Join Data with

Slides CC BY-SA RStudio

Details

⚠ The class of service you searched may not be available on one or more flights

BNA - ORD
Flight 1 of 2

ORD - YVR
Flight 2 of 2

Nashville, TN to Chicago, IL

Thursday, July 26, 2018

4:10 PM → 6:03 PM

AA 3246 ▪ CR7-Canadair RJ 700

Operated by SkyWest Airlines As American Eagle

Travel info

Travel time: **1h 53m**
Connection time: **2h 33m**

Performanc

On time: **52**
Late: **43%**

Main Cabin

Meals: Beverage service
Booking code: V
Class: Economy

Business

Meals: Beverage service
Booking code: I
Class: First

* This is based on information from the month of May 2018
** The on-time arrival percentage for the selected flight is based on arrival within 14 minutes after

Performance*

On time: **52%****

Late: **43%**

** The on-time arrival percentage for the selected flight is based on arrival within 14 minutes after the scheduled arrival as reported monthly to the U.S. Department of Transportation.

nycflights13

Data about every flight that departed La Guardia, JFK, or Newark airports in 2013

```
# install.packages("nycflights13")
library(nycflights13)
```

nycflights13

Flights

```
View(flights)
```

year	month	day	dep_time	sched_dep_time	dep_delay	arr_time	sched_arr_time	arr_delay	carrier	flight	tailnum	origin	dest	air_time	distance	hour	minute	time_hour
2013	1	1	517	515	2	830	819	11	UA	1545	N14228	EWR	IAH	227	1400	5	15	2013-01-01 05:00:00
2013	1	1	533	529	4	850	830	20	UA	1714	N24211	LGA	IAH	227	1416	5	29	2013-01-01 05:00:00
2013	1	1	542	540	2	923	850	33	AA	1141	N619AA	JFK	MIA	160	1089	5	40	2013-01-01 05:00:00
2013	1	1	544	545	-1	1004	1022	-18	B6	725	N804JB	JFK	BQN	183	1576	5	45	2013-01-01 05:00:00
2013	1	1	554	600	-6	812	837	-25	DL	461	N668DN	LGA	ATL	116	762	6	0	2013-01-01 06:00:00
2013	1	1	554	558	-4	740	728	12	UA	1696	N39463	EWR	ORD	150	719	5	58	2013-01-01 05:00:00
2013	1	1	555	600	-5	913	854	19	B6	507	N516JB	EWR	FLL	158	1065	6	0	2013-01-01 06:00:00
2013	1	1	557	600	-3	709	723	-14	EV	5708	N829AS	LGA	IAD	53	229	6	0	2013-01-01 06:00:00
2013	1	1	557	600	-3	838	846	-8	B6	79	N593JB	JFK	MCO	140	944	6	0	2013-01-01 06:00:00
2013	1	1	558	600	-2	753	745	8	AA	301	N3ALAA	LGA	ORD	138	733	6	0	2013-01-01 06:00:00
2013	1	1	558	600	-2	849	851	-2	B6	49	N793JB	JFK	PBI	149	1028	6	0	2013-01-01 06:00:00
2013	1	1	558	600	-2	853	856	-3	B6	71	N657JB	JFK	TPA	158	1005	6	0	2013-01-01 06:00:00
2013	1	1	558	600	-2	924	917	7	UA	194	N29129	JFK	LAX	345	2475	6	0	2013-01-01 06:00:00
2013	1	1	558	600	-2	923	937	-14	UA	1124	N53441	EWR	SFO	361	2565	6	0	2013-01-01 06:00:00
2013	1	1	559	600	-1	941	910	31	AA	707	N3DUAA	LGA	DFW	257	1389	6	0	2013-01-01 06:00:00
2013	1	1	559	559	0	702	706	-4	B6	1806	N708JB	JFK	BOS	44	187	5	59	2013-01-01 05:00:00
2013	1	1	559	600	-1	854	902	-8	UA	1187	N76515	EWR	LAS	337	2227	6	0	2013-01-01 06:00:00
2013	1	1	600	600	0	851	858	-7	B6	371	N595JB	LGA	FLL	152	1076	6	0	2013-01-01 06:00:00

Flights

What airlines have the longest delays?

carrier	avg_delay
9E	?
AA	?
AS	?
B6	?
DL	?
EV	?
F9	?
FL	?

name	avg_delay
AirTran Airways Corporation	?
Alaska Airlines Inc.	?
American Airlines Inc.	?
Delta Air Lines Inc.	?
Endeavor Air Inc.	?
Envoy Air	?
ExpressJet Airlines Inc.	?
Frontier Airlines Inc.	?

airlines

flights

carrier	avg_delay
9E	?
AA	?
AS	?
B6	?
DL	?
EV	?
F9	?
FL	?

name	avg_delay
AirTran Airways Corporation	?
Alaska Airlines Inc.	?
American Airlines Inc.	?
Delta Air Lines Inc.	?
Endeavor Air Inc.	?
Envoy Air	?
ExpressJet Airlines Inc.	?
Frontier Airlines Inc.	?

