Data Wrangling Workshop Slides

Christopher Kinson

What is data wrangling?

Data wrangling is the notion of cleaning and preparing a dataset for analysis. Wrangling may also be known as munging. Wrangling can often take a long time to complete and typically is the most critical aspect of data science.

Tidyverse for data wrangling

The **tidyverse** is a collection of R packages and a workflow for working with data. The tidyverse is supported by Posit (the company that made RStudio). See more details at https://www.tidyverse.org/.

The packages that are installed and loaded when running the **tidyverse** package are:

1. **dplyr**
2. **forcats**
3. **ggplot2**
4. **lubridate**
5. **purrr**
6. **readr**
7. **stringr**
8. **tibble**
9. **tidyr**
10. **magrittr**

First, we'll import the the City of Urbana's Rental Inspection Grades Listing Data with the `read_csv()` function and print a snapshot of it.

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.2     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
```

```
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
rentalsData <- read_csv("https://uofi.box.com/shared/static/y5rysysc7ycpe6nedwfzai58wkmqbtkw.csv",
    col_types = cols(`Inspection Date` = col_date(format = "%m/%d/%Y")))
rentalsData
```

```
## # A tibble: 1,738 x 7
##    `Property Address`   `Parcel Number` `Inspection Date` Grade `License Status`
##    <chr>                          <dbl> <date>            <chr> <chr>
##  1 1208 Beech Street       912107259011 2011-06-06        Clas~ Issued
##  2 1310 Ellis Drive        912107208017 2011-12-20        Clas~ Issued
##  3 310 Crystal Lake Dr~    912108404027 2010-11-09        Clas~ Issued
##  4 807 1/2 West Main S~    912108354004 2011-05-18        Clas~ Issued
##  5 1204 1/2 East Michi~    922116376028 2013-05-15        Clas~ Issued
##  6 601 A Glover Avenue     922116177015 2012-11-29        Clas~ Issued
##  7 108 L North Busey A~    912108360001 2010-10-12        Clas~ Expired
##  8 606 1/2 West Elm St~    922117110004 2013-06-12        Clas~ Issued
##  9 1206 1/2 East Michi~    922116376029 2013-05-15        Clas~ Issued
## 10 804 1/2 East Main S~    912109353015 2012-02-28        Clas~ Issued
## # i 1,728 more rows
## # i 2 more variables: `Expiration Date` <chr>, `Mappable Address` <chr>
```

One of the **tidyverse's** most useful operators is the pipe operator %>%. It is a concise way to achieve nested functions by taking the first code on the left side of the first pipe operator and places it into a pipeline of operations to be used with subsequent pipe operators.

For example, here's another way to import the `rentalsData` and print it with the pipe operator %>%.

```
read_csv("https://uofi.box.com/shared/static/y5rysysc7ycpe6nedwfzai58wkmqbtkw.csv") %>%
  print()
```

```
## Rows: 1738 Columns: 7
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (6): Property Address, Inspection Date, Grade, License Status, Expiratio...
## dbl (1): Parcel Number
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
## # A tibble: 1,738 x 7
##    `Property Address`   `Parcel Number` `Inspection Date` Grade `License Status`
##    <chr>                          <dbl> <chr>             <chr> <chr>
##  1 1208 Beech Street       912107259011 06/06/2011        Clas~ Issued
##  2 1310 Ellis Drive        912107208017 12/20/2011        Clas~ Issued
##  3 310 Crystal Lake Dr~    912108404027 11/09/2010        Clas~ Issued
##  4 807 1/2 West Main S~    912108354004 05/18/2011        Clas~ Issued
##  5 1204 1/2 East Michi~    922116376028 05/15/2013        Clas~ Issued
##  6 601 A Glover Avenue     922116177015 11/29/2012        Clas~ Issued
##  7 108 L North Busey A~    912108360001 10/12/2010        Clas~ Expired
##  8 606 1/2 West Elm St~    922117110004 06/12/2013        Clas~ Issued
```

```
##  9 1206 1/2 East Michi~    922116376029 05/15/2013        Clas~ Issued
## 10 804 1/2 East Main S~    912109353015 02/28/2012        Clas~ Issued
## # i 1,728 more rows
## # i 2 more variables: `Expiration Date` <chr>, `Mappable Address` <chr>
```

Data arrangement

Arranging a dataset involves organizing its columns and sorting the data by one or more of its columns. Some clients, analysts, and data scientists may request data to be arranged a certain way in order for them to do their work.

Organizing columns

By organizing the data, we may want certain columns to appear as the first column, second column, third column, etc. We use the `select()` function for this task.

Let's arrange the `rentalsData` such that:

- the first column is Parcel Number
- the second column is Property Address
- the third column is Mappable Address
- the fourth column is Inspection Date
- the fifth column is Expiration Date
- the sixth column is License Status
- the seventh column is Grade

