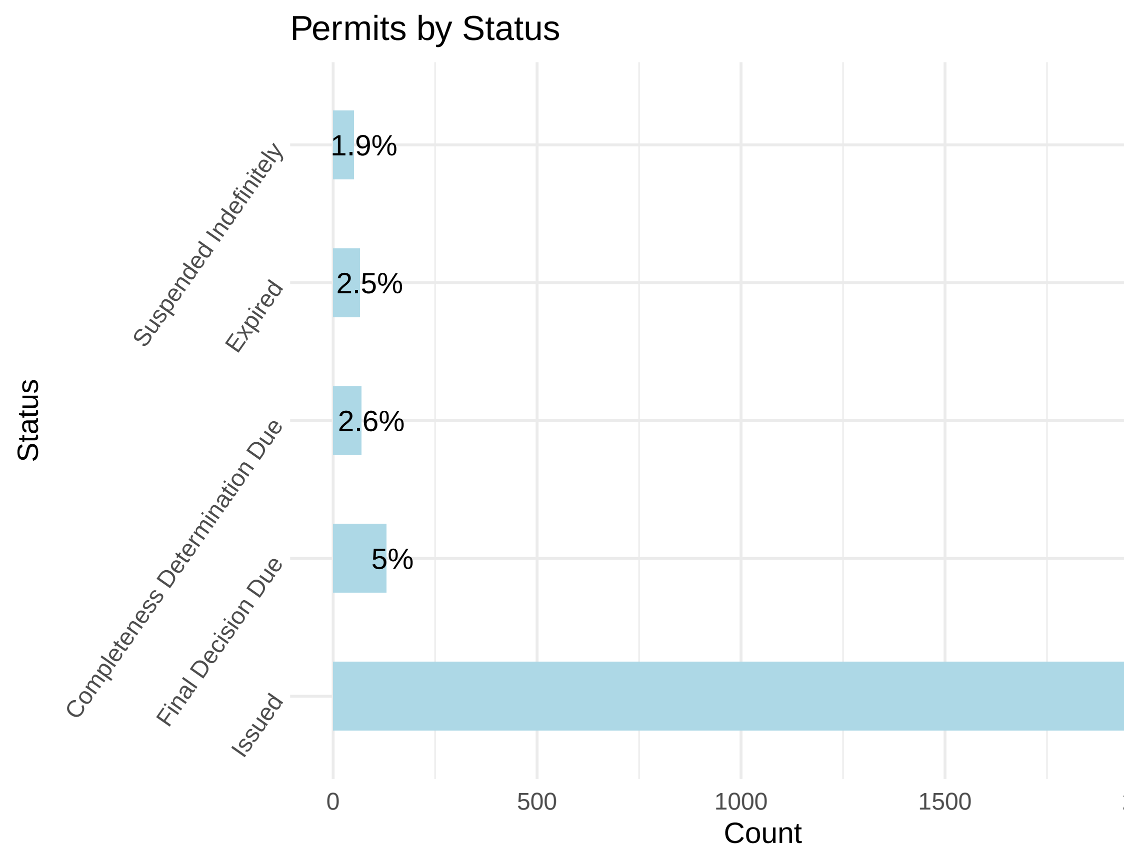
#### 4.0.0.8 Data Tables and Structure

```
(1) Table 1 - permit table: the purpose of this table is to have the most
recent permit information. This will have one row per permit.

(2) Table 2 - permit action table: the purpose of this table is to have a
table with every permit-action. This means there should be one row per
permit action.

(3) Table 3 - facility table: the purpose of this table is to have
information on the facility.

(4) Table 4 - applicant table: the purpose of this table is to have
information about the applicant.
```