Airline names

```
View(flights)
```

arr_delay	carrier
11	UA
20	UA
33	AA
-18	B6
-25	DL
12	UA

```
View(airlines)
```

carrier	name
9E	Endeavor Air Inc.
AA	American Airlines Inc.
AS	Alaska Airlines Inc.
B6	JetBlue Airways
DL	Delta Air Lines Inc.
EV	ExpressJet Airlines Inc.

Airline names

```
View(flights)
```

arr_delay	carrier	name
11	UA	Endeavor Air Inc.
20	UA	American Airlines Inc.
33	AA	Alaska Airlines Inc.
-18	B6	JetBlue Airways
-25	DL	Delta Air Lines Inc.
12	UA	ExpressJet Airlines Inc.

Airline names

```
View(flights)
```

arr_delay	carrier	name
11	UA	Endeavor Air Inc.
20	UA	American Airlines Inc.
33	AA	Alaska Airlines Inc.
-18	B6	JetBlue Airways
-25	DL	Delta Air Lines Inc.
12	UA	ExpressJet Airlines Inc.

mutating joins

Toy data

```r
band <- tribble(
   ~name,      ~band,
  "Mick",   "Stones",
  "John",  "Beatles",
  "Paul",  "Beatles"
)
```

band

name	band
Mick	Stones
John	Beatles
Paul	Beatles

```r
instrument <- tribble(
    ~name,    ~plays,
   "John",  "guitar",
   "Paul",    "bass",
  "Keith",  "guitar"
)
```

instrument

name	plays
John	guitar
Paul	bass
Keith	guitar

Toy data

band

name	band
Mick	Stones
John	Beatles
Paul	Beatles

instrument

name	plays
John	guitar
Paul	bass
Keith	guitar

The pipe operator %>%

```
instrument   %>%   left_join(_______, by = "name")
```

Passes result on left into first argument of function on right.
So, for example, these do the same thing. Try it.

```
left_join(instrument, by = "name")
instrument %>% left_join(by = "name")
```

left

```
band %>% left_join(instrument, by = "name")
```

band

name	band
Mick	Stones
John	Beatles
Paul	Beatles

+

instrument

name	plays
John	guitar
Paul	bass
Keith	guitar

=

name	band	plays
Mick	Stones	<NA>
John	Beatles	guitar
Paul	Beatles	bass

left

```
band %>% left_join(instrument, by = "name")
```

band

name	band
Mick	Stones
John	Beatles
Paul	Beatles

+

instrument

name	plays
John	guitar
Paul	bass
Keith	guitar

=

name	band	plays
Mick	Stones	<NA>
John	Beatles	guitar
Paul	Beatles	bass

right

```
band %>% right_join(instrument, by = "name")
```

band

name	band
Mick	Stones
John	Beatles
Paul	Beatles

+

instrument

name	plays
John	guitar
Paul	bass
Keith	guitar

=

name	band	plays
John	Beatles	guitar
Paul	Beatles	bass
Keith	<NA>	guitar

full

```
band %>% full_join(instrument, by = "name")
```

band

name	band
Mick	Stones
John	Beatles
Paul	Beatles

+

instrument

name	plays
John	guitar
Paul	bass
Keith	guitar

=

name	band	plays
Mick	Stones	<NA>
John	Beatles	guitar
Paul	Beatles	bass
Keith	<NA>	guitar

inner

```
band %>% inner_join(instrument, by = "name")
```

band

name	band
Mick	Stones
John	Beatles
Paul	Beatles

+

instrument

name	plays
John	guitar
Paul	bass
Keith	guitar

=

name	band	plays
John	Beatles	guitar
Paul	Beatles	bass

Airline names

```
View(flights)
```

arr_delay	carrier
11	UA
20	UA
33	AA
-18	B6
-25	DL
12	UA

```
View(airlines)
```

carrier	name
9E	Endeavor Air Inc.
AA	American Airlines Inc.
AS	Alaska Airlines Inc.
B6	JetBlue Airways
DL	Delta Air Lines Inc.
EV	ExpressJet Airlines Inc.