```
rentalsData %>%
  select(`Parcel Number`, `Property Address`, `Mappable Address`, `Inspection Date`, `Expiration Date`,
```

```
## # A tibble: 1,738 x 7
##    `Parcel Number` `Property Address`       `Mappable Address` `Inspection Date`
##              <dbl> <chr>                    <chr>              <date>
##  1    912107259011 1208 Beech Street        "1208 Beech Stree~ 2011-06-06
##  2    912107208017 1310 Ellis Drive         "1310 Ellis Drive~ 2011-12-20
##  3    912108404027 310 Crystal Lake Drive   "310 Crystal Lake~ 2010-11-09
##  4    912108354004 807 1/2 West Main Street "807 1 2 West Mai~ 2011-05-18
##  5    922116376028 1204 1/2 East Michigan ~ "1204 1 2 East Mi~ 2013-05-15
##  6    922116177015 601 A Glover Avenue      "601 A Glover Ave~ 2012-11-29
##  7    912108360001 108 L North Busey Avenue "108 L North Buse~ 2010-10-12
##  8    922117110004 606 1/2 West Elm Street  "606 1 2 West Elm~ 2013-06-12
##  9    922116376029 1206 1/2 East Michigan ~ "1206 1 2 East Mi~ 2013-05-15
## 10    912109353015 804 1/2 East Main Street "804 1 2 East Mai~ 2012-02-28
## # i 1,728 more rows
## # i 3 more variables: `Expiration Date` <chr>, `License Status` <chr>,
## #   Grade <chr>
```

PRACTICE: Organizing the `rentalsData` such that the Grade column appears as the first column, and all other columns appear afterwards in their original fashion.

```
rentalsData %>%
  select(Grade, everything())
```

```
## # A tibble: 1,738 x 7
##    Grade   `Property Address` `Parcel Number` `Inspection Date` `License Status`
##    <chr>   <chr>                        <dbl> <date>            <chr>
##  1 Class C 1208 Beech Street     912107259011 2011-06-06        Issued
##  2 Class A 1310 Ellis Drive      912107208017 2011-12-20        Issued
##  3 Class B 310 Crystal Lake ~    912108404027 2010-11-09        Issued
##  4 Class A 807 1/2 West Main~    912108354004 2011-05-18        Issued
##  5 Class B 1204 1/2 East Mic~    922116376028 2013-05-15        Issued
##  6 Class A 601 A Glover Aven~    922116177015 2012-11-29        Issued
##  7 Class C 108 L North Busey~    912108360001 2010-10-12        Expired
##  8 Class C 606 1/2 West Elm ~    922117110004 2013-06-12        Issued
##  9 Class B 1206 1/2 East Mic~    922116376029 2013-05-15        Issued
## 10 Class B 804 1/2 East Main~    912109353015 2012-02-28        Issued
## # i 1,728 more rows
## # i 2 more variables: `Expiration Date` <chr>, `Mappable Address` <chr>
```

```
rentalsData %>%
  select(Grade, everything())
```

Sorting columns

Sorting the data by the values in the columns involves the tidyverse's `arrange()` function. The default order within this function is ascending order. To achieve descending order, we enclose a column in the `desc()` function.

Now, let's sort the `rentalsData` such that the Parcel Number column is in descending order and print the result.

```
rentalsData %>%
  arrange(desc(`Parcel Number`)) %>%
  print()
```

```
## # A tibble: 1,738 x 7
##    `Property Address`   `Parcel Number` `Inspection Date` Grade `License Status`
##    <chr>                          <dbl> <date>            <chr> <chr>
##  1 1611 East Lexington~    932128406013 2017-12-05        Clas~ Issued
##  2 1708 East Lexington~    932128405023 2017-12-13        Clas~ Issued
##  3 1914 East Galena St~    932128276006 2017-11-29        Clas~ Issued
##  4 1705 East Trails Dr~    932128255024 2017-12-05        Clas~ Issued
##  5 1704 East Trails Dr~    932128253015 2018-02-12        Clas~ Issued
##  6 1704 East Trails Dr~    932128253014 2018-02-15        Clas~ Issued
##  7 1702 East Trails Dr~    932128253005 2017-12-08        Clas~ Issued
##  8 1905 East Summit Dr~    932128229003 2017-12-04        Clas~ Issued
##  9 2901 South Myra Rid~    932128226013 2018-03-01        Clas~ Issued
## 10 3007 A East Windsor~    932127200016 2017-12-14        Clas~ Issued
## # i 1,728 more rows
```

```
## # i 2 more variables: `Expiration Date` <chr>, `Mappable Address` <chr>
```

Let's sort such that the License Status column is in ascending order, then Parcel Number column is in descending order, and print the result.