#### 4.0.0.9 Data Considerations:

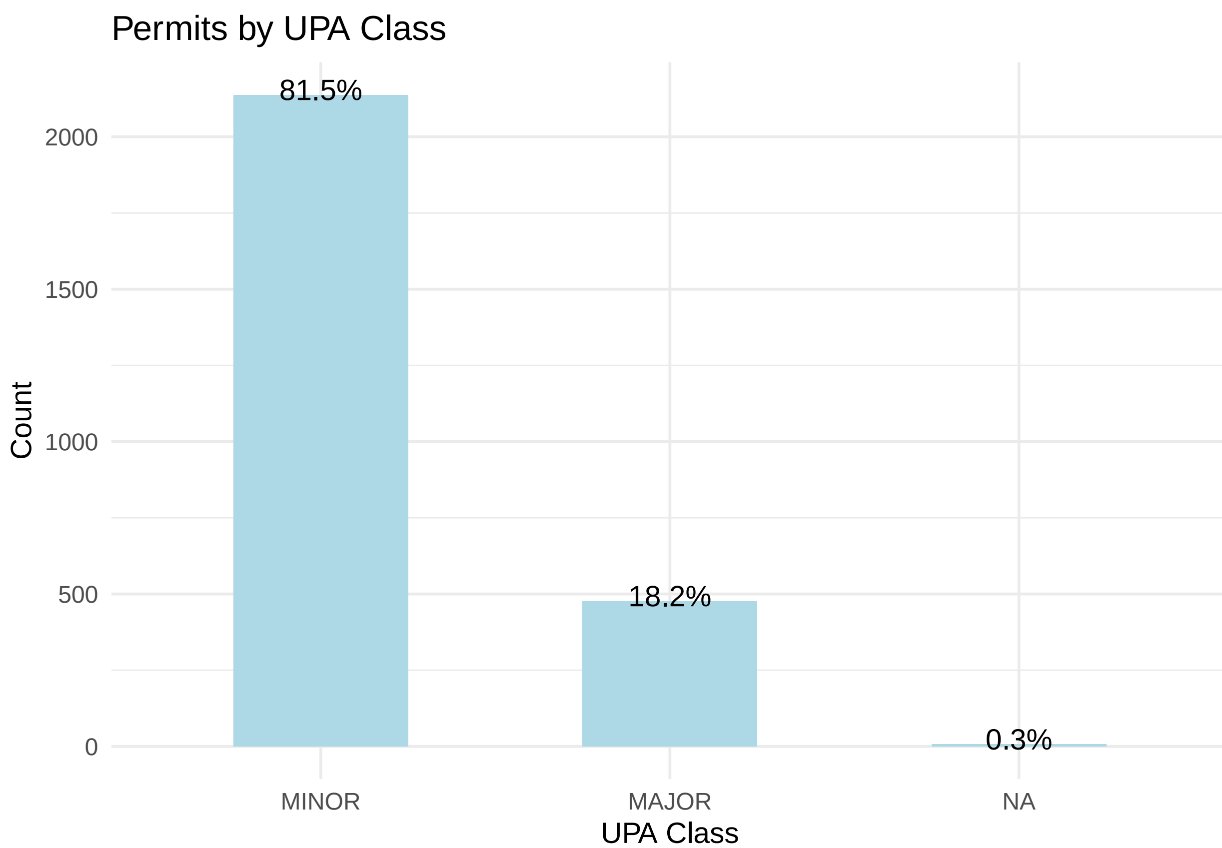
- There was missing data, such as NPDES IDs. This means that some permit information may not be available.
- There may be facilities that are listed as different facilities due to address changes. This information should be verified. Databases like EPA's Enforcement and Compliance History Online (ECHO)) may be helpful for verifying facility information.
- For entries that were made on the same day for a particular permit, it is not possible to identify which entry was made first. Permit transfer actions are largely affected by this. Due to this, duplicates and transfers are flagged for manual review.

```
permit_status <- tbl1_permit_lvl |>
group_by(status) |>
  summarize(
    Count = n(),
    Proportion = (n()/nrow(tbl1_permit_lvl))*100
  ) |>
  arrange(desc(Proportion)) |>
  head(5) |>
  rename("Status" = "status")

permit_status$Proportion <- paste0(round(permit_status$Proportion, digits=1),"%")

ggplot(permit_status,aes(x = reorder(Status, -Count), y= Count)) +
    geom_bar(stat="identity", fill="lightblue", width=0.5)+
    geom_text(aes(label=Proportion),
              hjust=.35)+
    theme_minimal()+
    labs(title="Permits by Status",x="Status")+
    theme(axis.text.y =element_text(angle = 55,hjust=1))+
    coord_flip()
```

