



Tidy data

country	year	cases	population
Afghanistan	1999	212258	1272915272
Afghanistan	2000	216766	128042583
Brazil	1999	37737	172006362
Brazil	2000	38488	174504898
China	1999	212258	1272915272
China	2000	216766	128042583

variables

country	year	cases	population
Afghanistan	1999	212258	1272915272
Afghanistan	2000	216766	128042583
Brazil	1999	37737	172006362
Brazil	2000	38488	174504898
China	1999	212258	1272915272
China	2000	216766	128042583

observations

country	year	cases	population
Afghanistan	99	212258	1272915272
Afghanistan	00	216766	128042583
Brazil	99	37737	172006362
Brazil	00	38488	174504898
China	99	212258	1272915272
China	00	216766	128042583

values

Tidy vs Untidy

Happy families are all alike; every unhappy family is unhappy in its own way

— Leo Tolstoy, Anna Karenina

```
## # A tibble: 317 × 7
##   artist      track      date.entered  wk1  wk2  wk3  wk4
##   <chr>      <chr>      <date>      <dbl> <dbl> <dbl> <dbl>
## 1 2 Pac      Baby Don't Cry (Keep... 2000-02-26    87    82    72    77
## 2 2Ge+her    The Hardest Part Of ... 2000-09-02    91    87    92    NA
## 3 3 Doors Down Kryptonite      2000-04-08    81    70    68    67
## 4 3 Doors Down Loser           2000-10-21    76    76    72    69
## 5 504 Boyz    Wobble Wobble   2000-04-15    57    34    25    17
## 6 98^0       Give Me Just One Nig... 2000-08-19    51    39    34    26
## 7 A*Teens    Dancing Queen   2000-07-08    97    97    96    95
## 8 Aaliyah    I Don't Wanna   2000-01-29    84    62    51    41
## 9 Aaliyah    Try Again       2000-03-18    59    53    38    28
## 10 Adams, Yolanda Open My Heart    2000-08-26    76    76    74    69
## # ... with 307 more rows
```



Modern data frames

Hadley Wickham / RStudio have a package that modifies data frames to be a bit more modern. The core features of tibbles is to have a nicer printing method as well as being "surly" and "lazy".

```
library(tibble)
```

```
iris
```

```
##      Sepal.Length Sepal.Width Petal.Length
## 1           5.1           3.5           1.4
## 2           4.9           3.0           1.4
## 3           4.7           3.2           1.3
## 4           4.6           3.1           1.5
## 5           5.0           3.6           1.4
## 6           5.4           3.9           1.7
## 7           4.6           3.4           1.4
## 8           5.0           3.4           1.5
## 9           4.4           2.9           1.4
## 10          4.9           3.1           1.5
## 11          5.4           3.7           1.5
## 12          4.8           3.4           1.6
```

```
(tbl_iris = as_tibble(iris))
```

```
## # A tibble: 150 × 5
##   Sepal.Length Sepal.Width Petal.Length
##         <dbl>         <dbl>         <dbl>
## 1           5.1           3.5           1.4
## 2           4.9           3.0           1.4
## 3           4.7           3.2           1.3
## 4           4.6           3.1           1.5
## 5           5.0           3.6           1.4
## 6           5.4           3.9           1.7
## 7           4.6           3.4           1.4
## 8           5.0           3.4           1.5
## 9           4.4           2.9           1.4
## 10          4.9           3.1           1.5
```

Tibbles are lazy

By default, subsetting tibbles always results in another tibble (`$` or `[[` can still be used to subset for a specific column).

```
tbl_iris[1,]
```

```
## # A tibble: 1 × 5
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##         <dbl>      <dbl>      <dbl>      <dbl> <fct>
## 1         5.1         3.5         1.4         0.2 setosa
```

```
tbl_iris[,1]
```

```
## # A tibble: 150 × 1
##   Sepal.Length
##         <dbl>
## 1         5.1
## 2         4.9
## 3         4.7
## 4         4.6
## 5         5
## 6         5.4
## 7         4.6
```

```
tbl_iris[[1]]
```

```
##   [1] 5.1 4.9 4.7 4.6 5.0 5.4 4.6 5.0 4.4 4.9 5.4 4.
##  [19] 5.7 5.1 5.4 5.1 4.6 5.1 4.8 5.0 5.0 5.2 5.2 4.
##  [37] 5.5 4.9 4.4 5.1 5.0 4.5 4.4 5.0 5.1 4.8 5.1 4.
##  [55] 6.5 5.7 6.3 4.9 6.6 5.2 5.0 5.9 6.0 6.1 5.6 6.
##  [73] 6.3 6.1 6.4 6.6 6.8 6.7 6.0 5.7 5.5 5.5 5.8 6.
##  [91] 5.5 6.1 5.8 5.0 5.6 5.7 5.7 6.2 5.1 5.7 6.3 5.
## [109] 6.7 7.2 6.5 6.4 6.8 5.7 5.8 6.4 6.5 7.7 7.7 6.
## [127] 6.2 6.1 6.4 7.2 7.4 7.9 6.4 6.3 6.1 7.7 6.3 6.
## [145] 6.7 6.7 6.3 6.5 6.2 5.9
```

