tidyverse

David John Baker

12/23/2018

Lesson Goals

- Be able to explain tidy data
- Explain the five tidyverse verbs
- Perform basic indexing
- Import and Export data from R

tidyverse + tidydata

One of the most important concepts data science and R is the idea of tidydata.

The idea behind tidy data is that...

1. Each variable forms a column
2. Each observation forms a row.
3. Each type of observation unit forms a table.

If your data is in this format, then you can do almost anything with the tidyverse.

In order to use the tidyverse, you first need to install it.

```
# install.packages("tidyverse") # Only need to do this once!
library(tidyverse)
```

```
## -- Attaching packages --------------------------------- tidyverse 1.2.1 --
## v ggplot2 3.1.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.8
## v tidyr   0.8.2     v stringr 1.3.1
## v readr   1.3.1     v forcats 0.3.0
## -- Conflicts ------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

Five Verbs

The five tidyverse verbs come from the dplyr package. More information on this package can be found here along with these descriptions.

- `mutate()` adds new variables that are functions of existing variables
- `select()` picks variables based on their names.
- `filter()` picks cases based on their values.
- `summarise()` reduces multiple values down to a single summary.
- `arrange()` changes the ordering of the rows.

We can think the verbs as happening in the logical order you would want to grab them. Each of the verbs is also going to be connected to one another with the pipe operator. The idea behind the pipe or '`%>%` is that the output of the last line is the first argument of the new function.

For example, if we wanted to make a small table that only had data from El Paso from 2011, then only get the first and fifth columns we would run the following code:

```
str(txhousing)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    8602 obs. of  9 variables:
##  $ city     : chr  "Abilene" "Abilene" "Abilene" "Abilene" ...
##  $ year     : int  2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 ...
##  $ month    : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ sales    : num  72 98 130 98 141 156 152 131 104 101 ...
##  $ volume   : num  5380000 6505000 9285000 9730000 10590000 ...
##  $ median   : num  71400 58700 58100 68600 67300 66900 73500 75000 64500 59300 ...
##  $ listings : num  701 746 784 785 794 780 742 765 771 764 ...
##  $ inventory: num  6.3 6.6 6.8 6.9 6.8 6.6 6.2 6.4 6.5 6.6 ...
##  $ date     : num  2000 2000 2000 2000 2000 ...
```

```
txhousing_only_el_paso <- txhousing[txhousing$city == "El Paso",]
iris_only_only_el_paso_2005_2011 <- txhousing_only_el_paso[txhousing_only_el_paso$year >= 2011,]
iris_only_only_el_paso_2005_2011[,c(1,5)]
```

```
## # A tibble: 55 x 2
##    city      volume
##    <chr>      <dbl>
##  1 El Paso 66136913
##  2 El Paso 44840808
##  3 El Paso 63884923
##  4 El Paso 74429226
##  5 El Paso 61624856
##  6 El Paso 71212091
##  7 El Paso 72366107
##  8 El Paso 86547783
##  9 El Paso 64083406
## 10 El Paso 67015215
## # ... with 45 more rows
```

Which is a bit verbose.

In order to do this with the tidyverse, you would start with the dataset, the run two verbs over it, connected with the pipe.

```
library(tidyverse)
iris_tibble <- as.tibble(iris)
```

```
txhousing %>%
  filter(city == "El Paso") %>%
  filter(year >= 2011) %>%
  select(1,5)
```

```
## # A tibble: 55 x 2
##    city      volume
##    <chr>      <dbl>
##  1 El Paso 66136913
##  2 El Paso 44840808
##  3 El Paso 63884923
##  4 El Paso 74429226
##  5 El Paso 61624856
##  6 El Paso 71212091
##  7 El Paso 72366107
##  8 El Paso 86547783
##  9 El Paso 64083406
## 10 El Paso 67015215
## # ... with 45 more rows
```

Both create the same output, but one is much easier to read.

We will now explore a dataset using the five verbs in the dplyr package. You use each of the five verbs as you would in English to think about how you want to manipulate your data.

They key to using the tidyverse is the %>% operator (the pipe operator). It works by taking output from what is before it and piping it to the next command.

Economics Data

The dataset here comes from housing sales data in Texas provided by the TAMU real estate centre.

Variable	Description
city	Name of MLS area
year,month,date	Date
sales	Number of sales
volume	Total value of sales
median	Median sale price
listings	Total active listings
inventory	"Months inventory" : amount of time it would take to sell all current listings at current pace of sales.

Select

The select command works by "Selecting" the columns you wish to work with. It can either take the index of the column using numbers, or the text. There are other options like asking for columns that start or end with certain text.

