An Introduction to dplyr

Kyle Burris

Duke University

April 5, 2016

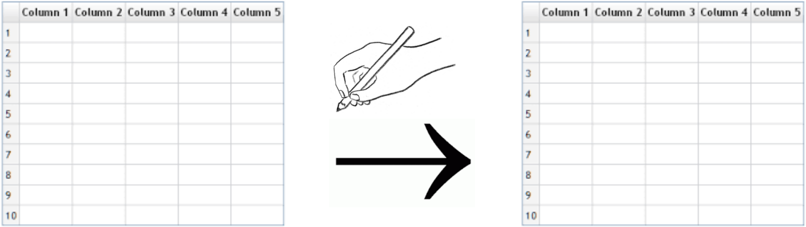
What is dplyr?

- R package designed to easily transform and manipulate data
- Developed by Hadley Wickham
- Why dplyr?
    - Speed
    - Readability
    - Ease

dplyr Function Rules

- The first argument is a data frame
- All other arguments dictate what to do with the data
- Returns a data frame

Note: dplyr does not modify data frames in place



	Column 1	Column 2	Column 3	Column 4	Column 5
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					

	Column 1	Column 2	Column 3	Column 4	Column 5
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					

Important Single Table Functions

- `filter()` : Select rows of a data frame based on criteria
- `select()` : Select specified columns of a data frame
- `slice()` : Select rows of a data frame by index
- `mutate()` : Create new variables
- `arrange()` : Sort data frame by specified variable
- `rename()` : Rename a variable
- `sample_n()` : Sample rows from data frame
- `group_by()` : Group by one or more variables
- `summarise()` : Calculate aggregate quantities

Example: 2011 Baseball Data

```
library(dplyr)
```

```
baseball <- read.csv("mlb11.csv")
```

	team	division	league	runs	at_bats	hits
1	Texas Rangers	AL West	American League	855	5659	1599
2	Boston Red Sox	AL East	American League	875	5710	1600
3	Detroit Tigers	AL Central	American League	787	5563	1540
...						
28	San Francisco Giants	NL West	National League	570	5486	1327
29	San Diego Padres	NL West	National League	593	5417	1284
30	Seattle Mariners	AL West	American League	556	5421	1263

Filter

```
filter(baseball, league == "National League")
```

	team	division	league	runs	at_bats	hits
1	St. Louis Cardinals	NL Central	National League	762	5532	1513
2	New York Mets	NL East	National League	718	5600	1477
3	Milwaukee Brewers	NL Central	National League	721	5447	1422
...						
13	Washington Nationals	NL East	National League	624	5441	1319
14	San Francisco Giants	NL West	National League	570	5486	1327
15	San Diego Padres	NL West	National League	593	5417	1284

Select

```
select(baseball, team, division)
```

	team	division
1	Texas Rangers	AL West
2	Boston Red Sox	AL East
3	Detroit Tigers	AL Central
...		
28	San Francisco Giants	NL West
29	San Diego Padres	NL West
30	Seattle Mariners	AL West

Arrange

```
arrange(baseball, league, desc(hits))
```

	team	division	league	runs	at_bats	hits
1	Boston Red Sox	AL East	American League	875	5710	1600
2	Texas Rangers	AL West	American League	855	5659	1599
3	Kansas City Royals	AL Central	American League	730	5672	1560
...						
28	Pittsburgh Pirates	NL Central	National League	610	5421	1325
29	Washington Nationals	NL East	National League	624	5441	1319
30	San Diego Padres	NL West	National League	593	5417	1284

Quiz: Question 1

- Write down a dplyr command that would return a data frame consisting only of the teams that scored more than 650 runs in 2011.

	team	division	league	runs	at_bats	hits
1	Texas Rangers	AL West	American League	855	5659	1599
2	Boston Red Sox	AL East	American League	875	5710	1600
3	Detroit Tigers	AL Central	American League	787	5563	1540
...						
28	San Francisco Giants	NL West	National League	570	5486	1327
29	San Diego Padres	NL West	National League	593	5417	1284
30	Seattle Mariners	AL West	American League	556	5421	1263

Interlude: Piping

- Often, we want to perform multiple operations on a given data frame
- We can do this in dplyr by a procedure called piping
- `f(g(x),y) = g(x) %>% f(y)`

```
baseball %>%
      select(team,division,runs) %>%
      filter(runs > 800)
```

	team	division	runs
1	Texas Rangers	AL West	855
2	Boston Red Sox	AL East	875
3	New York Yankees	AL East	867

Mutate

```
baseball %>%
    mutate(bat_avg = hits/at_bats) %>%
    select(team, hits, at_bats, bat_avg) %>%
    arrange(desc(bat_avg))
```

	team	hits	at_bats	bat_avg
1	Texas Rangers	1599	5659	0.283
2	Boston Red Sox	1600	5710	0.280
3	Detroit Tigers	1540	5563	0.277
...				
28	San Francisco Giants	1327	5486	0.242
29	San Diego Padres	1284	5417	0.237
30	Seattle Mariners	1263	5421	0.233

Quiz: Question 2

- Using piping, write down a dplyr command that creates a new variable called `runs_per_game` (there are 162 games in a season). Select only the variables `team` and `runs_per_game` and then sort by `runs_per_game` in ascending order

	team	division	league	runs	at_bats	hits
1	Texas Rangers	AL West	American League	855	5659	1599
2	Boston Red Sox	AL East	American League	875	5710	1600
3	Detroit Tigers	AL Central	American League	787	5563	1540
...						
28	San Francisco Giants	NL West	National League	570	5486	1327
29	San Diego Padres	NL West	National League	593	5417	1284
30	Seattle Mariners	AL West	American League	556	5421	1263

Summarise

```
baseball %>%
     summarise(total = n(), min_runs = min(runs))
```

	total	min_runs
1	30	556

Summarise (Group By)

```
baseball %>%
    group_by(division) %>%
    summarise(mean_bat_avg = mean(hits/at_bats),
        tot_runs = sum(runs),
        tot_teams = n_distinct(team))
```

	division	mean_bat_avg	tot_runs	tot_teams
1	AL Central	0.260	3494	5
2	AL East	0.259	3900	5
3	AL West	0.254	3338	5
4	NL Central	0.258	3482	5
5	NL East	0.250	3321	5
6	NL West	0.249	3273	5

Other Single Table Commands

```
rename(baseball, div = division)
```

```
sample_n(baseball, size = 10, replace = FALSE)
```

```
sample_frac(baseball, size = 1/3, replace = FALSE)
```

Other Single Table Commands

```
rename(baseball, div = division)
```

```
sample_n(baseball, size = 10, replace = FALSE)
```

```
sample_frac(baseball, size = 1/3, replace = FALSE)
```

Quiz Question 3:

- Using piping, write down a dplyr command that first, randomly samples 100 rows from `baseball` (with replacement), then counts how many occurrences of each team there are, and then sorts the teams by descending order of frequency.

Recap

- We reviewed a variety of single table commands in dplyr
- We introduced piping, which makes code much more readable

Base R

```
subset(baseball[sample(1:nrow(baseball),10),
      c("team","runs")], runs > 600)
```

dplyr

```
baseball %>%
      sample_n(10, replace = FALSE) %>%
      select(team, runs) %>%
      filter(runs > 600)
```

Next Time

- Special Functions
- Two-table functions (joining two data frames)
- In-class exercise with large data (bring your computers!)