

Lab 1 - Tidy Data Wrangling

SOLUTIONS

Data: Yearly statistics and standings for baseball teams

Today's data is all baseball statistics. The data is in the `Lahman` package.

View the data

Before doing any analysis, you will want to get quick view of the data. This is useful as part of the EDA process.

```
library(magrittr)
library(Lahman)
dim(Teams)
```

```
[1] 3015    48
```

Data dictionary

The variable definitions are found in the help for `Teams`, and are listed below.

```
?Teams
```

Column	Description
yearID	Year
lgID	League; a factor with levels AA AL FL NL PL UA
teamID	Team; a factor
franchID	Franchise (links to <code>TeamsFranchises</code> table)
divID	Team's division; a factor with levels C E W
Rank	Position in final standings

Column	Description
G	Games played
Ghome	Games played at home
W	Wins
L	Losses
DivWin	Division Winner (Y or N)
WCWin	Wild Card Winner (Y or N)
LgWin	League Champion(Y or N)
WSWin	World Series Winner (Y or N)
R	Runs scored
AB	At bats
H	Hits by batters
X2B	Doubles
X3B	Triples
HR	Homeruns by batters
BB	Walks by batters
SO	Strikeouts by batters
SB	Stolen bases
CS	Caught stealing
HBP	Batters hit by pitch
SF	Sacrifice flies
RA	Opponents runs scored
ER	Earned runs allowed
ERA	Earned run average
CG	Complete games
SHO	Shutouts
SV	Saves
IPouts	Outs Pitched (innings pitched x 3)
HA	Hits allowed
HRA	Homeruns allowed
BBA	Walks allowed
SOA	Strikeouts by pitchers
E	Errors
DP	Double Plays
FP	Fielding percentage
name	Team's full name
park	Name of team's home ballpark
attendance	Home attendance total
BPF	Three-year park factor for batters
PPF	Three-year park factor for pitchers
teamIDBR	Team ID used by Baseball Reference website
teamIDlahman45	Team ID used in Lahman database version 4.5

Column	Description
teamIDretro	Team ID used by Retrosheet

Exercises

Exercise 1

How many observations are in the `Teams` dataset? How many variables?

```
# take the first three rows and glimpse the data
Teams |> dplyr::slice_head(n=3) |> dplyr::glimpse()
```

```
Rows: 3
Columns: 48
$ yearID      <int> 1871, 1871, 1871
$ lgID        <fct> NA, NA, NA
$ teamID      <fct> BS1, CH1, CL1
$ franchID    <fct> BNA, CNA, CFC
$ divID       <chr> NA, NA, NA
$ Rank        <int> 3, 2, 8
$ G           <int> 31, 28, 29
$ Ghome       <int> NA, NA, NA
$ W           <int> 20, 19, 10
$ L           <int> 10, 9, 19
$ DivWin      <chr> NA, NA, NA
$ WCWin       <chr> NA, NA, NA
$ LgWin       <chr> "N", "N", "N"
$ WSWin       <chr> NA, NA, NA
$ R           <int> 401, 302, 249
$ AB          <int> 1372, 1196, 1186
$ H           <int> 426, 323, 328
$ X2B         <int> 70, 52, 35
$ X3B         <int> 37, 21, 40
$ HR          <int> 3, 10, 7
$ BB          <int> 60, 60, 26
$ SO          <int> 19, 22, 25
$ SB          <int> 73, 69, 18
$ CS          <int> 16, 21, 8
$ HBP         <int> NA, NA, NA
$ SF          <int> NA, NA, NA
```

```

$ RA          <int> 303, 241, 341
$ ER          <int> 109, 77, 116
$ ERA        <dbl> 3.55, 2.76, 4.11
$ CG          <int> 22, 25, 23
$ SHO        <int> 1, 0, 0
$ SV          <int> 3, 1, 0
$ IPouts      <int> 828, 753, 762
$ HA          <int> 367, 308, 346
$ HRA        <int> 2, 6, 13
$ BBA        <int> 42, 28, 53
$ SOA        <int> 23, 22, 34
$ E          <int> 243, 229, 234
$ DP          <int> 24, 16, 15
$ FP         <dbl> 0.834, 0.829, 0.818
$ name        <chr> "Boston Red Stockings", "Chicago White Stockings", "Cle~
$ park        <chr> "South End Grounds I", "Union Base-Ball Grounds", "Nati~
$ attendance  <int> NA, NA, NA
$ BPF         <int> 103, 104, 96
$ PPF         <int> 98, 102, 100
$ teamIDBR    <chr> "BOS", "CHI", "CLE"
$ teamIDlahman45 <chr> "BS1", "CH1", "CL1"
$ teamIDretro  <chr> "BS1", "CH1", "CL1"

```

SOLUTION

From the `dim(Teams)` statement used after `library(Lahman)`, there are 3015 observations and 48 variables.