```
txhousing %>%
  select(1,2)
```

```
## # A tibble: 8,602 x 2
##    city     year
```

```
##    <chr>   <int>
##  1 Abilene  2000
##  2 Abilene  2000
##  3 Abilene  2000
##  4 Abilene  2000
##  5 Abilene  2000
##  6 Abilene  2000
##  7 Abilene  2000
##  8 Abilene  2000
##  9 Abilene  2000
## 10 Abilene  2000
## # ... with 8,592 more rows
```

```
txhousing %>%
  select(1:3)
```

```
## # A tibble: 8,602 x 3
##    city     year month
##    <chr>   <int> <int>
##  1 Abilene  2000     1
##  2 Abilene  2000     2
##  3 Abilene  2000     3
##  4 Abilene  2000     4
##  5 Abilene  2000     5
##  6 Abilene  2000     6
##  7 Abilene  2000     7
##  8 Abilene  2000     8
##  9 Abilene  2000     9
## 10 Abilene  2000    10
## # ... with 8,592 more rows
```

```
txhousing %>%
  select(city, sales:median)
```

```
## # A tibble: 8,602 x 4
##    city    sales   volume median
##    <chr>   <dbl>    <dbl>  <dbl>
##  1 Abilene    72  5380000  71400
##  2 Abilene    98  6505000  58700
##  3 Abilene   130  9285000  58100
##  4 Abilene    98  9730000  68600
##  5 Abilene   141 10590000  67300
##  6 Abilene   156 13910000  66900
##  7 Abilene   152 12635000  73500
##  8 Abilene   131 10710000  75000
##  9 Abilene   104  7615000  64500
## 10 Abilene   101  7040000  59300
## # ... with 8,592 more rows
```

```
txhousing %>%
  select(starts_with("ci"))
```

```
## # A tibble: 8,602 x 1
##    city
##    <chr>
##  1 Abilene
```

```
##  2 Abilene
##  3 Abilene
##  4 Abilene
##  5 Abilene
##  6 Abilene
##  7 Abilene
##  8 Abilene
##  9 Abilene
## 10 Abilene
## # ... with 8,592 more rows
```

```
txhousing %>%
  select(-city)
```

```
## # A tibble: 8,602 x 8
##     year month sales   volume median listings inventory  date
##    <int> <int> <dbl>    <dbl>  <dbl>    <dbl>     <dbl> <dbl>
##  1  2000     1    72  5380000  71400      701       6.3 2000
##  2  2000     2    98  6505000  58700      746       6.6 2000.
##  3  2000     3   130  9285000  58100      784       6.8 2000.
##  4  2000     4    98  9730000  68600      785       6.9 2000.
##  5  2000     5   141 10590000  67300      794       6.8 2000.
##  6  2000     6   156 13910000  66900      780       6.6 2000.
##  7  2000     7   152 12635000  73500      742       6.2 2000.
##  8  2000     8   131 10710000  75000      765       6.4 2001.
##  9  2000     9   104  7615000  64500      771       6.5 2001.
## 10  2000    10   101  7040000  59300      764       6.6 2001.
## # ... with 8,592 more rows
```

Filter

Once we have the columns we want to work with, we can then pick the rows that are of interest. We do this with the filter function. Here when asking for matches of character strings, you need to use the ==. R will remind you if you forget. The filter command can be combined with the logical operators. Remember this includes negation operators.

