Data manipulation with dplyr

Martin Schmettow

TRUG meeting, Enschede, June 25, 2014

DPLYR

Basics

dplyr

- The new coup of **Hadley Wickham**, creator of
  - ggplot2
  - plyr

plyr

- ❖ functional programming paradigm
- ❖ Functions as parameters of <u>second level functions</u>
  - ⇨ llply(some.list, some.function) # returns list
  - ⇨ laply(some.list, some.function) # returns array
  - ⇨ aaply(some.array, some.function) # returns array
  - ⇨ ...
- ❖ replaces classic second level functions: apply, lapply, sapply, replicate

dplyr

next generation data.frame manipulation

- ❖ Simple interface
- ❖ Readable code
- ❖ Fast
- ❖ Can transparently deal with remote data
- ❖ interfaces well with → plyr and →ggplot

Basic elements of plyr

- ❖ Functions:
  - ⇨ Filter
  - ⇨ Select
  - ⇨ Mutate
  - ⇨ Group_by
  - ⇨ Summarize
  - ⇨ Arrange
- ❖ Operator
  - ⇨ Concatenation by %.% or %>%

Example 1: filter some observations

Classic

```
D[S$subject == 4 & D$trial == 10,]
```

dplyr

```
filter(D, subject == 4, trial == 10)
```

Example 2: select some variables

Classic

```
D[,c(“subject“, “trial“)]
```

dplyr

```
select(D, subject, trial)
```

Example 3: add a variable

Classic

```
D$freq <- D$count/D$time
```

dplyr

```
D <- mutate(D, freq = count/time)
```

Example 4: summarize data

Classic

aggregate(D$RT, list(subject = D$Subject), mean)

dplyr

group_by(D, subject) **%.%** summarize(totalcount = sum(count))

Example 5: order data

Classic

```
D[order(D$subject, D$trial),]
```

dplyr

```
arrange(D, subject, trial)
```

DPLYR

Pipelining

Piping commands with %.%

- select(D, subject, trial, count, time) %.%
  manipulate(freq = count/time) %.%
  group_by(subject) %.%
  summarize(avgfreq = mean(freq))
- You can even:
  D %.% select(subject, trial, count, time) %.% ...
- Or:
  read.spss(“D.sav“, to.data.frame = T) %.%
  select(subject, trial, count, time)

Interfacing with ggplot

```
D%.%
mutate(freq = count/time) %.%
ggplot(aes(participant, freq))
```

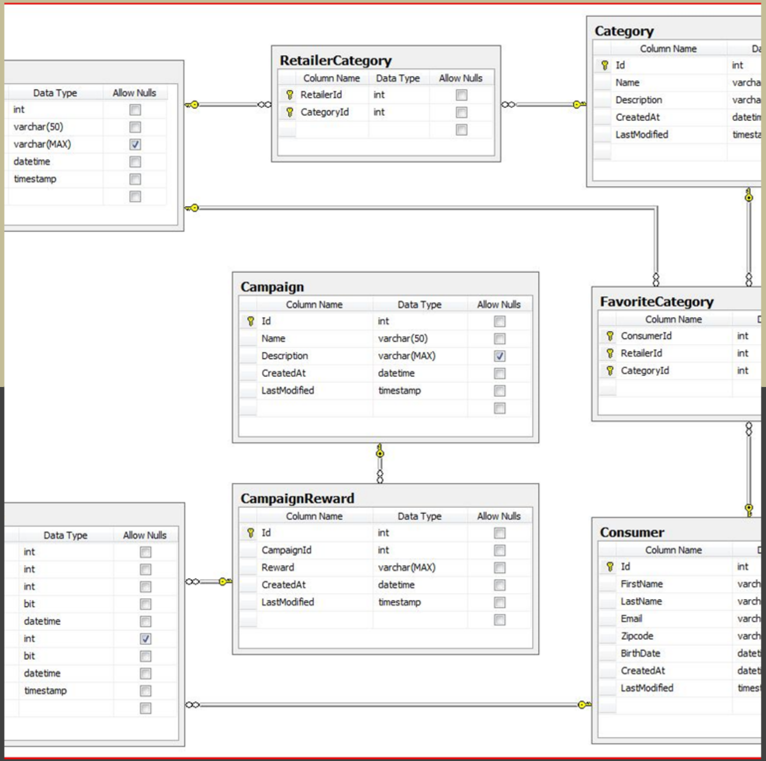
This seems to work for all functions that take the data.rame as first argument

