



JOINING DATA WITH DATA.TABLE IN R

# Complex keys

Scott Ritchie

Postdoctoral Researcher in Systems Genomics

# Misspecified joins

What happens when you don't use the correct columns for join keys?

- An error is thrown
- The result is a malformed `data.table`

# Column type mismatch

Using join key columns with different types will error

```
customers[web_visits, on = .(age = name)]
```

```
Error in bmerge(i, x, leftcols, rightcols, io, xo, roll, rollends,
nomatch,  :
  typeof x.age (double) != typeof i.name (character)
```

customers:

name	gender	age	address
Madeline Martin	F	54	5 Market lane
Madeline Bernard	F	45	4 Jacaranda crescent
George Dimakos	M	39	2a Park square

+

web\_visits:

name	date	duration
Madeline Martin	2018-05-02	5
Madeline Martin	2018-05-03	32
Madeline Bernard	2018-05-03	12
George Dimakos	2018-04-27	45

# Column type mismatch

```
customers[web_visits, on = .(id)]
```

```
Error in bmerge(i, x, leftcols, rightcols, io, xo, roll, rollends,
nomatch,  :
  typeof x.id (integer) != typeof i.id(character)
```

customers:

id	name	gender	age	address
1	"Madeline Martin"	"F"	54	"5 Market lane"
2	"Madeline Bernard"	"F"	45	"4 Jacaranda crescent"
3	"George Dimakos"	"M"	39	"2a Park square"

+

web\_visits:

id	name	date	duration
"1"	"Madeline Martin"	2018-05-02	5
"1"	"Madeline Martin"	2018-05-03	32
"2"	"Madeline Bernard"	2018-05-03	12
"3"	"George Dimakos"	2018-04-27	45

# Malformed full joins - no common key values

```
merge(customers, web_visits, by.x = "address", by.y = "name", all = TRUE)
```

customers:

name	gender	age	address
Madeline Martin	F	54	5 Market lane
Madeline Bernard	F	45	4 Jacaranda crescent
George Dimakos	M	39	2a Park square



web\_visits:

name	date	duration
Madeline Martin	2018-05-02	5
Madeline Martin	2018-05-03	32
Madeline Bernard	2018-05-03	12
George Dimakos	2018-04-27	45



address	name	gender	age	date	duration
2a Park square	George Dimakos	M	39	NA	NA
4 Jacaranda crescent	Madeline Bernard	F	45	NA	NA
5 Market lane	Madeline Martin	F	54	NA	NA
George Dimakos	NA	NA	NA	2018-04-27	45
Madeline Bernard	NA	NA	NA	2018-05-03	12
Madeline Martin	NA	NA	NA	2018-05-02	5
Madeline Martin	NA	NA	NA	2018-05-03	32

# Malformed right and left joins - no common key values

```
customers[web_visits, on = .(address = name)]
```

customers:

name	gender	age	address
Madeline Martin	F	54	5 Market lane
Madeline Bernard	F	45	4 Jacaranda crescent
George Dimakos	M	39	2a Park square

+

web\_visits:

name	date	duration
Madeline Martin	2018-05-02	5
Madeline Martin	2018-05-03	32
Madeline Bernard	2018-05-03	12
George Dimakos	2018-04-27	45

=

name	gender	age	address	date	duration
NA	NA	NA	Madeline Martin	2018-05-02	5
NA	NA	NA	Madeline Martin	2018-05-03	32
NA	NA	NA	Madeline Bernard	2018-05-03	12
NA	NA	NA	George Dimakos	2018-04-27	45

# Malformed inner joins - no common key values

```
customers[web_visits, on = .(address = name), nomatch = 0]
```

customers:

name	gender	age	address
Madeline Martin	F	54	5 Market lane
Madeline Bernard	F	45	4 Jacaranda crescent
George Dimakos	M	39	2a Park square

+

web\_visits:

name	date	duration
Madeline Martin	2018-05-02	5
Madeline Martin	2018-05-03	32
Madeline Bernard	2018-05-03	12
George Dimakos	2018-04-27	45

