Section 001
CDS 101 *Introduction to Computational and Data Sciences*

Class 9: Data wrangling II

February 20, 2018

General

Annoucements

- Reading for next class: *R for Data Science*
  - All of chapter 10
  - From chapter 11: sections 11.1, 11.2, 11.4.2, and 11.5
- Readings 7 and 8, to be completed for class on February 27th and March 1st, also posted
- Homework 1 due Friday, February 23rd by 11:59pm

`dplyr` package (continued)

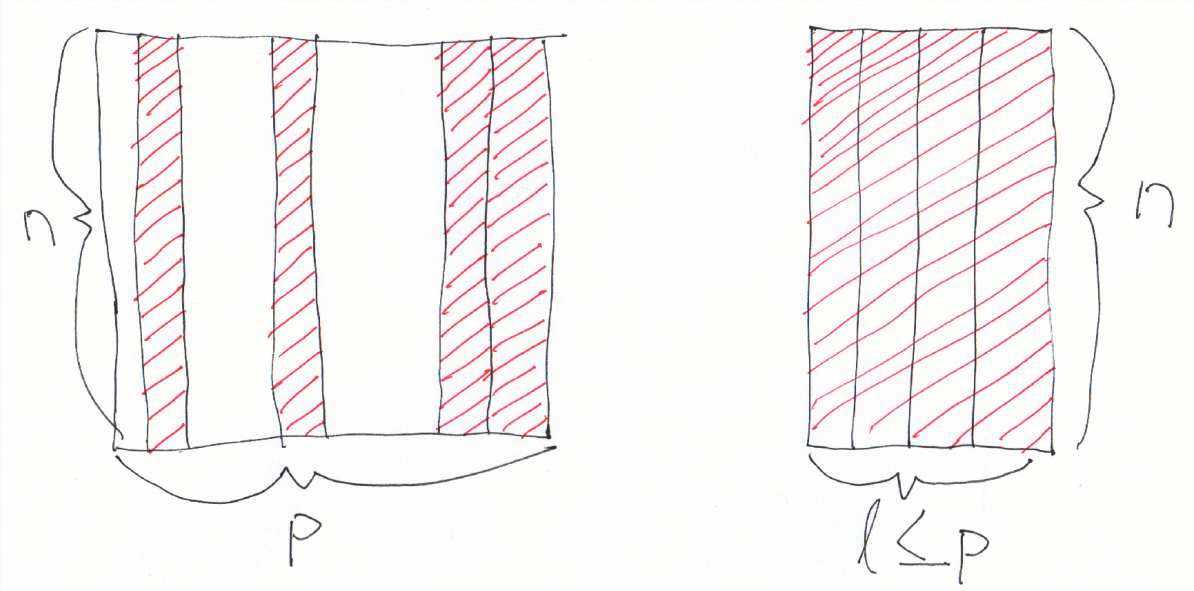
Get copy of `dplyr` demo repository

- Open RStudio and reload your **dplyr demo** repository from last Thursday's class
- If you were absent, find link to Github repository on class website, http://spring18.cds101.com/materials.html
- Follow along in the demos

