# Workshop 4:
# The tidyverse and beyond

- It's the end of base R as you know it


**Brendan Palmer,**
**Statistics & Data Analysis Unit,**
**Clinical Research Facility - Cork**

# The real world of data analysis (for most of us anyway)

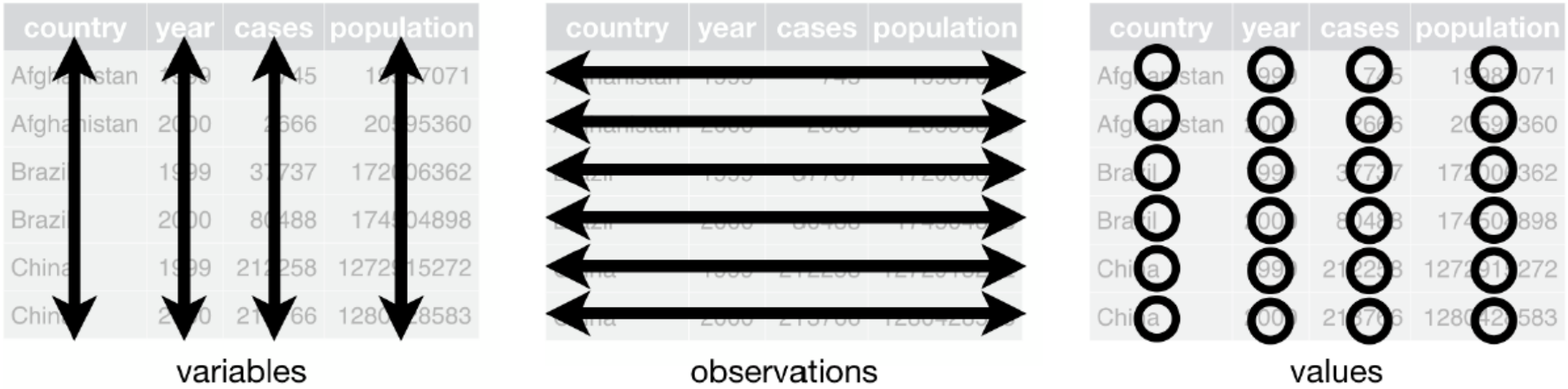
Before this evening

After this evening

# Data transformation with tidyr

- tidyr makes it easy to tidy your data and tidy data is easy to work with

- the two most important properties of tidy data are:
  1. Every column is a variable
  2. Every row is an observation



# Tidy data

- why isn't most data in the tidy format?
    - an obvious reason is that data collection is orientated around making its entry as easy as possible
- goal is to spend less time worrying about how to feed the output of one function to the input of another
- In this part of the workshop we will look at using some key tidyr functions:
        - separate()
        - gather()
        - spread()
        - unite()

# Recall: Basic code structure

Option 1:

```
new_object <- function(input_data, arguments)
```

Option 2:

```
new_object <- input_data %>%
    function(arguments)
```

new_object - assign the output to a new object

<- - the assign operator

%>% - the magrittr/pipe operator

function - the function you are calling on

input_data - the data supplied to the function

arguments - how you want to apply the function

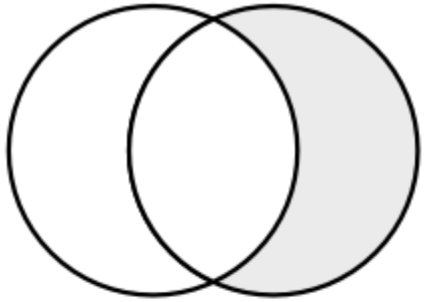
# Worksheet

# Open ws4_script1_working_with_tidyr.R

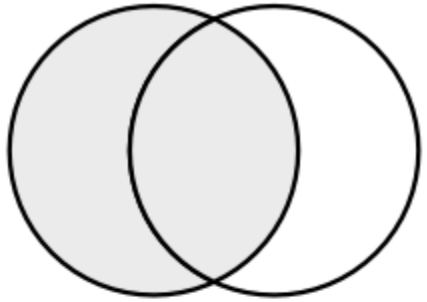
# Data transformation with dplyr

- Some key dplyr functions:
    - filter()
    - select()
    - mutate()
    - join()
    - summarise() or summarize()
        [depending on your grammatical upbringings!!]