```
rentalsData %>%
  arrange(`License Status`, desc(`Parcel Number`)) %>%
  print()
```

```
## # A tibble: 1,738 x 7
##    `Property Address`   `Parcel Number` `Inspection Date` Grade `License Status`
##    <chr>                          <dbl> <date>            <chr> <chr>
##  1 1012 North Geraldin~    912108278007 2021-11-23        Clas~ Approved
##  2 905 East Pennsylvan~    922116353003 2016-03-29        Clas~ Contract Sale
##  3 504 Stoughton Street    912108363020 2010-10-25        Clas~ Contract Sale
##  4 105 North Coler Ave~    912108360009 2010-10-25        Clas~ Contract Sale
##  5 3030 East Stillwate~    932122406018 2017-11-14        Clas~ Expired
##  6 1401 East Scovill S~    932121381011 2017-03-27        Clas~ Expired
##  7 1109 East Scovill S~    932121379008 2017-03-21        Clas~ Expired
##  8 2304 South Lynn Str~    932121376009 2017-04-17        Clas~ Expired
##  9 804 East Shurts Str~    932121355017 2017-03-02        Clas~ Expired
## 10 2207 South Philo Ro~    932121332017 2017-05-11        Clas~ Expired
## # i 1,728 more rows
## # i 2 more variables: `Expiration Date` <chr>, `Mappable Address` <chr>
```

Data Reduction

Data reduction may also be known as subsetting. Data reduction tasks are done by data workers so often that they become second nature. Data reduction tasks include filtering, slicing, and selecting.

Filtering rows

We can select rows or observations through conditions with the `filter()` function in the tidyverse. For example, we can filter only the rows that have Grade of "Class F" of `rentalsData`.

```
rentalsData %>%
  filter(Grade=="Class F")
```

```
## # A tibble: 2 x 7
##   `Property Address` `Parcel Number` `Inspection Date` Grade   `License Status`
##   <chr>                        <dbl> <date>            <chr>   <chr>
## 1 1304 Silver Street    932121181019 2013-02-25        Class F Expired
## 2 1302 Silver Street    932121181018 2013-02-25        Class F Expired
## # i 2 more variables: `Expiration Date` <chr>, `Mappable Address` <chr>
```

We can filter only the rows that have Grade of "Class A" or "Class F" (where the pipe key | means "or").

```
rentalsData %>%
  filter(Grade=="Class A" | Grade=="Class F")
```

```
## # A tibble: 163 x 7
##    `Property Address`   `Parcel Number` `Inspection Date` Grade `License Status`
```

```
##    <chr>                          <dbl> <date>           <chr> <chr>
##  1 1310 Ellis Drive        912107208017 2011-12-20       Clas~ Issued
##  2 807 1/2 West Main S~    912108354004 2011-05-18       Clas~ Issued
##  3 601 A Glover Avenue     922116177015 2012-11-29       Clas~ Issued
##  4 2108 East Michigan ~    912115306026 2018-02-23       Clas~ Issued
##  5 1208 South Lanore D~    922116477004 2018-03-15       Clas~ Issued
##  6 1209 North Busey Av~    912108152020 2010-12-29       Clas~ Issued
##  7 1806 South Glenwood~    932121108022 2017-09-07       Clas~ Issued
##  8 610 West Iowa Street    932117305027 2008-01-03       Clas~ Issued
##  9 402 East Kerr Avenue    912108277022 2021-10-27       Clas~ Issued
## 10 506 West Iowa Street    932117326017 2019-05-31       Clas~ Issued
## # i 153 more rows
## # i 2 more variables: `Expiration Date` <chr>, `Mappable Address` <chr>
```

We filter the rows corresponding to the Grade of "Class A" and Inspection Date value prior to year 2021.

```
rentalsData %>%
  filter(Grade=="Class A", `Inspection Date`<"2021-01-01")
```

```
## # A tibble: 141 x 7
##    `Property Address`   `Parcel Number` `Inspection Date` Grade `License Status`
##    <chr>                          <dbl> <date>           <chr> <chr>
##  1 1310 Ellis Drive        912107208017 2011-12-20       Clas~ Issued
##  2 807 1/2 West Main S~    912108354004 2011-05-18       Clas~ Issued
##  3 601 A Glover Avenue     922116177015 2012-11-29       Clas~ Issued
##  4 2108 East Michigan ~    912115306026 2018-02-23       Clas~ Issued
##  5 1208 South Lanore D~    922116477004 2018-03-15       Clas~ Issued
##  6 1209 North Busey Av~    912108152020 2010-12-29       Clas~ Issued
##  7 1806 South Glenwood~    932121108022 2017-09-07       Clas~ Issued
##  8 610 West Iowa Street    932117305027 2008-01-03       Clas~ Issued
##  9 506 West Iowa Street    932117326017 2019-05-31       Clas~ Issued
## 10 105 West Washington~    932117401012 2009-03-25       Clas~ Issued
## # i 131 more rows
## # i 2 more variables: `Expiration Date` <chr>, `Mappable Address` <chr>
```

PRACTICE: Filter the `rentalsData` such that the Inspection Dates are between Memorial Day and Veteran's Day of year 2015 and arrange the result such that Inspection Dates are sorted from oldest to newest.