`dplyr` so far

In the previous class, we reviewed the following `dplyr` commands

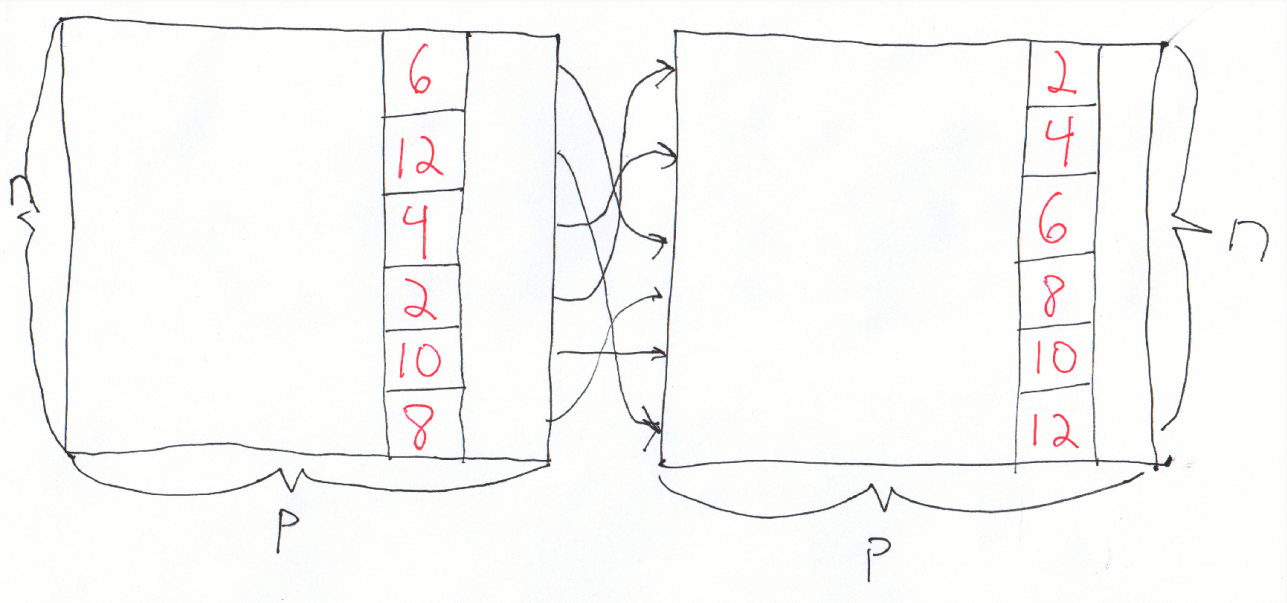
- `select()`



`dplyr` so far

In the previous class, we reviewed the following `dplyr` commands

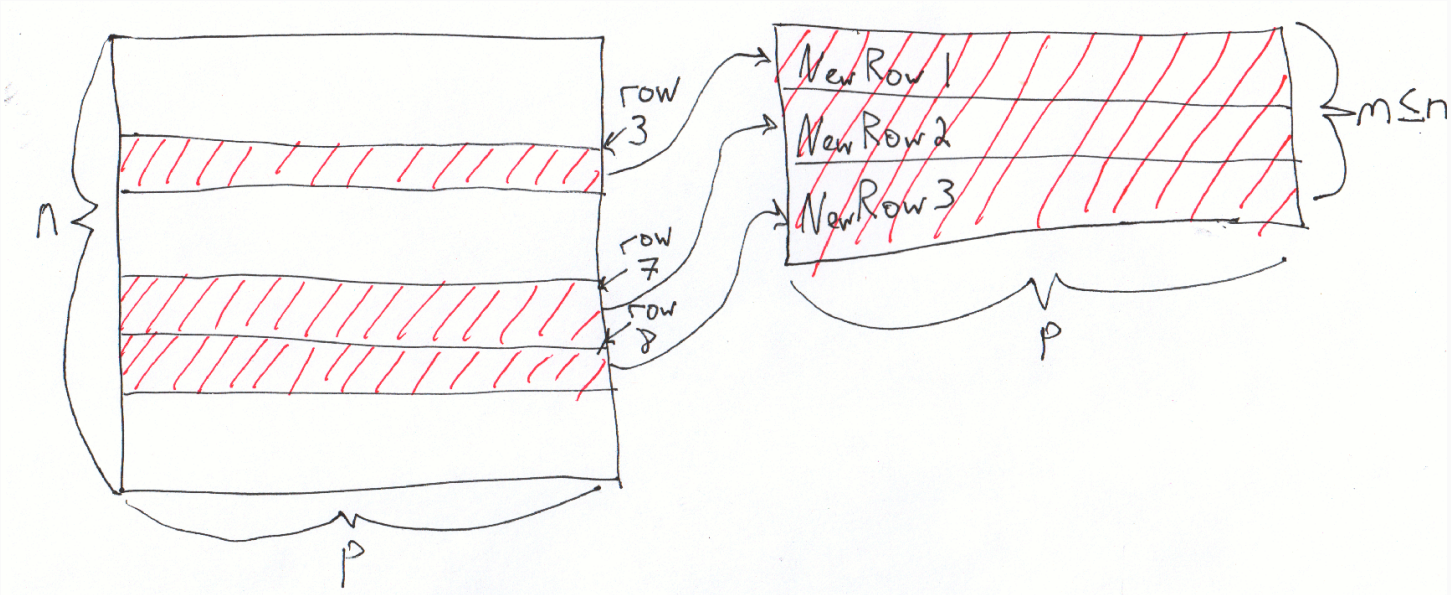
- `select()`