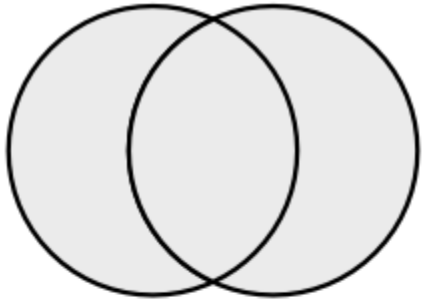
# Logical operators and conditional subsetting



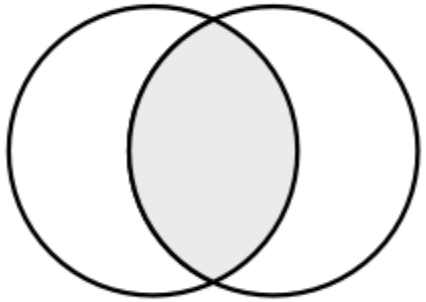
`y & !x`



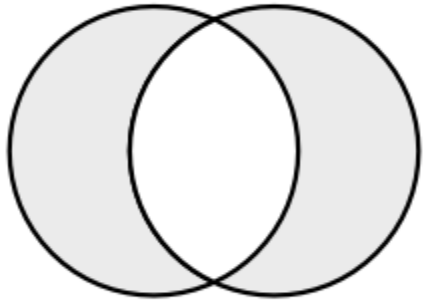
`x`



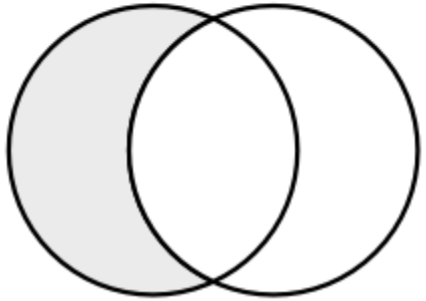
`x | y`



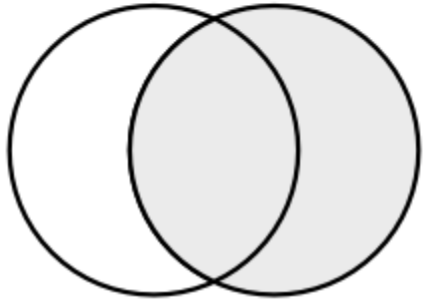
`x & y`



`xor(x, y)`



`x & !y`



`y`

- **`&` -> AND**
- **`|` -> OR (inclusive)**
- **`!` -> NOT**
- **`==` -> EQUAL**
- **`!=` -> NOT EQUAL**

Image: Hadley Wickham R for Data Science Chapter 3

# Worksheet

# Open ws4_script2_working_with_dplyr_partA.R

# A word of caution

- Computers use finite precision arithmetic

- Therefore all numbers are an approximation

- For this reason, instead of using == for numeric searches, use near()

- See lines 12-16 of the worksheet

# Last leg with dplyr

- summarise() or summarize() changes the analysis from the overall dataset to individual specified groups

```
> flights %>%
+ summarise(mean(dep_delay, na.rm = TRUE))
# A tibble: 1 x 1
  `mean(dep_delay, na.rm = TRUE)`
                            <dbl>
1                        12.63907
> |
```

- works best in conjunction with group_by()

```
> flights %>%
+ group_by(month) %>%
+ summarise(mean(dep_delay, na.rm = TRUE))
# A tibble: 12 x 2
   month `mean(dep_delay, na.rm = TRUE)`
   <int>                           <dbl>
 1     1                        10.036665
 2     2                        10.816843
 3     3                        13.227076
 4     4                        13.938038
 5     5                        12.986859
 6     6                        20.846332
 7     7                        21.727787
 8     8                        12.611040
 9     9                         6.722476
10    10                         6.243988
11    11                         5.435362
12    12                        16.576688
> |
```