```
rentalsData %>%
  filter(`Inspection Date`>="2015-05-25", `Inspection Date`<="2015-11-11") %>%
  arrange(`Inspection Date`)
```

```
## # A tibble: 25 x 7
##    `Property Address`   `Parcel Number` `Inspection Date` Grade `License Status`
##    <chr>                          <dbl> <date>           <chr> <chr>
##  1 1209 East Main Stre~    922116127006 2015-05-28       Clas~ Temporarily Not~
##  2 402 Grove Street        922117245001 2015-05-29       Clas~ Issued
##  3 2308 Lantern Hill D~    912115302007 2015-06-03       Clas~ Issued
##  4 402 South Cottage G~    922116130007 2015-06-03       Clas~ Issued
##  5 1109 1/2 East Main ~    922116126007 2015-06-05       Clas~ Issued
##  6 706 East Main Street    912109352012 2015-06-11       Clas~ Issued
##  7 207 South Cottage G~    922116104035 2015-06-11       Clas~ Issued
##  8 401 South Webber St~    922116107004 2015-06-12       Clas~ Issued
```

```
##  9 909 South Cottage G~    922116304018 2015-06-12        Clas~ Issued
## 10 102 Grove Street        922117232001 2015-06-12        Clas~ Issued
## # i 15 more rows
## # i 2 more variables: `Expiration Date` <chr>, `Mappable Address` <chr>
```

```
rentalsData %>%
  filter(`Inspection Date`>="2015-05-25", `Inspection Date`<="2015-11-11") %>%
  arrange(`Inspection Date`)
```

Slicing rows

If we struggle to recall any data columns and their values, we can reduce the data using values that represent the row location via the `slice()` function. For example, we can slice rows 1000 to 1111 of the `rentalsData`.

```
rentalsData %>%
  slice(1000:1111)
```

```
## # A tibble: 112 x 7
##    `Property Address`   `Parcel Number` `Inspection Date` Grade `License Status`
##    <chr>                          <dbl> <date>            <chr> <chr>
##  1 1009 South Country ~    922116426019 2014-08-08        Clas~ Issued
##  2 708 West Illinois S~    922117108009 2007-08-20        Clas~ Issued
##  3 307 East Pennsylvan~    932117478003 2021-07-15        Clas~ Expired
##  4 2103 East Lantern H~    912115304004 2018-02-12        Clas~ Issued
##  5 805 East High Street    922116110007 2012-04-10        Clas~ Issued
##  6 802 East Shurts Str~    932121355016 2017-04-11        Clas~ Issued
##  7 503 West California~    922117177005 2008-04-16        Clas~ Expired
##  8 814 Church Street       912108302014 2014-02-07        Clas~ Issued
##  9 1609 East Washingto~    922116402004 2014-07-21        Clas~ Issued
## 10 401 West Green Stre~    922117133005 2008-06-13        Clas~ Issued
## # i 102 more rows
## # i 2 more variables: `Expiration Date` <chr>, `Mappable Address` <chr>
```

We can show the first few rows with `slice_head()`.

```
rentalsData %>%
  slice_head(n=10)
```

```
## # A tibble: 10 x 7
##    `Property Address`   `Parcel Number` `Inspection Date` Grade `License Status`
##    <chr>                          <dbl> <date>            <chr> <chr>
##  1 1208 Beech Street       912107259011 2011-06-06        Clas~ Issued
##  2 1310 Ellis Drive        912107208017 2011-12-20        Clas~ Issued
##  3 310 Crystal Lake Dr~    912108404027 2010-11-09        Clas~ Issued
##  4 807 1/2 West Main S~    912108354004 2011-05-18        Clas~ Issued
##  5 1204 1/2 East Michi~    922116376028 2013-05-15        Clas~ Issued
##  6 601 A Glover Avenue     922116177015 2012-11-29        Clas~ Issued
##  7 108 L North Busey A~    912108360001 2010-10-12        Clas~ Expired
##  8 606 1/2 West Elm St~    922117110004 2013-06-12        Clas~ Issued
##  9 1206 1/2 East Michi~    922116376029 2013-05-15        Clas~ Issued
## 10 804 1/2 East Main S~    912109353015 2012-02-28        Clas~ Issued
## # i 2 more variables: `Expiration Date` <chr>, `Mappable Address` <chr>
```

We can show the last few rows with `slice_tail()`.