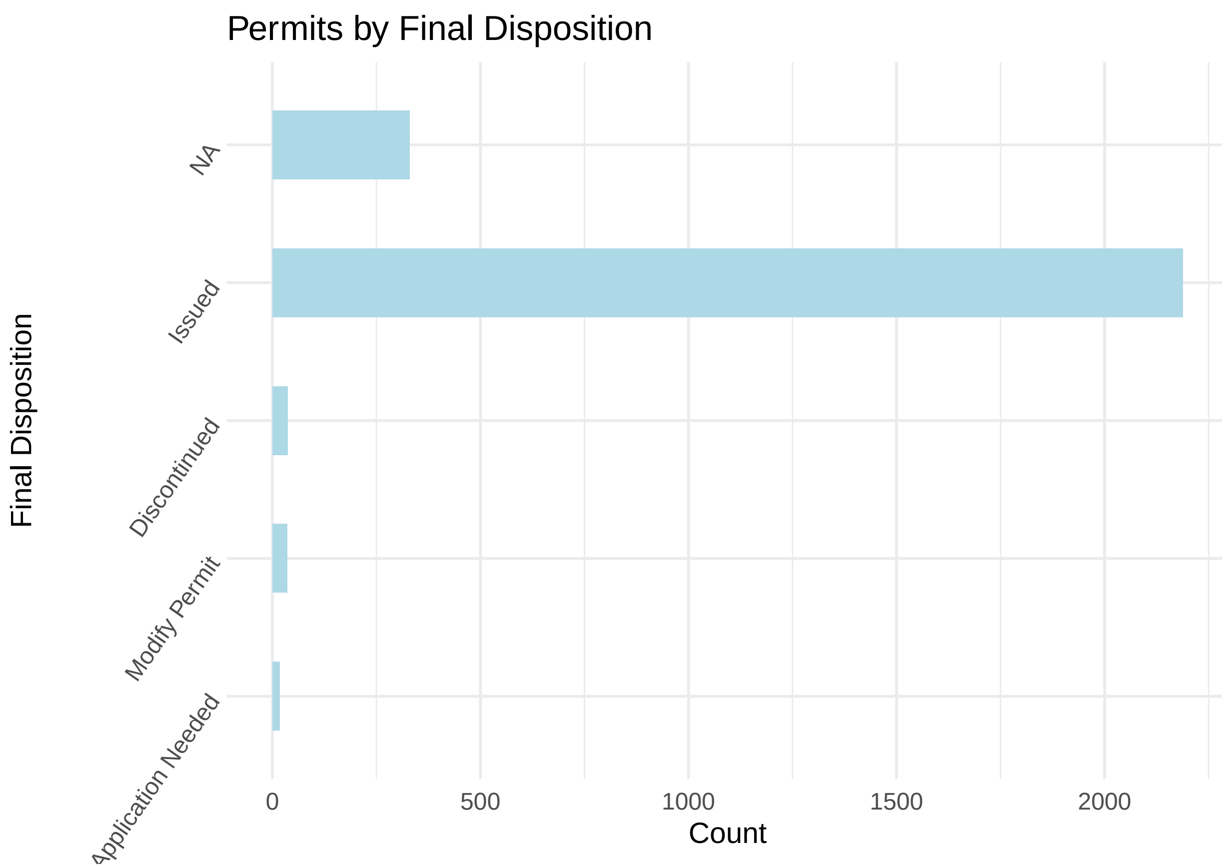



#### 4.0.0.10 Analysis

```
upa_class <- tbl1_permit_lvl |>
group_by(upa_class) |>
  summarize(
    Count = n(),
    Proportion = (n()/nrow(tbl1_permit_lvl))*100
  )

upa_class$Proportion <- paste0(round(upa_class$Proportion, digits=1),"%")

ggplot(upa_class,aes(x = reorder(upa_class, -Count), y= Count)) +
    geom_bar(stat="identity", fill="lightblue", width=0.5)+
    geom_text(aes(label=Proportion),
              hjust=.5,
              vjust=0.25)+
    theme_minimal()+
    labs(title="Permits by UPA Class",x="UPA Class")
```




```r
final_dis <- tbl1_permit_lvl |>
group_by(final_disposition) |>
  summarize(
    Count = n(),
    Proportion = (n()/nrow(tbl1_permit_lvl))*100
  ) |>
  arrange(desc(Count)) |>
  head(5)

final_dis$Proportion <- paste0(round(final_dis$Proportion, digits=1),"%")
final_dis$Count <- as.numeric(final_dis$Count)

final_dis <- final_dis |>
  clean_names("title")

ggplot(final_dis,aes(x =reorder(`Final Disposition`,`Count`, .desc = TRUE), y= Count)) +
    geom_bar(stat="identity", fill="lightblue", width=0.5)+
    theme_minimal()+
    labs(title="Permits by Final Disposition",x="Final Disposition")+
    theme(axis.text.y =element_text(angle = 55,hjust=1))+
    coord_flip()
```




```
app_type <- tbl2_permit_act_lvl |>
group_by(application_type) |>
  summarize(
    Count = n(),
    Proportion = n()/nrow(tbl2_permit_act_lvl)
  ) |>
  clean_names("title") |>
  arrange(desc(Count))

knitr::kable(app_type, format ="markdown")
```

| Application Type | Count | Proportion |
|---|---:|---:|
| Renewal Treat as New | 2565 | 0.7753930 |
| Modification Treat as New | 274 | 0.0828295 |
| Minor Modification | 185 | 0.0559250 |
| New | 160 | 0.0483676 |
| Modification | 48 | 0.0145103 |
| DIM Treat as New | 47 | 0.0142080 |
| Department Initiated Modification | 29 | 0.0087666 |

```r
short_desc <- tbl2_permit_act_lvl |>
  mutate(c_fast_track=coalesce(str_count(short_description,"fast track"),0)) |>
  summarize(
    "Fast Tracked Renewal Actions" = sum(c_fast_track),
    "Total Actions" = nrow(tbl2_permit_act_lvl),
    Proportion = sum(c_fast_track)/nrow(tbl2_permit_act_lvl)
  ) |>
  clean_names("title")

knitr::kable(short_desc, format ="markdown")
```

| Fast Tracked Renewal Actions | Total Actions | Proportion |
|---:|---:|---:|
| 2204 | 3308 | 0.6662636 |