More laziness - partial matching

Tibbles do not use partial matching when the \$ operator is used.

```
head( iris$Sp )
```

```
## [1] setosa setosa setosa setosa setosa setosa  
## Levels: setosa versicolor virginica
```

```
head( iris$Species )
```

```
## [1] setosa setosa setosa setosa setosa setosa  
## Levels: setosa versicolor virginica
```

```
head( tbl_iris$Sp )
```

```
## Warning: Unknown or uninitialised column: `Sp`.  
## NULL
```

```
head( tbl_iris$Species )
```

```
## [1] setosa setosa setosa setosa setosa setosa  
## Levels: setosa versicolor virginica
```

More laziness - stringsAsFactors

Tibbles also have always had `stringsAsFactors = FALSE` as default behavior.

```
(t = tibble(  
  x = 1:3,  
  y = c("A", "B", "C"),  
  z = factor(c("X", "Y", "Z"))  
))
```

```
## # A tibble: 3 × 3  
##       x y     z  
##   <int> <chr> <fct>  
## 1     1 A     X  
## 2     2 B     Y  
## 3     3 C     Z
```

```
str(t)
```

```
## tibble [3 × 3] (S3: tbl_df/tbl/data.frame)  
## $ x: int [1:3] 1 2 3  
## $ y: chr [1:3] "A" "B" "C"  
## $ z: Factor w/ 3 levels "X","Y","Z": 1 2 3
```

```
(d = data.frame(  
  x = 1:3,  
  y = c("A", "B", "C"),  
  z = factor(c("X", "Y", "Z")),  
  stringsAsFactors = TRUE  
))
```

```
##   x y z  
## 1 1 A X  
## 2 2 B Y  
## 3 3 C Z
```

```
str(d)
```

```
## 'data.frame':   3 obs. of  3 variables:  
## $ x: int  1 2 3  
## $ y: Factor w/ 3 levels "A","B","C": 1 2 3  
## $ z: Factor w/ 3 levels "X","Y","Z": 1 2 3
```


Tibbles and length coercion

```
data.frame(x = 1:4, y = 1)
```

```
##   x y  
## 1 1 1  
## 2 2 1  
## 3 3 1  
## 4 4 1
```

```
data.frame(x = 1:4, y = 1:2)
```

```
##   x y  
## 1 1 1  
## 2 2 2  
## 3 3 1  
## 4 4 2
```

```
data.frame(x = 1:4, y = 1:3)
```

```
## Error in data.frame(x = 1:4, y = 1:3): argument
```

```
tibble(x = 1:4, y = 1)
```

```
## # A tibble: 4 × 2  
##       x     y  
##   <int> <dbl>  
## 1     1     1  
## 2     2     1  
## 3     3     1  
## 4     4     1
```

```
tibble(x = 1:4, y = 1:2)
```

```
## Error: Tibble columns must have compatible sizes.  
## * Size 4: Existing data.  
## * Size 2: Column `y`.  
## i Only values of size one are recycled.
```

```
tibble(x = 1:4, y = 1:3)
```

```
## Error: Tibble columns must have compatible sizes.  
## * Size 4: Existing data.
```

Tibbles and S3

```
t = tibble(  
  x = 1:3,  
  y = c("A", "B", "C")  
)
```

```
class(t)
```

```
## [1] "tbl_df"      "tbl"        "data.frame"
```

```
d = data.frame(  
  x = 1:3,  
  y = c("A", "B", "C")  
)
```

```
class(d)
```

```
## [1] "data.frame"
```

```
methods(class="tbl_df")
```

```
## [1] [          [[          [[<-         [<-          $  
## [6] $<-        as.data.frame coerce        initialize  names<-  
## [11] Ops         row.names<-  show        slotsFromS3 str  
## [16] tbl_sum  
## see '?methods' for accessing help and source code
```

```
methods(class="tbl")
```

```
## [1] [[<-        [<-          $<-          coerce       format       glimpse  
## [7] initialize  Ops          print        show         slotsFromS3 tbl_sum  
## see '?methods' for accessing help and source code
```



magrittr

What is a pipe

In software engineering, a pipeline consists of a chain of processing elements (processes, threads, coroutines, functions, etc.), arranged so that the output of each element is the input of the next;

- [Wikipedia - Pipeline \(software\)](#)

Magrittr's pipe is a new infix operator that allows us to link two functions together in a way that is readable from left to right.