```
rentalsData %>%
  slice_tail(n=10)
```

```
## # A tibble: 10 x 7
##    `Property Address`   `Parcel Number` `Inspection Date` Grade `License Status`
##    <chr>                          <dbl> <date>            <chr> <chr>
##  1 1006 South Kinch St~    912115303003 2018-02-19        Clas~ Issued
##  2 1104 Lanore Drive       922116432002 2015-01-15        Clas~ Issued
##  3 3 Rainbow Court         922116430016 2015-01-09        Clas~ Issued
##  4 804 North Lincoln A~    912108301003 2010-07-20        Clas~ Issued
##  5 904 North Broadway ~    912108257026 2008-02-08        Clas~ Expired
##  6 905 Wabash Avenue       922116303018 2012-07-26        Clas~ Issued
##  7 802 Ohio Street         932117302011 2009-07-16        Clas~ Issued
##  8 409 West Griggs Str~    912108380002 2021-08-11        Clas~ Issued
##  9 606 East Elm Street     922117232006 2007-07-30        Clas~ Issued
## 10 1105 West Main Stre~    912107478006 2009-04-28        Clas~ Issued
## # i 2 more variables: `Expiration Date` <chr>, `Mappable Address` <chr>
```

We can show the smallest value for a specific column such as, Parcel Number, with the `slice_min()` or its largest value with `slice_max()`.

```
rentalsData %>%
  slice_min(`Parcel Number`)
```

```
## # A tibble: 1 x 7
##   `Property Address`    `Parcel Number` `Inspection Date` Grade `License Status`
##   <chr>                          <dbl> <date>            <chr> <chr>
## 1 2412 North Somerset ~    912103126029 2021-10-29        Clas~ Issued
## # i 2 more variables: `Expiration Date` <chr>, `Mappable Address` <chr>
```

We can show its largest value with `slice_max()`.

```
rentalsData %>%
  slice_max(`Parcel Number`)
```

```
## # A tibble: 1 x 7
##   `Property Address`    `Parcel Number` `Inspection Date` Grade `License Status`
##   <chr>                          <dbl> <date>            <chr> <chr>
## 1 1611 East Lexington ~    932128406013 2017-12-05        Clas~ Issued
## # i 2 more variables: `Expiration Date` <chr>, `Mappable Address` <chr>
```

The `slice_sample()` function slices a random set of rows.

```
rentalsData %>%
  slice_sample(n=10)
```

```
## # A tibble: 10 x 7
##    `Property Address`   `Parcel Number` `Inspection Date` Grade `License Status`
##    <chr>                          <dbl> <date>            <chr> <chr>
##  1 1504 Philo Road         922116453002 2013-04-30        Clas~ Issued
```

```
##  2 1712 James Cherry D~    922116456026 2013-05-06        Clas~ Issued
##  3 912 South Lierman A~    922116402042 2014-07-21        Clas~ Issued
##  4 112 South Poplar St~    922116127010 2012-06-20        Clas~ Expired
##  5 905 Springfield Ave~    932118227009 2009-01-28        Clas~ Issued
##  6 1211 Beech Street       912107260001 2011-06-17        Clas~ Issued
##  7 910 East Water Stre~    912109354005 2012-03-06        Clas~ Issued
##  8 409 East Kerr Avenue    912108280012 2021-10-19        Clas~ Issued
##  9 503 Urbana Avenue       922117276004 2007-04-10        Clas~ Issued
## 10 705 West Green Stre~    922117107006 2012-10-30        Clas~ Issued
## # i 2 more variables: `Expiration Date` <chr>, `Mappable Address` <chr>
```

Selecting columns

We can select certain columns using the `select()` function to reduce the dimension of a dataset. Selecting can be helpful when we don't need all data columns.

Working with the `rentalsData`, suppose we want to select only the Parcel Number and Mappable Address columns.

```
rentalsData %>%
  select(`Parcel Number`,`Mappable Address`)
```

```
## # A tibble: 1,738 x 2
##    `Parcel Number` `Mappable Address`
##              <dbl> <chr>
##  1    912107259011 "1208 Beech Street\nUrbana, IL\n(-88.2251, 40.1225)"
##  2    912107208017 "1310 Ellis Drive\nUrbana, IL\n(-88.2269, 40.1242)"
##  3    912108404027 "310 Crystal Lake Drive\nUrbana, IL\n(-88.2044, 40.1191)"
##  4    912108354004 "807 1 2 West Main Street\nUrbana, IL\n(40.114488, -88.21840~
##  5    922116376028 "1204 1 2 East Michigan Avenue\nUrbana, IL\n(40.101585, -88.~
##  6    922116177015 "601 A Glover Avenue\nUrbana, IL\n(40.108517, -88.193331)"
##  7    912108360001 "108 L North Busey Avenue\nUrbana, IL\n(40.113198, -88.21773~
##  8    922117110004 "606 1 2 West Elm Street\nUrbana, IL\n(40.111499, -88.215277~
##  9    922116376029 "1206 1 2 East Michigan Avenue\nUrbana, IL\n(40.101555, -88.~
## 10    912109353015 "804 1 2 East Main Street\nUrbana, IL\n(40.112998, -88.19879~
## # i 1,728 more rows
```

We can de-select (or remove) columns with the `minus` sign or with the `!` sign which represents negation. For example, we can remove the Mappable Address and License Status columns.