=

name	gender	age	address	date	duration
------	--------	-----	---------	------	----------

# Malformed joins - coincidental common key values

```
customers[web_visits, on = .(age = duration), nomatch = 0]
```

customers:

name	gender	age	address
Madeline Martin	F	54	5 Market lane
Madeline Bernard	F	45	4 Jacaranda crescent
George Dimakos	M	39	2a Park square

+

web\_visits:

name	date	duration
Madeline Martin	2018-05-02	5
Madeline Martin	2018-05-03	32
Madeline Bernard	2018-05-03	12
George Dimakos	2018-04-27	45

=

name	gender	age	address	i.name	date
Madeline Bernard	F	45	4 Jacaranda crescent	George Dimakos	2018-04-27

# Avoiding misspecified joins

*Learning what each column represents before joins will help you avoid errors*

# Keys with different column names

customers:

name	gender	age	address
Madeline Martin	F	54	5 Market lane
Madeline Bernard	F	45	4 Jacaranda crescent
George Dimakos	M	39	2a Park square

web\_visits:

person	date	duration
Madeline Martin	2018-05-02	5
Madeline Martin	2018-05-03	32
Madeline Bernard	2018-05-03	12
George Dimakos	2018-04-27	45

```
merge(customers, web_visits, by.x = "name", by.y = "person")
```

```
customers[web_visits, on = .(name = person)]
```

```
customers[web_visits, on = c("name" = "person")]
```

```
key <- c("name" = "person")
```

```
customers[web_visits, on = key]
```



# Multi-column keys

customers:

first	last	gender	age	address
Madeline	Martin	F	54	5 Market lane
Madeline	Bernard	F	45	4 Jacaranda crescent
George	Dimakos	M	39	2a Park square

web\_visits:

first	last	date	duration
Madeline	Martin	2018-05-02	5
Madeline	Martin	2018-05-03	32
Madeline	Bernard	2018-05-03	12
George	Dimakos	2018-04-27	45



# Multi-column keys

purchases:

name	date	item	units	price
Madeline Martin	2018-05-03	book	2	\$15.00
Arthur Smith	2018-05-03	shelf	1	\$30.00
Jaqueline Mary	2018-05-03	CD	1	\$12.00
George Dimakos	2018-05-03	plant	3	\$16.00
George Dimakos	2018-04-27	shelf	1	\$30.00

web\_visits:

name	date	duration
Madeline Martin	2018-05-02	5
Madeline Martin	2018-05-03	32
Madeline Bernard	2018-05-03	12
George Dimakos	2018-04-27	45

# Specifying multiple keys with merge()

```
merge(purchases, web_visits, by = c("name", "date"))
```

```
merge(purchases, web_visits,  
      by.x = c("name", "date"),  
      by.y = c("person", "date"))
```

# Specifying multiple keys with the data.table syntax

```
purchases[web_visits, on = .(name, date)]  
purchases[web_visits, on = c("name", "date")]
```

```
purchases[web_visits, on = .(name = person, date)]  
purchases[web_visits, on = c("name" = "person", "date")]
```



JOINING DATA WITH DATA.TABLE IN R

**Final Slide**



JOINING DATA WITH DATA.TABLE IN R

# Problem columns

Scott Ritchie

Postdoctoral Researcher in Systems Genomics

# Common column names

parents:

name	gender	age
Sarah	F	41
Max	M	43
Qin	F	36

children:

parent	name	gender	age
Sarah	Oliver	M	5
Max	Sebastian	M	8
Qin	Kai-lee	F	7

# Common column names

## Using the `data.table` syntax

```
parents[children, on = .(name = parent)]
```

	name	gender	age	i.name	i.gender	i.age
1:	Sarah	F	41	Oliver	M	5
2:	Max	M	43	Sebastian	M	8
3:	Qin	F	36	Kai-lee	F	7

# Common column names with merge()

Using the `merge()` function

```
merge(x = children, y = parents, by.x = "parent", by.y = "name")
```

	parent	name	gender.x	age.x	gender.y	age.y
1:	Max	Sebastian	M	8	M	43
2:	Qin	Kai-lee	F	7	F	36
3:	Sarah	Oliver	M	5	F	41