---

# 5 Cheese Dataset:

#### 5.0.0.1 Import Data

Read in raw csv file as data frame

```
data <- read.csv("https://raw.githubusercontent.com/rfordatascience/tidytuesday/refs/hea
```

Row_Count

Column_Count

Null_Count

None_Str_Count

1187

19

7133

0

#### 5.0.0.2 Data Handling

- select columns needed to tidy and for analysis
- fill empty strings and null values with 'None' string

```
fill_empty_str = function(x){if_else(x=="", 'None' ,x)}

df = data |>
  select(cheese, milk, country, texture, aroma, flavor) |>
  mutate_all(fill_empty_str) |>
  mutate_all(replace_na, "None")
```

Row_Count

Column_Count

Null_Count

None_Str_Count

1187

6

0

340

#### 5.0.0.3 Tidy Data

Tidy data by ensuring each value has its own cell

- split out each row with listed values (milk, texture, aroma, flavor, country) into individual rows and lengthen the dataframe

```
df = df |>
  mutate(cheese_id = row_number()) |>
  separate_rows(country, sep = ', ') |>
  separate_rows(milk, sep = ', ') |>
  separate_rows(texture, sep = ', ') |>
  separate_rows(aroma, sep = ', ') |>
  separate_rows(flavor, sep = ', ')
```

Row_Count

Column_Count

Null_Count

None_Str_Count

14394

7

0

2050

#### 5.0.0.4 Normalize Data

Normalize data to reduce redundancy and allow for more efficent analysis

- create a data frame for each column and create an associated id column for each
- replace all column values with respective id value in core data frame

```
create_id_dfs = function(id_prefix, col, df) {
  id_df = df |>
    select(all_of(col)) |>
    distinct() |>
    arrange(col) |>
    mutate(id = paste0(id_prefix, row_number()))
  return(id_df)
}

cheese_df = df |>
  select(cheese, cheese_id) |>
  distinct()
```

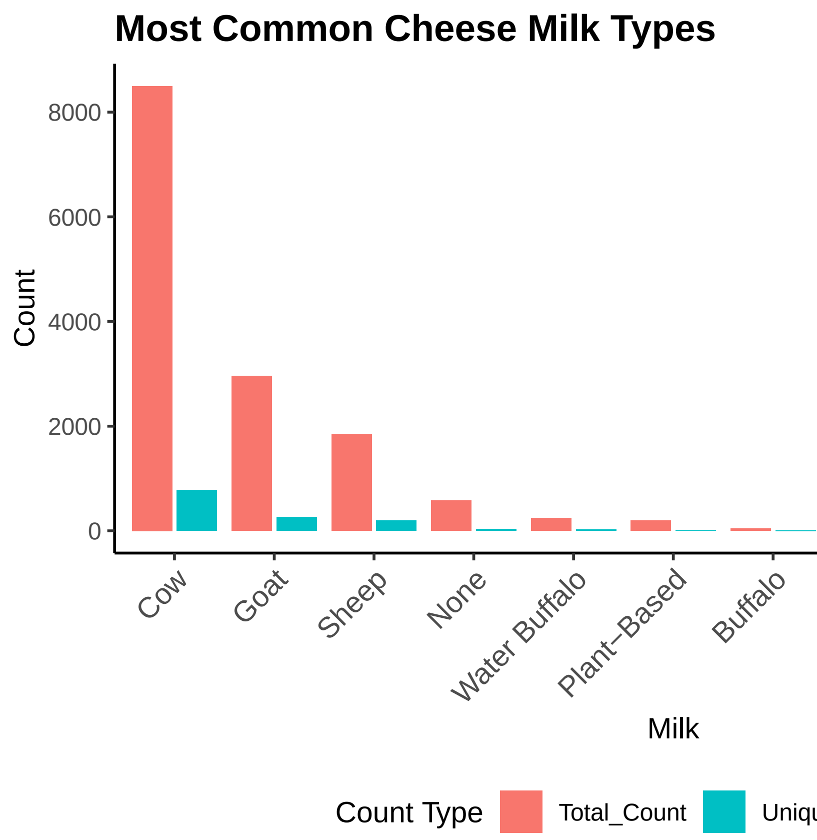
```r
country_df = create_id_dfs('C', 'country', df)
colnames(country_df) = c('country', 'country_id')

milk_df = create_id_dfs('M', 'milk', df)
colnames(milk_df) = c('milk', 'milk_id')

texture_df = create_id_dfs('T', 'texture', df)
colnames(texture_df) = c('texture', 'texture_id')

aroma_df = create_id_dfs('A', 'aroma', df)
colnames(aroma_df) = c('aroma', 'aroma_id')

flavor_df = create_id_dfs('F', 'flavor', df)
colnames(flavor_df) = c('flavor', 'flavor_id')

df = left_join(df, country_df, by = join_by(country))
df = left_join(df, milk_df, by = join_by(milk))
df = left_join(df, texture_df, by = join_by(texture))
df = left_join(df, aroma_df, by = join_by(aroma))
df = left_join(df, flavor_df, by = join_by(flavor))

df = df |>
  select(cheese_id, country_id, milk_id, texture_id, aroma_id, flavor_id)
```