Your Turn 1

Which airlines had the largest arrival delays? Work in groups to complete the code below.

```
flights %>%
  filter(!is.na(arr_delay)) %>%
  ______________________ %>%
  group_by(_________) %>%
  ______________________ %>%
  arrange(_______)
```

1. Join airlines to flights

2. Compute and order the average arrival delays by airline. Display full names, no codes.

```r
flights %>%
  filter(!is.na(arr_delay)) %>%
  left_join(airlines, by = "carrier") %>%
  group_by(name) %>%
  summarise(delay = mean(arr_delay)) %>%
  arrange(delay)
```

```
## # A tibble: 16 × 2
##                          name      delay
##                         <chr>      <dbl>
## 1         Alaska Airlines Inc. -9.9308886
## 2       Hawaiian Airlines Inc. -6.9152047
## 3       American Airlines Inc.  0.3642909
## 4         Delta Air Lines Inc.  1.6443409
## 5               Virgin America  1.7644644
```

Toy data

```r
band <- tribble(
   ~name,      ~band,
  "Mick",   "Stones",
  "John",  "Beatles",
  "Paul",  "Beatles"
)
```

band

name	band
Mick	Stones
John	Beatles
Paul	Beatles

```r
instrument2 <- tribble(
    ~artist,    ~plays,
   "John",  "guitar",
   "Paul",    "bass",
  "Keith",  "guitar"
)
```

instrument2

artist	plays
John	guitar
Paul	bass
Keith	guitar

What if the names do not match?

Use a named vector to match on variables with different names.

```
band %>% left_join(instrument2, by = c("name" = "artist"))
```

A named vector

The name of the element = the column name in the first data set

The value of the element = the column name in the second data set

What if the names do not match?

Use a named vector to match on variables with different names.

```
band %>% left_join(instrument2, by = c("name" = "artist"))
```

band

name	band
Mick	Stones
John	Beatles
Paul	Beatles

+

instrument2

artist	plays
John	guitar
Paul	bass
Keith	guitar

=

name	band	plays
Mick	Stones	<NA>
John	Beatles	guitar
Paul	Beatles	bass

Airport names

```
flights %>% select(14:15)
```

dest <chr>	air_time <dbl>
IAH	227
IAH	227
MIA	160
BQN	183
ATL	116
ORD	150
FLL	158
IAD	53

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

faa <chr>	name <chr>
04G	Lansdowne Airport
06A	Moton Field Municipal Airport
06C	Schaumburg Regional
06N	Randall Airport
09J	Jekyll Island Airport
0A9	Elizabethton Municipal Airport
0G6	Williams County Airport
0G7	Finger Lakes Regional Airport

common syntax - matching names

```
flights %>% left_join(airports, by = c("dest" = "faa"))
```

dest <chr>	air_time <dbl>
IAH	227
IAH	227
MIA	160
BQN	183
ATL	116
ORD	150
FLL	158
IAD	53

faa <chr>	name <chr>
04G	Lansdowne Airport
06A	Moton Field Municipal Airport
06C	Schaumburg Regional
06N	Randall Airport
09J	Jekyll Island Airport
0A9	Elizabethton Municipal Airport
0G6	Williams County Airport
0G7	Finger Lakes Regional Airport

Your Turn 2

Join **flights** and **airports** by **dest** and **faa**.

Then **for each name**, compute the **distance** from NYC and the average **arr_delay**. Hint: use first() to get the first value of distance.

Order by average delay, worst to best.

```r
flights %>%
  filter(!is.na(arr_delay)) %>%
  left_join(airports, by = c("dest" = "faa")) %>%
  group_by(name) %>%
  summarise(distance = first(distance),
    delay = mean(arr_delay)) %>%
  arrange(desc(delay))
## # A tibble: 101 × 3
##                                 name  distance    delay
##                                <chr>    <dbl>    <dbl>
## 1           Columbia Metropolitan       602 41.76415
## 2                     Tulsa Intl      1215 33.65986
## 3               Will Rogers World      1325 30.61905
```

filtering joins

semi

```
band %>% semi_join(instrument, by = "name")
```

band

name	band
Mick	Stones
John	Beatles
Paul	Beatles

+

instrument

name	plays
John	guitar
Paul	bass
Keith	guitar

=

name	band
John	Beatles
Paul	Beatles

anti

```
band %>% anti_join(instrument, by = "name")
```

band

name	band
Mick	Stones
John	Beatles
Paul	Beatles

+

instrument

name	plays
John	guitar
Paul	bass
Keith	guitar

=

name	band
Mick	Stones

Airport names

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

faa <chr>	name <chr>
04G	Lansdowne Airport
06A	Moton Field Municipal Airport
06C	Schaumburg Regional
06N	Randall Airport
09J	Jekyll Island Airport
0A9	Elizabethton Municipal Airport
0G6	Williams County Airport
0G7	Finger Lakes Regional Airport

```
flights %>% select(14:15)
```

dest <chr>	air_time <dbl>
IAH	227
IAH	227
MIA	160
BQN	183
ATL	116
ORD	150
FLL	158
IAD	53

Your Turn 3

How many airports in **airports** are serviced by flights in **flights**? (i.e. how many places can you fly to direct from New York?)