```
txhousing %>%
  filter(city == "El Paso")
```

```
## # A tibble: 187 x 9
##    city     year month sales   volume median listings inventory  date
##    <chr>   <int> <int> <dbl>    <dbl>  <dbl>    <dbl>     <dbl> <dbl>
##  1 El Paso  2000     1   306 31525000  82100     2512       5.8 2000
##  2 El Paso  2000     2   346 32300000  76600     2572       5.9 2000.
##  3 El Paso  2000     3   492 47505000  77100     2549       5.8 2000.
##  4 El Paso  2000     4   382 37915000  79400     2525       5.9 2000.
##  5 El Paso  2000     5   459 43335000  80100     2552       5.9 2000.
##  6 El Paso  2000     6   486 47880000  83200       NA      NA   2000.
##  7 El Paso  2000     7   422 42925000  82600     2685       6.4 2000.
##  8 El Paso  2000     8   538 53800000  81700     3396       8   2001.
##  9 El Paso  2000     9   382 36775000  78300     2661       6.3 2001.
## 10 El Paso  2000    10   392 40535000  81900     2704       6.5 2001.
## # ... with 177 more rows
```

```
txhousing %>%
  filter(city == "El Paso" | city == "San Antonio")
```

```
## # A tibble: 374 x 9
##    city     year month sales   volume median listings inventory  date
##    <chr>   <int> <int> <dbl>    <dbl>  <dbl>    <dbl>     <dbl> <dbl>
##  1 El Paso  2000     1   306 31525000  82100     2512       5.8 2000
##  2 El Paso  2000     2   346 32300000  76600     2572       5.9 2000.
##  3 El Paso  2000     3   492 47505000  77100     2549       5.8 2000.
##  4 El Paso  2000     4   382 37915000  79400     2525       5.9 2000.
##  5 El Paso  2000     5   459 43335000  80100     2552       5.9 2000.
##  6 El Paso  2000     6   486 47880000  83200       NA      NA   2000.
##  7 El Paso  2000     7   422 42925000  82600     2685       6.4 2000.
##  8 El Paso  2000     8   538 53800000  81700     3396       8   2001.
##  9 El Paso  2000     9   382 36775000  78300     2661       6.3 2001.
## 10 El Paso  2000    10   392 40535000  81900     2704       6.5 2001.
## # ... with 364 more rows
```

```
txhousing %>%
  filter(city == "El Paso" | city == "San Antonio") %>%
  filter(year >= 2004)
```

```
## # A tibble: 278 x 9
##    city     year month sales   volume median listings inventory  date
##    <chr>   <int> <int> <dbl>    <dbl>  <dbl>    <dbl>     <dbl> <dbl>
##  1 El Paso  2004     1   435 48330000  93500     3028       5.8 2004
##  2 El Paso  2004     2   441 48215000  89600     3162       6   2004.
##  3 El Paso  2004     3   551 60105000  89000     3288       6.2 2004.
##  4 El Paso  2004     4   579 64980000  89900     3320       6.2 2004.
##  5 El Paso  2004     5   576 68890000  97500     3271       6.1 2004.
##  6 El Paso  2004     6   586 72925000  97200       NA      NA   2004.
##  7 El Paso  2004     7   619 77745000  99900       NA      NA   2004.
##  8 El Paso  2004     8   462 54045000  95400       NA      NA   2005.
##  9 El Paso  2004     9   294 29910000  85800       NA      NA   2005.
## 10 El Paso  2004    10   484 56625000  94800       NA      NA   2005.
## # ... with 268 more rows
```

```
txhousing %>%
  filter(city == "El Paso" | city == "San Antonio") %>%
  filter(year >= 2004) %>%
  filter(month != 1)
```

```
## # A tibble: 254 x 9
##    city     year month sales   volume median listings inventory  date
##    <chr>   <int> <int> <dbl>    <dbl>  <dbl>    <dbl>     <dbl> <dbl>
##  1 El Paso  2004     2   441 48215000  89600     3162       6   2004.
##  2 El Paso  2004     3   551 60105000  89000     3288       6.2 2004.
##  3 El Paso  2004     4   579 64980000  89900     3320       6.2 2004.
##  4 El Paso  2004     5   576 68890000  97500     3271       6.1 2004.
##  5 El Paso  2004     6   586 72925000  97200       NA      NA   2004.
##  6 El Paso  2004     7   619 77745000  99900       NA      NA   2004.
##  7 El Paso  2004     8   462 54045000  95400       NA      NA   2005.
##  8 El Paso  2004     9   294 29910000  85800       NA      NA   2005.
##  9 El Paso  2004    10   484 56625000  94800       NA      NA   2005.
## 10 El Paso  2004    11   470 55690000  96200       NA      NA   2005.
```

```
## # ... with 244 more rows
```

Mutate

The mutate command will create new variables.