#### 5.0.0.5 Analysis Analysis Requested in Discussion Post:

1. What are the most common milks used?
2. What are the more common textures associated with cheese?
3. Is there a country or region that produces more cheese?
4. Are there common aromas or flavors across cheeses made by different milks?

```r
df |>
  select(cheese_id, milk_id) |>
  left_join(cheese_df, by = join_by(cheese_id)) |>
  left_join(milk_df, by = join_by(milk_id)) |>
  mutate(milk = str_to_title(milk)) |>
  group_by(milk) |>
  summarise(
    Total_Count = n(),
    Unique_Cheese_Count = n_distinct(cheese)) |>
```

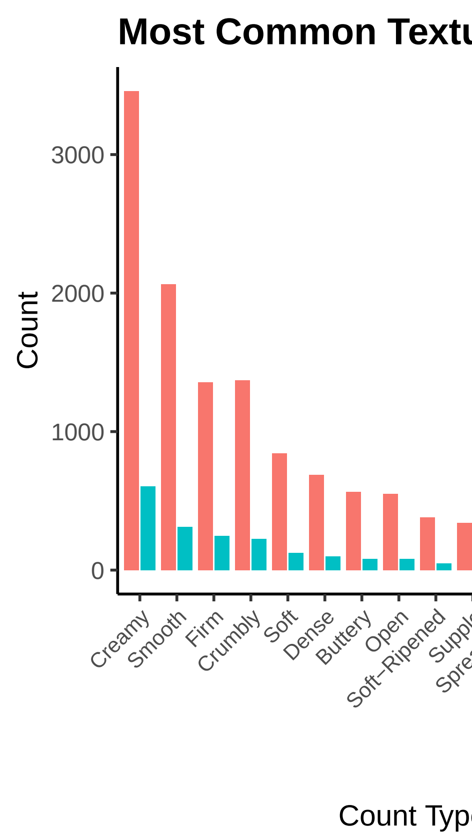
```
  pivot_longer(cols = c(Total_Count, Unique_Cheese_Count)) |>
  ggplot(aes(x = reorder(milk, -value), y = value, fill = name)) +
  geom_col(position = position_dodge2(width = 0.3, preserve = "single")) +
  labs(
    title = "Most Common Cheese Milk Types",
    x = "Milk",
    y = "Count",
    fill = 'Count Type'
  ) +
  theme_classic() +
  theme(
    axis.text.x = element_text(size = 11, angle = 45, vjust = 1, hjust=1),
    plot.title = element_text(size = 14, face = "bold"),
    legend.position = "bottom")
```