# Adding context with your own suffixes

The `suffixes` argument can add useful context:

```
merge(children, parents, by.x = "parent", by.y = "name",
      suffixes = c(".child", ".parent"))
```

	parent	name	gender.child	age.child	gender.parent	age.parent
1:	Max	Sebastian	M	8	M	43
2:	Qin	Kai-lee	F	7	F	36
3:	Sarah	Oliver	M	5	F	41

# Renaming columns

Rename all columns using `setnames()`

```
setnames(parents, c("parent", "parent.gender", "parent.age"))
```

```
setnames(parents, old = c("gender", "age"),  
          new = c("parent.gender", "parent.age"))
```

```
parents  
  
  parent parent.gender parent.age  
1:  Sarah             F           41  
2:   Max             M           43  
3:   Qin             F           36
```

# Joining with data.frames

Join keys for `data.frames` may be in the rownames

```
# A data.frame
parents
  gender age
Sarah   F  41
Max     M  43
Qin     F  36
```

```
parents <- as.data.table(parents, keep.rownames = "parent")
parents
  parent gender age
1: Sarah   F  41
2:  Max    M  43
3:  Qin    F  36
```



## JOINING DATA WITH DATA.TABLE IN R

**Let's practice!**



JOINING DATA WITH DATA.TABLE IN R

# Duplicate matches

Scott Ritchie

Postdoctoral Researcher in Systems Genomics

# Join key duplicates

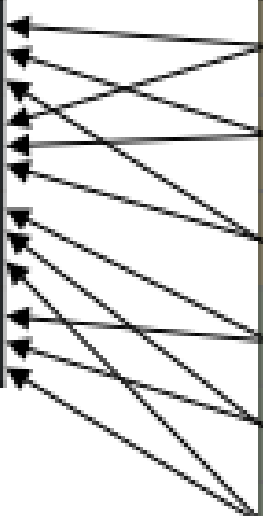
```
# Which bacteria could be found at both sites using any method?  
site1_ecology[site2_ecology, on = .(genus)]
```

site1\_ecology:

genus	count	method
<i>Nitrosomonas</i>	500	WGS
<i>Nitrosomonas</i>	620	16S
<i>Rhizobium</i>	360	WGS
<i>Rhizobium</i>	300	16S

site2\_ecology:

genus	present	method
<i>Nitrosomonas</i>	TRUE	WGS
<i>Nitrosomonas</i>	TRUE	16S
<i>Nitrosomonas</i>	TRUE	Culture
<i>Rhizobium</i>	TRUE	WGS
<i>Rhizobium</i>	TRUE	16S
<i>Rhizobium</i>	FALSE	Culture





# Error from multiplicative matches

```
site1_ecology[site2_ecology, on = .(genus)]
```

```
Error in vecseq(f__, len__, if (allow.cartesian || notjoin ||
```

```
!anyDuplicated(f__, :
```

```
Join results in 12 rows; more than 10 = nrow(x)+nrow(i). Check for  
duplicate key values in i each of which join to the same group in x over  
and over again. If that's ok, try by=.EACHI to run j for each group to  
avoid the large allocation. If you are sure you wish to proceed, rerun  
with allow.cartesian=TRUE. Otherwise, please search for this error message  
in the FAQ, Wiki, Stack Overflow and data.table issue tracker for advice.
```

# Allowing multiplicative matches

`allow.cartesian = TRUE` allows the join to proceed:

```
# data.table syntax  
site1_ecology[site2_ecology, on = .(genus), allow.cartesian = TRUE]
```

```
# merge()  
merge(site1_ecology, site2_ecology, by = "genus", allow.cartesian = TRUE)
```