- `arrange()`



`dplyr` so far

In the previous class, we reviewed the following `dplyr` commands

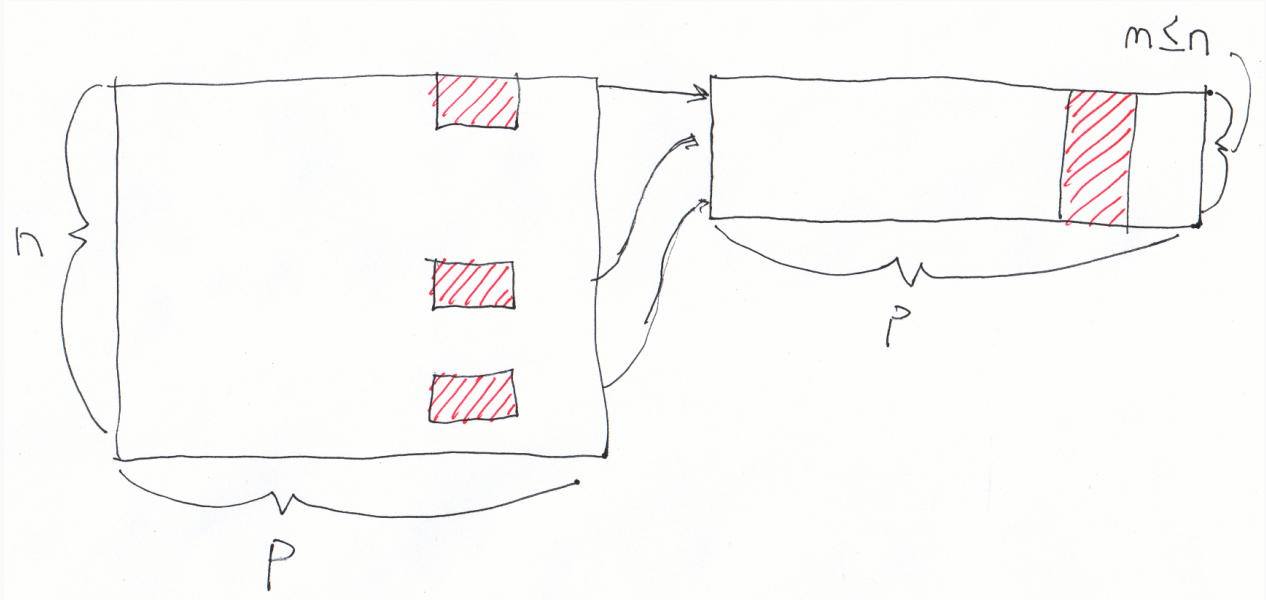
- `select()`
- `arrange()`
- `slice()`



`dplyr` so far

In the previous class, we reviewed the following `dplyr` commands