# Worksheet

# Open ws4_script3_working_with_dplyr_partB.R

# Joining data frames

- Important to understand the chain of relations between the tables

- Variables used to connect each pair of tables are called **keys**

  - primary keys
  - foreign keys

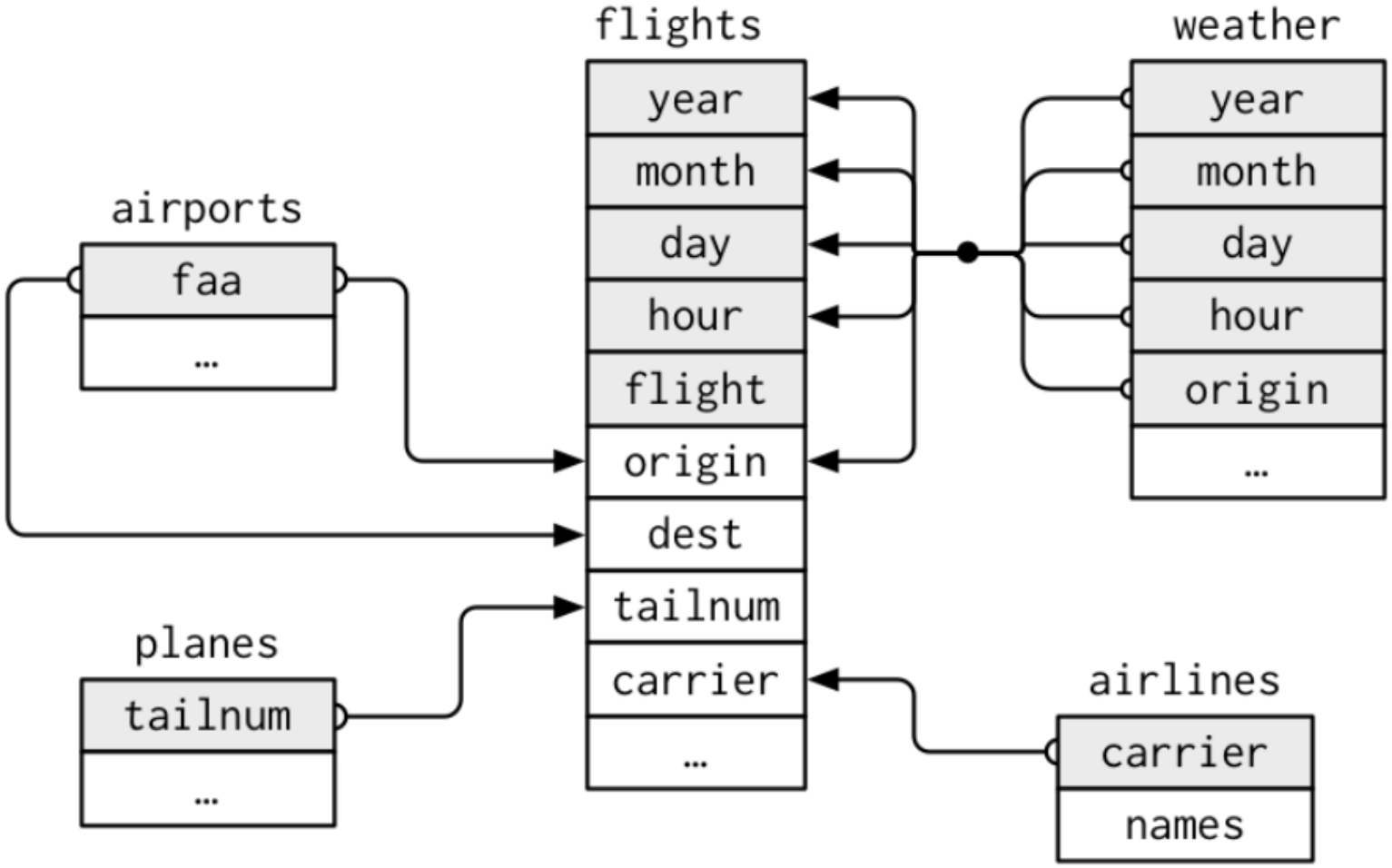
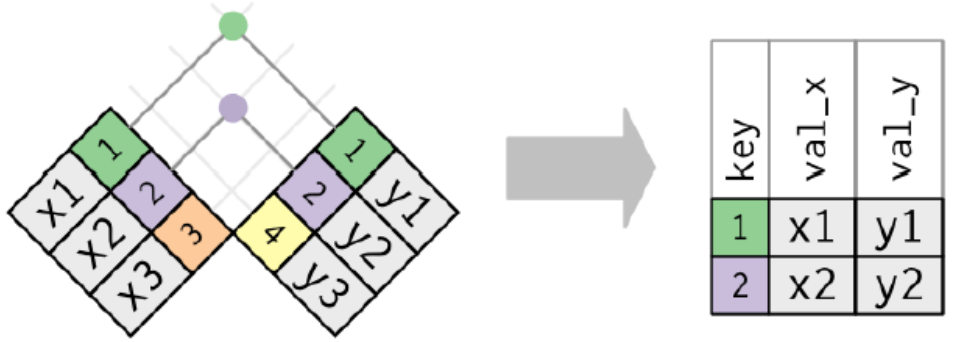