Exercise 2

[Ben Baumer](#) worked for the [New York Mets](#) from 2004 to 2012. What was the team W/L record during those years? Use `filter()` and `select()` to quickly identify only those pieces of information that we care about.

SOLUTION:

```

# filter to use only rows where teamID equals "NYN"
mets <- Teams %>%
  dplyr::filter(teamID == "NYN")
# filter to use only rows where yearID is >= 2004 and <= 2012
# you could also write dplyr::filter(yearID %in% 2004:2012)
my_mets <- mets %>%
  dplyr::filter(yearID >= 2004 & yearID <= 2012)
# the dataset needs to have at least the year and the won (W) loss (L) record for that y
my_mets %>%
  dplyr::select(teamID,yearID,W,L)

```

	teamID	yearID	W	L
1	NYN	2004	71	91
2	NYN	2005	83	79
3	NYN	2006	97	65
4	NYN	2007	88	74
5	NYN	2008	89	73
6	NYN	2009	70	92
7	NYN	2010	79	83
8	NYN	2011	77	85
9	NYN	2012	74	88

Overall, the won-loss record was as follows:

```

my_mets %>%
  dplyr::select(teamID,yearID,W,L) %>%
  dplyr::summarize(
    "2004-2012 wins" = sum(W)
    , "2004-2012 losses" = sum(L)
  )

```

	2004-2012 wins	2004-2012 losses
1	728	730

Exercise 3

The model estimates the expected winning percentage as follows:

$$\hat{W}_{\text{pct}} = \frac{1}{1 + \left(\frac{RA}{RS}\right)^2}$$

where RA is the number of runs the team allows to be scored, RS is the number of runs that the team scores, and \hat{W}_{pct} is the team's expected winning percentage. The runs scored and allowed are present in the `Teams` table, so we start by selecting them.

SOLUTION:

```
mets_ben <- Teams |>
  # select to get the columns you want
  dplyr::select(teamID, yearID, W, L, R, RA) |>
  # filter to get the rows you want
  dplyr::filter(teamID == "NYN" & yearID %in% 2004:2012)
```

The column name can be changed with the `dplyr::rename` function (Use `new_name = old_name` to rename selected variables). Alternatively, you can rename the column directly in the select statement above, like this:

```
dplyr::select(teamID, yearID, W, L, RS = R, RA)
```

```
mets_ben <- mets_ben |>
  dplyr::rename(RS = R)    # new name = old name
mets_ben
```

	teamID	yearID	W	L	RS	RA
1	NYN	2004	71	91	684	731
2	NYN	2005	83	79	722	648
3	NYN	2006	97	65	834	731
4	NYN	2007	88	74	804	750
5	NYN	2008	89	73	799	715
6	NYN	2009	70	92	671	757
7	NYN	2010	79	83	656	652
8	NYN	2011	77	85	718	742
9	NYN	2012	74	88	650	709

Exercise 4

Next, we need to compute the team's actual winning percentage in each of these seasons. Thus, we need to add a new column to our data frame, and we do this with the `mutate()` command.

SOLUTION:

```

mets_ben <- mets_ben |>
  # once we have the data, we mutate to add a new value (column), using the formula
  dplyr::mutate( WPct = 1/(1 + (RA/RS)^2 ) )
mets_ben

```

	teamID	yearID	W	L	RS	RA	WPct
1	NYN	2004	71	91	684	731	0.4668211
2	NYN	2005	83	79	722	648	0.5538575
3	NYN	2006	97	65	834	731	0.5655308
4	NYN	2007	88	74	804	750	0.5347071
5	NYN	2008	89	73	799	715	0.5553119
6	NYN	2009	70	92	671	757	0.4399936
7	NYN	2010	79	83	656	652	0.5030581
8	NYN	2011	77	85	718	742	0.4835661
9	NYN	2012	74	88	650	709	0.4566674

The expected number of wins is then equal to the product of the expected winning percentage times the number of games.

```

mets_ben <- mets_ben |>
  # once we have calculated the expected winning percentage,
  # the expected number of wins is the percentage times the total number of games played
  dplyr::mutate( W_hat = WPct * (W+L) )
mets_ben