```
txhousing %>%
  mutate(zSales = scale(sales))
```

```
## # A tibble: 8,602 x 10
##    city     year month sales  volume median listings inventory  date zSales
##    <chr>   <int> <int> <dbl>   <dbl>  <dbl>    <dbl>     <dbl> <dbl>  <dbl>
##  1 Abilene  2000     1    72  5.38e6  71400      701       6.3 2000  -0.430
##  2 Abilene  2000     2    98  6.50e6  58700      746       6.6 2000. -0.407
##  3 Abilene  2000     3   130  9.28e6  58100      784       6.8 2000. -0.378
##  4 Abilene  2000     4    98  9.73e6  68600      785       6.9 2000. -0.407
##  5 Abilene  2000     5   141  1.06e7  67300      794       6.8 2000. -0.368
##  6 Abilene  2000     6   156  1.39e7  66900      780       6.6 2000. -0.354
##  7 Abilene  2000     7   152  1.26e7  73500      742       6.2 2000. -0.358
##  8 Abilene  2000     8   131  1.07e7  75000      765       6.4 2001. -0.377
##  9 Abilene  2000     9   104  7.62e6  64500      771       6.5 2001. -0.401
## 10 Abilene  2000    10   101  7.04e6  59300      764       6.6 2001. -0.404
## # ... with 8,592 more rows
```

```
txhousing %>%
  filter(city == "El Paso" | city == "San Antonio") %>%
  filter(year >= 2004) %>%
  filter(month != 1) %>%
  mutate(zScale = scale(sales))
```

```
## # A tibble: 254 x 10
##    city     year month sales  volume median listings inventory  date zScale
##    <chr>   <int> <int> <dbl>   <dbl>  <dbl>    <dbl>     <dbl> <dbl>  <dbl>
##  1 El Paso  2004     2   441  4.82e7  89600     3162         6   2004. -0.989
##  2 El Paso  2004     3   551  6.01e7  89000     3288       6.2 2004. -0.846
##  3 El Paso  2004     4   579  6.50e7  89900     3320       6.2 2004. -0.809
##  4 El Paso  2004     5   576  6.89e7  97500     3271       6.1 2004. -0.813
##  5 El Paso  2004     6   586  7.29e7  97200       NA        NA  2004. -0.800
##  6 El Paso  2004     7   619  7.77e7  99900       NA        NA  2004. -0.757
##  7 El Paso  2004     8   462  5.40e7  95400       NA        NA  2005. -0.962
##  8 El Paso  2004     9   294  2.99e7  85800       NA        NA  2005. -1.18
##  9 El Paso  2004    10   484  5.66e7  94800       NA        NA  2005. -0.933
## 10 El Paso  2004    11   470  5.57e7  96200       NA        NA  2005. -0.952
## # ... with 244 more rows
```

Arrange

Arrange will sort our data.

```
txhousing %>%
  filter(city == "El Paso" | city == "San Antonio") %>%
  filter(year >= 2004) %>%
  filter(month != 1) %>%
  mutate(zScale = scale(sales)) %>%
  arrange(sales)
```

```
## # A tibble: 254 x 10
##    city     year month sales  volume median listings inventory  date zScale
##    <chr>   <int> <int> <dbl>   <dbl>  <dbl>    <dbl>     <dbl> <dbl>  <dbl>
##  1 El Paso  2011     2   287  4.48e7 134500     3023       6.5 2011.  -1.19
##  2 El Paso  2008     4   292  4.45e7 126700     4840      10.9 2008.  -1.18
##  3 El Paso  2004     9   294  2.99e7  85800       NA      NA   2005.  -1.18
##  4 El Paso  2009     2   326  4.97e7 130900     4530      11.4 2009.  -1.14
##  5 El Paso  2008     2   328  5.34e7 133800     4374       9   2008.  -1.14
##  6 El Paso  2013     2   350  5.42e7 139000     3425       7.3 2013.  -1.11
##  7 El Paso  2005     9   356  4.23e7 111300       NA      NA   2006.  -1.10
##  8 El Paso  2007    12   362  5.56e7 133500     4625       8.9 2008.  -1.09
##  9 El Paso  2008    11   362  5.65e7 132800     4773      12   2009.  -1.09
## 10 El Paso  2008    12   363  5.80e7 136300     4454      11.2 2009.  -1.09
## # ... with 244 more rows
```