```
rentalsData %>%
  select(-`Mappable Address`, -`License Status`)
```

```
## # A tibble: 1,738 x 5
##    `Property Address`  `Parcel Number` `Inspection Date` Grade `Expiration Date`
##    <chr>                         <dbl> <date>            <chr> <chr>
##  1 1208 Beech Street      912107259011 2011-06-06        Clas~ <NA>
##  2 1310 Ellis Drive       912107208017 2011-12-20        Clas~ <NA>
##  3 310 Crystal Lake D~    912108404027 2010-11-09        Clas~ 10/14/2022
##  4 807 1/2 West Main ~    912108354004 2011-05-18        Clas~ 10/14/2020
##  5 1204 1/2 East Mich~    922116376028 2013-05-15        Clas~ 10/14/2018
##  6 601 A Glover Avenue    922116177015 2012-11-29        Clas~ 10/14/2020
```

```
##  7 108 L North Busey ~    912108360001 2010-10-12        Clas~ <NA>
##  8 606 1/2 West Elm S~    922117110004 2013-06-12        Clas~ <NA>
##  9 1206 1/2 East Mich~    922116376029 2013-05-15        Clas~ <NA>
## 10 804 1/2 East Main ~    912109353015 2012-02-28        Clas~ <NA>
## # i 1,728 more rows
```

```
rentalsData %>%
  select(!c(`Mappable Address`, `License Status`))
```

```
## # A tibble: 1,738 x 5
##    `Property Address`  `Parcel Number` `Inspection Date` Grade `Expiration Date`
##    <chr>                         <dbl> <date>            <chr> <chr>
##  1 1208 Beech Street      912107259011 2011-06-06        Clas~ <NA>
##  2 1310 Ellis Drive       912107208017 2011-12-20        Clas~ <NA>
##  3 310 Crystal Lake D~    912108404027 2010-11-09        Clas~ 10/14/2022
##  4 807 1/2 West Main ~    912108354004 2011-05-18        Clas~ 10/14/2020
##  5 1204 1/2 East Mich~    922116376028 2013-05-15        Clas~ 10/14/2018
##  6 601 A Glover Avenue    922116177015 2012-11-29        Clas~ 10/14/2020
##  7 108 L North Busey ~    912108360001 2010-10-12        Clas~ <NA>
##  8 606 1/2 West Elm S~    922117110004 2013-06-12        Clas~ <NA>
##  9 1206 1/2 East Mich~    922116376029 2013-05-15        Clas~ <NA>
## 10 804 1/2 East Main ~    912109353015 2012-02-28        Clas~ <NA>
## # i 1,728 more rows
```

PRACTICE: Select only the following columns: Property Address, Parcel Number, Inspection Date, Grade, License Status, and Expiration Date.

```
rentalsData %>%
  select(`Property Address`:`Expiration Date`)
```

```
## # A tibble: 1,738 x 6
##    `Property Address`   `Parcel Number` `Inspection Date` Grade `License Status`
##    <chr>                          <dbl> <date>            <chr> <chr>
##  1 1208 Beech Street       912107259011 2011-06-06        Clas~ Issued
##  2 1310 Ellis Drive        912107208017 2011-12-20        Clas~ Issued
##  3 310 Crystal Lake Dr~    912108404027 2010-11-09        Clas~ Issued
##  4 807 1/2 West Main S~    912108354004 2011-05-18        Clas~ Issued
##  5 1204 1/2 East Michi~    922116376028 2013-05-15        Clas~ Issued
##  6 601 A Glover Avenue     922116177015 2012-11-29        Clas~ Issued
##  7 108 L North Busey A~    912108360001 2010-10-12        Clas~ Expired
##  8 606 1/2 West Elm St~    922117110004 2013-06-12        Clas~ Issued
##  9 1206 1/2 East Michi~    922116376029 2013-05-15        Clas~ Issued
## 10 804 1/2 East Main S~    912109353015 2012-02-28        Clas~ Issued
## # i 1,728 more rows
## # i 1 more variable: `Expiration Date` <chr>
```

```
rentalsData %>%
  select(`Property Address`:`Expiration Date`)
```

Data expansion

Data expansion means making a dataset larger in some way, usually by adding new columns of information. We can create new columns that exist in addition to the original columns of a dataset.

Mutating columns

The `mutate()` function creates new columns and is often a function of an existing column in the data. Hence, mathematics may be involved in the `mutate()`.

Working with the `rentalsData`, suppose we want to create a new column called "New Parcel Number" which represents the Parcel Number divided by 100 and then select only the Parcel Number and New Parcel Number columns.