The two code examples below are equivalent,

```
f(g(x=1, y=2), n=2)
```

```
g(x=1, y=2) %>% f(n=2)
```

Readability

Consider the following sequence of actions that describe the process of getting to campus in the morning:

I need to find my key, then unlock my car, then start my car, then drive to school, then park.

Expressed as a set of nested functions in R pseudocode this would look like:

```
park(drive(start_car(find("keys")), to="campus"))
```

Writing it out using pipes give it a more natural (and easier to read) structure:

```
find("keys") %>%  
  start_car() %>%  
  drive(to="campus") %>%  
  park()
```

Approaches

All of the following are fine, it comes down to personal preference:

Nested:

```
h( g( f(x), y=1), z=1 )
```

Piped:

```
f(x) %>%  
  g(y=1) %>%  
  h(z=1)
```

Intermediate:

```
res = f(x)  
res = g(res, y=1)  
res = h(res, z=1)
```

What about other arguments?

Sometimes we want to send our results to a function argument other than the first one or we want to use the previous result for multiple arguments. In these cases we can refer to the previous result using `..`

```
data.frame(a = 1:3, b = 3:1) %>% lm(a~b, data=.)
```

```
##  
## Call:  
## lm(formula = a ~ b, data = .)  
##  
## Coefficients:  
## (Intercept)          b  
##           4          -1
```

```
data.frame(a = 1:3, b = 3:1) %>% .[[1]]
```

```
## [1] 1 2 3
```

```
data.frame(a = 1:3, b = 3:1) %>% .[[length(.)]]
```

```
## [1] 3 2 1
```

The base R pipe

As of R v4.1.0 a pipe operator has been added to the base language in R, it is implemented as `|>`.

```
1:10 |> cumsum()
```

```
## [1] 1 3 6 10 15 21 28 36 45 55
```

```
1:10 |> cumsum() |> mean()
```

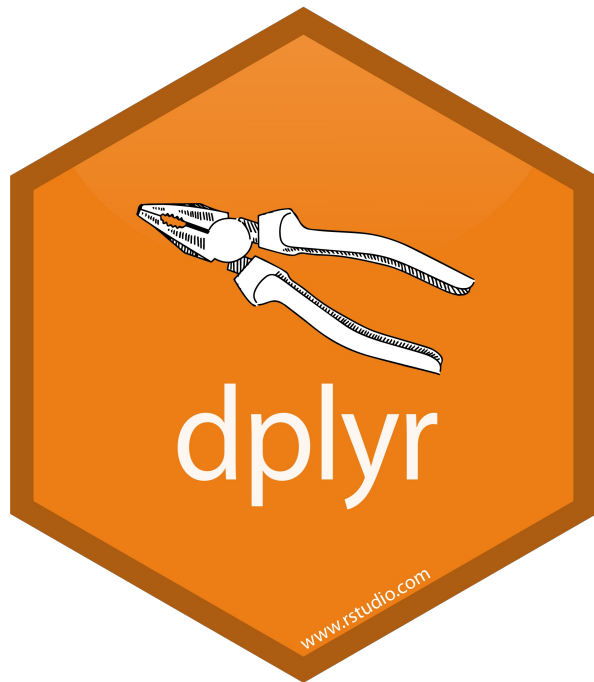
```
## [1] 22
```

The current version of RStudio on the departmental servers is v4.0.5 but you can install a newer version on your personal machine if you want to try it out.

For this reason and a couple of other caveats listed below we will be relying on the magrittr pipe for this course.

Base R pipe considerations:

- Depending on an R version ≥ 4.1 is a harder dependency than depending on the magrittr



A Grammar of Data Manipulation

dplyr is based on the concepts of functions as verbs that manipulate data frames.

Core single data frame functions / verbs:

- `filter()` / `slice()`: pick rows based on criteria
- `select()` / `rename()`: select columns by name
- `pull()`: grab a column as a vector
- `arrange()`: reorder rows
- `mutate()` / `transmute()`: create or modify columns
- `distinct()`: filter for unique rows
- `summarise()` / `count()`: reduce variables to values
- `group_by()` / `ungroup()`: modify other verbs to act on subsets
- `relocate()`: change column order

dplyr rules

1. First argument is always a data frame
2. Subsequent arguments say what to do with that data frame
3. Always return a data frame
4. Don't modify in place
5. Lazy evaluation magic

Example Data

We will demonstrate dplyr's functionality using the nycflights13 data.

```
library(dplyr)
library(nycflights13)
```

```
flights
```

```
## # A tibble: 336,776 × 19
##   year month  day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>         <dbl>     <int>         <int>
## 1  2013     1     1     517           515           2         830           819
## 2  2013     1     1     533           529           4         850           830
## 3  2013     1     1     542           540           2         923           850
## 4  2013     1     1     544           545          -1        1004          1022
## 5  2013     1     1     554           600          -6         812           837
## 6  2013     1     1     554           558          -4         740           728
## 7  2013     1     1     555           600          -5         913           854
## 8  2013     1     1     557           600          -3         709           723
## 9  2013     1     1     557           600          -3         838           846
## 10 2013     1     1     558           600          -2         753           745
## # ... with 336,766 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

filter() - March flights

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