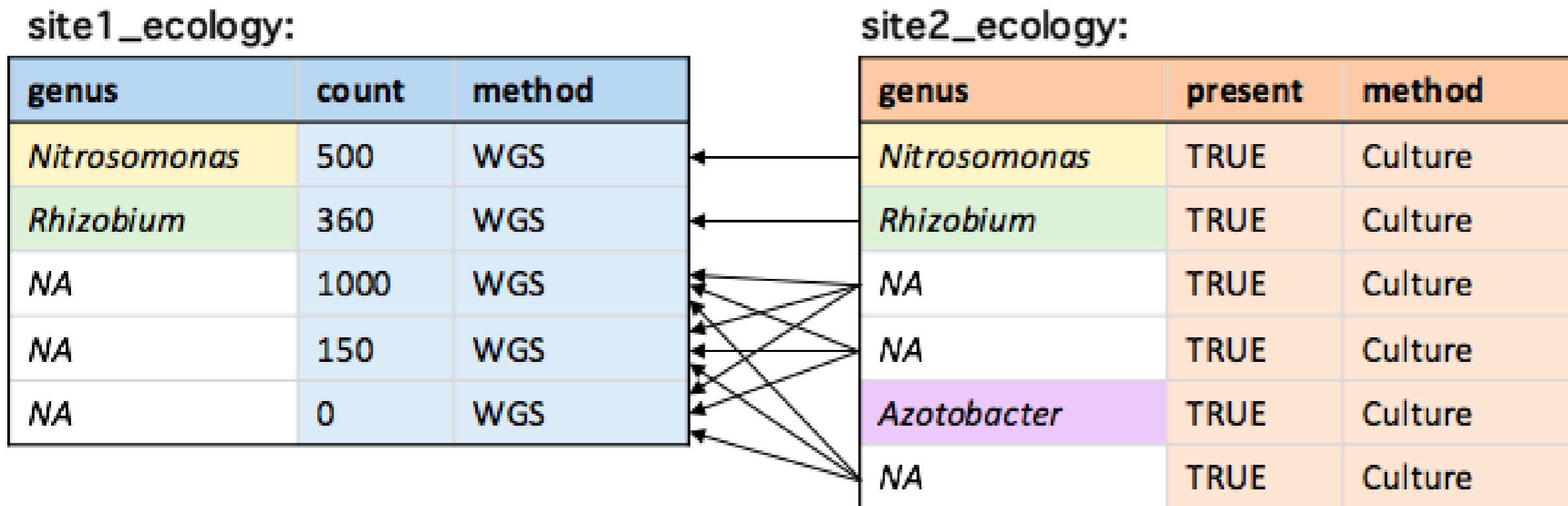
# Allowing multiplicative matches

```
site1_ecology[site2_ecology, on = .(genus), allow.cartesian = TRUE]
```

	genus	count	method	present	i.method
1:	Nitrosomonas	500	WGS	TRUE	WGS
2:	Nitrosomonas	620	16S	TRUE	WGS
3:	Nitrosomonas	500	WGS	TRUE	16S
4:	Nitrosomonas	620	16S	TRUE	16S
5:	Nitrosomonas	500	WGS	TRUE	Culture
6:	Nitrosomonas	620	16S	TRUE	Culture
7:	Rhizobium	360	WGS	TRUE	WGS
8:	Rhizobium	300	16S	TRUE	WGS
9:	Rhizobium	360	WGS	TRUE	16S
10:	Rhizobium	300	16S	TRUE	16S
11:	Rhizobium	360	WGS	FALSE	Culture
12:	Rhizobium	300	16S	FALSE	Culture

# Missing values

Missing values (`NA`) will match all other missing values:



# Filtering missing values

`!is.na()` can be used to filter rows with missing values

```
site1_ecology <- site1_ecology[!is.na(genus)]  
site1_ecology
```

	genus	count	method
1:	Nitrosomonas	500	WGS
2:	Rhizobium	360	WGS

```
site2_ecology <- site2_ecology[!is.na(genus)]  
site2_ecology
```

	genus	present	method
1:	Nitrosomonas	TRUE	Culture
2:	Rhizobium	TRUE	Culture
3:	Azotobacter	TRUE	Culture