```

	teamID	yearID	W	L	RS	RA	WPct	W_hat
1	NYN	2004	71	91	684	731	0.4668211	75.62501
2	NYN	2005	83	79	722	648	0.5538575	89.72491
3	NYN	2006	97	65	834	731	0.5655308	91.61600
4	NYN	2007	88	74	804	750	0.5347071	86.62255
5	NYN	2008	89	73	799	715	0.5553119	89.96053
6	NYN	2009	70	92	671	757	0.4399936	71.27896
7	NYN	2010	79	83	656	652	0.5030581	81.49541
8	NYN	2011	77	85	718	742	0.4835661	78.33771
9	NYN	2012	74	88	650	709	0.4566674	73.98012

Exercise 5

In this case, the Mets' fortunes were better than expected in three of these seasons, and worse than expected in the other six.

We can confirm this as follows:

SOLUTION:

```
mets_ben %>%  
  # first check that the assertion above is correct  
  dplyr::summarize('better then expected' = sum(W >= W_hat), 'worse than expected' = sum  
  
better then expected worse than expected  
1                      3                      6
```

To see how the Mets did over all seasons we can repeat our calculation

```
Teams |>  
  # here we repeat our prior calculation (all steps combined) for all the years in the d  
  dplyr::select(teamID, yearID, W, L, RS = R, RA) |>  
  dplyr::filter(teamID == "NYN") |>  
  dplyr::mutate(  
    WPct = 1/(1 + (RA/RS)^2 )  
    , W_hat = WPct * (W+L)  
  ) |>  
  dplyr::summarize(  
    "better then expected" = sum(W >= W_hat)  
    , 'worse than expected' = sum(W < W_hat)  
  )  
  
better then expected worse than expected  
1                      22                      39
```

Exercise 6

Naturally, the Mets experienced ups and downs during Ben's time with the team. Which seasons were best? To figure this out, we can simply sort the rows of the data frame by number of wins.

SOLUTION:

```
# for this we just need to sort the number of wins in descending order  
mets_ben |> dplyr::arrange(desc(W))
```


	teamID	yearID	W	L	RS	RA	WPct	W_hat
1	NYN	2006	97	65	834	731	0.5655308	91.61600
2	NYN	2008	89	73	799	715	0.5553119	89.96053
3	NYN	2007	88	74	804	750	0.5347071	86.62255
4	NYN	2005	83	79	722	648	0.5538575	89.72491
5	NYN	2010	79	83	656	652	0.5030581	81.49541
6	NYN	2011	77	85	718	742	0.4835661	78.33771
7	NYN	2012	74	88	650	709	0.4566674	73.98012
8	NYN	2004	71	91	684	731	0.4668211	75.62501
9	NYN	2009	70	92	671	757	0.4399936	71.27896

Exercise 7

In 2006, the Mets had the best record in baseball during the regular season and nearly made the *World Series*. How do these seasons rank in terms of the team's performance relative to our model?

SOLUTION:

```
mets_ben %>%
  # add a column with the difference between wins (W) and expected wins (W_hat)
  dplyr::mutate(Diff = W - W_hat) |>
  # then sort the result
  dplyr::arrange(desc(Diff))
```

	teamID	yearID	W	L	RS	RA	WPct	W_hat	Diff
1	NYN	2006	97	65	834	731	0.5655308	91.61600	5.38400315
2	NYN	2007	88	74	804	750	0.5347071	86.62255	1.37744558
3	NYN	2012	74	88	650	709	0.4566674	73.98012	0.01988152
4	NYN	2008	89	73	799	715	0.5553119	89.96053	-0.96052803
5	NYN	2009	70	92	671	757	0.4399936	71.27896	-1.27895513
6	NYN	2011	77	85	718	742	0.4835661	78.33771	-1.33770571
7	NYN	2010	79	83	656	652	0.5030581	81.49541	-2.49540821
8	NYN	2004	71	91	684	731	0.4668211	75.62501	-4.62501135
9	NYN	2005	83	79	722	648	0.5538575	89.72491	-6.72490937

In the years 2006, 2007 and 2012, the Mets had more wins than expected by the model. In all other seasons they performed worse than predicted by the model.

Exercise 8

We can summarize the Mets performance as follows:

SOLUTION:

```
mets_ben |>
  dplyr::summarize(
    num_years = dplyr::n(), # number of years
    total_W = sum(W),       # total number of wins
    total_L = sum(L),       # total number of losses
    total_WPct = total_W / (total_W + total_L) # win percentage
  )

  num_years total_W total_L total_WPct
1         9      728      730  0.4993141
```

In these nine years, the Mets had a combined record of 728 wins and 730 losses, for an overall winning percentage of 49.93%.