```
## # A tibble: 28,834 × 19
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>         <dbl>   <int>         <int>
## 1  2013     3     1         4           2159          125     318             56
## 2  2013     3     1        50           2358           52     526            438
## 3  2013     3     1       117           2245          152     223           2354
## 4  2013     3     1       454           500            -6     633            648
## 5  2013     3     1      505           515           -10     746            810
## 6  2013     3     1      521           530            -9     813            827
## 7  2013     3     1      537           540            -3     856            850
## 8  2013     3     1      541           545            -4    1014           1023
## 9  2013     3     1      549           600           -11     639            703
## 10 2013     3     1      550           600           -10     747            801
## # ... with 28,824 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

filter() - Flights in the first 7 days of March

```
flights %>% filter(month == 3, day <= 7)
```

```
## # A tibble: 6,530 × 19
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>         <dbl>   <int>         <int>
## 1  2013     3     1         4           2159          125     318             56
## 2  2013     3     1        50           2358           52     526            438
## 3  2013     3     1       117           2245          152     223           2354
## 4  2013     3     1       454           500            -6     633            648
## 5  2013     3     1      505           515           -10     746            810
## 6  2013     3     1      521           530            -9     813            827
## 7  2013     3     1      537           540            -3     856            850
## 8  2013     3     1      541           545            -4    1014           1023
## 9  2013     3     1      549           600           -11     639            703
## 10 2013     3     1      550           600           -10     747            801
## # ... with 6,520 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

filter() - Flights to LAX or JFK in March

```
flights %>% filter(dest == "LAX" | dest == "JFK", month==3)
```

```
## # A tibble: 1,178 × 19
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>         <dbl>   <int>         <int>
## 1  2013     3     1     607           610           -3     832           925
## 2  2013     3     1     629           632           -3     844           952
## 3  2013     3     1     657           700           -3     953          1034
## 4  2013     3     1     714           715            -1     939          1037
## 5  2013     3     1     716           710            6     958          1035
## 6  2013     3     1     727           730           -3    1007          1100
## 7  2013     3     1     836           840           -4    1111          1157
## 8  2013     3     1     857           900           -3    1202          1221
## 9  2013     3     1     903           900            3    1157          1220
## 10 2013     3     1     904           831            33    1150          1151
## # ... with 1,168 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

slice() - First 10 flights

```
flights %>% slice(1:10)
```

```
## # A tibble: 10 × 19
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>         <dbl>   <int>         <int>
## 1  2013     1     1     517           515           2     830           819
## 2  2013     1     1     533           529           4     850           830
## 3  2013     1     1     542           540           2     923           850
## 4  2013     1     1     544           545          -1    1004          1022
## 5  2013     1     1     554           600          -6     812           837
## 6  2013     1     1     554           558          -4     740           728
## 7  2013     1     1     555           600          -5     913           854
## 8  2013     1     1     557           600          -3     709           723
## 9  2013     1     1     557           600          -3     838           846
## 10 2013     1     1     558           600          -2     753           745
## # ... with 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dtm>
```


slice() - Last 5 flights

```
flights %>% slice((n()-4):n())
```

```
## # A tibble: 5 × 19
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>         <dbl>   <int>         <int>
## 1  2013     9    30     NA           1455           NA     NA           1634
## 2  2013     9    30     NA           2200           NA     NA           2312
## 3  2013     9    30     NA           1210           NA     NA           1330
## 4  2013     9    30     NA           1159           NA     NA           1344
## 5  2013     9    30     NA            840           NA     NA           1020
## # ... with 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dtm>
```

```
flights %>% slice_tail(n = 5)
```

```
## # A tibble: 5 × 19
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>         <dbl>   <int>         <int>
## 1  2013     9    30     NA           1455           NA     NA           1634
## 2  2013     9    30     NA           2200           NA     NA           2312
## 3  2013     9    30     NA           1210           NA     NA           1330
## 4  2013     9    30     NA           1159           NA     NA           1344
## 5  2013     9    30     NA            840           NA     NA           1020
## # ... with 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dtm>
```

select() - Individual Columns

```
flights %>% select(year, month, day)
```

```
## # A tibble: 336,776 × 3
##   year month   day
##   <int> <int> <int>
## 1  2013     1     1
## 2  2013     1     1
## 3  2013     1     1
## 4  2013     1     1
## 5  2013     1     1
## 6  2013     1     1
## 7  2013     1     1
## 8  2013     1     1
## 9  2013     1     1
## 10 2013     1     1
## # ... with 336,766 more rows
```