Interfacing with lm

lm(formula, data) ← data.frame not first arg

New in dplyr: **%>%** (magrittr)

```
D%>%
mutate(freq = count/time) %>%
lm(freq ~ age, data = .) %>%
summary()
```



Data Type	Allow Nulls
int	
varchar(50)	
varchar(MAX)	✓
datetime	
timestamp	

RetailerCategory

	Column Name	Data Type	Allow Nulls
🔑	RetailerId	int	
🔑	CategoryId	int	

Category

	Column Name	Da
🔑	Id	int
	Name	varcha
	Description	varcha
	CreatedAt	datetim
	LastModified	timesta

Campaign

	Column Name	Data Type	Allow Nulls
🔑	Id	int	
	Name	varchar(50)	
	Description	varchar(MAX)	✓
	CreatedAt	datetime	
	LastModified	timestamp	

FavoriteCategory

	Column Name	D
🔑	ConsumerId	int
🔑	RetailerId	int
🔑	CategoryId	int

Data Type	Allow Nulls
int	
int	
int	
bit	
datetime	
int	✓
bit	
datetime	
timestamp	

CampaignReward

	Column Name	Data Type	Allow Nulls
🔑	Id	int	
	CampaignId	int	
	Reward	varchar(MAX)	
	CreatedAt	datetime	
	LastModified	timestamp	

Consumer

	Column Name	D
🔑	Id	int
	FirstName	varch
	LastName	varch
	Email	varch
	Zipcode	varch
	BirthDate	datet
	CreatedAt	datet
	LastModified	times

PRINCIPLES OF DATA MODELLING

Tables, Keys, Redundancy

Subj	Gender	Word	Picture	RT
1	m	speak	robo 1	352
1	m	speak	robo 2	789
1	m	beep	robo 1	435
1	m	beep	robo 2	978
2	f	speak	robo 1	1423
2	f	speak	robo 2	1453
2	f	beep	robo 1	983
2	f	beep	robo 2	1234

- **Key:** smallest combination of variables that identifies an observation
- **Redundancy:** value of one column is strictly determined by another column

Avoiding redundancy

Subj		Word	Picture	RT
1		speak	robo 1	352
1		speak	robo 2	789
1		beep	robo 1	435
1		beep	robo 2	978
2		speak	robo 1	1423
2		speak	robo 2	1453
2		beep	robo 1	983
2		beep	robo 2	1234

Subj	Gender
1	m
2	f

re-joining

```
join(ExperimentalData,
     SubjectData,
     by = Subj)
```

In data modeling speak, this is a 1:n relation with Subj as foreign key

join commands supplied by plyr

Subj	Gender	Word	Picture	RT
1	m	speak	robo 1	352
1	m	speak	robo 2	789
1	m	beep	robo 1	435
1	m	beep	robo 2	978
2	f	speak	robo 1	1423
2	f	speak	robo 2	1453
2	f	beep	robo 1	983
2	f	beep	robo 2	1234

Tipps & Tricks

plyr and dplyr

- Always load plyr first
  library(plyr)
  library(dplyr)
- If accidentally done wrong: Restart R
- plyr even warns you

Renaming a column

Classic

```
D <- read.spss(“D.sav”, to.data.frame = T)

D$participant <- D$proefpersoon

D$proefpersoon <- NULL
```

dplyr

```
D <- read.spss(“D.sav”, to.data.frame = T) %.%

      select(participant = proefpersoon, trial:time)
```