#### 5.0.0.6 1. Most Common Milks Used

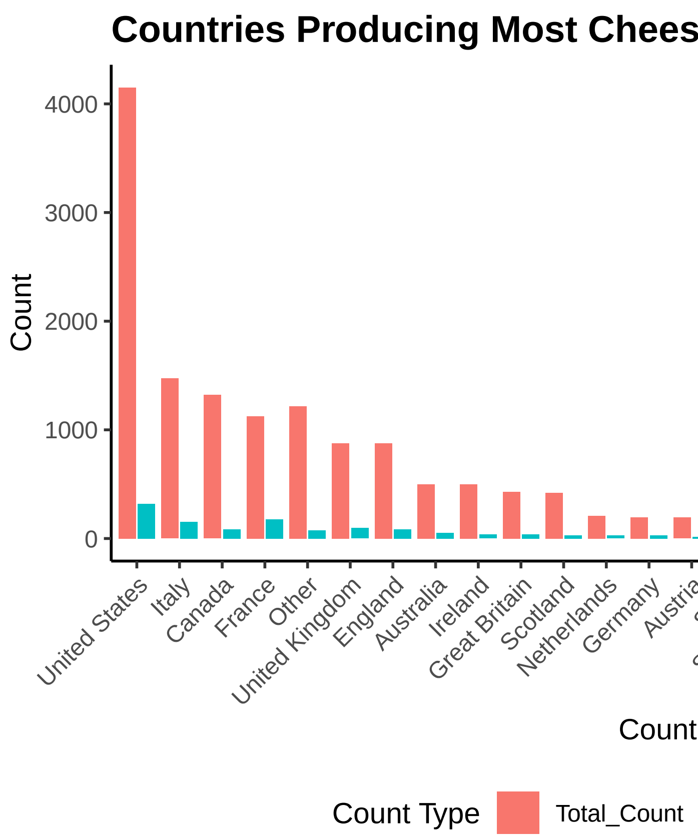
```
df |>
  select(cheese_id, texture_id) |>
  left_join(cheese_df, by = join_by(cheese_id)) |>
```

```r
  left_join(texture_df, by = join_by(texture_id)) |>
  mutate(texture = str_to_title(texture)) |>
  group_by(texture) |>
  summarise(
    Total_Count = n(),
    Unique_Cheese_Count = n_distinct(cheese)) |>
  pivot_longer(cols = c(Total_Count, Unique_Cheese_Count)) |>
  ggplot(aes(x = reorder(texture, -value), y = value, fill = name)) +
  geom_col(position = position_dodge2(width = 0.2, preserve = "single")) +
  labs(
    title = "Most Common Textures Associated with Cheeses",
    x = "Texture",
    y = "Count",
    fill = 'Count Type'
  ) +
  theme_classic() +
  theme(
    axis.text.x = element_text(angle = 45, vjust = 1, hjust=1, size = 8),
    plot.title = element_text(size = 14, face = "bold"),
    legend.position = "bottom")
```




#### 5.0.0.7 2. Most Common Textures Associated with Cheeses

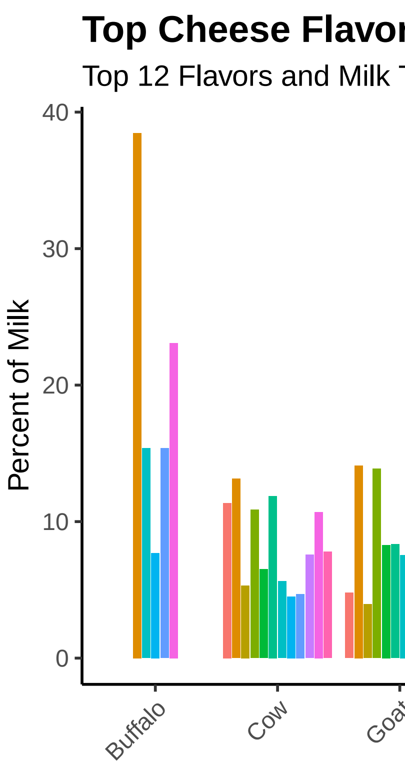
```
df |>
  select(cheese_id, country_id) |>
  left_join(cheese_df, by = join_by(cheese_id)) |>
  left_join(country_df, by = join_by(country_id)) |>
  mutate(country = str_to_title(country)) |>
  group_by(country) |>
  summarise(
    Total_Count = n(),
    Unique_Cheese_Count = n_distinct(cheese)) |>
  mutate(country = ifelse(Unique_Cheese_Count <= 5, 'Other', country)) |>
  group_by(country) |>
  summarise(
    Total_Count = sum(Total_Count),
    Unique_Cheese_Count = sum(Unique_Cheese_Count)) |>
  pivot_longer(cols = c(Total_Count, Unique_Cheese_Count)) |>
  ggplot(aes(x = reorder(country, -value), y = value, fill = name)) +
  geom_col(position = position_dodge2(width = 0.2, preserve = "single")) +
  labs(
    title = "Countries Producing Most Cheese",
    x = "Country",
    y = "Count",
    fill = 'Count Type'
  ) +
  theme_classic() +
  theme(
    axis.text.x = element_text(angle = 45, vjust = 1, hjust=1),
    plot.title = element_text(size = 14, face = "bold"),
    legend.position = "bottom")
```