select() - Exclude Columns

```
flights %>% select(-year, -month, -day)
```

```
## # A tibble: 336,776 × 16
##   dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier
##   <int>      <int>      <dbl>   <int>      <int>      <dbl> <chr>
## 1     517         515         2     830         819         11 UA
## 2     533         529         4     850         830         20 UA
## 3     542         540         2     923         850         33 AA
## 4     544         545        -1    1004        1022        -18 B6
## 5     554         600        -6     812         837        -25 DL
## 6     554         558        -4     740         728         12 UA
## 7     555         600        -5     913         854         19 B6
## 8     557         600        -3     709         723        -14 EV
## 9     557         600        -3     838         846         -8 B6
## 10    558         600        -2     753         745          8 AA
## # ... with 336,766 more rows, and 9 more variables: flight <int>, tailnum <chr>,
## #   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
## #   minute <dbl>, time_hour <dtm>
```

select() - Ranges

```
flights %>% select(year:day)
```

```
## # A tibble: 336,776 × 3
##   year month   day
##   <int> <int> <int>
## 1  2013     1     1
## 2  2013     1     1
## 3  2013     1     1
## 4  2013     1     1
## 5  2013     1     1
## 6  2013     1     1
## 7  2013     1     1
## 8  2013     1     1
## 9  2013     1     1
## 10 2013     1     1
## # ... with 336,766 more rows
```

select() - Exclusion Ranges

```
flights %>% select(-(year:day))
```

```
## # A tibble: 336,776 × 16
##   dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier
##   <int>      <int>      <dbl>   <int>         <int>         <dbl> <chr>
## 1     517         515         2     830           819         11 UA
## 2     533         529         4     850           830         20 UA
## 3     542         540         2     923           850         33 AA
## 4     544         545        -1    1004          1022        -18 B6
## 5     554         600        -6     812           837        -25 DL
## 6     554         558        -4     740           728         12 UA
## 7     555         600        -5     913           854         19 B6
## 8     557         600        -3     709           723        -14 EV
## 9     557         600        -3     838           846         -8 B6
## 10    558         600        -2     753           745          8 AA
## # ... with 336,766 more rows, and 9 more variables: flight <int>, tailnum <chr>,
## #   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
## #   minute <dbl>, time_hour <dtm>
```

select() - Matching

```
flights %>% select(contains("dep"),
                  contains("arr"))
```

```
## # A tibble: 336,776 × 7
##   dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier
##   <int>      <int>      <dbl>   <int>      <int>      <dbl> <chr>
## 1     517         515         2     830         819         11 UA
## 2     533         529         4     850         830         20 UA
## 3     542         540         2     923         850         33 AA
## 4     544         545        -1    1004        1022        -18 B6
## 5     554         600        -6     812         837        -25 DL
## 6     554         558        -4     740         728         12 UA
## 7     555         600        -5     913         854         19 B6
## 8     557         600        -3     709         723        -14 EV
## 9     557         600        -3     838         846         -8 B6
## 10    558         600        -2     753         745          8 AA
## # ... with 336,766 more rows
```

```
flights %>% select(starts_with("dep"),
                  starts_with("arr"))
```

```
## # A tibble: 336,776 × 4
##   dep_time dep_delay arr_time arr_delay
##   <int>     <dbl>   <int>     <dbl>
## 1     517         2     830         11
## 2     533         4     850         20
## 3     542         2     923         33
## 4     544        -1    1004        -18
## 5     554        -6     812        -25
## 6     554        -4     740         12
## 7     555        -5     913         19
## 8     557        -3     709        -14
## 9     557        -3     838         -8
## 10    558         -2     753          8
## # ... with 336,766 more rows
```

Other helpers provide by tidyselct:

starts_with, ends_with, everything, matches, num_range, one_of, everything, last_col.

select() + where() - Get numeric columns

```
flights %>% select(where(is.numeric))
```

```
## # A tibble: 336,776 × 14
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>         <dbl>   <int>         <int>
## 1  2013     1     1     517           515           2     830           819
## 2  2013     1     1     533           529           4     850           830
## 3  2013     1     1     542           540           2     923           850
## 4  2013     1     1     544           545          -1    1004          1022
## 5  2013     1     1     554           600          -6     812           837
## 6  2013     1     1     554           558          -4     740           728
## 7  2013     1     1     555           600          -5     913           854
## 8  2013     1     1     557           600          -3     709           723
## 9  2013     1     1     557           600          -3     838           846
## 10 2013     1     1     558           600          -2     753           745
## # ... with 336,766 more rows, and 6 more variables: arr_delay <dbl>,
## #   flight <int>, air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>
```

```
flights %>% select(where(function(x) !is.numeric(x)))
```

```
## # A tibble: 336,776 × 5
##   carrier tailnum origin dest   time_hour
##   <chr>   <chr>   <chr> <chr> <dtm>
## 1 UA      N14228   EWR   IAH   2013-01-01 05:00:00
```