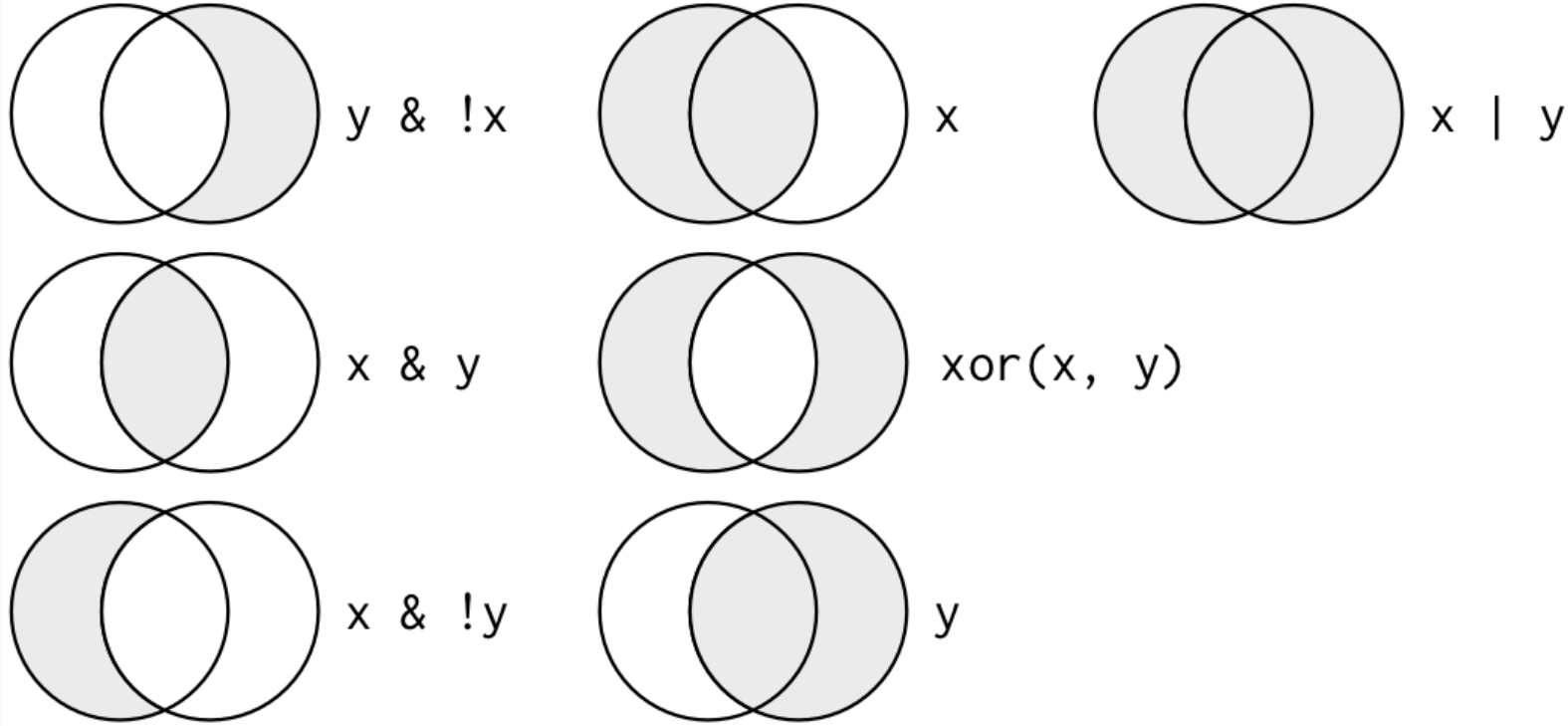
- `select()`
- `arrange()`
- `slice()`
- `filter()`



Use comparisons for filtering

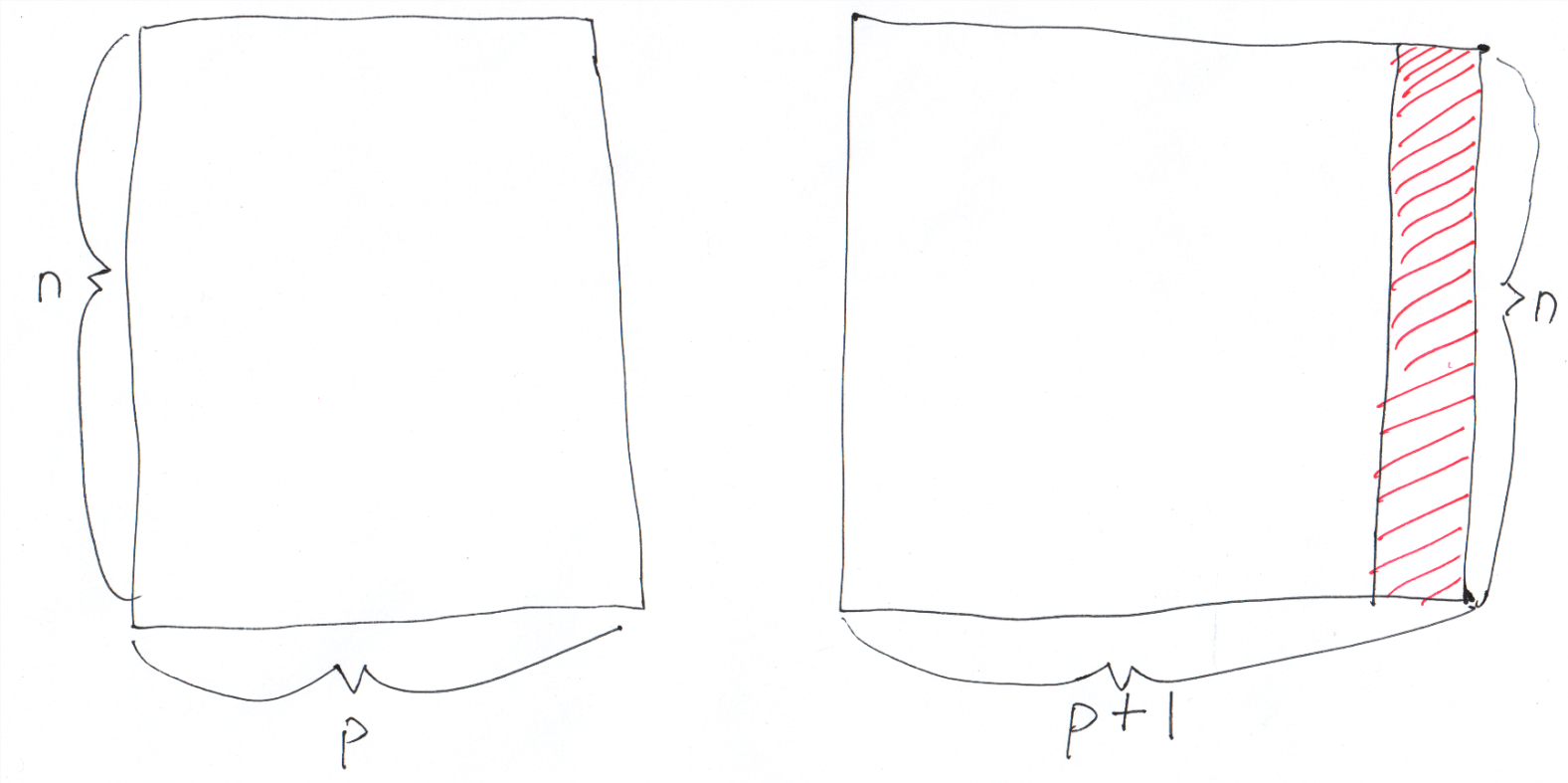
- `>`: greater than
- `>=`: greater than or equal to
- `<`: less than
- `<=`: less than or equal to
- `!=`: not equal
- `==`: equal