```
rentalsData %>%
  mutate(`New Parcel Number` = `Parcel Number`/10000) %>%
  select(`Parcel Number`, `New Parcel Number`)
```

```
## # A tibble: 1,738 x 2
##    `Parcel Number` `New Parcel Number`
##              <dbl>               <dbl>
##  1    912107259011           91210726.
##  2    912107208017           91210721.
##  3    912108404027           91210840.
##  4    912108354004           91210835.
##  5    922116376028           92211638.
##  6    922116177015           92211618.
##  7    912108360001           91210836.
##  8    922117110004           92211711.
##  9    922116376029           92211638.
## 10    912109353015           91210935.
## # i 1,728 more rows
```

Suppose we want to create a new column called "Truncated Parcels" which represents the floor of the last four digits of the Parcel Number column in `rentalsData`. Only print the Parcel Number and Truncated Parcels columns. Then we might require the following functions or operators in addition to `mutate()`: floor, `%%`, `*`, or `/`.

```
rentalsData %>%
  mutate(`Truncated Parcels` = floor(`Parcel Number`%%10000)) %>%
  select(`Parcel Number`,`Truncated Parcels`)
```

```
## # A tibble: 1,738 x 2
##    `Parcel Number` `Truncated Parcels`
##              <dbl>               <dbl>
##  1    912107259011                9011
##  2    912107208017                8017
##  3    912108404027                4027
##  4    912108354004                4004
##  5    922116376028                6028
##  6    922116177015                7015
##  7    912108360001                   1
##  8    922117110004                   4
```

```
##  9    922116376029               6029
## 10    912109353015               3015
## # i 1,728 more rows
```

Suppose we want to create a new column called "Numeric Grade" representing the Grade column as numeric values instead of character values, such that 1 is the lowest grade and 6 represents the highest grade.

```
rentalsData %>%
  mutate(grade_n = if_else(Grade=="Class N", "1", Grade),
         grade_f = if_else(Grade=="Class F", "2", grade_n),
         grade_d = if_else(Grade=="Class D", "3", grade_f),
         grade_c = if_else(Grade=="Class C", "4", grade_d),
         grade_b = if_else(Grade=="Class B", "5", grade_c),
         `Numeric Grade` = as.numeric(if_else(Grade=="Class A", "6", grade_b)))
```

```
## # A tibble: 1,738 x 13
##    `Property Address`   `Parcel Number` `Inspection Date` Grade `License Status`
##    <chr>                          <dbl> <date>            <chr> <chr>
##  1 1208 Beech Street       912107259011 2011-06-06        Clas~ Issued
##  2 1310 Ellis Drive        912107208017 2011-12-20        Clas~ Issued
##  3 310 Crystal Lake Dr~    912108404027 2010-11-09        Clas~ Issued
##  4 807 1/2 West Main S~    912108354004 2011-05-18        Clas~ Issued
##  5 1204 1/2 East Michi~    922116376028 2013-05-15        Clas~ Issued
##  6 601 A Glover Avenue     922116177015 2012-11-29        Clas~ Issued
##  7 108 L North Busey A~    912108360001 2010-10-12        Clas~ Expired
##  8 606 1/2 West Elm St~    922117110004 2013-06-12        Clas~ Issued
##  9 1206 1/2 East Michi~    922116376029 2013-05-15        Clas~ Issued
## 10 804 1/2 East Main S~    912109353015 2012-02-28        Clas~ Issued
## # i 1,728 more rows
## # i 8 more variables: `Expiration Date` <chr>, `Mappable Address` <chr>,
## #   grade_n <chr>, grade_f <chr>, grade_d <chr>, grade_c <chr>, grade_b <chr>,
## #   `Numeric Grade` <dbl>
```

PRACTICE: Create a new column called "Numeric Grade" representing the Grade column as numeric values instead of character values, such that 1 is the lowest grade and 6 represents the highest grade. Print the original seven columns of `rentalsData` and "Numeric Grade".

```
rentalsData %>%
  mutate(grade_n = if_else(Grade=="Class N", "1", Grade),
         grade_f = if_else(Grade=="Class F", "2", grade_n),
         grade_d = if_else(Grade=="Class D", "3", grade_f),
         grade_c = if_else(Grade=="Class C", "4", grade_d),
         grade_b = if_else(Grade=="Class B", "5", grade_c),
         `Numeric Grade` = as.numeric(if_else(Grade=="Class A", "6", grade_b))) %>%
  select(`Property Address`:`Mappable Address`,`Numeric Grade`)
```

```
## # A tibble: 1,738 x 8
##    `Property Address`   `Parcel Number` `Inspection Date` Grade `License Status`
##    <chr>                          <dbl> <date>            <chr> <chr>
##  1 1208 Beech Street       912107259011 2011-06-06        Clas~ Issued
##  2 1310 Ellis Drive        912107208017 2011-12-20        Clas~ Issued
##  3 310 Crystal Lake Dr~    912108404027 2010-11-09        Clas~ Issued
##  4 807 1/2 West Main S~    912108354004 2011-05-18        Clas~ Issued
```