Notice that the column to filter on is named **faa** in the **airports** dataset and **dest** in the **flights** dataset.

05:00

```
airports %>%
  semi_join(flights, by = c("faa" = "dest")) %>%
  select(faa)
```

faa <chr>
IAH
MIA
ATL
ORD
FLL
IAD
MCO
PBI
TPA
LAX

1–10 of 101 rows

Previous 1 2 3 4 5 6 ... 11 Next

Recap: Two table verbs

left_join() retains all cases in **left** data set

right_join() retains all cases in **right** data set

full_join() retains all cases in **either** data set

inner_join() retains only cases in **both** data sets

semi_join() extracts cases that **have a match**

anti_join() extracts cases that **do not have a match**

Two table verbs

Data Transformation cheatsheet

on back

Combine Tables

COMBINE VARIABLES

x

A	B	C
a	t	1
b	u	2
c	v	3

+

y

A	B	D
a	t	3
b	u	2
d	w	1

=

A	B	C	A	B	D
a	t	1	a	t	3
b	u	2	b	u	2
c	v	3	d	w	1

Use **bind_cols()** to paste tables beside each other as they are.

bind_cols(...) Returns tables placed side by side as a single table.
BE SURE THAT ROWS ALIGN.

Use a "**Mutating Join**" to join one table to columns from another, matching values with the rows that they correspond to. Each join retains a different combination of values from the tables.

A	B	C	D
a	t	1	3
b	u	2	2
c	v	3	NA

left_join(x, y, by = NULL, copy=FALSE, suffix=c(".x",".y"),...)
Join matching values from y to x.

A	B	C	D
a	t	1	3
b	u	2	2
d	w	NA	1

right_join(x, y, by = NULL, copy = FALSE, suffix=c(".x",".y"),...)
Join matching values from x to y.

A	B	C	D
a	t	1	3
b	u	2	2

inner_join(x, y, by = NULL, copy = FALSE, suffix=c(".x",".y"),...)
Join data. Retain only rows with matches.

A	B	C	D
a	t	1	3
b	u	2	2
c	v	3	NA
d	w	NA	1

full_join(x, y, by = NULL, copy=FALSE, suffix=c(".x",".y"),...)
Join data. Retain all values, all rows.

A	B.x	C	B.y	D
a	t	1	t	3
b	u	2	u	2
c	v	3	NA	NA

Use **by = c("col1", "col2", ...)** to specify one or more common columns to match on.
left_join(x, y, by = "A")

A.x	B.x	C	A.y	B.y
a	t	1	d	w
b	u	2	b	u
c	v	3	a	t

Use a named vector, **by = c("col1" = "col2")**, to match on columns that have different names in each table.
left_join(x, y, by = c("C" = "D"))

A1	B1	C	A2	B2
a	t	1	d	w
b	u	2	b	u
c	v	3	a	t

Use **suffix** to specify the suffix to give to unmatched columns that have the same name in both tables.
left_join(x, y, by = c("C" = "D"), suffix = c("1", "2"))

COMBINE CASES

x

A	B	C
a	t	1
b	u	2
c	v	3

+ y

A	B	C
c	v	3
d	w	4

Use **bind_rows()** to paste tables below each other as they are.

DF	A	B	C
x	a	t	1
x	b	u	2
x	c	v	3
z	c	v	3
z	d	w	4

bind_rows(..., .id = NULL)
Returns tables one on top of the other as a single table. Set .id to a column name to add a column of the original table names (as pictured)

A	B	C
c	v	3

intersect(x, y, ...)
Rows that appear in both x and y.

A	B	C
a	t	1
b	u	2

setdiff(x, y, ...)
Rows that appear in x but not y.

A	B	C
a	t	1
b	u	2
c	v	3
d	w	4

union(x, y, ...)
Rows that appear in x or y. (Duplicates removed). union_all() retains duplicates.

Use **setequal()** to test whether two data sets contain the exact same rows (in any order).

EXTRACT ROWS

x

A	B	C
a	t	1
b	u	2
c	v	3

+

y

A	B	D
a	t	3
b	u	2
d	w	1

=

Use a "**Filtering Join**" to filter one table against the rows of another.

A	B	C
a	t	1
b	u	2

semi_join(x, y, by = NULL, ...)
Return rows of x that have a match in y. USEFUL TO SEE WHAT WILL BE JOINED.

A	B	C
c	v	3

anti_join(x, y, by = NULL, ...)
Return rows of x that do not have a match in y. USEFUL TO SEE WHAT WILL NOT BE JOINED.

Join Data with

Slides CC BY-SA RStudio