Image: Hadley Wickham R for Data Science Chapter 10

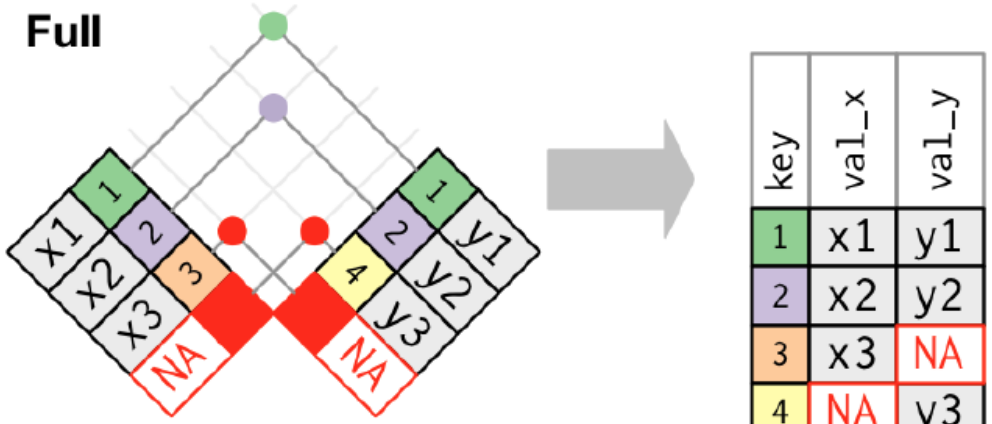
# Types of join

- Inner join: Matches pairs of observations whenever their keys are equal
- Unmatched rows are not included




| key | val_x | val_y |
|---|---|---|
| 1 | x1 | y1 |
| 2 | x2 | y2 |

- Outer joins:
- left join keeps all the observations in x
- right join keeps all the observations in y
- full join keeps all the observations in a and y




| key | val_x | val_y |
|---|---|---|
| 1 | x1 | y1 |
| 2 | x2 | y2 |
| 3 | x3 | NA |
| 4 | NA | y3 |

Image: Hadley Wickham R for Data Science Chapter 10

# Worksheet

# Open ws4_script4_working_with_dplyr_partC.R

# Introductory R Workshops

~~**Week 4 (6th March):**~~
~~**It's the end of base R as you know it**~~
~~- introduction to the tidyverse packages tidyr and dplyr~~

**Week 5 (13th March):**
**Welcome to the ggungle**
- analysis and visualisation of data

**Week 6 (27th March):**
**Don't look back in anger**
- writing clear code and making your work reproducible