```
##  5 1204 1/2 East Michi~    922116376028 2013-05-15        Clas~ Issued
##  6 601 A Glover Avenue     922116177015 2012-11-29        Clas~ Issued
##  7 108 L North Busey A~    912108360001 2010-10-12        Clas~ Expired
##  8 606 1/2 West Elm St~    922117110004 2013-06-12        Clas~ Issued
##  9 1206 1/2 East Michi~    922116376029 2013-05-15        Clas~ Issued
## 10 804 1/2 East Main S~    912109353015 2012-02-28        Clas~ Issued
## # i 1,728 more rows
## # i 3 more variables: `Expiration Date` <chr>, `Mappable Address` <chr>,
## #   `Numeric Grade` <dbl>
```

```
rentalsData %>%
  mutate(grade_n = if_else(Grade=="Class N", "1", Grade),
         grade_f = if_else(Grade=="Class F", "2", grade_n),
         grade_d = if_else(Grade=="Class D", "3", grade_f),
         grade_c = if_else(Grade=="Class C", "4", grade_d),
         grade_b = if_else(Grade=="Class B", "5", grade_c),
         `Numeric Grade` = as.numeric(if_else(Grade=="Class A", "6", grade_b))) %>%
  select(`Property Address`:`Mappable Address`,`Numeric Grade`)
```

Data summarization

Summarizing (or aggregating) data is the idea of generalizing specific instances into big-picture numeric descriptors. This may also be considered as applying summary functions, such as the sum `sum()`, mean `mean()`, median `median()`, variance `var()`, or standard deviation `sd()`. Some of these summary values are good to know for groups in a dataset. We can summarize grouped data to learn about numerical behaviors or patterns within and among those groups.

We can summarize data with the `summarize()` (also written as `summarise()`) function. With the `rentalsData`, let's show the mean and median of Parcel Number as a data frame.

```
rentalsData %>%
  summarize(mean(`Parcel Number`), median(`Parcel Number`))
```

```
## # A tibble: 1 x 2
##   `mean(\`Parcel Number\`)` `median(\`Parcel Number\`)`
##                       <dbl>                       <dbl>
## 1             921079495298.               922116429522.
```

Summarizing grouped data

In order to summarize grouped data, we need two actions to work in tandem: `group_by()` and `summarize()`. The `group_by()` function allows us to do data processing separately for each group in the data frame. The group processing takes place once we combine this function with other tidyverse functions, especially `summarize()`. First, we `group_by()` then `summarize()`.

Beginning with the `rentalsData`, suppose we want to compute "Grade Proportion", which represents the proportion of each grade of the Grade column. We can use the `n()` function to compute the number of each group of a specific column.

```
rentalsData %>%
group_by(Grade) %>%
  summarize(`Grade Proportion` = n()/nrow(rentalsData))
```

```
## # A tibble: 6 x 2
##   Grade   `Grade Proportion`
##   <chr>                <dbl>
## 1 Class A           0.0926
## 2 Class B           0.828
## 3 Class C           0.0765
## 4 Class D           0.00115
## 5 Class F           0.00115
## 6 Class N           0.000575
```

PRACTICE: Summarize with a new column called "Mean Numeric Grade" representing the average numeric grade for each License Status group.

```
rentalsData %>%
  mutate(grade_n = if_else(Grade=="Class N", "1", Grade),
         grade_f = if_else(Grade=="Class F", "2", grade_n),
         grade_d = if_else(Grade=="Class D", "3", grade_f),
         grade_c = if_else(Grade=="Class C", "4", grade_d),
         grade_b = if_else(Grade=="Class B", "5", grade_c),
         `Numeric Grade` = as.numeric(if_else(Grade=="Class A", "6", grade_b))) %>%
  group_by(`License Status`) %>%
  summarize(`Mean Numeric Grade` = mean(`Numeric Grade`))
```

```
## # A tibble: 7 x 2
##   `License Status`          `Mean Numeric Grade`
##   <chr>                                    <dbl>
## 1 Approved                                  5
## 2 Contract Sale                             5
## 3 Expired                                   4.95
## 4 Fee Exempt Registration                   5.4
## 5 Issued                                    5.02
## 6 Temporarily Not a Rental                  5
## 7 Under Review                              5
```

```
rentalsData %>%
  mutate(grade_n = if_else(Grade=="Class N", "1", Grade),
         grade_f = if_else(Grade=="Class F", "2", grade_n),
         grade_d = if_else(Grade=="Class D", "3", grade_f),
         grade_c = if_else(Grade=="Class C", "4", grade_d),
         grade_b = if_else(Grade=="Class B", "5", grade_c),
         `Numeric Grade` = as.numeric(if_else(Grade=="Class A", "6", grade_b))) %>%
  group_by(`License Status`) %>%
  summarize(`Mean Numeric Grade` = mean(`Numeric Grade`))
```