Logical operators



Source: Digital image of logical operations, *R for Data Science website*, accessed September 20, 2017, http://r4ds.had.co.nz/transform.html#logical-operators

mutate()



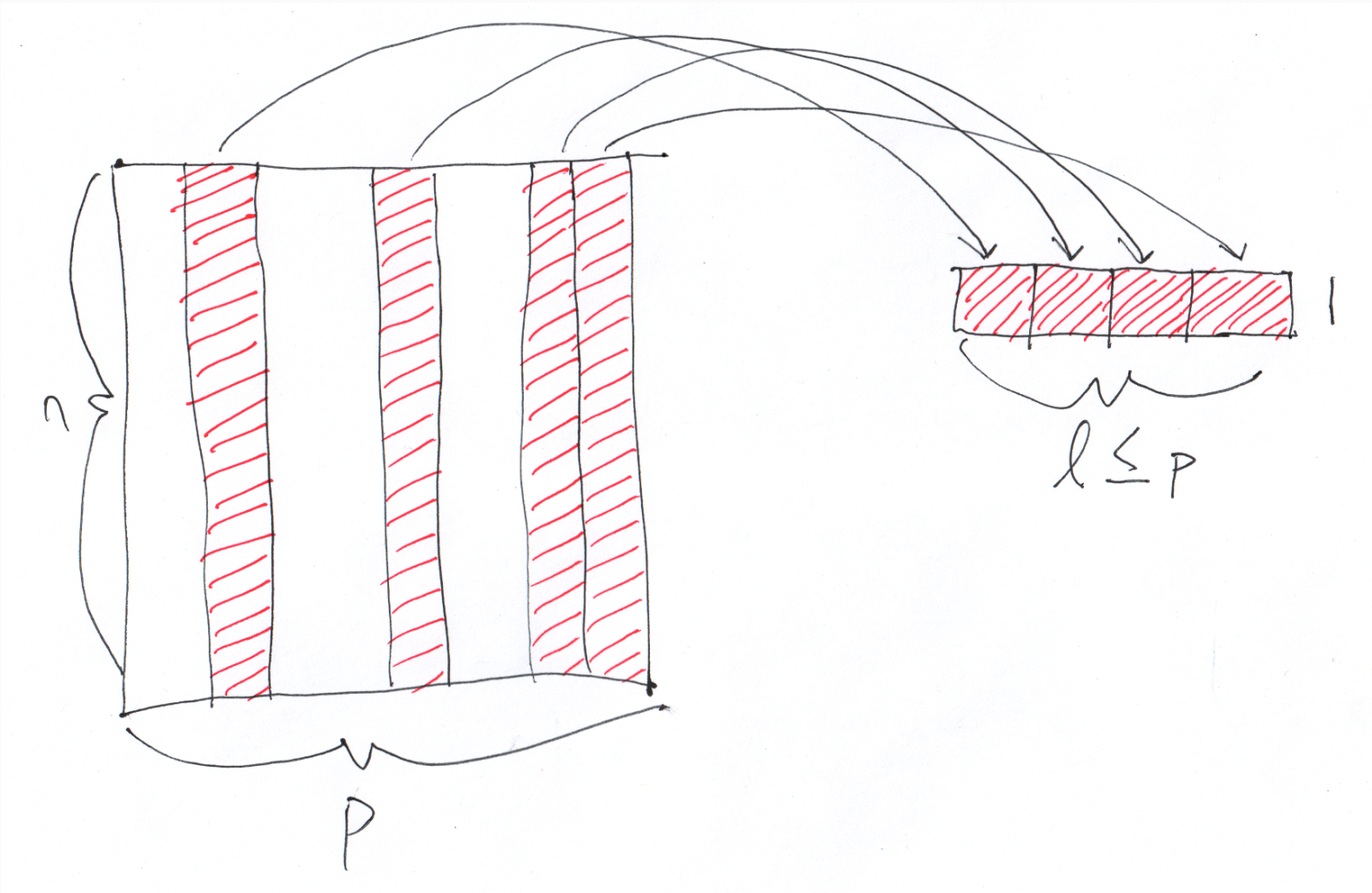
Using `mutate()`

- Many different operators and functions can be used with `mutate()`
- **Arithmetic operators:** `+`, `-`, `*`, `/`, `^`
- **Modular arithmetic**
  - `%/%`: integer division
  - `%%`: remainder
- **Logs:** `log()`
- **Logical comparisons:** `<`, `<=`, `>`, `>=`, `!=`

`mutate()` demo

Follow along in RStudio

group_by() and summarize()



Using `summarize()`

- `n()`: Counts number of rows in a group
- `sum()`: For numerical variables, sums rows within a group
- **statistical**: `mean()`, `median()`, `sd()`, `min()`, `max()`
- Counts and proportions of logical values: `sum(x > 10)`, `mean(y == 0)`

`group_by()` and `summarize()` demo

Follow along in RStudio

Other helpful `dplyr` verbs

- `transmute()`: Like `mutate()`, except the transformed output is placed in a new data frame
- `pull()`: Extract column into the base R `vector` data type
- `rename()`: Convenient way to change the name of a variable (column)
- `distinct()`: Finds unique rows in the dataset
- `count()`: Group by category and count the number of group members

`transmute()` example

```
presidential %>%
  transmute(term_length = interval(start, end) / dyears(1))
```

```