relocate - to the front

```
flights %>% relocate(carrier, origin, dest)
```

```
## # A tibble: 336,776 × 19
##   carrier origin dest   year month   day dep_time sched_dep_time dep_delay
##   <chr>   <chr> <chr> <int> <int> <int>   <int>         <int>         <dbl>
## 1 UA      EWR    IAH    2013     1     1     517           515           2
## 2 UA      LGA    IAH    2013     1     1     533           529           4
## 3 AA      JFK    MIA    2013     1     1     542           540           2
## 4 B6      JFK    BQN    2013     1     1     544           545          -1
## 5 DL      LGA    ATL    2013     1     1     554           600          -6
## 6 UA      EWR    ORD    2013     1     1     554           558          -4
## 7 B6      EWR    FLL    2013     1     1     555           600          -5
## 8 EV      LGA    IAD    2013     1     1     557           600          -3
## 9 B6      JFK    MCO    2013     1     1     557           600          -3
## 10 AA     LGA    ORD    2013     1     1     558           600          -2
## # ... with 336,766 more rows, and 10 more variables: arr_time <int>,
## #   sched_arr_time <int>, arr_delay <dbl>, flight <int>, tailnum <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

relocate - to the end

```
flights %>%  
  relocate(year, month, day, .after = last_col())
```

```
## # A tibble: 336,776 × 19  
##   dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier  
##   <int>      <int>      <dbl>   <int>      <int>      <dbl> <chr>  
## 1     517         515         2     830         819         11 UA  
## 2     533         529         4     850         830         20 UA  
## 3     542         540         2     923         850         33 AA  
## 4     544         545        -1    1004        1022        -18 B6  
## 5     554         600        -6     812         837        -25 DL  
## 6     554         558        -4     740         728         12 UA  
## 7     555         600        -5     913         854         19 B6  
## 8     557         600        -3     709         723        -14 EV  
## 9     557         600        -3     838         846         -8 B6  
## 10    558         600        -2     753         745          8 AA  
## # ... with 336,766 more rows, and 12 more variables: flight <int>, tailnum <chr>,  
## #   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,  
## #   minute <dbl>, time_hour <dtm>, year <int>, month <int>, day <int>
```

rename() - Change column names

```
flights %>% rename(tail_number = tailnum)
```

```
## # A tibble: 336,776 × 19
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>         <dbl>   <int>         <int>
## 1  2013     1     1     517           515           2     830           819
## 2  2013     1     1     533           529           4     850           830
## 3  2013     1     1     542           540           2     923           850
## 4  2013     1     1     544           545          -1    1004          1022
## 5  2013     1     1     554           600          -6     812           837
## 6  2013     1     1     554           558          -4     740           728
## 7  2013     1     1     555           600          -5     913           854
## 8  2013     1     1     557           600          -3     709           723
## 9  2013     1     1     557           600          -3     838           846
## 10 2013     1     1     558           600          -2     753           745
## # ... with 336,766 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tail_number <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

select() vs. rename()

```
flights %>% select(tail_number = tailnum)
```

```
## # A tibble: 336,776 × 1
##   tail_number
##   <chr>
## 1 N14228
## 2 N24211
## 3 N619AA
## 4 N804JB
## 5 N668DN
## 6 N39463
## 7 N516JB
## 8 N829AS
## 9 N593JB
## 10 N3ALAA
## # ... with 336,766 more rows
```

```
flights %>% rename(tail_number = tailnum)
```

```
## # A tibble: 336,776 × 19
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>         <dbl>   <int>         <int>
## 1  2013     1     1     517             515           2       830           819
## 2  2013     1     1     533             529           4       850           830
```

pull()

```
names(flights)
```

```
## [1] "year"           "month"          "day"            "dep_time"  
## [5] "sched_dep_time" "dep_delay"     "arr_time"      "sched_arr_time"  
## [9] "arr_delay"     "carrier"       "flight"        "tailnum"  
## [13] "origin"        "dest"          "air_time"      "distance"  
## [17] "hour"          "minute"        "time_hour"
```

```
flights %>% pull("year") %>% head()
```

```
## [1] 2013 2013 2013 2013 2013 2013
```

```
flights %>% pull(1) %>% head()
```

```
## [1] 2013 2013 2013 2013 2013 2013
```

```
flights %>% pull(-1) %>% head()
```

```
## [1] "2013-01-01 05:00:00 EST" "2013-01-01 05:00:00 EST"  
## [3] "2013-01-01 05:00:00 EST" "2013-01-01 05:00:00 EST"  
## [5] "2013-01-01 06:00:00 EST" "2013-01-01 05:00:00 EST"
```

```
flights %>% .[["year"]] %>% head()
```