Exercise 9

Note

This question was incomplete and will not be counted.

Exercise 10

Discretize the years into three chunks: one for each of the three general managers under whom Ben worked. [Jim Duquette](#) was the Mets' *general manager* in 2004, [Omar Minaya](#) from 2005 to 2010, and [Sandy Alderson](#) from 2011 to 2012.

```
mets_ben %>%
  # this questions requires a logic for deciding
  # which years each general manager worked
  dplyr::mutate(
    # nested ifelse statements are OK for this logic,
    # but are only practical for about three cases
    gm = ifelse(
```

```

    yearID == 2004,
    'Jim Duquette',
    ifelse(
      yearID >= 2011,
      'Sandy Alderson',
      'Omar Minaya')
  )
)

```

	teamID	yearID	W	L	RS	RA	WPct	W_hat	gm
1	NYN	2004	71	91	684	731	0.4668211	75.62501	Jim Duquette
2	NYN	2005	83	79	722	648	0.5538575	89.72491	Omar Minaya
3	NYN	2006	97	65	834	731	0.5655308	91.61600	Omar Minaya
4	NYN	2007	88	74	804	750	0.5347071	86.62255	Omar Minaya
5	NYN	2008	89	73	799	715	0.5553119	89.96053	Omar Minaya
6	NYN	2009	70	92	671	757	0.4399936	71.27896	Omar Minaya
7	NYN	2010	79	83	656	652	0.5030581	81.49541	Omar Minaya
8	NYN	2011	77	85	718	742	0.4835661	78.33771	Sandy Alderson
9	NYN	2012	74	88	650	709	0.4566674	73.98012	Sandy Alderson

Alternatively, we can use the `case_when` function

```

mets_ben <- mets_ben |>
  dplyr::mutate(
    # same problem, but case_when is easier to work with
    gm = dplyr::case_when(
      yearID == 2004 ~ 'Jim Duquette',
      yearID >= 2011 ~ 'Sandy Alderson',
      TRUE ~ 'Omar Minaya' # this is the default case
    )
  )
mets_ben

```

	teamID	yearID	W	L	RS	RA	WPct	W_hat	gm
1	NYN	2004	71	91	684	731	0.4668211	75.62501	Jim Duquette
2	NYN	2005	83	79	722	648	0.5538575	89.72491	Omar Minaya
3	NYN	2006	97	65	834	731	0.5655308	91.61600	Omar Minaya
4	NYN	2007	88	74	804	750	0.5347071	86.62255	Omar Minaya
5	NYN	2008	89	73	799	715	0.5553119	89.96053	Omar Minaya
6	NYN	2009	70	92	671	757	0.4399936	71.27896	Omar Minaya
7	NYN	2010	79	83	656	652	0.5030581	81.49541	Omar Minaya

```
8   NYN    2011 77 85 718 742 0.4835661 78.33771 Sandy Alderson
9   NYN    2012 74 88 650 709 0.4566674 73.98012 Sandy Alderson
```

Exercise 11

Use the `gm` function to define the manager groups with the `group_by()` operator, and run the summaries again, this time across the manager groups.

SOLUTION:

```
mets_ben
```

```
mets_ben %>%
  # group by managers
  dplyr::group_by(gm) %>%
  # summarize - one row for each manager
  dplyr::summarize(
    num_years = dplyr::n(),
    total_W = sum(W),
    total_L = sum(L),
    total_WPct = total_W / (total_W + total_L)
  ) %>%
  # gt:: functions create and format tables
  gt::gt('gm') %>%                # create the table from the data,
                                   # and use 'gm' as the rownames
  gtExtras::gt_theme_espn() # apply a theme to the format
```

	num_years	total_W	total_L	total_WPct
Jim Duquette	1	71	91	0.4382716
Omar Minaya	6	506	466	0.5205761
Sandy Alderson	2	151	173	0.4660494

Resources for additional practice (optional)

- [Chapter 2: Get Started](#) *Data Visualization by Kieran Healy*
- [Chapter 3: Data visualization](#) in *R for Data Science* by Hadley Wickham
- RStudio Cloud Primers
 - Visualization Basics: <https://rstudio.cloud/learn/primers/1.1>
 - Work with Data: <https://rstudio.cloud/learn/primers/2>

- Visualize Data: <https://rstudio.cloud/learn/primers/3>