#### 5.0.0.8 3. Countries Producing Most Cheese

```
top_flavor_df = df |>
  left_join(cheese_df, by = join_by(cheese_id)) |>
  left_join(flavor_df, by = join_by(flavor_id)) |>
  group_by(flavor, flavor_id) |>
  summarise(cnt = n(), .groups = 'keep') |>
  arrange(desc(cnt)) |>
  head(12)

milk_cnt_df = df |>
  left_join(cheese_df, by = join_by(cheese_id)) |>
  inner_join(top_flavor_df, by = join_by(flavor_id)) |>
  left_join(milk_df, by = join_by(milk_id)) |>
  group_by(milk, milk_id) |>
  summarise(milk_cnt = n(), .groups = 'keep') |>
  filter(milk_cnt>10)

df |>
  select(cheese_id, milk_id, flavor_id) |>
```

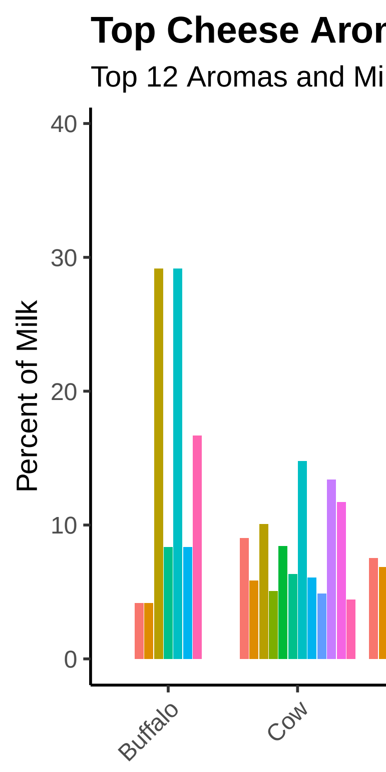
```r
left_join(cheese_df, by = join_by(cheese_id)) |>
inner_join(top_flavor_df, by = join_by(flavor_id)) |>
inner_join(milk_cnt_df, by = join_by(milk_id)) |>
mutate(
  flavor = str_to_title(flavor),
  milk = str_to_title(milk)
  ) |>
group_by(milk) |>
mutate(y= n()) |>
group_by(milk, flavor) |>
mutate(x = n()) |>
mutate(Unique_Cheese_Prct =  (x / y)*100) |>
select(milk, flavor, Unique_Cheese_Prct, x, milk_cnt, y) |>
distinct() |>
ggplot(aes(x = milk, y = Unique_Cheese_Prct, fill = flavor)) +
geom_col(position = position_dodge2(width = 0.3, preserve = "single")) +
labs(
  title = "Top Cheese Flavors and Milk Distribution",
  subtitle = 'Top 12 Flavors and Milk Types with a Count of at Least 10',
  x = "Milk",
  y = "Percent of Milk",
  fill = 'Flavor'
) +
theme_classic() +
theme(
  axis.text.x = element_text(angle = 45, vjust = 1, hjust=1),
  plot.title = element_text(size = 14, face = "bold"))
```




#### 5.0.0.9 4a. Common Flavors Across Cheeses By Different Milks

```
top_aroma_df = df |>
  left_join(cheese_df, by = join_by(cheese_id)) |>
  left_join(aroma_df, by = join_by(aroma_id)) |>
  group_by(aroma, aroma_id) |>
  summarise(cnt = n(), .groups = 'keep') |>
  arrange(desc(cnt)) |>
  head(12)

df |>
  select(cheese_id, milk_id, aroma_id) |>
  left_join(cheese_df, by = join_by(cheese_id)) |>
  inner_join(top_aroma_df, by = join_by(aroma_id)) |>
  inner_join(milk_cnt_df, by = join_by(milk_id)) |>
  mutate(
    aroma = str_to_title(aroma),
    milk = str_to_title(milk)
    ) |>
  group_by(milk) |>
```

```
mutate(y= n()) |>
group_by(milk, aroma) |>
mutate(x = n()) |>
mutate(Unique_Cheese_Prct =  (x / y)*100) |>
select(milk, aroma, Unique_Cheese_Prct, x, milk_cnt, y) |>
distinct() |>
ggplot(aes(x = milk, y = Unique_Cheese_Prct, fill = aroma)) +
geom_col(position = position_dodge2(width = 0.3, preserve = "single")) +
labs(
  title = "Top Cheese Aromas and Milk Distribution",
  subtitle = 'Top 12 Aromas and Milk Types with a Count of at Least 10',
  x = "Milk",
  y = "Percent of Milk",
  fill = 'Aroma'
) +
theme_classic() +
theme(
  axis.text.x = element_text(angle = 45, vjust = 1, hjust=1),
  plot.title = element_text(size = 14, face = "bold"))
```