# Keeping only the first match

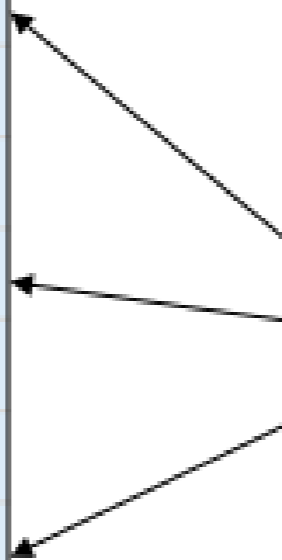
```
site1_ecology[site2_ecology, on = .(genus), mult = "first"]
```

site1\_ecology:

genus	Year	count	method
<i>Nitrosomonas</i>	2018	620	16S
<i>Nitrosomonas</i>	2017	603	16S
<i>Nitrosomonas</i>	2016	591	16S
<i>Rhizobium</i>	2018	290	16S
<i>Rhizobium</i>	2017	300	16S
<i>Rhizobium</i>	2016	280	16S
<i>Azotobacter</i>	2018	1230	16S
<i>Azotobacter</i>	2017	0	16S
<i>Azotobacter</i>	2016	0	16S

site2\_ecology:

genus	present	method
<i>Nitrosomonas</i>	TRUE	WGS
<i>Rhizobium</i>	TRUE	WGS
<i>Azotobacter</i>	FALSE	WGS





# Keeping only the last match

```
children[parents, on = .(parent = name), mult = "last"]
```

site1\_ecology:

genus	Year	count	method
<i>Nitrosomonas</i>	2018	620	16S
<i>Nitrosomonas</i>	2017	603	16S
<i>Nitrosomonas</i>	2016	591	16S
<i>Rhizobium</i>	2018	290	16S
<i>Rhizobium</i>	2017	300	16S
<i>Rhizobium</i>	2016	280	16S
<i>Azotobacter</i>	2018	1230	16S
<i>Azotobacter</i>	2017	0	16S
<i>Azotobacter</i>	2016	0	16S

site2\_ecology:

genus	present	method
<i>Nitrosomonas</i>	TRUE	WGS
<i>Rhizobium</i>	TRUE	WGS
<i>Azotobacter</i>	FALSE	WGS

# Identifying and removing duplicates

`duplicated()`: what rows are duplicates?

`unique()`: **filter** a `data.table` to just unique rows

# The duplicated() function

Using values in all columns:

```
duplicated(site1_ecology)
[1] FALSE FALSE FALSE FALSE
```

Using values in a subset of columns:

```
duplicated(site1_ecology,
           by = "genus")
[1] FALSE TRUE FALSE TRUE
```

site1\_ecology:

genus	count	method
<i>Nitrosomonas</i>	500	WGS
<i>Nitrosomonas</i>	620	16S
<i>Rhizobium</i>	360	WGS
<i>Rhizobium</i>	300	16S

# The unique() function

```
unique(site1_ecology, by = "genus")
```

site1\_ecology:

genus	count	method
<i>Nitrosomonas</i>	500	WGS
<i>Nitrosomonas</i>	620	16S
<i>Rhizobium</i>	360	WGS
<i>Rhizobium</i>	300	16S

X

X

genus	count	method
<i>Nitrosomonas</i>	500	WGS
<i>Rhizobium</i>	360	WGS

# Changing the search order

`fromLast = TRUE` changes the direction of the search to start from the last row

```
 duplicated(site1_ecology, by = "genus", fromLast = TRUE)
 [1] TRUE FALSE TRUE FALSE
 unique(site1_ecology, by = "genus", fromLast = TRUE)
```

site1\_ecology:

genus	count	method
<i>Nitrosomonas</i>	500	WGS
<i>Nitrosomonas</i>	620	16S
<i>Rhizobium</i>	360	WGS
<i>Rhizobium</i>	300	16S

genus	count	method
<i>Nitrosomonas</i>	620	16S
<i>Rhizobium</i>	300	16S



## JOINING DATA WITH DATA.TABLE IN R

**Let's practice!**