arrange() - Sort data

```
flights %>% filter(month==3,day==2) %>% arrange(origin, dest)
```

```
## # A tibble: 765 × 19
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>         <dbl>   <int>         <int>
## 1  2013     3     2    1336           1329             7     1426           1432
## 2  2013     3     2     628             629            -1     837            849
## 3  2013     3     2     637             640            -3     903            915
## 4  2013     3     2     743             745            -2     945           1010
## 5  2013     3     2     857             900            -3    1117           1126
## 6  2013     3     2    1027            1030            -3    1234           1247
## 7  2013     3     2    1134            1145           -11    1332           1359
## 8  2013     3     2    1412            1415            -3    1636           1630
## 9  2013     3     2    1633            1636            -3    1848           1908
## 10 2013     3     2    1655            1700            -5    1857           1924
## # ... with 755 more rows, and 11 more variables: arr_delay <dbl>, carrier <chr>,
## #   flight <int>, tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>,
## #   distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

arrange() & desc() - Descending order

```
flights %>%  
  filter(month==3, day==2) %>%  
  arrange(desc(origin), dest) %>%  
  select(origin, dest, tailnum)
```

```
## # A tibble: 765 × 3  
##   origin dest tailnum  
##   <chr> <chr> <chr>  
## 1 LGA    ATL    N928AT  
## 2 LGA    ATL    N623DL  
## 3 LGA    ATL    N680DA  
## 4 LGA    ATL    N996AT  
## 5 LGA    ATL    N510MQ  
## 6 LGA    ATL    N663DN  
## 7 LGA    ATL    N942DL  
## 8 LGA    ATL    N511MQ  
## 9 LGA    ATL    N910DE  
## 10 LGA   ATL    N902DE  
## # ... with 755 more rows
```


mutate() - Modify columns

```
flights %>%  
  select(year:day) %>%  
  mutate(date = paste(year, month, day, sep="/"))
```

```
## # A tibble: 336,776 × 4  
##   year month   day date  
##   <int> <int> <int> <chr>  
## 1  2013     1     1 2013/1/1  
## 2  2013     1     1 2013/1/1  
## 3  2013     1     1 2013/1/1  
## 4  2013     1     1 2013/1/1  
## 5  2013     1     1 2013/1/1  
## 6  2013     1     1 2013/1/1  
## 7  2013     1     1 2013/1/1  
## 8  2013     1     1 2013/1/1  
## 9  2013     1     1 2013/1/1  
## 10 2013     1     1 2013/1/1  
## # ... with 336,766 more rows
```

distinct() - Find unique rows

```
flights %>%  
  select(origin, dest) %>%  
  distinct() %>%  
  arrange(origin,dest)
```

```
## # A tibble: 224 × 2  
##   origin dest  
##   <chr> <chr>  
## 1 EWR    ALB  
## 2 EWR    ANC  
## 3 EWR    ATL  
## 4 EWR    AUS  
## 5 EWR    AVL  
## 6 EWR    BDL  
## 7 EWR    BNA  
## 8 EWR    BOS  
## 9 EWR    BQN  
## 10 EWR   BTV  
## # ... with 214 more rows
```

summarise()

```
flights %>%  
  summarize(n(), min(dep_delay), max(dep_delay))
```

```
## # A tibble: 1 × 3  
##   `n()` `min(dep_delay)` `max(dep_delay)`  
##   <int>         <dbl>         <dbl>  
## 1 336776           NA             NA
```

```
flights %>%  
  summarize(  
    n = n(),  
    min_dep_delay = min(dep_delay, na.rm = TRUE),  
    max_dep_delay = max(dep_delay, na.rm = TRUE)  
  )
```

```
## # A tibble: 1 × 3  
##       n min_dep_delay max_dep_delay  
##   <int>         <dbl>         <dbl>  
## 1 336776          -43           1301
```

group_by()

```
flights %>% group_by(origin)
```

```
## # A tibble: 336,776 × 19
## # Groups:   origin [3]
##   year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>   <int>         <int>         <dbl>   <int>         <int>
## 1  2013     1     1     517             515           2     830           819
## 2  2013     1     1     533             529           4     850           830
## 3  2013     1     1     542             540           2     923           850
## 4  2013     1     1     544             545          -1    1004          1022
## 5  2013     1     1     554             600          -6     812           837
## 6  2013     1     1     554             558          -4     740           728
## 7  2013     1     1     555             600          -5     913           854
## 8  2013     1     1     557             600          -3     709           723
## 9  2013     1     1     557             600          -3     838           846
## 10 2013     1     1     558             600          -2     753           745
## # ... with 336,766 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dtm>
```

summarise() with group_by()