```
txhousing %>%
  filter(city == "El Paso" | city == "San Antonio") %>%
  filter(year >= 2004) %>%
  filter(month != 1) %>%
  mutate(zScale = scale(sales)) %>%
  arrange(sales, -year)
```

```
## # A tibble: 254 x 10
##    city     year month sales  volume median listings inventory  date zScale
##    <chr>   <int> <int> <dbl>   <dbl>  <dbl>    <dbl>     <dbl> <dbl>  <dbl>
##  1 El Paso  2011     2   287  4.48e7 134500     3023       6.5 2011.  -1.19
##  2 El Paso  2008     4   292  4.45e7 126700     4840      10.9 2008.  -1.18
##  3 El Paso  2004     9   294  2.99e7  85800       NA      NA   2005.  -1.18
##  4 El Paso  2009     2   326  4.97e7 130900     4530      11.4 2009.  -1.14
##  5 El Paso  2008     2   328  5.34e7 133800     4374       9   2008.  -1.14
##  6 El Paso  2013     2   350  5.42e7 139000     3425       7.3 2013.  -1.11
##  7 El Paso  2005     9   356  4.23e7 111300       NA      NA   2006.  -1.10
##  8 El Paso  2008    11   362  5.65e7 132800     4773      12   2009.  -1.09
##  9 El Paso  2007    12   362  5.56e7 133500     4625       8.9 2008.  -1.09
## 10 El Paso  2010     2   363  5.16e7 126300     3321       7.4 2010.  -1.09
## # ... with 244 more rows
```

Group By and Sumemrise

Often we also want to perform the same type of calculation on a group in our dataset. For this we need to group our data, then use the summarize command. We can also use the `n()` function to count the number of observations in each group.

```
txhousing %>%
  filter(city == "El Paso" | city == "San Antonio") %>%
  filter(year >= 2004) %>%
  group_by(year) %>%
  summarise(mean = mean(sales))
```

```
## # A tibble: 12 x 2
##     year  mean
##    <int> <dbl>
##  1  2004 1102.
##  2  2005 1224.
##  3  2006 1381.
```

```
##  4  2007 1257.
##  5  2008 1006.
##  6  2009 1004.
##  7  2010 1000.
##  8  2011  980.
##  9  2012 1090.
## 10  2013 1246.
## 11  2014 1323.
## 12  2015 1461.
```

```
txhousing %>%
  filter(city == "El Paso" | city == "San Antonio") %>%
  filter(year >= 2004) %>%
  group_by(year) %>%
  summarise(mean = mean(sales), n = n())
```

```
## # A tibble: 12 x 3
##     year  mean     n
##    <int> <dbl> <int>
##  1  2004 1102.    24
##  2  2005 1224.    24
##  3  2006 1381.    24
##  4  2007 1257.    24
##  5  2008 1006.    24
##  6  2009 1004.    24
##  7  2010 1000.    24
##  8  2011  980.    24
##  9  2012 1090.    24
## 10  2013 1246.    24
## 11  2014 1323.    24
## 12  2015 1461.    14
```

Work Time

We will now explore the dataset using some guided questions.

In a new script, create following:

- Create a table with only the the first four counties in the dataset.
- Next, run the same command and run that only using one argument that adds in counties that have the work "County" in the title
- Create any new table using a single logical operator
- Create a table with a two logical operators
- Create a table that has no observations from either Paris or Waco.
- Create a new variable based on two other variables
- Find the month with the highest average scales in Tyler county for the year 2015
- Create a table with data from Austin and Galveston, using only the last three years of the dataset. Group the sales by county and then calculate z scores for each county.
- Save your new table to a csv file