## # A tibble: 11 x 1
##    term_length
##          <dbl>
##  1        8.01
##  2        2.84
##  3        5.17
##  4        5.55
##  5        2.45
##  6        4.00
##  7        8.01
##  8        4.00
##  9        8.01
## 10        8.01
## 11        8.01
```

pull() example

```
presidential %>%
  pull(name)
```

```
##  [1] "Eisenhower" "Kennedy"    "Johnson"    "Nixon"      "Ford"
##  [6] "Carter"     "Reagan"     "Bush"       "Clinton"    "Bush"
## [11] "Obama"
```

rename() example

```
presidential %>%
  rename(term_begin = start, term_end = end)
```

```
## # A tibble: 11 x 4
##    name       term_begin term_end   party
##    <chr>      <date>     <date>     <chr>
##  1 Eisenhower 1953-01-20 1961-01-20 Republican
##  2 Kennedy    1961-01-20 1963-11-22 Democratic
##  3 Johnson    1963-11-22 1969-01-20 Democratic
##  4 Nixon      1969-01-20 1974-08-09 Republican
##  5 Ford       1974-08-09 1977-01-20 Republican
##  6 Carter     1977-01-20 1981-01-20 Democratic
##  7 Reagan     1981-01-20 1989-01-20 Republican
##  8 Bush       1989-01-20 1993-01-20 Republican
##  9 Clinton    1993-01-20 2001-01-20 Democratic
## 10 Bush       2001-01-20 2009-01-20 Republican
## 11 Obama      2009-01-20 2017-01-20 Democratic
```

distinct() example

```
presidential %>%
  distinct(name)
```

```
## # A tibble: 10 x 1
##    name
##    <chr>
##  1 Eisenhower
##  2 Kennedy
##  3 Johnson
##  4 Nixon
##  5 Ford
##  6 Carter
##  7 Reagan
##  8 Bush
##  9 Clinton
## 10 Obama
```

count() example

```
presidential %>%
  count(party)
```

```
## # A tibble: 2 x 2
##   party          n
##   <chr>      <int>
## 1 Democratic     5
## 2 Republican     6
```