```
flights %>%
  group_by(origin) %>%
  summarize(
    n = n(),
    min_dep_delay = min(dep_delay, na.rm = TRUE),
    max_dep_delay = max(dep_delay, na.rm = TRUE)
  )
```

```
## # A tibble: 3 × 4
##   origin    n min_dep_delay max_dep_delay
##   <chr> <int>      <dbl>      <dbl>
## 1 EWR    120835         -25         1126
## 2 JFK    111279         -43         1301
## 3 LGA    104662         -33          911
```

Groups after summarise

```
flights %>%
  group_by(origin) %>%
  summarize(
    n = n(),
    min_dep_delay = min(dep_delay, na.rm = TRUE),
    max_dep_delay = max(dep_delay, na.rm = TRUE),
    .groups = "drop_last"
  )
```

```
## # A tibble: 3 × 4
##   origin      n min_dep_delay max_dep_delay
##   <chr>    <int>         <dbl>         <dbl>
## 1 EWR     120835          -25           1126
## 2 JFK     111279          -43           1301
## 3 LGA     104662          -33            911
```

```
flights %>%
  group_by(origin) %>%
  summarize(
    n = n(),
    min_dep_delay = min(dep_delay, na.rm = TRUE),
    max_dep_delay = max(dep_delay, na.rm = TRUE),
    .groups = "keep"
  )
```

```
## # A tibble: 3 × 4
## # Groups:   origin [3]
##   origin      n min_dep_delay max_dep_delay
##   <chr>    <int>         <dbl>         <dbl>
## 1 EWR     120835          -25           1126
## 2 JFK     111279          -43           1301
## 3 LGA     104662          -33            911
```

```
flights %>%
  group_by(origin, carrier) %>%
  summarize(
    n = n(),
    min_dep_delay = min(dep_delay, na.rm = TRUE),
    max_dep_delay = max(dep_delay, na.rm = TRUE)
  ) %>%
  filter(n > 10000)
```

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

```
## # A tibble: 10 × 5
## # Groups:   origin [3]
##   origin carrier      n min_dep_delay max_dep_delay
##   <chr>   <chr>   <int>         <dbl>         <dbl>
## 1 EWR     EV     43939          -25           548
## 2 EWR     UA     46087          -18           424
## 3 JFK     9E     14651          -24           747
## 4 JFK     AA     13783          -15          1014
## 5 JFK     B6     42076          -43           453
## 6 JFK     DL     20701          -18           960
## 7 LGA     AA     15459          -24           803
## 8 LGA     DL     23067          -33           911
## 9 LGA     MQ     16928          -26           366
## 10 LGA    US     13136          -18           500
```

count()

```
flights %>%  
  group_by(origin, carrier) %>%  
  summarize(n = n(), .groups = "drop")
```

```
## # A tibble: 35 × 3  
##   origin carrier     n  
##   <chr> <chr>   <int>  
## 1 EWR    9E     1268  
## 2 EWR    AA     3487  
## 3 EWR    AS      714  
## 4 EWR    B6     6557  
## 5 EWR    DL     4342  
## 6 EWR    EV    43939  
## 7 EWR    MQ     2276  
## 8 EWR    OO        6  
## 9 EWR    UA    46087  
## 10 EWR   US     4405  
## # ... with 25 more rows
```

```
flights %>%  
  count(origin, carrier)
```

```
## # A tibble: 35 × 3  
##   origin carrier     n  
##   <chr> <chr>   <int>  
## 1 EWR    9E     1268  
## 2 EWR    AA     3487  
## 3 EWR    AS      714  
## 4 EWR    B6     6557  
## 5 EWR    DL     4342  
## 6 EWR    EV    43939  
## 7 EWR    MQ     2276  
## 8 EWR    OO        6  
## 9 EWR    UA    46087  
## 10 EWR   US     4405  
## # ... with 25 more rows
```


mutate() with group_by()

```
flights %>% group_by(origin) %>%  
  mutate(  
    n = n(),  
  ) %>%  
  select(origin, n)
```

```
## # A tibble: 336,776 × 2  
## # Groups:   origin [3]  
##   origin      n  
##   <chr>   <int>  
## 1 EWR    120835  
## 2 LGA    104662  
## 3 JFK    111279  
## 4 JFK    111279  
## 5 LGA    104662  
## 6 EWR    120835  
## 7 EWR    120835  
## 8 LGA    104662  
## 9 JFK    111279  
## 10 LGA    104662  
## # ... with 336,766 more rows
```

Demos

1. How many flights to Los Angeles (LAX) did each of the legacy carriers (AA, UA, DL or US) have in May from JFK, and what was their average duration?
2. What was the shortest flight out of each airport in terms of distance? In terms of duration?

Exercise 1

1. Which plane (check the tail number) flew out of each New York airport the most?
2. Which date should you fly on if you want to have the lowest possible average departure delay? What about arrival delay?