#### 5.0.0.10 4b. Common Aromas Across Cheeses By Different Milks

---

# 6 Exporting Processed Data

```r
# Export cleaned Emissions dataset
write.csv(yearly_emissions_by_area, "yearly_emissions_by_area_cleaned.csv", row.names =
write.csv(yearly_emissions, "yearly_emissions_cleaned.csv", row.names = FALSE)
write.csv(emissions_by_area, "emissions_by_area_cleaned.csv", row.names = FALSE)

# Export cleaned DART water permit dataset
write.csv(tbl1_permit_lvl, "tbl1_permit_lvl_cleaned.csv", row.names = FALSE)
write.csv(tbl2_permit_act_lvl, "tbl2_permit_act_lvl_cleaned.csv", row.names = FALSE)
write.csv(tbl3_facility_lvl, "tbl3_facility_lvl_cleaned.csv", row.names = FALSE)
write.csv(tbl4_app_lvl, "tbl4_app_lvl_cleaned.csv", row.names = FALSE)

# Export cleaned Cheese Quality dataset
write.csv(cheese_df, "cheese_df_cleaned.csv", row.names = FALSE)
write.csv(country_df, "country_df_cleaned.csv", row.names = FALSE)
write.csv(milk_df, "milk_df_cleaned.csv", row.names = FALSE)
write.csv(texture_df, "texture_df_cleaned.csv", row.names = FALSE)
write.csv(aroma_df, "aroma_df_cleaned.csv", row.names = FALSE)
write.csv(flavor_df, "flavor_df_cleaned.csv", row.names = FALSE)
write.csv(df, "final_cheese_data_cleaned.csv", row.names = FALSE)

# Confirm that the files were saved successfully
list.files(pattern = "*.csv")
```

```
##  [1] "aroma_df_cleaned.csv"
##  [2] "cheese_df_cleaned.csv"
##  [3] "country_df_cleaned.csv"
##  [4] "dart_2020_2025.csv"
##  [5] "dart_3.7.2025.csv"
##  [6] "DATA607_Project2Data_cheeses.csv"
##  [7] "emissions_by_area_cleaned.csv"
##  [8] "final_cheese_data_cleaned.csv"
##  [9] "flavor_df_cleaned.csv"
## [10] "milk_df_cleaned.csv"
## [11] "project2-emissions-data.csv"
## [12] "tbl1_permit_lvl_cleaned.csv"
## [13] "tbl2_permit_act_lvl_cleaned.csv"
## [14] "tbl3_facility_lvl_cleaned.csv"
## [15] "tbl4_app_lvl_cleaned.csv"
## [16] "texture_df_cleaned.csv"
## [17] "yearly_emissions_by_area_cleaned.csv"
## [18] "yearly_emissions_cleaned.csv"
```

### 6.0.1 Why Exporting Matters?

- **Preserving Cleaning Efforts:** Once data transformation is complete, saving the cleaned versions prevents the need to redo preprocessing each time.
- **Improving Reproducibility:** The structured datasets can be shared with other analysts or data scientists for further analysis.
- **Facilitating Advanced Analytics:** The exported `.csv` files are now ready for machine learning models, visualization dashboards, and predictive analytics.

---

# 7 Conclusion

This project focused on transforming and analyzing three diverse datasets, demonstrating the importance of data wrangling techniques in preparing raw information for meaningful insights. By leveraging `tidyr` and `dplyr`, we efficiently cleaned, structured, and transformed the datasets into a tidy format, making them suitable for downstream analysis.

### 7.0.1 Key Takeaways from Each Dataset:

1. **Emissions Data:**

   - The data was reshaped to a long format, making it easier to analyze changes over time.
   - Trends in emissions were identified, providing insights into pollution levels and their environmental implications.

2. **DART Water Permits Data:**

   - The dataset was transformed to facilitate trend analysis in water permit issuance from 2020 to 2025.
   - Cleaning and standardization helped address inconsistencies, ensuring accurate comparisons across years.

3. **Cheese Quality Data:**

   - The dataset was normalized to separate variables, improving its usability.
   - Various transformations allowed us to explore relationships between cheese characteristics, quality ratings, and production factors.

### 7.0.2 Overall Insights and Future Applications:

- **Data Preparation Matters:**

The process of tidying and transforming data is crucial for accurate and meaningful analysis. Well-structured data improves efficiency in both visualization and modeling.

- **Standardization & Normalization:**

Converting datasets to tidy formats ensured that values were easily accessible for statistical computation.

- **Potential for Further Analysis:**

These datasets can now be used for deeper predictive modeling, trend forecasting, and policy recommendations based on their respective domains.

Through this project, we reinforced the significance of data wrangling techniques in real-world data science applications. The ability to tidy, transform, and analyze raw data is a fundamental skill that enhances decision-making and unlocks valuable insights across different industries.