Practicing with `nycflights13`

`nycflights13` dataset

- The utility of the `dplyr` functions becomes more obvious when we are working with a larger dataset
- Install the `nycflights13` dataset by typing the following in your RStudio *Console* window:

```
install.packages("nycflights13")
```

- To load it, create an `R` code block and run it:

```
library(tidyverse)
```

First glimpse of nycflights13

```
flights %>% glimpse()
```

First glimpse of nycflights13

```
flights %>% glimpse()
```

```
## Observations: 336,776
## Variables: 19
## $ year           <int> 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013,...
## $ month          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ day            <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ dep_time       <int> 517, 533, 542, 544, 554, 554, 555, 557, 557, 55...
## $ sched_dep_time <int> 515, 529, 540, 545, 600, 558, 600, 600, 600, 60...
## $ dep_delay      <dbl> 2, 4, 2, -1, -6, -4, -5, -3, -3, -2, -2, -2, -2...
## $ arr_time       <int> 830, 850, 923, 1004, 812, 740, 913, 709, 838, 7...
## $ sched_arr_time <int> 819, 830, 850, 1022, 837, 728, 854, 723, 846, 7...
## $ arr_delay      <dbl> 11, 20, 33, -18, -25, 12, 19, -14, -8, 8, -2, -...
## $ carrier        <chr> "UA", "UA", "AA", "B6", "DL", "UA", "B6", "EV",...
## $ flight         <int> 1545, 1714, 1141, 725, 461, 1696, 507, 5708, 79...
## $ tailnum        <chr> "N14228", "N24211", "N619AA", "N804JB", "N668DN...
## $ origin         <chr> "EWR", "LGA", "JFK", "JFK", "LGA", "EWR", "EWR"...
## $ dest           <chr> "IAH", "IAH", "MIA", "BQN", "ATL", "ORD", "FLL"...
## $ air_time       <dbl> 227, 227, 160, 183, 116, 150, 158, 53, 140, 138...
## $ distance       <dbl> 1400, 1416, 1089, 1576, 762, 719, 1065, 229, 94...
## $ hour           <dbl> 5, 5, 5, 5, 6, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 5,...
## $ minute         <dbl> 15, 29, 40, 45, 0, 58, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ time_hour      <dttm> 2013-01-01 05:00:00, 2013-01-01 05:00:00, 2013...
```

Practice Question 1

What command should I run to answer the following question:

"Which airlines flew into or out of the New York City airports in 2013?"

Practice Question 1

What command should I run to answer the following question:

"Which airlines flew into or out of the New York City airports in 2013?"

```
flights %>%
  distinct(carrier)
```

Practice Question 1

What command should I run to answer the following question:

"Which airlines flew into or out of the New York City airports in 2013?"

```
flights %>%
  distinct(carrier)
```

Carriers 1 through 8

```
## # A tibble: 8 x 1
##   carrier
##   <chr>
## 1 UA
## 2 AA
## 3 B6
## 4 DL
## 5 EV
## 6 MQ
## 7 US
## 8 WN
```

Carriers 9 through 16

```
## # A tibble: 8 x 1
##   carrier
##   <chr>
## 1 VX
## 2 FL
## 3 AS
## 4 9E
## 5 F9
## 6 HA
## 7 YV
## 8 OO
```

Practice Question 2

What command should I run to answer the following question:

"Which flights departed either during the month of March or the month of June?"

Practice Question 2

What command should I run to answer the following question:

"Which flights departed either during the month of March or the month of June?"

```
flights %>%
  filter(month == 3 | month == 6)
```

Practice Question 2

```
## Observations: 57,077
## Variables: 19
## $ year           <int> 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013,...
## $ month          <int> 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,...
## $ day            <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ dep_time       <int> 4, 50, 117, 454, 505, 521, 537, 541, 549, 550, ...
## $ sched_dep_time <int> 2159, 2358, 2245, 500, 515, 530, 540, 545, 600,...
## $ dep_delay      <dbl> 125, 52, 152, -6, -10, -9, -3, -4, -11, -10, -8...
## $ arr_time       <int> 318, 526, 223, 633, 746, 813, 856, 1014, 639, 7...
## $ sched_arr_time <int> 56, 438, 2354, 648, 810, 827, 850, 1023, 703, 8...
## $ arr_delay      <dbl> 142, 48, 149, -15, -24, -14, 6, -9, -24, -14, -...
## $ carrier        <chr> "B6", "B6", "B6", "US", "UA", "UA", "AA", "B6",...
## $ flight         <int> 11, 707, 608, 1117, 475, 1714, 1141, 725, 2114,...
## $ tailnum        <chr> "N706JB", "N794JB", "N328JB", "N177US", "N527UA...
## $ origin         <chr> "JFK", "JFK", "JFK", "EWR", "EWR", "LGA", "JFK"...
## $ dest           <chr> "FLL", "SJU", "PWM", "CLT", "IAH", "IAH", "MIA"...
## $ air_time       <dbl> 166, 198, 48, 79, 199, 213, 173, 191, 31, 89, 1...
## $ distance       <dbl> 1069, 1598, 273, 529, 1400, 1416, 1089, 1576, 1...
## $ hour           <dbl> 21, 23, 22, 5, 5, 5, 5, 5, 6, 6, 5, 6, 6, 6, 6,...
## $ minute         <dbl> 59, 58, 45, 0, 15, 30, 40, 45, 0, 0, 59, 0, 0, ...
## $ time_hour      <dttm> 2013-03-01 21:00:00, 2013-03-01 23:00:00, 2013...
```

Practice Question 2

What command should I run to answer the following question:

"Which flights departed either during the month of March or the month of June?"

```
flights %>%
  filter(month == 3 | month == 6)
```

To check that both months are present, we can use `distinct()`:

Practice Question 2

What command should I run to answer the following question:

"Which flights departed either during the month of March or the month of June?"

```
flights %>%
  filter(month == 3 | month == 6)
```

To check that both months are present, we can use `distinct()`:

```
flights %>%
  filter(month == 3 | month == 6) %>%
  distinct(month)
```

```
## # A tibble: 2 x 1
##   month
##   <int>
## 1     3
## 2     6
```

Practice Question 3

What command should I run to convert the time duration in the arrival delay (`arr_delay`) column from minutes to seconds?

Practice Question 3

What command should I run to convert the time duration in the arrival delay (`arr_delay`) column from minutes to seconds?

```
flights %>%
  mutate(delay_in_seconds = arr_delay * 60)
```

Practice Question 3

```
## Observations: 336,776
## Variables: 20
## $ year             <int> 2013, 2013, 2013, 2013, 2013, 2013, 2013, 201...
## $ month            <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
## $ day              <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
## $ dep_time         <int> 517, 533, 542, 544, 554, 554, 555, 557, 557, ...
## $ sched_dep_time   <int> 515, 529, 540, 545, 600, 558, 600, 600, 600, ...
## $ dep_delay        <dbl> 2, 4, 2, -1, -6, -4, -5, -3, -3, -2, -2, -2, ...
## $ arr_time         <int> 830, 850, 923, 1004, 812, 740, 913, 709, 838,...
## $ sched_arr_time   <int> 819, 830, 850, 1022, 837, 728, 854, 723, 846,...
## $ arr_delay        <dbl> 11, 20, 33, -18, -25, 12, 19, -14, -8, 8, -2,...
## $ carrier          <chr> "UA", "UA", "AA", "B6", "DL", "UA", "B6", "EV...
## $ flight           <int> 1545, 1714, 1141, 725, 461, 1696, 507, 5708, ...
## $ tailnum          <chr> "N14228", "N24211", "N619AA", "N804JB", "N668...
## $ origin           <chr> "EWR", "LGA", "JFK", "JFK", "LGA", "EWR", "EW...
## $ dest             <chr> "IAH", "IAH", "MIA", "BQN", "ATL", "ORD", "FL...
## $ air_time         <dbl> 227, 227, 160, 183, 116, 150, 158, 53, 140, 1...
## $ distance         <dbl> 1400, 1416, 1089, 1576, 762, 719, 1065, 229, ...
## $ hour             <dbl> 5, 5, 5, 5, 6, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, ...
## $ minute           <dbl> 15, 29, 40, 45, 0, 58, 0, 0, 0, 0, 0, 0, 0, 0...
## $ time_hour        <dttm> 2013-01-01 05:00:00, 2013-01-01 05:00:00, 20...
## $ delay_in_seconds <dbl> 660, 1200, 1980, -1080, -1500, 720, 1140, -84...
```